OANet: Learning Two-View Correspondences and Geometry Using Order-Aware Network

Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, *Member, IEEE*, Hongkai Chen, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao, *Senior Member, IEEE*

Abstract—Establishing correct correspondences between two images should consider both local and global spatial context. Given putative correspondences of feature points in two views, in this paper, we propose Order-Aware Network, which infers the probabilities of correspondences being inliers and regresses the relative pose encoded by the essential or fundamental matrix. Specifically, this proposed network is built hierarchically and comprises three operations. First, to capture the local context of sparse correspondences, the network clusters unordered input correspondences by learning a soft assignment matrix. These clusters are in canonical order and invariant to input permutations. Next, the clusters are spatially correlated to encode the global context of correspondences. After that, the context-encoded clusters are interpolated back to the original size and position to build a hierarchical architecture. We intensively experiment on both outdoor and indoor datasets. The accuracy of the two-view geometry and correspondences are significantly improved over the state-of-the-arts. Besides, based on the proposed method and advanced local feature, we won the first place in CVPR 2019 image matching workshop challenge and also achieve state-of-the-art results in the Visual Localization benchmark. Code is available at https://github.com/zjhthu/OANet

Index Terms—Sparse matching, graph neural network, two-view geometry, structure-from-motion, visual localization

1 Introduction

Two-view geometry estimation is a fundamental problem in computer vision, which plays an important role in Structure from Motion (SfM) [1], [2], visual Simultaneous Localization and Mapping (SLAM) [3], visual localization [4], image stitching [5], *etc* By finding correct correspondences between images, the relative camera pose can be recovered through well-studied pose estimation solvers. The qualities of these downstream applications are heavily influenced by the accuracy of initial two-view geometry. Though various excellent hand-crafted [6], [7], [8] or learned local features [9], [10], [11], [12] have been proposed, false matching is still inevitable due to severe viewpoint/illumination change, repetitive texture, etc Therefore, distinguishing these outliers is crucial for accurate two-view pose estimation, which is the topic of this paper.

Putative correspondences distribution has intrinsic patterns because of motion smoothness, which means neighboring true matches usually have similar motion while outliers usually scatter [13]. There are various methods to leverage the motion smoothness property for outlier rejection, such as energy optimization [13], statistics [14], graphical model [15], *etc* Recently, this topic has also been revisited by deep learning methods [16], [17]. Giving unordered putative correspondences, recent works [16], [17] exploit PointNet-like architecture [18] and Context Normalization [16], [19] to classify putative correspondences. Although appealing results have been achieved, these works still have the following drawbacks: (1) PointNet-like architecture applies Multi-Layer Perceptrons (MLPs) on each point individually. Hence it cannot capture the local context [20], e.g., similar motion shared by neighboring pixels [14], which has been shown to be beneficial for outlier rejection [14], [15]. (2) Context Normalization is used to encode the global context which normalizes the feature maps by their mean and variance. However, this simple operation overlooks the underlying complex relations among different points and may hinder the overall performance.

One of the challenges in mitigating the above limitations is how to exploit neighboring cues to encode local context. Unlike 3D point clouds, sparse matches could have multiple heuristic definitions for the neighboring relationship, e.g, this issue is previously tackled in the bilateral domain [13] (2D spatial domain and 2D motion domain) or by a graphical model [15]. But it is not clear which definition is optimal for learning-based sparse matching. Besides, another challenge is how to model the complex relations between correspondences since they are unordered and have no stable relations to be captured. Only global relations captured by the Context Normalization might be not enough.

To address the above two problems, we draw inspiration from hierarchical representations of Graph Neural Network (GNN). In particular, we generalize the Differentiable Pooling

- *Jiahui Zhang and Hongen Liao are with the Department of Biomedical Engineering, Tsinghua University, Beijing 100084, China. E-mail: jiahui-z15@mails.tsinghua.edu.cn, liao@tsinghua.edu.cn.*
- *Dawei Sun, Anbang Yao, and Yurong Chen are with the Intel Labs China, Beijing, China. E-mail: {dawei.sun, anbang.yao, yurong.chen}@intel.com.*
- *Zixin Luo and Lei Zhou are with the Hong Kong University of Science and Technology, Hong Kong, and also with the Shenzhen Zhuke Innovation Technology (Altizure), Hong Kong. E-mail: {zluoag, lzhouai}@cse.ust.hk.*
- *Hongkai Chen, Tianwei Shen, and Long Quan are with the Hong Kong University of Science and Technology, Hong Kong. E-mail: {hchencf, tianwei, quan}@cse.ust.hk.*

Manuscript received 25 Sept. 2019; revised 9 Nov. 2020; accepted 14 Dec. 2020. Date of publication 29 Dec. 2020; date of current version 5 May 2022.
(Corresponding author: Hongen Liao.)
Recommended for acceptance by V. Lepetit.
Digital Object Identifier no. 10.1109/TPAMI.2020.3048013

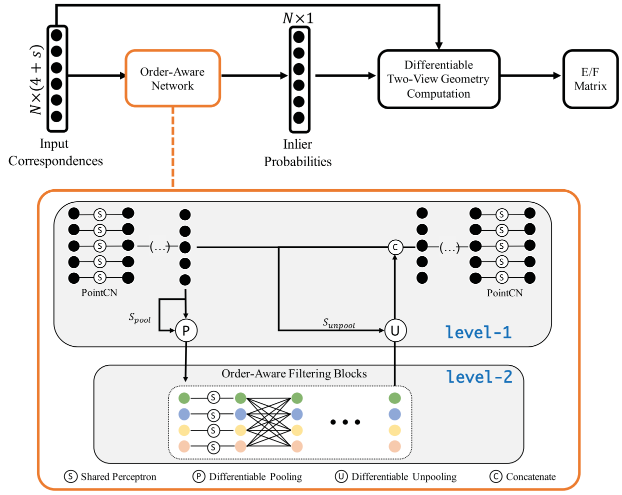


Fig. 1. The Order-Aware Network to learn two-view correspondences and geometry. Inputting correspondences and side information, our network predicts the inlier probabilities and E/F matrix. PointCN blocks are used to process unordered input. Besides, we exploit three operations to exploit the local and global context: the DiffPool and DiffUnpool layer to capture local context and the Order-Aware Filtering block for global context.

(DiffPool) [21] operator, which is permutation-invariant and originally designed for GNN, into a PointNet-like framework to capture the local context. Specifically, as shown in Fig. 1, DiffPool maps input nodes to a set of clusters. “Neighboring” nodes “fall into” the same cluster. This clustering process is learned rather than hand-crafted as in [15], [22]. We find these learned clusters can capture meaningful neighbors. Besides, we will show that the permutation-invariant DiffPool essentially yields canonical order for the resulting clusters. Being in canonical order further enables us to exploit the relations of clusters with more power spatially-correlated operators, i.e., the proposed Order-Aware Filtering block shown in Fig. 1, to capture more complex global context. This canonical order is learned by the network rather than sorted using heuristic methods as [22], [23], [24]. Finally, to build a hierarchical architecture and assign per-correspondence predictions, we develop the Differentiable Unpooling (DiffUnpool) layer to upsample these clusters to the original size, which is a transposed version of DiffPool layer. It is noteworthy that the proposed DiffUnpool operator is specially designed to be order-aware so as to precisely align the upsampled features with the original input correspondences.

Our models can be well generalized to unknown scenes, we conduct experiments on both large scale indoor and outdoor datasets with diverse scenes and improve the mean Average Precision (mAP) for relative pose estimation by a large margin over previous state-of-the-arts.

Our main contributions are threefold:

- We exploit the DiffPool layer and develop DiffUnpool layers to capture the local context of unordered sparse correspondences in a learnable manner.
- By the collaborative use of DiffPool operator, we propose Order-Aware Filtering block which exploits the complex global context of sparse correspondences.
- Our method significantly improves the relative pose estimation accuracy on both outdoor and indoor datasets, and also benefits the SfM and visual localization pipelines.

Compared to our preliminary conference version [25], we extend it in the following aspects: (1) fundamental matrix estimation; (2) more comprehensive evaluations; (3) applications in SfM and visual localization.

2 Related Work

2.1 Learning Based Matching

With the emergence of deep learning, many works attempt to employ learning-based methods to solve geometric matching tasks, including both dense methods [26], [27], [28], [29] and sparse methods [9], [10], [11], [30], [31], [32]. For these sparse methods, most of them focus on interest point extraction and description with Convolutional Neural Network (CNN) to replace handcrafted features [6], [7], [8]. They either replace the descriptors [11], [12], [32], or the detectors [33], [34], or both [9], [10], [30], [31], [35], [36]. Meanwhile, some works [16], [17], [37] also attempt to solve the outlier rejection problem with learning-based methods, which is also the topic of this work. In the dense feature matching side, NC-Net [38] filters the matches in 4D space, but the usage of 4D volume limits input resolution.

2.2 Outlier Rejection

Typically, putative correspondences established by handcrafted or learned features contain many outliers, e.g in the wide baseline case, so that outlier rejection is necessary to improve relative pose estimation accuracy. RANSAC [39] and its variants [40], [41], [42], [43], [44], [45], [46], [47], [48], [49] are the standard and still the most popular outlier rejection methods. Some of them exploit motion consistency to help find correct matches, e.g by sampling potential matches in nearby areas [47] or groups [49], or reject potential outlier by checking spatial consistency [48]. Besides RANSAC variants, many traditional methods also utilize motion smoothness to establish correct matches. BF [13] optimizes piecewise smoothness energy on the bilateral domain to filter outliers. GMS [14] formulates the motion smoothness in a statistical manner by counting matches between regions. RMBP [15] defines a graphical model that describes the spatial organization of matches and then uses belief propagation to reason the correct matches.

Besides, in the context of spatial verification of image retrieval [50], [51], [52], [53], [54], there are also many works use feature matches group information to identify potential outliers. These approaches use matches to vote for all possible weak transformations or motions between images [51]. The voting information can then be used for computing matching strength and rejecting outliers. The grouping of transformation used in these works is similar to the clustering in our work, while the latter is learned by the network rather than predefined.

In the deep learning era, DSAC [37] mimics the behavior of RANSAC and proposes a differentiable counterpart using probabilistic selection. PointCN [16] reformulates the outlier rejection task as an inlier/outlier classification problem and an essential matrix regression problem. DFE [17] also uses PointNet-like architecture and Context Normalization but adopts a different loss function and an iterative network. N^3 Net [55] inserts soft k-nearest neighbors (KNN) layer to augment PointCN. Our work is also built on

PointCN but puts the effort on improving the local and global contexts which are shown very important by traditional methods [13], [14], [15], [56]. Based on the neighbors defined in traditional handcrafted methods, a concurrent work NM-Net [22] captures local information by mining compatibility-specific neighbors. Unlike NM-Net, we let the network itself learn how to define neighbors, which can get better performance. Besides, our method eschews the need for local affine information for detectors, so it can be used together with any off-the-shelf local feature detectors such as SIFT, SuperPoint, *etc*

2.3 Geometric Deep Learning

Geometric Deep Learning deals with data on non-euclidean domains, such as graphs [57], [58], [59], [60] and manifolds [18], [61], [62], [63], [64]. PointNet-like architecture can be regarded as a special case of Graph Neural Network which processes graphs without edges. Different from 3D point clouds, sparse correspondences have no deterministic definition of neighbors , which is also a difficulty faced by many tasks on graphs [21]. Instead of defining heuristic neighbors for correspondences as done in previous works [13], [15], [22], we exploit Differentiable Pooling [21] to cluster nodes in a learnable manner to capture the local context. However, the original DiffPool Network is not applicable in our case because it does not give a full-size prediction. Hence, we propose the DiffUnpool layer to upsample the coarsened feature maps and build a hierarchical architecture. Moreover, we introduce the Order-Aware Filtering block with spatial connections to capture the global context.

3 Order-Aware Network

We will present Order-Aware Network for learning two-view correspondences and geometry, which contains three operations: Differentiable Pooling layer, Order-Aware Differentiable Unpooling layer, and Order-Aware Filtering block. The formulation of our problem is first introduced, and then these submodules successively.

3.1 Problem Formulation

Given image pairs, our goal is to remove outliers from putative correspondences and recover the relative pose. More specifically, after extracting keypoints and their descriptors in each image using handcrafted features [6], [7], [8] or learned features [9], [30], putative correspondences can be established by finding their nearest neighbors in the other image. Then outlier rejection methods can be applied to establish geometrically consistent correspondences. Finally, a fundamental matrix can be recovered from the inlier correspondences by a closed-form solution [16], [65], or an essential matrix when intrinsics are known.

The overview of our workflow is illustrated in Fig. 1. The input to the outlier rejection process is a set of putative correspondences $\mathbf{C} \in \mathcal{R}^{N\times 4}$ and possibly side information $\mathbf{S} \in \mathcal{R}^{N\times s}$, where

$$\mathbf{C} = [c_1; c_2; \ldots; c_N], c_i = (x_1^i, y_1^i, x_2^i, y_2^i), \tag{1}$$

c_i is a correspondence and (x_1^i, y_1^i), (x_2^i, y_2^i) are the coordinates of keypoints in these two images. Side information can be the ratio to the second-best match [6], mutual nearest neighbor check, and results from the previous iteration. Other information such as descriptor can also be used but is not covered in this paper. The coordinates are normalized using camera intrinsics [16] when computing essential matrix, or using Hartley's classic normalization method [66] when computing fundamental matrix.

We formulate the two-view geometry estimation task as an inlier classification problem and an essential or fundamental (E/F) matrix regression problem. A neural network is used to predict the probability of each correspondence to be an inlier. After that, we apply the weighted eight-point algorithm [16] to directly regress the E/F matrix. The overall architecture can be written as

$$\mathbf{z} = f_\phi(\mathbf{C}, \mathbf{S}), \tag{2}$$

$$\mathbf{w} = \tanh(\mathrm{ReLU}(\mathbf{z})), \tag{3}$$

$$\hat{\mathbf{M}} = g(\mathbf{w}, \mathbf{C}), \tag{4}$$

where $\mathbf{z}$ is the logit values for inlier classification. $f_\phi(\cdot)$ is a permutation-equivariant neural network and ϕ denotes the network parameters. $\mathbf{w}$ is the weights of correspondences. For each weight $w_i \in [0, 1)$, $w_i = 0$ means an outlier. tanh and ReLU are applied to easily remove outliers [16]. $g(\cdot,\cdot)$ in Eq. (4) is the weighted eight-point algorithm and $\hat{\mathbf{M}}$ is the regressed E/F matrix. $g(\cdot,\cdot)$ takes more than eight correspondences and their weights, then regresses the E/F matrix by computing the eigenvector associated to the smallest eigenvalue of a self-adjoint matrix. The weighted eight-point algorithm can be more robust than the traditional eight-point algorithm [67] because it has eliminated the influence of outliers. Besides, it is differentiable with respect to $\mathbf{w}$ which makes it possible to regress the E/F matrix in an end-to-end manner.

The optimization objective of this neural network is to minimize a classification loss and an regression matrix loss as follows:

$$loss = l_{cls}(\mathbf{z}, \mathbf{l}) + \alpha l_{reg}(\mathbf{M},\hat{} M), \tag{5}$$

where l_{cls} is a binary cross entropy loss for the classification term. $\mathbf{l}$ denotes weakly supervised labels for correspondences, which are derived using the bellow geometric error in normalized coordinates, and a threshold of 10^{-4} is used to determine correct correspondences. l_{reg} is the regression loss between the predicted E/F matrix $\hat{\mathbf{M}}$ and the ground truth E/F matrix. It can be an $L2$ loss [16]

$$loss_{L2} = \min\{\|\hat{\mathbf{M}} \pm M\|\}, \tag{6}$$

or a geometry loss [17], [67]

$$loss_{geo} = \frac{(p_2^T \hat{\mathbf{M}} p_1)^2}{\|\mathbf{M}\mathbf{p_1}\|_{[1]}^2 + \|\mathbf{M}\mathbf{p_1}\|_{[2]}^2 + \|\mathbf{M^T}\mathbf{p_2}\|_{[1]}^2 + \|\mathbf{M^T}\mathbf{p_2}\|_{[2]}^2}, \tag{7}$$

where $\mathbf{p_1}, \mathbf{p_2}$ are virtual correspondences generated by ground truth E/F matrix and $\mathbf{t}_{[i]}$ denotes the ith element of vector $\mathbf{t}$. α is the weight to balance these two losses.

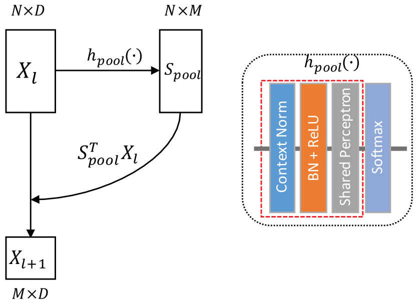
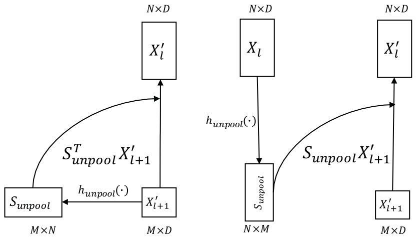


Fig. 2. Differentiable Pooling layer. DiffPool maps nodes $\mathbf{X}_l$ to clusters $\mathbf{X}_{l+1}$ in a soft assignment manner. The soft assignment matrix is learned by $h_{pool}(\cdot)$ which contains one PointCN block (in dashed red box) and one softmax layer.



(a) Plain DiffUnpool

(b) Order-Aware DiffUnpool

Fig. 3. Designs of Differentiable Unpooling layer, which upsamples $\mathbf{X}'_{l+1}$ to $\mathbf{X}'_l$. (a) Plain DiffUnpool layer. It learns a soft assignment matrix using features at level $l+1$. (b) Order-Aware DiffUnpool layer. It learns a soft assignment matrix using features at level l which can encode the order information of nodes at level l.

3.2 Differentiable Pooling Layer

The unordered input correspondences require network $f_\phi(\cdot)$ to be permutation-equivariant. So PointNet-like architecture was used in previous works [16], [17]. Each block in the PointNet-like architecture [16] comprises one Context Normalization layer, one Batch Normalization layer with ReLU, and one shared Perceptron layer. This so-called PointCN block is shown in Figs. 1 and 2. The proposed Context Normalization layer [16] normalizes features of each sample using their global statistics and can largely boost the performance.

However, PointNet-like architecture has the drawback in capturing the local context because there is no direct interaction between points. In order to capture the local context for sparse correspondences, we draw the idea from the DiffPool layer [21] to learn to cluster nodes to a coarser representation, as shown in Fig. 2. The DiffPool layer is analogous to the Pooling layer in CNN which assigns nodes to different clusters. Rather than employing a hard assignment for each node, the DiffPool layer learns a soft assignment matrix. Denoting the assignment matrix as $\mathbf{S}_{pool} \in \mathcal{R}^{N\times M}$, DiffPool layer maps N nodes to M clusters

$$\mathbf{X}_{l+1} = \mathbf{S}^T_{pool}\mathbf{X}_l, \tag{8}$$

where $\mathbf{X}_l \in \mathcal{R}^{N\times D}$ and $\mathbf{X}_{l+1} \in \mathcal{R}^{M\times D}$ are the features at level l and level $l+1$ respectively. D is the dimension of features, and typically $M < N$, e.g $N = 2000, M = 500$.

As we have mentioned before, the assignment matrix is learned rather than pre-defined. More specifically, taking the features at level l, we directly generate the assignment matrix using a permutation-equivariant network as follows:

$$\mathbf{S}_{pool} = \mathrm{softmax}(h_{pool}(\mathbf{X}_l)), \tag{9}$$

where the permutation-equivariant function $h_{pool}(\cdot)$ is one PointCN block here. It maps features from $N \times D$ to $N \times M$. A softmax layer is applied to normalize the assignment matrix along the row dimension. So these clusters can be viewed as weighted average results of nodes in the previous level.

Permutation-Invariance. DiffPool is a permutation-invariant[1] operation [21], which is a desired property in graph neural network. Assuming permuting $\mathbf{X}_l$ with a permutation matrix $\mathbf{P} \in \{0,1\}^{N\times N}$, Eq. (9) becomes

$$\tilde{\mathbf{S}}_{pool} = \mathrm{softmax}(h_{pool}(\mathbf{P}\mathbf{X}_l)) = \mathbf{P}\mathbf{S}_{pool}, \tag{10}$$

because both $h_{pool}(\cdot)$ and softmax are permutation-equivariant functions. So, according to Eq. (8), features at level $l+1$ become

$$\mathbf{X}_{l+1} = \tilde{\mathbf{S}}^T_{pool}\mathbf{P}\mathbf{X}_l = \mathbf{S}^T_{pool}\mathbf{P}^T\mathbf{P}\mathbf{X}_l = \mathbf{S}^T_{pool}\mathbf{X}_l, \tag{11}$$

since $\mathbf{P}^T\mathbf{P} = \mathbf{I}$ holds for every permutation matrix. Eqs. (11) and (8) prove the permutation-invariance property of DiffPool layer.

The permutation-invariance property also means that, once the network is learned, no matter how the input is permuted, they will be mapped into clusters in a particular *learned canonical order* by the DiffPool layer. This canonical order is only determined by the parameters of $h_{pool}(\cdot)$ and has nothing to do with the input. Being in canonical order also means that: (1) the pooled features have lost the order of input nodes. (2) the pooled features are ordered rather than unordered as the previous input nodes. These two things lead to our following designs and we will give more explanations. Since order plays an important role in our designs, we refer our network to Order-Aware Network (OANet).

3.3 Differentiable Unpooling Layer

DiffPool Network was used to predict the label for an entire graph [21]. However, it is not applicable for our sparse matching problem, since we need to give predictions for all correspondences. So, we develop a Differentiable Unpooling layer inspired by the DiffPool layer to upsample the coarse representation and build a hierarchical architecture.

A straightforward way to implement the DiffUnpool layer is reversing the behavior of DiffPool layer, as shown in Fig. 3a. More specifically, similar to Eqs. (8) and (9), an unpooling assignment matrix $\mathbf{S}_{unpool} \in \mathcal{R}^{M\times N}$ is first predicted taking features $\mathbf{X}'_{l+1}$ through

$$\mathbf{S}_{unpool} = \mathrm{softmax}(h_{unpool}(\mathbf{X}'_{l+1})), \tag{12}$$

where $\mathbf{X}'_{l+1} \in \mathcal{R}^{M\times D}$ denotes new features at the same level of $\mathbf{X}_{l+1}$, and it is computed from $\mathbf{X}_{l+1}$ through a permutation-equivariant operation. We can then map features $\mathbf{X}'_{l+1}$ to a new embedding $\mathbf{X}'_l \in \mathcal{R}^{N\times D}$ at level l as follows:

$$\mathbf{X}'_l = \mathbf{S}^T_{unpool}\mathbf{X}'_{l+1}. \tag{13}$$

[1] Equivariance means applying a transformation to the input equals to applying the same transformation to the output, while invariance means applying a transformation to input generates the same output.

However, we find the above implementation is not optimal because it cannot align the unpooled features $\mathbf{X}_l^{'}$ with features $\mathbf{X}_l$ in the previous stage (see experiments in Section 4.4). The point is that DiffPool is a permutation-invariant operation, which means one $\mathbf{X}_{l+1}$ can correspond to various input $\mathbf{X}_l$. In the other words, features $\mathbf{X}_{l+1}$ and $\mathbf{X}_{l+1}^{'}$ at level $l+1$ have lost the spatial order information of features $\mathbf{X}_l$ at level l. We cannot expect the learned assignment matrix as in Eq. (12) can recover the original spatial order of $\mathbf{X}_l$ or generate features which can be precisely aligned with $\mathbf{X}_l$, since $\mathbf{S}_{unpool}$ in Eq. (12) only utilizes information at level $l+1$.

Keeping this in mind, we propose an Order-Aware DiffUnpool layer as shown in Fig. 3b, which can be aware of the particular order (position) of nodes in the previous level. Different from the above implementation, the assignment matrix for unpooling is learned from features at level l which has stored the input order information. The process is as follows:

$$\mathbf{S}_{unpool} = \mathrm{softmax}(h_{unpool}(\mathbf{X}_l)). \tag{14}$$

$h_{unpool}(\cdot)$ is also a PointCN block and it maps features from $N \times D$ to $N \times M$. With this unpooling assignment matrix $\mathbf{S}_{unpool} \in \mathcal{R}^{N \times M}$, we can map features at level $l+1$ to level l by

$$\mathbf{X}_l^{'} = \mathbf{S}_{unpool}\mathbf{X}_{l+1}^{'}. \tag{15}$$

We apply the softmax along the column dimension this time,[2] so the unpooled features can be viewed as weighted average results of different clusters. Since each row in this $\mathbf{S}_{unpool} \in \mathcal{R}^{N \times M}$ corresponds to one node in $\mathbf{X}_l$, it has already encoded the particular order information of $\mathbf{X}_l$ and ensures the unpooled features can well aligned to the previous stage. The mapping in Eq. (15) also requires the learned assignment matrix to be aware of the order of $\mathbf{X}_{l+1}^{'}$. But it is much easier for the network this time since the feature $\mathbf{X}_{l+1}^{'}$ is in canonical order. $\mathbf{X}_l^{'}$ is then concatenated with $\mathbf{X}_l$ to fuse shallow features.

Another advantage of the proposed Order-Aware DiffUnpool layer is that it does not require a fixed-size input. When there are less than or more than 2000 keypoints in images, we can still pool nodes to fixed 500 clusters and then upsample clusters back to the same size. This is useful and necessary in practice since usually more keypoints can bring better results in testing.

3.4 Order-Aware Filtering Block

With the DiffPool and DiffUnpool layers, a multiscale network can be built which is a common practice in CNN. We can apply PointCN blocks repeatedly to process these newly generated clusters. However, as we have discussed above, PointCN may have weaknesses in modeling the complex global context because it ignores the relations between nodes. Here we propose a simple but more effective operation than the PointCN block, which is called the Spatial Correlation layer to explicitly model relations between different nodes and capture the complex global context.

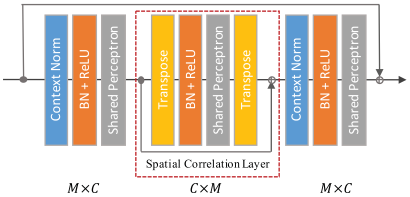


Fig. 4. Order-Aware Filtering block. We insert the Spatial Correlation layer to PointCN ResNet block. This layer is complementary to PointCN and can help capture the global context effectively. Sizes of feature maps are also marked.

As we have shown above, the pooled features for all samples are in a canonical order after the DiffPool layer. This is a useful property but PointNet-like architecture cannot make full use of it. Our Spatial Correlation layer applies weight-sharing perceptrons directly on the spatial dimension to establish connections between nodes, as shown in Fig. 1. Note that before the DiffPool layer, we cannot apply the Spatial Correlation layer on the feature maps because the input data is unordered and there are no stable spatial relations to be captured. This operation is different from the fully connected layer because the weights are shared along the channel dimension, which can help to prevent overfitting. The Spatial Correlation layer is orthogonal to PointCN, one is along the spatial dimension and the other is along the channel dimension. Since these two operations are complementary, we assemble them into one block to better capture the global context as shown in Fig. 4.

Spatial Correlation layer is implemented by transposing the spatial and channel dimensions of features. After a weight-sharing perceptrons layer, we transpose features back. Residual connection and batch normalization with ReLU are also used. We insert the Spatial Correlation layer to the middle of PointCN ResNet block and call this composite module Order-Aware Filtering block which can process data in canonical order. This simple block is only applied at the level after the DiffPool layer and we find it can significantly boost the performance.

4 Experiments

We first conduct experiments about relative pose estimation on outdoor and indoor datasets, more specifically, the YFCC100M [68] dataset and the SUN3D [69] dataset. Then, in order to test the generalization ability of the proposed network and its benefit in more complex tasks, we also conduct experiments on FM-Bench [70], SfM task [71] and visual localization task [4]. Experiment results and network interpretation are as follows.

4.1 Two-View Datasets

Outdoor Scenes. We use the Yahoo's YFCC100M dataset [68], which contains 100 million photos from internet. The authors of [72] later generated 72 3D reconstructions of tourist landmarks from a subset of the collections. We use four sequences [16] as unknown scenes to test generalization ability. For training sequences, different from PointCN [16], we use the remaining 68 sequences for training, while they use only two sequences. Our setting is not prone to overfitting and has better generalization ability on unknown

[2] Actually we find changing the normalization directions in Eqs. (9) and (14) has little influence on results. They do not even need to be orthogonal.

TABLE 1
Performances of Baseline Network [16] on YFCC100M Unknown Sequences

threshold	S	L	mAP5°(%)	mAP10°(%)	mAP20°(%)
0.01	✓		17.5/12.5	27.6/21.2	42.1/34.2
0.001	✓		44.5/12.5	54.5/21.2	65.3/34.2
		✓	48.0/23.6	58.1/36.6	68.7/53.1

Results with/without *RANSAC under error thresholds of* 5°, 10° *and* 20° *are all reported. Changing the inlier threshold in RANSAC and using more data can significantly boost the performance.* ***S****: using only sequences 'Saint Peter's' and 'brown_bm_3_05' as [16].* ***L****: using 68 sequences.*

sequences as shown in Table 1. To have a fair comparison, we re-train all models on the same data.

A minimum visual overlap is required if pairs are selected into the dataset. For outdoor scenes, the visual overlap is the number of sparse 3D points in the reconstructed model which can be both seen by the image pairs. We use the camera poses and sparse models provided by [72] to generate ground-truth.

Indoor Scenes. We use the SUN3D dataset [69] for indoor scenes, which is an RGBD video dataset with camera poses computed by generalized bundle adjustment. Following [26] we split the dataset into 253 scenes for training and 15 as unknown scenes for testing. This splitting can ensure there is no spatial overlap between training and testing datasets. We find some sequences in the training set do not provide camera poses, so we drop these sequences and finally get 239 sequences for training. We subsample videos every 10 frames. The visual overlap for indoor scenes is computed by projecting the depth map to the other image.

Following [16], we test on both known scenes and unknown scenes. The known scenes are the training sequences. We split them into disjoint subsets for training (60 percent), validation (20 percent) and testing (20 percent). The unknown sequences are the test sequences described above.

4.2 Evaluation Metrics

For essential matrix estimation, we use the angular differences between ground truth and predicted vectors for both rotation and translation as the error metric. mAP results with and without RANSAC post-processing are reported. When using RANSAC-variants post-processing, we filter out correspondences whose weights are smaller than 0 according to Eq. (3). We find the inlier threshold of OpenCV function `findEssentialMat()` used in [16] is not optimal. Changing the threshold from 0.01 to 0.001 will largely improve results with RANSAC, as shown in Table 1. We will use mAP under 5° as the default metric since it is more usable in 3D reconstruction context.

4.3 Implementation Details

The Order-Aware Network has 12 PointCN ResNet blocks as [16] in the first level. We add one DiffPool layer and one DiffUnpool layer to construct a hierarchical architecture. Another 6 Order-Aware Filtering blocks are used at the second level as shown in Fig. 1. The channel dimensions are all 128 in these blocks. The inputs to the network are $N \times 4$ putative correspondences established using SIFT feature. $N = 2000$ in our experiments if there is no special declaration. After the DiffPool layer, the number of nodes is reduced to 500 which gives the best performance. Besides, we also use an iterative network as [17] which takes residuals and weights of the previous stage as additional inputs. This can further improve the performance. Our network is implemented with Pytorch [73]. We use Adam solver with a learning rate of 10^{-3} and batch size 32. Weight α is 0 during the first 20k iterations and then 0.1 in the rest 480k iterations as in [16] when using $L2$ loss, and is 0.5 when using geometry loss.

TABLE 2
Ablation Study on YFCC100M

PointCN	UnA	UnB	OF	L3	Geo	Iter	Known	Unknown
✓							34.4/13.9	48.0/23.6
✓	✓						34.4/14.0	47.9/24.1
✓		✓					36.3/17.9	49.7/28.8
✓		✓	✓				40.8/25.9	51.6/32.6
✓		✓	✓	✓			39.7/26.0	50.7/30.5
✓		✓	✓		✓		40.8/28.4	51.1/33.7
✓		✓	✓		✓	✓	**42.5**/33.1	**52.2**/39.3

mAP (%) on both known and unknown scenes are reported with/without *RANSAC post-processing.* UnA*: the plain DiffUnpool layer.* UnB*: the Order-Aware DiffUnpool layer.* OF*: using the Order-Aware Filtering blocks in the second level rather than PointCN blocks.* L3*: a larger model with three levels.* Geo*: using geometry loss rather than L2 loss.* Iter*: using the iterative network.*

4.4 Ablation Studies

In this section, we will give ablation studies about the proposed operations, loss functions and network architecture on the YFCC100M dataset in essential matrix estimation. Besides, we also study the generalization ability regarding the number of input points.

DiffUnpool Layer Design. To demonstrate the efficacy of the DiffUnpool layer, we add DiffPool and DiffUnpool layers to the baseline PointCN model [16]. Both plain DiffUnpool and Order-Aware DiffUnpool described in Section 3.3 are tested. After the DiffPool layer, another six PointCN ResNet blocks are used. Features after the DiffUnpool layer are concatenated to the previous stage. As shown in Table 2, our Order-Aware DiffUnpool *(PointCN + UnB)* achieves an improvement of 5.2 percent over the baseline on unknown scenes when without RANSAC, while the plain DiffPool *(PointCN + UnA)* gives a negligible improvement over the baseline. Besides, we also give a comparison with a naive nonparametric DiffUnpool operation in the Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2020.3048013, which simply transposes the assignment matrix of DiffPool layer rather than learning a new one. The proposed layer achieves superior results over the nonparametric one.

Plain PointCN block versus Order-Aware Filtering block. We replace the PointCN blocks at the second level with Order-Aware Filtering blocks described in Section 3.4, which can better exploit the spatial relationships within the clusters. As shown in Table 2, the proposed block *(PointCN + UnB + OF)* can significantly boost the performance over simple PointCN block *(PointCN + UnB)*, achieving an improvement of 3.8 percent on unknown scenes without RANSAC.

Does a Larger Model Help? We train a larger model which use a U-Net style architecture [74] with three levels. 12 PointCN ResNet blocks are used at the first level, 12 and 6 Order-Aware Filtering blocks are used at the second and third levels. The number of nodes in the second and third levels is pooled to 500 and 125 respectively. However, we find this larger model *(PointCN+UnB+OF+L3)* even has an accuracy drop on unknown scenes, as shown in Table 2. This might show that the representational ability of Order-Aware Filtering block suffices to capture the global context. So we use the two-level network in our rest experiments.

Essential Matrix Loss. $L2$ loss is used as the essential matrix loss in previous experiments. However, the $L2$ loss is not geometric meaningful. So we replace the $L2$ loss with the Gold Standard geometry loss [17], [67]. We clamp the geometry loss values to make the loss more robust as [17], where the clamping threshold is set to 0.1 which works best in our case. Using the geometry loss helps a little for both known and unknown scenes as shown in Table 2 when without RANSAC.

Iterative Network. As in [17], we iteratively refine the estimation by passing the estimated weights and geometric residuals to the next stage. This additional information can guide the estimation process. Here we use one initialization network and one refinement network. Each network has 6 PointCN ResNet blocks and 3 Order-Aware Filtering blocks to keep almost the same amount of parameters. We find it is really necessary to detach the gradients from the latter stage to make the training convergence. Table 2 shows that the iterative network can largely improve the mAP from 33.7 to 39.3 percent without RANSAC on unknown scenes.

Input Point Numbers. Since the number of input points in testing can be very different from the number in training, we expect the network to have robustness regarding the input numbers. So we train the network using 1,000/2,000/4,000/8,000 points and test it with different numbers. As shown in Fig. 6, comparing to training with more points, training with only 1,000 points will severely degrade the performance when testing with more points. On the contrary, training with more points gets similar and even better results than training with fewer points, which demonstrates that network training with more points might have better generalization ability. Due to computation resource constraints, models on two-view relative pose estimation experiments are all trained using 2,000 points. Models in the middle scale SfM experiments and visual localization experiments are trained using 8,000 points to get better results.

4.5 Comparison to Other Baselines

We compare our network with other state-of-the-art methods from [16], [17], [18], [22], [55] on both outdoor and indoor datasets. Only results on unknown scenes are reported.

For the traditional methods, we set up a much stronger baseline than [16] by adopting ratio test [6] and mutual nearest neighbor check before RANSAC, which we refer to RANSAC++ in Table 3. Both these two strategies can boost the performance and their combination gives the best results. For these learned methods, all these models are trained under the same settings. For N^3 Net [55], we use the official implementation. We find N^3 Net is unstable during training, so we run it three times and give the best results here. PointNet++ [18] is an extension of PointNet which also aims to improve the capability in capturing the local context of point sets. As we have discussed before, it may not be optimal for our sparse matching problem because correspondences have no deterministic definition of neighbor . Here we implement a 4D-version PointNet++ which exploits the 4D euclidean space as the underlying metric space. DFE [17] is a concurrent work with [16] and has similar core designs. We implement [17] based on [16] by adopting their loss formulation and iterative network with the authors' help. For our method, we present three models trained with different settings: *Our* is trained with $L2$ loss as PointCN [16]. *Our++* is trained with geometry loss and iterative network as DFE [17]. *Our+++* is trained with additional two-dimension side information, including the feature distance ratio to the second-best match and a binary value indicating whether they are mutual nearest neighbors. For NM-Net [22], the comparison is conducted on the COLMAP dataset used in their paper [2], [22], which is much smaller. For a fair comparison, we use the same loss as they and do not use the iterative network. The comparison results are reported using their metric.

TABLE 3
Comparison With Other Baselines on Unknown Scenes of YFCC100M and SUN3D

		YFCC100M	SUN3D
		mAP5°(%)	mAP5°(%)
SIFT	RANSAC	9.1/-	2.9/-
	RANSAC++	45.9/-	15.9/-
	PointCN [16]	48.0/23.6	16.0/9.4
	PointNet++ [18]	46.2/14.0	15.6/5.6
	N^{3}Net [54]	49.1/23.2	15.4/7.1
	DFE [17]	49.5/29.7	16.5/12.5
	Our	51.6/32.6	16.5/12.5
	Our++	52.2/39.3	17.5/16.4
	Our+++	**55.5**/43.7	17.9/**17.9**
SuperPoint	RANSAC	22.0/-	14.6/-
	PointCN [16]	43.2/24.8	18.5/10.2
	Our	**46.3**/32.2	**19.0**/12.1

Hessian -Affine		Precision(%)	Recall(%)	F-score(%)
	NM-Net [22]	31.9	**56.0**	39.1
	Our	**39.5**	53.9	**44.4**

mAPs (%) (with/without *RANSAC post-processing) are reported.* Ours *use the L2 loss.* Ours++ *uses the geometry loss and iterative network while* Ours *not uses.* Ours+++ *uses additional side information based on* Ours++. *Comparison with NM-Net is conducted on COLMAP dataset [2], [22].*

As shown in Table 3, the gap between the traditional method and learned methods still exists but is much smaller when compared with the stronger traditional baseline. Comparing to other methods, our method achieves the best results in almost all settings. Comparing to PointCN [16], the model *Our* shows an improvement 9.0 and 3.1 percent on both outdoor and indoor unknown scenes without RANSAC and still works well with strong RANSAC post-processing. Comparing to the stronger baseline DFE [17] which also uses geometry loss and iterative network, *Our++* is 9.6 and 3.9 percent higher on these two datasets. Using side information can further largely improve the results, bringing an improvement of 3.3 percent even with RANSAC on outdoor scenes. Comparing to the concurrent NM-Net [22] which also aims to utilize local context information, our method achieves much better results in precision and F-score. Besides, our method can be used with any local feature detector, not just Hessian-Affine detector [8].

We also evaluate learned features such as SuperPoint [9] as shown in Table 3. It is surprising to find SuperPoint gives worse results on the outdoor scenes than SIFT when using learned outlier rejection methods, although it performs much better than SIFT when only using RANSAC. It might

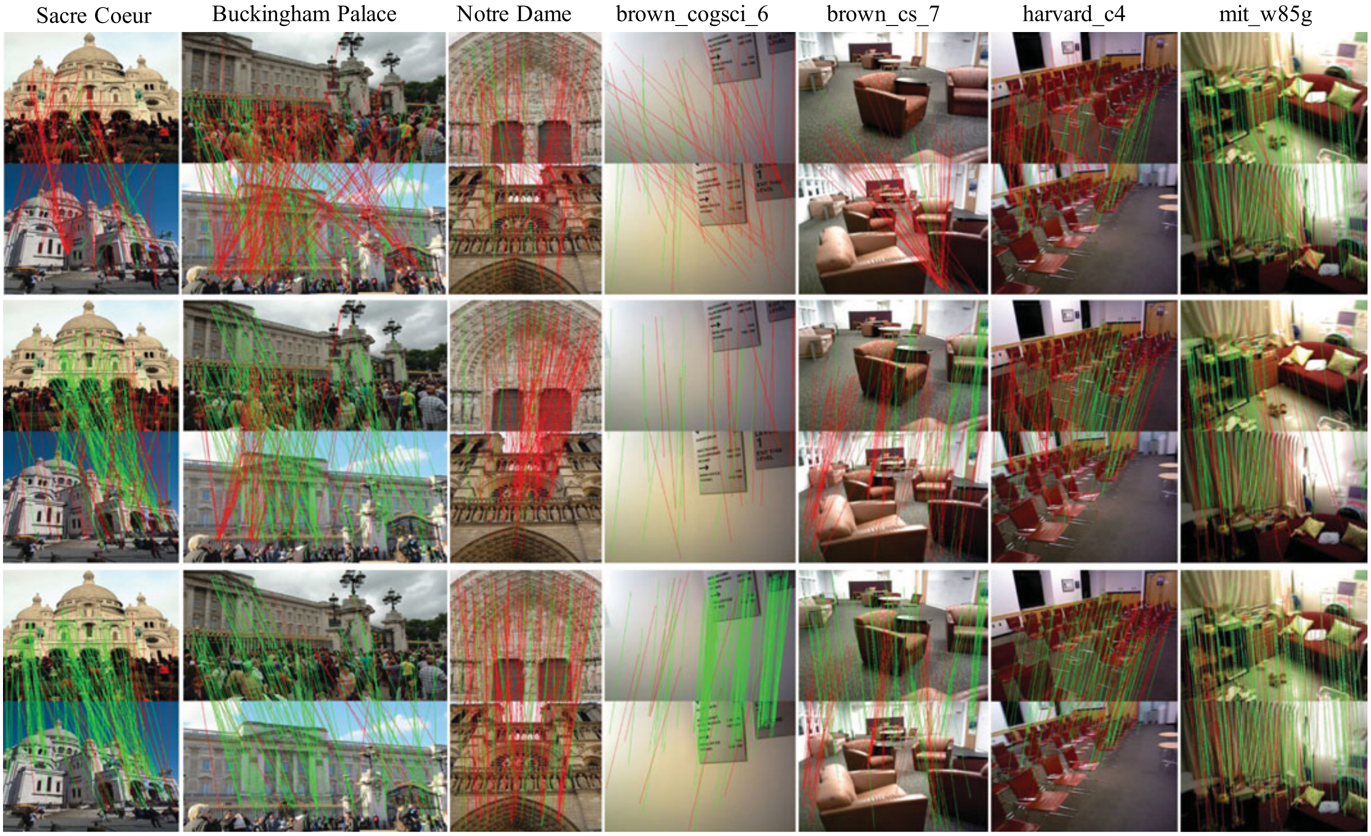


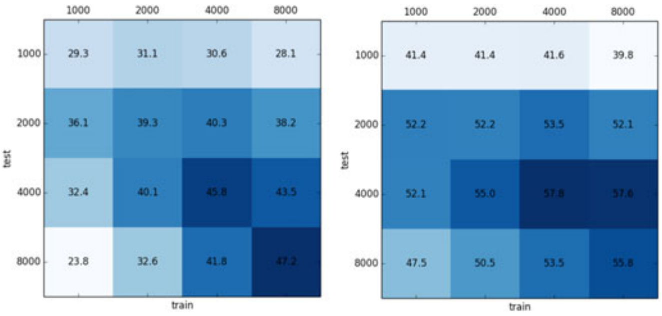
Fig. 5. Matching results using RANSAC (top), PointCN [16] (middle) and our method (bottom). Images are taken from test set of YFCC100M and SUN3D datasets. Correpondences are in green if they conform the ground truth essential matrix (true positives), and in red otherwise (false positives). *Best viewed in color*.

demonstrate that SuperPoint has better descriptors but less accurate keypoints. It can give putative correspondences with a higher initial inlier ratio thus has better performance when only using RANSAC. But the bottleneck may become keypoint accuracy when the inlier ratio is largely improved, in which situation, SuperPoint performs worse.

Fig. 5 shows the visualization results of our method and other baselines. It can be found that our method can give better results on several difficult scenes such as wide baselines, textureless objects, repetitive structures, and large illumination changes.

4.6 Fundamental Matrix Estimation

Fundamental matrix estimation is a more general problem which does not assume camera intrinsics are known, and it is widely used in current SfM pipelines [2]. Estimating the fundamental matrix is very similar to estimate the essential matrix except that the inputs are raw image coordinates rather than intrinsic normalized coordinates. So we utilize the normalized eight-point algorithm [66] to eliminate the influence of image resolution and make the optimization better behaved. After normalization, the origin is at the centroid of input points and the average distance to the origin is equal to $\sqrt{2}$. We denormalize the estimated fundamental matrix after the weighted eight-point method. We use the ratio test as side information and set the clamping threshold in geometry loss to 0.03.



(a) mAP5° results without RANSAC

(b) mAP5° results with RANSAC

Fig. 6. Training and testing in different numbers of input points. Model trained with more points has better generalization ability.

Here we also provide comparisons with RANSAC-variants, including the OpenCV plain RANSAC, the MAGSAC [46] and the USAC [44]. For RANSAC variants, we use ratio test with a threshold of 0.8 to prune outliers. For MAGSAC and USAC, we set the maximum iteration to 10000. The inlier thresholds for RANSAC/MAGSAC/USAC are set to 0.80px/0.70px/0.25px and 1.3px/1.5px/0.5px for YFCC100M and SUN3D respectively. As shown in Table 4, on the outdoor

TABLE 4
Fundamental Matrix Estimation Results of Different Methods

	Outdoor			Indoor		
	mAP5°	mAP10°	mAP20°	mAP5°	mAP10°	mAP20°
Ratio Test + RANSAC	21.3	27.0	35.5	7.6	13.1	21.9
Ratio Test + USAC [43]	23.8	29.8	38.4	8.7	14.2	23.5
Ratio Test + MAGSAC [45]	27.9	32.8	40.1	11.0	17.1	26.0
Our + RANSAC	33.4	41.7	52.5	11.1	18.9	30.6
Our + MAGSAC [45]	**38.3**	**45.7**	55.2	12.9	20.7	32.0
Our	31.3	44.0	**58.4**	**15.0**	**24.9**	**38.6**

TABLE 5
Results on FM-Bench [70]

	TUM [74] (indoor SLAM settings)			KITTI [75] (driving settings)		
Methods	%Recall	%Inliers	%Inlier-m	%Recall	%Inliers	#Inlier-m
GMS [14]	59.2	76.2	70.0	91.7	98.6	95.6
LPM [80]	58.9	75.8	64.4	91.5	98.3	92.5
LC [16]	54.1	76.0	71.3	89.7	**99.4**	**97.5**
Ours	66.1	76.2	73.6	**92.3**	98.3	95.4
GC-RANSAC [44]	70.6	75.5	59.2	90.6	98.0	87.4
GC-RANSAC [44] + Ours	**74.0**	**76.5**	**75.9**	88.6	98.5	97.3
	T&T [76] (wide baseline reconstruction)			CPC [77] (web images)		
GMS [14]	80.9	**84.4**	**77.7**	43.0	**85.9**	**82.4**
LPM [80]	80.7	81.6	67.0	39.40	78.2	66.0
LC [16]	76.6	84.0	72.3	39.4	84.0	72.2
Ours	84.0	80.6	66.8	47.0	81.1	67.2
GC-RANSAC [44]	89.1	83.0	53.2	61.1	83.8	48.1
GC-RANSAC [44] + Ours	**90.3**	83.1	74.4	**68.9**	85.6	74.8

%Recall reflects the overall performance. %Inliers and %Inlier-m are inlier ratio after and before RANSAC.

dataset, the combination of our method and MAGSAC post-processing can outperform RANSAC-variants by a large margin. Moreover, we find our method without RANSAC-variants post-processing can even achieve better results than using post-processing in a mild threshold of mAP20°. This might because that weighted eight-point method can make full use of all inliers and their confidence, while RANSAC-variants drop the confidence and only select subset of inliers, which might be not optimal. On the SUN3D dataset, our method without post-processing is consistently better than using RANSAC post processing.

For run-time comparison. Our network takes 0.02s on a GTX 1080 GPU, or 0.10s if on a CPU, while MAGSAC needs 0.09s on a CPU.

4.7 Generalization Ability

In order to resolve this concern about generalization ability of learning-based methods, here we provide a comparison on the FM-Bench [70] for fundamental matrix estimation, which contains four datasets, including TUM [75], KITTI [76], Tank and Temple [77], and Community Photo Collection [78]. We train one model on the GL3D dataset [79] and test on these four datasets without tuning any dataset-specific parameter. We compare our method with other outlier pruning methods and get the best results, as shown in Table 5. Note we exclude the results of CODE [80] in FM-Bench because it [80] uses different settings: ASIFT *vs* SIFT and more points.

4.8 Network Visualization

In order to understand the mechanism of the proposed Order-Aware Network, we visualize the assignment matrix $S_{unpool} \in \mathcal{R}^{N \times M}$ of the DiffUnpool layer which reflects the spatial relationships between different nodes in the first level. More specifically, we visualize the top k responses in each column of S_{unpool}. Each column in S_{unpool} represents one cluster and each row corresponds to one putative correspondence. These top k correspondences are "clustered" together because they all have a strong response to the same cluster. We find DiffUnpool can capture meaningful context for sparse matching. Fig. 7 shows that different clusters might correspond to different local motions. More surprisingly, we find the corresponding motions of a particular cluster are roughly consistent in different pairs and even in different scenes as shown in Fig. 8, which supports our view that the pooled features are in canonical order.

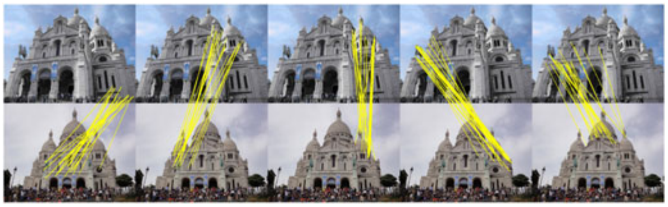


Fig. 7. DiffUnpool layer visualization. Top 15 responses in different clusters are visualized in the same image pair. Different clusters might correspond to different motions in different areas. *Best viewed in color with 200 percent zoom in.*

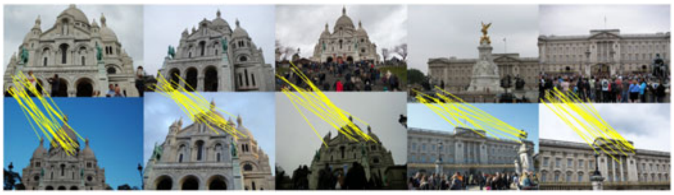


Fig. 8. DiffUnpool layer visualization. Top 20 responses of one particular are visualized in different image pairs. Motions in different pairs are roughly consistent. *Best viewed in color with 200 percent zoom in.*

TABLE 6
Part of the Results of the CVPR 2019
Image Matching Challenge

	Ims(%)	#Pts	SR	TL	mAP5°	mAP15°
ContextDesc [11]+Our	**98.6**	6126.0	97.5	3.44	**0.5755**	**0.7389**
ContextDesc [11]+PointCN [16]	98.4	6045.8	97.8	3.43	0.5553	0.7169
ContextDesc [11]+MutualNN	98.1	6472.1	**98.0**	3.34	0.4287	0.6017
SuperPoint [9]+PointCN [16]	96.5	1349.6	92.1	**3.87**	0.4826	0.6458
HardNet [12]	97.9	6552.9	97.2	3.36	0.3894	0.5481
LogPol [81]	97.6	**6831.3**	96.9	3.29	0.3827	0.5427
HesAffNet [34]+HardNet [12]	96.8	5418.9	95.7	3.43	0.3716	0.5284
L2Net [82]	97.3	6082.5	96.3	3.27	0.3513	0.5087
HarrisZ+RsGLOH2 [83]	96.6	1727.7	94.0	3.31	0.3388	0.5040

MutualNN: *Mutual nearest neighbor matcher.* Ims: *Ratios of registered images.* #Pts: *Dense points.* SR: *Success ratio in the 3D reconstruction.* TL: *Track length. Ranking using mAP*15°*.*

The mechanism of our network possibly is similar to traditional methods such as GMS [14] or Hough pyramid matching [51]. The network learns to enumerate possible motions between pairs and each cluster represents one motion. Then the matches in each cluster can vote for these clusters. The motion which has more supported matches is more likely to be correct than those having fewer supported matches. Then matches following the correct motions are more likely to be inliers.

4.9 SfM Experiments

We first present small scale SfM results in the multi-view track of CVPR 2019 Image Matching Challenge.[3] The datasets used in this challenge is also from YFCC100M. In order to evaluate the performance, the organizers build SfM reconstructions [2] from subsets (≤ 25 images) of test sequences, and compare the obtained pose with the ground truth reconstructed from a much larger set. We train a fundamental matrix estimation model using the provided training sequences with the advanced local feature ContextDesc [11]. 8,000 keypoints are extracted for each image when testing in this challenge. As shown in Table 6, comparing with PointCN [16] and mutual

[3] https://image-matching-workshop.github.io/leaderboard/

TABLE 7
3D Reconstruction Results on Middle Scale Datasets [71]

		#Images	#Reg.	#Sparse	#Dense	#Observations	TL	Reproj.
Almao*	R+M	2915	**963**	198433	**3737516**	2437084	**12.28**	**0.65px**
	Our	2915	897	**294612**	3023898	**2908379**	9.87	0.77px
Alamo	R+M	2915	774	127965	2878738	1263904	**9.88**	**0.52px**
	Our	2915	**823**	**226022**	**3073049**	**1767174**	7.82	0.70px
Gendarmenmarkt	R+M	1463	987	175817	4217246	1002271	**5.70**	**0.63px**
	Our	1463	**1039**	**380840**	**4478160**	**1772267**	4.65	0.78px
Madrid-Metropolis	R+M	1344	414	61291	**1534875**	416715	**6.80**	**0.54px**
	Our	1344	**480**	**124452**	1443902	**690655**	5.55	0.69px
Roman Forum	R+M	2364	**1497**	259578	8758676	1842726	**7.10**	**0.61px**
	Our	2364	1478	**507703**	**9071424**	**2958493**	5.83	0.72px
Tower of London	R+M	1576	634	130338	2817384	908289	**6.97**	**0.52px**
	Our	1576	**648**	**246003**	**2840489**	**1359929**	5.53	0.65px

Alamo* *is result in Alamo dataset using standard setup.* R+M*: ratio test + mutual check.* #Reg.*: registered images number.* #Sparse*: sparse points number.* #Dense*: dense points number.* TL*: mean track length.* Reproj.*: reprojection error.*

nearest neighbor check which also use ContextDesc, our matcher obtains the best results in almost all metrics. Our solution outperformed several recent works and won the first place.

Besides, we also evaluate our method on some middle scale (1,000∼3,000 images) datasets [71] whose images are collected from the internet and contain many distractor images. Evaluations are conducted follow [71] and we report the number of registered images, sparse points of SfM and dense points of Multi-View Stereo (MVS). We use OpenCV SIFT local feature as a baseline and 8,000 keypoints are extracted for each image. Our fundamental matrix estimation model is trained with 8,000 points and the *Our+++* network described in Section 4.5 is used. We use the Bag of Word (BoW) model to retrieves the top 20 similar images as done in [85] rather than top 100 images [71] as matching candidates.

We find in the top-100 setting, the added distractor images result in more noise and incomplete reconstruction and make it harder to give a comparison, as shown Table 7 and Fig. 9. In the top-20 setting, our method consistently gets more sparse points compared to the traditional method which utilizes ratio test and mutual nearest neighbor check. But it also brings shorter mean track length and larger reprojection error. Moreover, as visualized in Fig. 9, we can even successfully reconstruct the challenging scene where the traditional method fails due to false matching.

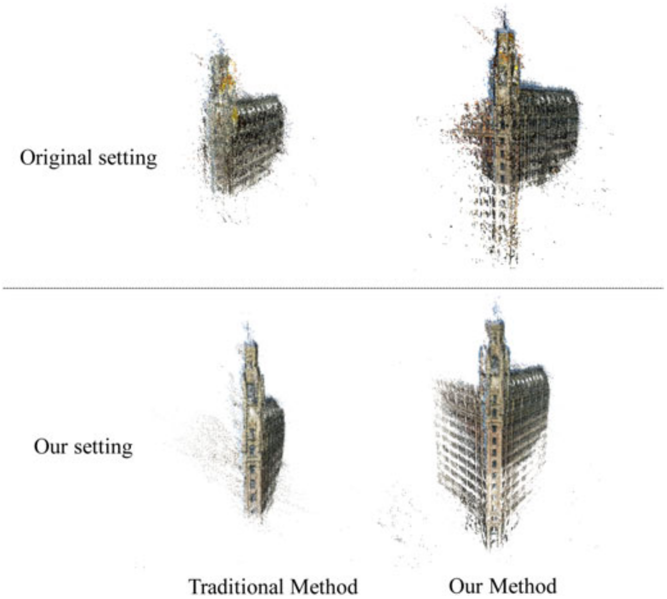


Fig. 9. Reconstructions from the *Alamo* dataset. Traditional method fails but our method successes in this challenging scene.

TABLE 8
Results of Visual Localization Experiments on the Aachen Day-Night Dataset [4], [86]

	0.5m, 0.2°(%)	1m, 5°(%)	5m, 10°(%)
ContextDesc [11]+MutalNN+Our	**48.0**	63.3	**88.8**
ContextDesc [11]+MutualNN	41.8	57.1	79.6
D2-Net [35]	45.9	**68.4**	**88.8**
R2D2 [86]	45.9	66.3	**88.8**
DELF [87]	39.8	61.2	85.7
SaliencyRankingNet	44.9	59.2	77.6
SuperPoint [9]	42.9	57.1	77.6
HesAffNet [34]+HardNet [12]	37.8	54.1	75.5
RootSIFT	33.7	52	65.3

4.10 Visual Localization Experiments

Finally, we also test our method on the visual localization task. Experiments are conducted on the Aachen Day-Night dataset [4], [86], which has severe illumination changes. Each nighttime query image in the dataset is given several relevant daytime images with known poses and 3D model. We use the ContextDesc [11] and our matcher again to set up reliable sparse correspondences between the query and reference images. The model is trained without side information on YFCC100M datasets. 10,000 keypoints are extracted for each image when testing. We find the mixed matcher works best which first applies mutual nearest neighbor check and then our learned matcher. Following [35], we use COLMAP [2] to register the query images and recover their poses. The percentages of correctly localized images under certain thresholds are reported as shown in Table 8.[4] Our method can give an improvement of 6.2/6.2/9.2 percent upon mutual nearest neighbor check. We achieve almost equal results with state-of-the-art method [35], [87] and get the best results in the most strict metric.

5 Conclusion

In this work, we proposed the Order-Aware Network for learning two-view correspondences and geometry. The introduced DiffPool layer and Order-Aware DiffUnpool layer can learn to cluster meaningful nodes to capture local context. Besides, we develop Order-Aware Filtering blocks to capture the global context. These operations can significantly improve relative pose estimation accuracy across different datasets and tasks.

Acknowledgments

This work was supported by the Natural Science Foundation of China (82027807, 81771940, and 81427803), National Key Research and Development Program of China (2017YFC0108000), National Beijing National Science Foundation (L172003), and Soochow-Tsinghua Innovation Project (2016SZ0206). This work was also supported by Hong Kong RGC GRF 16206819 and 16203518. Part of the work was done when Jiahui Zhang was an intern at ILC supervised by Anbang Yao and visiting HKUST. The authors would also like to thank Vladlen Koltun, René Ranftl and David Hafner for helping us reimplement their work. They also thank Dániel Baráth for the

[4] Results are taken from the Local feature track of Visual Localization Benchmark. https://www.visuallocalization.net/workshop/cvpr/2019/

help in GC-RANSAC and MAGSAC experiments. Jiahui Zhang and Dawei Sun contributed equally to this work.

References

[1] C. Wu, "VisualSFM: A visual structure from motion system," 2011. [Online]. Available: http://ccwu.me/vsfm/

[2] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4104–4113.

[3] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: A versatile and accurate monocular SLAM system," *IEEE Trans. Robot.*, vol. 31, no. 5, pp. 1147–1163, Oct. 2015.

[4] T. Sattler *et al.*, "Benchmarking 6DOF outdoor visual localization in changing conditions," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8601–8610.

[5] M. Brown and D. G. Lowe, "Automatic panoramic image stitching using invariant features," *Int. J. Comput. Vis.*, vol. 74, pp. 59–73, 2007.

[6] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, pp. 91–110, 2004.

[7] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. Int. Conf. Comput. Vis.*, 2011, pp. 2564–2571.

[8] K. Mikolajczyk and C. Schmid, "Scale & affine invariant interest point detectors," *Int. J. Comput. Vis.*, vol. 60, pp. 63–86, 2004.

[9] D. DeTone, T. Malisiewicz, and A. Rabinovich, "SuperPoint: Self-supervised interest point detection and description," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops*, 2018, pp. 337–33712.

[10] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "LF-Net: Learning local features from images," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 6237–6247.

[11] Z. Luo *et al.*, "ContextDesc: Local descriptor augmentation with cross-modality context," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 2522–2531.

[12] A. Mishchuk, D. Mishkin, F. Radenovic, and J. Matas, "Working hard to know your neighbor's margins: Local descriptor learning loss," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 4829–4840.

[13] W.-Y. D. Lin, M.-M. Cheng, J. Lu, H. Yang, M. N. Do, and P. Torr, "Bilateral functions for global motion modeling," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 341–356.

[14] J. Bian, W.-Y. Lin, Y. Matsushita, S.-K. Yeung, T.-D. Nguyen, and M.-M. Cheng, "GMS: Grid-based motion statistics for fast, ultra-robust feature correspondence," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2828–2837.

[15] L. Zhou *et al.*, "Learning and matching multi-view descriptors for registration of point clouds," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 527–544.

[16] K. Moo Yi, E. Trulls, Y. Ono, V. Lepetit, M. Salzmann, and P. Fua, "Learning to find good correspondences," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 2666–2674.

[17] R. Ranftl and V. Koltun, "Deep fundamental matrix estimation," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 292–309.

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 77–85.

[19] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, 2017, pp. 4105–4113.

[20] 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. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 5105–5114.

[21] Z. Ying, J. You, C. Morris, X. Ren, W. Hamilton, and J. Leskovec, "Hierarchical graph representation learning with differentiable pooling," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 4805–4815.

[22] C. Zhao, Z. Cao, C. Li, X. Li, and J. Yang, "NM-Net: Mining reliable neighbors for robust feature correspondences," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 215–224.

[23] M. Niepert, M. Ahmed, and K. Kutzkov, "Learning convolutional neural networks for graphs," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 2014–2023.

[24] M. Zhang, Z. Cui, M. Neumann, and Y. Chen, "An end-to-end deep learning architecture for graph classification," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 4438–4445.

[25] J. Zhang *et al.*, "Learning two-view correspondences and geometry using order-aware network," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 5844–5853.

[26] B. Ummenhofer *et al.*, "DeMoN: Depth and motion network for learning monocular stereo," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5622–5631.

[27] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6612–6619.

[28] R. Li, S. Wang, Z. Long, and D. Gu, "UnDeepVO: Monocular visual odometry through unsupervised deep learning," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 7286–7291.

[29] I. Rocco, R. Arandjelovic, and J. Sivic, "Convolutional neural network architecture for geometric matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 39–48.

[30] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua, "LIFT: Learned invariant feature transform," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 467–483.

[31] D. DeTone, T. Malisiewicz, and A. Rabinovich, "Self-improving visual odometry," 2018, *arXiv: 1812.03245*.

[32] Z. Luo *et al.*, "GeoDesc: Learning local descriptors by integrating geometry constraints," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 170–185.

[33] N. Savinov, A. Seki, L. Ladicky, T. Sattler, and M. Pollefeys, "Quad-networks: Unsupervised learning to rank for interest point detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 3929–3937.

[34] J. Matas, D. Mishkin, and F. Radenovic, "Repeatability is not enough: Learning discriminative affine regions via discriminability," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 287–304.

[35] M. Dusmanu *et al.*, "D2-Net: A trainable CNN for joint detection and description of local features," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8084–8093.

[36] Z. Luo *et al.*, "ASLFeat: Learning local features of accurate shape and localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 6589–6598.

[37] E. Brachmann *et al.*, "DSAC – Differentiable RANSAC for camera localization," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2492–2500.

[38] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic, "Neighbourhood consensus networks," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1651–1662.

[39] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, pp. 381–395, 1981.

[40] O. Chum and J. Matas, "Matching with PROSAC-progressive sample consensus," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2005, pp. 220–226.

[41] K. Lebeda, J. Matas, and O. Chum, "Fixing the locally optimized RANSAC–full experimental evaluation," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–11.

[42] O. Chum, J. Matas, and J. Kittler, "Locally optimized RANSAC," in *Proc. Joint Pattern Recognit. Symp.*, 2003, pp. 236–243.

[43] O. Chum and J. Matas, "Optimal randomized RANSAC," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 8, pp. 1472–1482, Aug. 2008.

[44] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J.-M. Frahm, "USAC: A universal framework for random sample consensus," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 2022–2038, Aug. 2013.

[45] D. Barath and J. Matas, "Graph-cut RANSAC," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 6733–6741.

[46] D. Barath, J. Matas, and J. Noskova, "MAGSAC: Marginalizing sample consensus," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 10 197–10 205.

[47] D. Myatt, P. Torr, J. Bishop, R. Craddock, and S. Nasuto, "NAPSAC: High noise, high dimensional robust estimation - it's in the bag," in *Proc. Brit. Mach. Vis. Conf.*, 2002, pp. 44.1–44.10, doi: 10.5244/C.16.44.

[48] T. Sattler, B. Leibe, and L. Kobbelt, "SCRAMSAC: Improving RANSAC's efficiency with a spatial consistency filter," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 2090–2097.

[49] K. Ni, H. Jin, and F. Dellaert, "GroupSAC: Efficient consensus in the presence of groupings," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 2193–2200.

[50] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman, "Object retrieval with large vocabularies and fast spatial matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[51] Y. Avrithis and G. Tolias, "Hough pyramid matching: Speeded-up geometry re-ranking for large scale image retrieval," *Int. J. Comput. Vis.*, vol. 107, no. 1, pp. 1–19, 2014.

[52] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 304–317.

[53] X. Li, M. Larson, and A. Hanjalic, "Pairwise geometric matching for large-scale object retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 5153–5161.

[54] X. Wu and K. Kashino, "Robust spatial matching as ensemble of weak geometric relations," in *Proc. Brit. Mach. Vis. Conf.*, 2015, pp. 25.1–25.12.

[55] T. Plötz and S. Roth, "Neural nearest neighbors networks," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 1095–1106.

[56] A. Albarelli, S. R. Bulo, A. Torsello, and M. Pelillo, "Matching as a non-cooperative game," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, 2009, pp. 1319–1326.

[57] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations*, 2017.

[58] F. Monti, D. Boscaini, J. Masci, E. Rodola, J. Svoboda, and M. M. Bronstein, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5425–5434.

[59] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[60] M. Fey, J. Eric Lenssen, F. Weichert, and H. Müller, "SplineCNN: Fast geometric deep learning with continuous B-spline kernels," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 869–877.

[61] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "SpiderCNN: Deep learning on point sets with parameterized convolutional filters," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 90–105.

[62] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3D point cloud models," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 863–872.

[63] B. Graham, M. Engelcke, and L. van der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9224–9232.

[64] J. Zhang, H. Zhao, A. Yao, Y. Chen, L. Zhang, and H. Liao, "Efficient semantic scene completion network with spatial group convolution," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 749–765.

[65] H. C. Longuet-Higgins, "A computer algorithm for reconstructing a scene from two projections," *Nature*, vol. 293, pp. 133–135, 1981.

[66] R. I. Hartley, "In defence of the 8-point algorithm," in *Proc. IEEE Int. Conf. Comput. Vis.*, 1995, pp. 1064–1070.

[67] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[68] B. Thomee *et al.*, "YFCC100M: The new data in multimedia research," *Commun. ACM*, vol. 59, pp. 64–73, 2016.

[69] J. Xiao, A. Owens, and A. Torralba, "SUN3D: A database of big spaces reconstructed using SfM and object labels," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2013, pp. 1625–1632.

[70] J.-W. Bian *et al.*, "An evaluation of feature matchers for fundamental matrix estimation," in *Proc. Brit. Mach. Vis. Conf.*, vol. 2, 2019, pp. 25, [Online]. Available: https://bmvc2019.org/wp-content/uploads/papers/0450-paper.pdf

[71] J. L. Schonberger, H. Hardmeier, T. Sattler, and M. Pollefeys, "Comparative evaluation of hand-crafted and learned local features," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6959–6968.

[72] J. Heinly, J. L. Schonberger, E. Dunn, and J.-M. Frahm, "Reconstructing the world* in six days*(as captured by the Yahoo 100 million image dataset)," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3287–3295.

[73] A. Paszke **et al.**, "Automatic differentiation in pytorch," in *Proc. Int. Conf. Neural Inf. Process. Syst. Workshops*, 2017.

[74] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Med. Image Comput. Comput.-Assisted Intervention*, 2015, pp. 234–241.

[75] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 573–580.

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

[77] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–13, 2017.

[78] K. Wilson and N. Snavely, "Robust global translations with 1DSfM," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 61–75.

[79] T. Shen *et al.*, "Matchable image retrieval by learning from surface reconstruction," in *Proc. Asian Conf. Comput. Vis.*, 2018, pp. 415–431.

[80] W.-Y. Lin *et al.*, "CODE: Coherence based decision boundaries for feature correspondence," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 1, pp. 34–47, Jan. 2018.

[81] J. Ma, J. Zhao, J. Jiang, H. Zhou, and X. Guo, "Locality preserving matching," *Int. J. Comput. Vis.*, vol. 127, pp. 512–531, 2019.

[82] P. Ebel, A. Mishchuk, K. M. Yi, P. Fua, and E. Trulls, "Beyond cartesian representations for local descriptors," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019, pp. 253–262.

[83] Y. Tian, B. Fan, and F. Wu, "L2-net: Deep learning of discriminative patch descriptor in euclidean space," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 6128–6136.

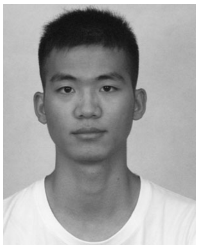
[84] F. Bellavia and C. Colombo, "Rethinking the sGLOH descriptor," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 4, pp. 931–944, Apr. 2018.

[85] M. Dusmanu, J. L. Schönberger, and M. Pollefeys, "Multi-view optimization of local feature geometry," in *Proc. Eur. Conf. Comput. Vis.*, A. Vedaldi, H. Bischof, T. Brox, J. M. Frahm, Eds., Lecture Notes Comput. Sci., Springer, Cham, vol. 12346, 2020. [Online]. Available: https://doi.org/10.1007/978-3-030-58452-8_39

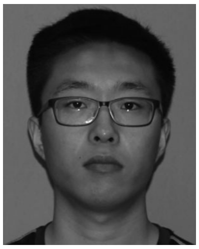
[86] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt, "Image retrieval for image-based localization revisited," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 76.1–76.12.

[87] J. Revaud, P. Weinzaepfel, C. R. de Souza, and M. Humenberger, "R2D2: Repeatable and reliable detector and descriptor," in *Advances Neural Inf. Process. Syst. NeurIPS*, vol. 32, 2019, pp. 12405−12415.

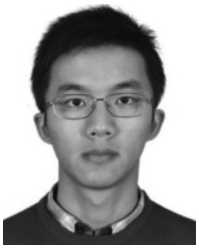
[88] H. Noh, A. Araujo, J. Sim, T. Weyand, and B. Han, "Large-scale image retrieval with attentive deep local features," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 3456–3465.



Jiahui Zhang received the BS degree in mechanical engineering from Tsinghua University, China, in 2015. He is currently working toward the PhD degree from the Department of Biomedical Engineering, Tsinghua University, Beijing, China. His research interests include 3D computer vision and light field display.



Dawei Sun receieved the BE degree from the Department of Automation, Tsinghua University, China. He is currently working toward the PhD degree from the ECE Department, University of Illinois at Urbana-Champaign, Champaign, Illinois. His research interests include machine learning, computer vision, and formal methods.



Zixin Luo received the bachelor's degree from the Department of Automation, Tsinghua University, Beijing, China. He is currently working toward the PhD degree at the Hong Kong University of Science and Technology, Hong Kong under the supervision of Prof. Long Quan. His research interests include matching tasks in 3D computer vision.

Anbang Yao (Member, IEEE) received the PhD degree from Tsinghua University, China, in 2010. He is currently a senior staff research scientist at Intel Labs China, where he leads the research efforts on developing omni-scale high-performance intelligent vision systems. His research interests include deep neural network compression, efficient CNN architecture engineering, 2D/3D scene understanding and multimodal emotion analysis. He has more than 80 PCT/US/EP patent applications got granted/filed, which are widely used in Intel AI related HW accelerators and SW toolkits. As the first/corresponding author, he has published more than 30 top-tier research papers in NeurIPS, ICLR, CVPR, ICCV, ECCV, etc. He was recognized with numerous Awards at Intel, including Intel Global Innovator of 2018, 3 times of annual Intel Labs Gordy Awards and 2 times of annual Intel China Awards. He led the team and won the Winner of the prestigious ACM ICMI EmotiW challenges in 2015 and 2017. He is also a member of ACM.

Hongkai Chen rececived the bachelor's degree from the School of Information Science and Technology, University of Science and Technology of China, China. He is currently working toward the PhD degree from the Hong Kong University of Science and Technology, Hong Kong under the supervision of Prof. Long Quan. His research interests include matching tasks related to SfM in computer vision.

Lei Zhou received the bachelor's degree in information science and electronic engineering from Zhejiang University, China, in 2015, and the PhD degree in computer science and engineering from the Hong Kong University of Science and Engineering, Hong Kong, in 2019. His research interests include image matching & localization, structure from motion, SLAM, and 3D reconstruction.

Tianwei Shen received the bachelor's degree from Peking University, China, double major in machine intelligence (EECS) and psychology. He is currently working toward the PhD degree in the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong, advised by Professor Long Quan. His research interests include large-scale 3D reconstruction and geometric learning problems in 3D vision.

Yurong Chen received the BS and PhD degrees from Tsinghua University, China, in 1998 and 2002 respectively. He joined Intel, in 2004 after completing the postdoctoral research in the Institute of Software, CAS, where he is currently a principal research scientist and director of Cognitive Computing Lab, Intel Labs China, responsible for leading visual cognition and machine learning research for Intel platforms. He received one "Intel China Award" and three Intel Labs Gordy Awards. He has published more than 60 papers and holds more than 50 issued/pending patents.

Long Quan received the PhD degree in computer science at INRIA, France, in 1989. He is currently a professor at the Department of Computer Science and Engineering, Hong Kong University of Science and Technology, Hong Kong. He is an IEEE fellow of the Computer Society. He has served in all the major computer vision journals, as an associate editor of the *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, a regional editor of the *Image and Vision Computing Journal (IVC)*, an editorial board member of the *International Journal of Computer Vision (IJCV)*, an editorial board member of the *Electronic Letters on Computer Vision and Image Analysis (ELCVIA)*, an associate editor of the *Machine Vision and Applications (MVA)*, and an editorial member of the *Foundations and Trends in Computer Graphics and Vision*.

Hongen Liao (Senior Member, IEEE) received the BS degree from Peking University, China, in 1996, and the ME and PhD degrees from the University of Tokyo, Japan, in 2000 and 2003 respectively. He was a research fellow of the Japan Society for the Promotion of Science. He became an associate professor of Graduate School of Engineering, University of Tokyo, Japan, in 2007. He has been selected as a National Thousand Talents distinguished professor, National Recruitment Program of Global Experts, China, since 2010, and is currently a full professor of and the vice director in the Department of Biomedical Engineering, Tsinghua University, China. He is the author and co-author of more than 270 peer-reviewed articles and proceedings papers, as well as more than 50 patents, 290 abstracts and numerous invited lectures. He is an associate editor of the IEEE EMBS Conference, and he has been the organization chair of Medical Imaging and Augmented Reality Conference (MIAR) 2008, the program chair of the Asian Conference on Computer-Aided Surgery Conference (ACCAS) 2008 and 2009, the tutorial co-chair of the Medical Image Computing and Computer Assisted Intervention Conference (MICCAI) 2009, the publicity chair of MICCAI 2010, the general chair of MIAR 2010 and ACCAS 2012, the program chair of MIAR 2013, the workshop chair of MICCAI 2013 and MICCAI 2019, and the general co-chair of MIAR 2016 and ACCAS 2018. He has served as a president of Asian Society for Computer Aided Surgery and co-chair of Asian-Pacific Activities Working Group, International Federation for Medical and Biological Engineering (IFMBE).