T-Net: Effective Permutation-Equivariant Network for Two-View Correspondence Learning

Zhen Zhong[1], Guobao Xiao[1,*], Linxin Zheng[1], Yan Lu[1] and Jiayi Ma[2]
[1]College of Computer and Control Engineering, Minjiang University, China
[2]Electronic Information School, Wuhan University, China

Abstract

We develop a conceptually simple, flexible, and effective framework (named T-Net) for two-view correspondence learning. Given a set of putative correspondences, we reject outliers and regress the relative pose encoded by the essential matrix, by an end-to-end framework, which is consisted of two novel structures: “ – ” structure and “|” structure. “ – ” structure adopts an iterative strategy to learn correspondence features. “|” structure integrates all the features of the iterations and outputs the correspondence weight. In addition, we introduce Permutation-Equivariant Context Squeeze-and-Excitation module, an adapted version of SE module, to process sparse correspondences in a permutation-equivariant way and capture both global and channel-wise contextual information. Extensive experiments on outdoor and indoor scenes show that the proposed T-Net achieves state-of-the-art performance. On outdoor scenes (YFCC100M dataset), T-Net achieves an mAP of 52.28%*, a* 34.22% *precision increase from the best-published result* (38.95%)*. On indoor scenes (SUN3D dataset), T-Net* (19.71%) *obtains a* 21.82% *precision increase from the best-published result* (16.18%)*. Source code:* `https://github.com/x-gb/T-Net`.

1. Introduction

Two-view feature matching is the core of many fundamental computer vision problems [15, 12], including Structure from Motion (SfM) [32, 25], visual Simultaneous Localization and Mapping [17, 3] and image retrieval [29, 18]. However, establishing reliable correspondences is not a trivial task, due to a large number of false correspondences (i.e. outliers), caused by the large viewpoint and lighting changes, occlusion, blur, and lack of texture.

Recently, learning-based outlier rejection algorithms [16, 22, 26, 36] obtain superior matching performance due to the powerful ability in the feature extraction and nonlinear expression. The most popular networks often adopt an iterative network [22, 36], where the latter iteration inherits the weights and residuals of the previous iteration, since it can extremely boost the performance for outlier rejection. Nevertheless, we find that a large amount of information in the previous iteration is not fully exploited, and only the last iteration result could be used as the predicted weight, which may lead to sub-optimal performance.

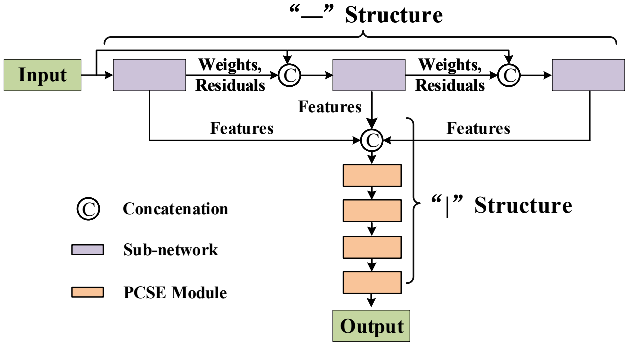

Figure 1. The proposed T-Net architecture.

To further improve the performance, we develop a novel network (named T-Net), which integrates the features of all the iterations to comprehensively utilize all the information of iterations. For convenience understanding, we regard all iterations as a whole network and each iteration as the feature learning of the sub-network in that network. As shown in Fig. 1, T-Net consists of two structures: “ – ” structure and “|” structure. “ – ” structure iteratively learns correspondence features in each sub-network, and “|” structure integrates all the features of sub-networks and learns the integrated features.

Moreover, previous works [16, 22, 36] often rely on a PointNet-like architecture with Context Normalization to learn the features. Although that module is an effective module to process unordered and sparse data (e.g., sparse correspondences), that module is not very robust to outliers [26]. To address this issue, we introduce a novel Permutation-Equivariant Context Squeeze-and-Excitation

[*] Corresponding author

(PCSE) module, which can replace PointCN [16] to capture both global and channel-wise contextual information, thus, it can extremely boost the performance.

The contributions of our work are summarized as follows:

- We propose a simple and effective framework, called T-Net, which not only learns two-view correspondences by an iterative strategy but also synthesizes the different information of each iteration.
- We propose a reformulation of SE module in the context of sparse correspondences, to capture contextual information in an equivariant manner.
- We achieve state-of-the-art performance for two-view correspondence learning. On YFCC100M unknown dataset, T-Net achieves an mAP of 52.28%, a 34.22% precision increase from the best-published result (38.95%). On SUN3D dataset, T-Net (19.71%) obtains a 21.82% precision increase from the best-published result (16.18%).

2. Related Work

2.1. Handcrafted Methods

Typically, the most popular formulation of the handcrafted methods is RANSAC [7] and its variations [30, 4, 21, 2]. The common idea of these methods uses a hypothesize-and-verify approach to seek the largest co-consistent correspondence set. Inspired by RANSAC, many methods have been proposed. For instance, MLESAC [30] uses likelihood instead of reprojection and shows improvements on image geometry problems. DEGENSAC [4] employs homographies for degeneracy checking. USA [21] integrates multiple advancements into a unified framework. GC-RANSAC [2] uses local optimization to distinguish inliers and outliers. Those methods perform well and often are regarded as a standard solution for establishing correspondences. However, those methods rely on the reliability of sampled subsets, making it limited or even failed when the data involves a large number of outliers.

2.2. Learning-Based Methods

Recent works [34, 5, 19, 6, 23] are proposed to improve handcrafted features (e.g., SIFT [14]) for local features detection. However, they inevitably generate correspondences that contain numerous outliers in real-world applications.

Other approaches generate correspondences by graph neural networks [37, 24], which treat the matching problems as an assignment problem or an optimal transport problem. Such as, CMPNN [37] proposes a graph neural network to transform coordinates of feature points into local features. SuperGlue [24] adopts a context aggregation mechanism based on attention to infer underlying 3D scenes and feature assignments jointly. Although graph neural network provides a new view on feature matching, those methods require an excessive memory footprint and huge network parameters. They often fail when the data involves a large number of correspondences per image pairs.

As a new direction, some works [16, 22, 26, 36] attempt to establish reliable correspondences by formulating the matching problem as an inlier/outlier classification problem. For instance, CNe-Net [16] introduces a PointNet-like architecture and Context Normalization, which we call PointCN, to classify the putative correspondences, and adopts the weighted eight-point algorithm to regress the essential matrix. DFE [22] not only uses PointCN but also adopts an iterative strategy to drastically improve performance. ACNe-Net [26] proposes Attentive Context Normalization to establish reliable correspondences. OA-Net++ [36] uses a Differentiable Pooling layer, Order and Aware Filtering block, and Differentiable Unpooling layer, which we refer to as DP&OA&DUP module, to capture local and global spatial context. Moreover, OA-Net++ adopts an iterative network and achieves a significant performance improvement on pose estimation. In this paper, our network is also based on the iterative network. However, unlike DFE and OA-Net++, which only use the information of residuals and weights from the last sub-network, we gather all the features of the iterative sub-networks and predict weights based on gathered features. In addition, we also propose a novel module (i.e., PCSE module), which can capture more context information and boost the matching performance over PointCN.

2.3. Channel Attention

Recently, the channel attention mechanism has achieved significant success in the deep convolutional neural networks. For instance, SE-Net [11] proposes the “Squeeze-and-Excitation” (SE) block and achieves promising performance. SK-Net [13] proposes a dynamic selection mechanism in CNNs which can adaptively adjust its receptive field. MobileNetV3 [10] adopts SE block and a hard-swish activation function to build lightweight attention modules. ECA-Net [31] employs an adaptive kernel size to replace FC layers in SE block. However, all of the above methods focus on regular grid data, such as image data. In contrast, our PCSE module aims to handle sparse and unordered data in a permutation-equivariant way.

3. T-Net

In this section, we introduce the details of the proposed T-Net for learning two-view correspondences and geometry. Specifically, we first describe the formulation of our problem in Sec. 3.1. Then, we develop a T-Structure network to synthesize the information of all the iterative sub-networks

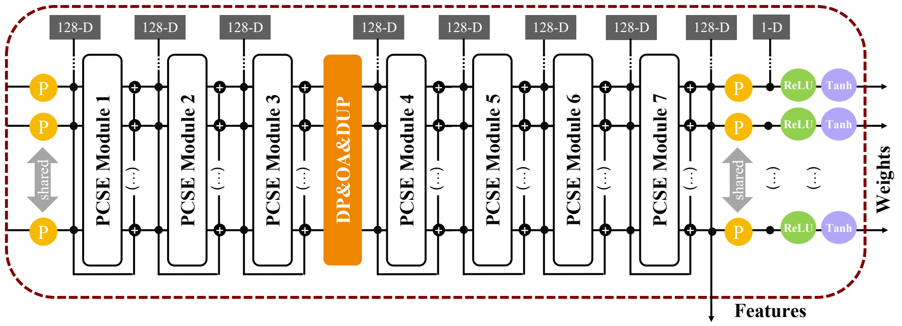

Figure 2. The proposed sub-network architecture.

in Sec. 3.2. After that, in Sec. 3.3, we propose a novel basic module (named PCSE module) that can efficiently capture both global and channel-wise contextual information. Finally, we describe the network architecture in Sec. 3.4.

3.1. Problem Formulation

Given a pair of images in the same scene, our goal is to establish reliable correspondences and recover the relative pose. Specifically, we first adopt local features (e.g., SIFT [14]) to detect keypoints and extract descriptors. Then the descriptors are computed by nearest neighbor search to create a set of N putative correspondences:

$$\begin{aligned} C &= [c_1; c_2; ......; c_N], \\ c_i &= [u_1^i, v_1^i, u_2^i, v_2^i], \end{aligned} \tag{1}$$

where c_i represents a putative correspondence; (u_1^i, v_1^i) and (u_2^i, v_2^i) are keypoint coordinates from an image pairs, respectively. Following [16], the keypoint coordinates are normalized by camera intrinsics. After that, we treat the two-view geometry estimation task as an outlier/inlier classification problem and an essential matrix regression problem. As shown in Fig. 2, our T-Net takes the putative correspondence set C as input and output the weight set W:

$$W = [w_1; w_2; w_3; ...; w_N], \tag{2}$$

where $w_i \in [0, 1)$ is the output weight of correspondences c_i. $w_i > 0$ indicates an inlier, and outlier otherwise. Finally, we employ the weighted eight-point algorithm [16] to regress the essential matrix based on the weight set W. The whole architecture can be written as:

$$W = f_\varphi(C), \tag{3}$$

$$\hat{E} = g(W, C), \tag{4}$$

where $f_\varphi(\cdot)$ represents a permutation-equivariant neural network. φ denotes the network parameters. $\hat{E}$ represents the regressed essential matrix. $g(\cdot, \cdot)$ is the weighted eight-point algorithm to compute the essential matrix $\hat{E}$ via self-adjoint eigen-decomposition.

3.2. T-Structure Network

As reported by previous works [22, 36], there are two important discoveries about the iterative network: 1) The iterative network can significantly promote the network performance for outlier rejection. 2) A network with more sub-networks (i.e., iterations) means the network tends to have superior experiment performance. However, in the iterative network of previous works [22, 36], only the last sub-network result is used as the predicted weight, while a large amount of information in the previous sub-network is ignored. This kind of operation will cause a significant loss of information.

In this section, we develop a novel structure (called T-Structure), which includes two structures: " $-$ " structure and "|" structure. " $-$ " structure consists of a series of iterative sub-networks where the latter sub-network inherits the weights and residuals of the previous sub-network. "|" structure comprises three novel operations (i.e., feature extraction, feature concatenation and feature learning).

Feature extraction: "|" structure extracts the features of the last built block in each sub-network. This operation is used to capture valuable information from each sub-network.

Feature concatenation: "|" structure adopts concatenate strategy to integrate the features of each sub-network:

$$F_{all} = F_1 \bigoplus F_2 \cdots \bigoplus F_S, \tag{5}$$

where F_S represents the feature in S sub-network. $\bigoplus$ indicates the concatenate operation.

Feature learning: Following feature concatenation operation, we employ four PCSE modules to learn the concatenated features:

$$F_{final} = f_{|}(F_{all}), \tag{6}$$

where F_{final} is the output of the feature learning operation. $f_{|}(\cdot)$ denotes the feature learning module which consists of four PCSE modules. At last, we achieve the final weights by two activation functions $W = tanh(ReLU(F_{final}))$. Based on the above operations, our T-Structure will enhance

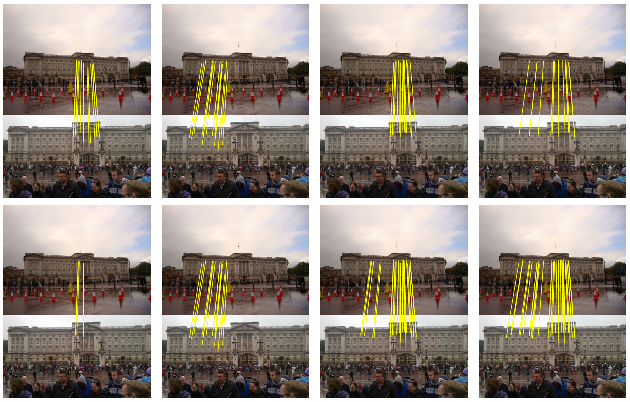

Figure 3. Visual results of top 15 response correspondences (**the top row**) and a fixed threshold of $w > 0.9$ (**the bottom row**) in the same image pair from the different sub-networks. From left to right: the results of the first sub-network, the second sub-network, the third sub-network and the final output. We draw the correspondences in yellow if they conform to the ground truth epipolar geometry.

the model. That is because our T-Structure not only depends on the last sub-network results, but also captures valuable information from every single sub-network.

To further illustrate our network, we show an example to visualize the output in each sub-network and the final result of our network in Fig. 3. For the top row, we visualize the top 15 responses in each sub-network and our final output. We can see that, the focus of each sub-network is different, but our T-Net will combine the focus of each sub-network. Compared with other sub-networks, T-Net has wider points of interest. For the bottom row, we visualize matches with a weight greater than 0.9. After collecting all features, T-Net will assign higher weights to inliers compared with other sub-networks.

3.3. PCSE module

SE module is a key basic module for many neural network architectures [11, 27, 10]. The standard SE module involves three convolution layers with 1×1 or 3×3 kernels and a squeeze-and-excitation Block. In particular, the 3×3 convolution layer, which is used to extract local information, is a crucial convolution layer for SE module. However, the 3×3 convolution kernel will mix putative correspondences that are sparse and unordered in the form of point clouds. Thus, SE module is ill-suited for processing unordered correspondences. To address the above issues, we propose the PCSE module (i.e., a permutation-equivariant module) to learn sparse and unordered correspondences. As illustrated in Fig. 4, all the convolution layers use 1×1 convolution kernel, therefore, this framework will learn correspondence features in a particular canonical order. Moreover, we introduce Context Normalization [16] after every convolution layer. That operation can embed global context information for each correspondence, which is critical for sparse correspondences.

Formally, let $f_i^c \in R^{C^c}$ be the output of c-th channel C for the i correspondence, Context Normalization is formulated as:

$$CN(f_i^c) = \frac{f_i^c - u^l}{o^c}, \tag{7}$$

where

$$u^c = \frac{1}{N}\sum_{i=1}^{N} f_i^c,\ o^c = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_i^c - u^c)}. \tag{8}$$

Following Context Normalization, we adopt Batch Normalization and the ReLU activation function to process feature maps. All processes are formulated as:

$$f_{out} = \delta(BN(CN(f_{in}))), \tag{9}$$

where f_{in} denotes the output of convolution layer. BN represents Batch Normalization. δ is the ReLU. For capturing the contextual information in different channels, we adopt a squeeze and excitation strategy [11], after the second ReLU layer. Specifically, given the c-th feature map f_c, we first adopt the global average pooling to capture channel-wise global contextual information gc:

$$gc_c = \frac{1}{N}\sum_{i=1}^{N} f_c i, \tag{10}$$

where c is the c-th layer feature-map. Next, for learning the contextual information in different channels, we employ a bottleneck with two fully connected (FC) layers and a softmax operator to process global contextual and get gc'. After that, a weighted fusion of the c-th feature-map V_c is:

$$V_c = gc_c' \times FG_c. \tag{11}$$

At last, we obtain the output V of PCSE layer by concatenating all the feature-maps (i.e., $V = cat(V_1, V_2, V_3, \ldots, V_C)$). Note that, we also adopt a residual structure for PCSE module to prevent network degradation:

$$f_y = f_x + F_{PCSE}(f_x), \tag{12}$$

where f_x and f_y represent the input and output features, respectively. $F_{PCSE}(\cdot)$ is the PCSE module.

3.4. Network Architecture

In the subsection, we describe our T-Net in detail. As shown in Fig. 1, T-Net consists of two structures: " $-$ " structure and "|" structure. " $-$ " structure contains three sub-networks. "|" structure collects the feature from each

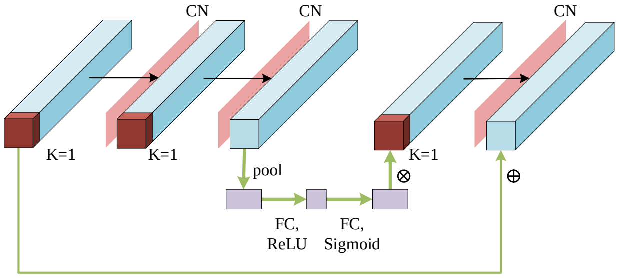

Figure 4. The proposed PCSE module architecture.

sub-network and learns the feature by four PCSE modules. For each sub-network (see Fig. 2), seven PCSE modules are stacked. The detailed structure of PESA module (Fig. 4) is C(128, 1) - C(128,1) - P(128) - FC(64) - FC(128) - C(128,1), where C(c, k) represents a convolution layer with c output channels and convolution kernel = $k \times k$. P(c) and FC(c) are global pooling layer and fully connected layer, respectively. Moreover, the DP&OA&DUP module follows the 3-th PCSE module.

3.4.1 DP&OA&DUP module

The DP&OA&DUP module [36] can effectively extract local and global information of correspondences. In T-Net, we employ the DP&OA&DUP module in each sub-network to help extract the local and global features. Here, we briefly introduce DP&OA&DUP module.

Specifically, DP&OA&DUP module involves three parts: Differentiable Pooling layer, Order and Aware Filtering block, and Differentiable Unpooling layer. The Differentiable Pooling Layer is first proposed by Graph Neural Network [35], which is permutation-invariant and originally designed for GNN. OA-Net++ generalizes it to capture the local information of correspondences. The Differentiable Pooling Layer adopts a soft assignment matrix to map the correspondence features into a set of clusters and employs a Softmax function to normalize the clusters. For the Order-Aware Filtering Block, it exploits the cluster relation with spatially-correlated operators by applying Multi Layer Perceptrons (MLPs) directly on the spatial dimension. The Differentiable Unpooling layer first reverses the behavior of the DiffPool layer and recovers the clusters to the original size. Next, it embeds cluster features by the channel-wise multiplication with the input feature of the P&OA&UP module.

3.4.2 Loss Function

Following [8, 22], we employ a hybrid loss function to optimize T-Net:

$$Loss = l_c(W, L) + \alpha l_e(\hat{E}, E), \tag{13}$$

where $l_c(\cdot,\cdot)$ denotes the binary cross entropy loss for the classification term. W is the predicted weights. In particular, L represents weakly supervised labels, which are evaluated by geometric error [22], with a threshold of 10^{-4} used to determine valid correspondences. $l_g(\cdot,\cdot)$ is the essential matrix loss. $\hat{E}_q$ and E_q are the predicted essential matrix and the ground truth essential matrix, respectively. α is a weight to balance the binary cross entropy loss and the essential matrix loss.

For the essential matrix loss, we calculate it by:

$$l_e(\hat{E}, E) = \frac{\left(p'^T \hat{E} p\right)^2}{||Ep||_{[1]}^2 + ||Ep||_{[2]}^2 + ||Ep'||_{[1]}^2 + ||Ep'||_{[2]}^2}, \tag{14}$$

where p and p' denote two keypoint positions forming the correspondence. $A_{[i]}$ indicates the i-th element of vector A.

4. Experiments

In the section, we evaluate the performance of T-Net and compare it with recent state-of-the-art methods on outdoor YFCC100M [28] and indoor SUN3D [33], for the camera pose estimation and outlier rejection tasks. In addition, we also test on different local features and report the results. In the following: we first introduce the datasets and evaluation metrics, and then we report the implementation details and experiment results. Finally, we analyze the ablation study.

4.1. Datasets

4.1.1 Outdoor Scenes

We use Yahoo's YFCC100M dataset [28], an outdoor dataset composed of 100 million photos from the internet. The authors of [9] splits it into 72 image sequences from different tourist spots. Following [36], we use 4 sequences (i.e., Buckingham palace, Sacre Coeur, Reichstag and Notre dame front facade) as unknown scenes to test generalization ability and the remaining 68 sequences as training sequences. In addition, we use [9] to recover the camera poses and generate ground truth.

4.1.2 Indoor Scenes

For the indoor dataset, we evaluate on the SUN3D dataset [33], which includes a series of RGB-D videos with camera poses. We sub-sample videos in every 10 frames to generate an image. In addition, we preserve the same setting as [36] (i.e., 15 scenes as unknown scenes for testing and 239 scenes for training).

In this work, we re-train all models in the same setting and test both known scenes and unknown scenes. The unknown sequences are the testing sequences introduced above. For the known scenes, we split the training sequences into three sets, i.e., the training set (60%), valida-

Table 1. Performance comparison for camera pose estimation on YFCC100M and SUN3D datasets. Results **without/with** RANSAC post-processing under error thresholds of 5° and 20° are reported. The best result of each dataset is boldfaced.

Local Features	Datasets	YFCC100M(%)				SUN3D(%)			
	Matcher	Known Scene		Unknown Scene		Known Scene		Unknown Scene	
		5°	20°	5°	20°	5°	20°	5°	20°
SuperPoint	RANSAC	-/12.85	-/31.22	-/17.47	-/38.83	-/14.93	-/38.16	-/12.15	-/33.01
	Point-Net++	11.87/28.46	33.35/51.01	17.95/38.83	49.32/64.04	11.40/21.19	31.96/47.03	9.38/17.08	31.16/40.13
	CNe-Net	12.18/30.25	34.75/52.13	24.25/42.57	52.70/66.89	12.63/21.81	32.46/46.96	10.68/17.36	32.68/40.66
	DFE	18.79/31.72	40.53/53.56	29.13/43.00	58.41/67.51	13.35/22.57	35.45/47.84	12.04/17.41	33.62/40.99
	OA-Net++	29.52 /35.72	53.76/57.75	35.27/45.45	66.81/70.99	20.01/24.43	45.97/49.77	15.62/18.56	40.95/42.66
	ACNe-Net	26.72/31.16	49.29/56.68	32.98/45.34	62.68/69.19	18.35/21.12	43.97/48.76	13.82/18.05	37.73/41.78
	T-Net	35.73/**37.99**	58.95/**59.01**	40.62/**46.37**	70.62/**71.01**	21.62/**24.66**	47.60/**50.17**	17.18/**19.09**	42.83/**43.25**
SIFT	RANSAC	-/5.81	-/16.88	-/9.07	-/22.92	-/4.52	-/15.46	-/2.84	-/11.19
	Point-Net++	10.49/33.78	31.17/56.24	16.48/46.25	42.09/67.53	10.58/19.17	35.75/44.06	8.10/15.29	30.97/35.83
	CNe-Net	13.81/34.55	35.20/57.27	23.95/48.03	52.44/69.10	11.55/20.60	36.12/44.33	9.30/16.40	31.32/37.23
	DFE	19.13/36.46	42.03/59.15	30.27/51.16	59.18/70.88	14.05/21.32	39.12/44.67	12.06/16.26	36.17/37.72
	OA-Net++	32.57/41.53	56.89/63.91	38.95/52.59	66.85/72.99	20.86/22.31	48.06/47.17	16.18/17.18	41.83/39.39
	ACNe-Net	29.17/40.32	52.59/62.11	33.06/50.89	62.91/71.25	18.86/22.12	46.35/46.90	14.12/16.99	39.17/39.01
	T-Net	44.49/**47.00**	66.75/**68.30**	52.28/**56.08**	**75.85**/75.46	**24.96**/23.81	**52.69**/48.46	**19.71**/18.00	**46.33**/40.75

tion set (20%), and testing set (20%), and pick the testing set as known scenes.

4.2. Evaluation Metrics

We use two set different evaluation metrics for two different tests, i.e., camera pose estimation and outlier rejection. For camera pose estimation, we employ the mean average precision (mAP) both on rotation and translation to evaluate the angular difference between the predicted vectors and ground truth. For outlier rejection, we adopt three popular metrics, including precision (P), recall (R), and F-measure (F) to verify performance.

4.3. Implementation Details

T-Net is implemented in PyTorch. We train our net with a batch size of 32. During training, we adopt Adam [1] optimizer with the learning rate of 10^{-3} to minimize the loss. In addition, the parameter α is 0 during the first $2k$ iterations and 0.1 in the rest $480k$. All experiments are performed on Linux $3.10.0$ with NVIDIA TESLA P100 GPUs.

4.4. Camera Pose Estimation

Camera pose estimation is an extremely challenging test. When testing on the outdoor dataset, lighting changes, and occlusion often limit the performance of matchers. For the indoor dataset, lacking texture and large viewpoint changes are the main challenges. Here, we test our network on both outdoor and indoor datasets with five state-of-the-art baselines. As shown in the experimental results, T-Net can overcome these challenges and achieve the best performance.

4.4.1 Baselines

We evaluate T-Net and six state-of-the-art baselines (i.e., RANSAC [7], Point-Net++ [20], CNe-Net [16], DFE [22], OA-Net++ [36] and ACNe-Net [26]) using both handcrafted features (i.e., SIFT [14]) and learned features (i.e., SuperPoint [5]). For Point-Net++, we replace 3D Euclidean space to 4D to search neighbors. For CNe-Net [16] and DFE [22], we replace their original loss formulation to our hybrid loss function. For OA-Net++, we employ the official implementation. For ACNe-Net, we implement it on the PyTorch version with the authors' help. To have a fair comparison, we train all these models at the same settings.

4.4.2 Results

We show the performance comparison for camera pose estimation on YFCC100M and SUN3D datasets in Table 1. We can see that, when evaluating on SIFT, our network outperforms state-of-the-art baselines under all settings. T-Net achieves the mean average precision (mAP) of 52.28% and 19.71% on both outdoor and indoor unknown scenes at 5° error threshold without RANSAC, which are the 13.33% and 3.53% improvements than OA-Net++. In addition, we also achieve a clear improvement gain the baseline OA-Net++ on both outdoor and indoor unknown scenes with RANSAC post-processing. These results demonstrate the effectiveness of our T-Net. One of the reasons for the effectiveness is the using of our T-Structure. Different from the iterative network, we collect all the features and reduce the feature loss from the sub-network as much as possible.

Fig. 5 shows some typical results of our network and other baselines. Moreover, we observe that RANSAC post-

Table 2. Comparative results of outlier rejection on the YFCC100M and SUN3D datasets.

Datasets	YFCC100M(%)						SUN3D(%)					
Matcher	Known Scene			Unknown Scene			Known Scene			Unknown Scene		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
RANSAC	47.35	52.39	49.74	43.55	50.65	46.83	51.87	56.27	53.98	44.87	48.82	46.76
Point-Net++	49.62	86.19	62.98	46.39	84.17	59.81	52.89	86.25	65.57	46.30	82.72	59.37
CNe-Net	54.43	86.88	66.93	52.84	85.68	65.37	53.70	87.03	66.42	46.11	83.92	59.52
DFE	56.72	87.16	68.72	54.00	85.56	66.21	53.96	87.23	66.68	46.18	84.01	59.60
OA-Net++	60.03	89.31	71.80	55.78	85.93	67.65	54.30	88.54	67.32	46.15	84.36	59.66
ACNe-Net	60.02	88.99	71.69	55.62	85.47	67.39	54.11	88.46	67.15	46.16	84.01	59.58
T-Net	**62.14**	**91.70**	**74.08**	**57.48**	**88.39**	**69.66**	**54.98**	**88.82**	**67.92**	**46.94**	**84.53**	**60.36**

procession may harm performance, especially on SUN3D dataset, which is an extremely challenging dataset. This is because the SUN3D dataset contains large viewpoint changes, lack of texture, and a large amount of self-similarity, which is difficult for SIFT descriptors to provide effective information, resulting in SIFT generating a large number of outliers. Through the feature fusion of different sub-networks and the rich contextual information extracted by PCSE, our network can retain many key inliers, but RANSAC mainly focuses on the largest collective set and may remove some key inliers. The bottom part in Table 1 can support our point. RANSAC can improve performance when evaluating SuperPoint of the SUN3D dataset. Note that, our network gets the best result in the extremely challenging scenes, including lighting changes, occlusion, lacking texture and large viewpoint changes.

When evaluating on SuperPoint, our network ranks the 1-st on both outdoor and indoor scenes and surpasses all baselines both with and without RANSAC post-processing. In addition, as reported by Table 1, we observe that SuperPoint can improve the performance when the method performs worse, but degrades the performance when the method performs well. The main reason is that SuperPoint has better descriptors but suffers from the accuracy of keypoint position. The better descriptors can offer higher inlier ratio in putative correspondence set. However, the less keypoint accuracy will mainly limit the final performance when the method rejects enough outliers, thus, SuperPoint performs worse.

4.5. Outlier Rejection

Outlier rejection is a critical step in two-view matching. In the test, we further evaluate the outlier rejection performance of T-Net. We test the outdoor and indoor dataset with SIFT local features and set the comparison methods as the same as the camera pose estimation task. As shown in Table 2, our T-Net gets the best result in all evaluation metrics (i.e., precision, recall, and F-measure). The learning-based method clearly outperforms the classical method (i.e., RANSAC), which demonstrates that the learning-based method can effectively reject outliers. Additionally, as reported in Tables 1 and 2, the performance of relative pose estimation is positively related to the performance of outlier removal.

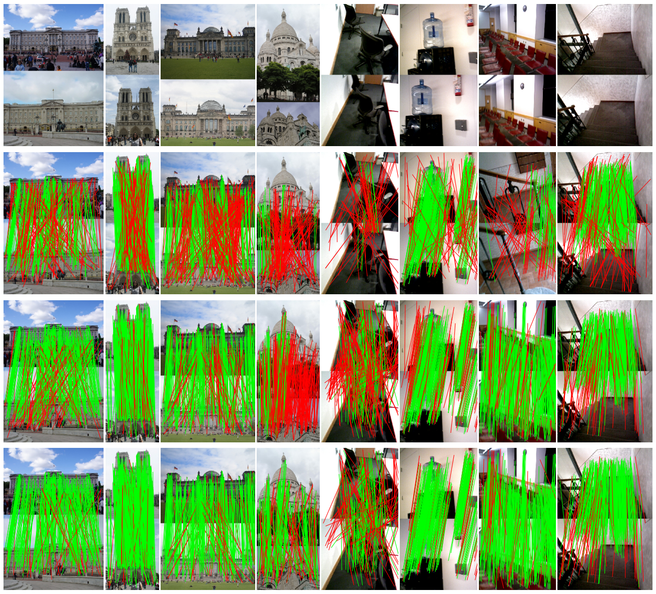

Figure 5. Visualization results on two challenging datasets, i.e., YFCC100M, SUN3D. From left to right: Buckingham-palace, Notre-dame-front-facade, Reichstag, Sacre-coeur, Te-mit1, Te-brown1, Te-harvard1 and Te-hotel1. From top to bottom: Original image pairs, and the results of RANSAC [7], OA-Net++ [36] and our network. We draw the correspondences in green if they conform to the ground-truth epipolar geometry, and in red otherwise.

4.6. Ablation Study

The core of our network is two key ideas: a novel architecture (T-Structure) extensively collects all the features of each sub-network and outputs the weight, and a PCSE module captures contextual information not only global-wise but also channel-wise to boost the performance. For exam-

Table 3. Ablation study on YFCC100M. The result is mAP (%) under error thresholds of 5° on both known and unknown scenes with weighted 8-point algorithm. **P&O&U**: using the D-P&OA&UDP module. **Iter**: using the iterative network. **SE**: using the SE module. **SE-P**: using the permutation-equivariant version for SE module. **PCSE**: using the PCSE module. **T**: T-Structure network.

PointCN	P&O&U	Iter	SE	SE-P	PCSE	T	Known	Unknown
✓							21.79	30.08
✓	✓						31.99	36.95
✓	✓	✓					36.16	40.88
	✓	✓	✓				25.55	32.68
	✓	✓		✓			31.00	36.85
	✓	✓			✓		40.43	47.73
	✓				✓	✓	**44.49**	**52.28**

ining the impact of these two choices, we implement each network with 25 build blocks to make sure their parameters almost the same, and use the weighted 8-point algorithm to compute the essential matrix. In addition, we will compare with Graph Neural Networks on YFCC100M with SIFT detectors.

4.6.1 Plain Network vs Iterations vs T-Structure

We consider three different settings: one plain network without iterations or T-Structure, one refinement network with iterative structure, and one network with T-Structure. From the results, we find that our T-Structure can work well for the task. Although the iterative structure performs much better than the pain network, they will lose a mass of information in the previous sub-network. Table 3 reports that our T-Structure can largely improve the mAP of iterative structure from 47.73% to 52.28% on unknown scenes without RANSAC.

4.6.2 PointCN vs SE module vs SE-P module vs PCSE module

We replace PointCN with PCSE module which can extract both global and channel-wise contextual information. In addition, we also compare with SE module, a popular architecture on regular data (i.e., image data), and SE-P module, a permutation-equivariant version for SE module which simply replaces the 3×3 convolution kernel as 1×1. From Table 3, we observe that SE module performs worse than SE-P module. It demonstrates that permutation-equivariant architecture is very important for unordered and sparse correspondences. Moreover, both SE module and SE-P module perform worse than PointCN. That is because SE module and SE-P module cannot capture enough contexture information which is critical for sparse correspondences. In contrast to SE module and SE-P module, our PCSE module not only learns features in a permutation-equivariant manner but also extracts more contextual information, thus, PCSE module achieves better improvement over other methods.

Table 4. Comparison with Graph Neural Networks. The results of mAP (%) under error thresholds of 5° on indoor unknown scenes (with/without RANSAC processing) are reported.

Methods	5°	10°	Parameters	GFLOPs
RANSAC	-/12.07	-/19.86	-	-
SuperGlue	6.30/16.09	11.21/25.96	12.02M	19.59
T-Net	**14.45/16.35**	**24.08/26.23**	**3.73M**	**1.01**

4.6.3 Comparison with Graph Neural Networks

We compare our T-Net with the state-of-the-art Graph Neural Network, i.e. SuperGlue [24], for feature matching. As mentioned in Sec. 2.2, the Graph Neural Network directly generates reliable correspondences from local features. The evaluation metrics include both effectiveness (i.e., the mean average precision (mAP)) and efficiency (i.e., network parameters and floating-point operations). For comparison, we test again on an extremely challenging dataset (i.e., SUN3D) with 512 SuperPoint keypoints. To have a fair comparison, both T-Net and SuperGlue use the pretrained model provided by previous tests and official implementations, respectively. The result is reported in Table 4. We observe that SuperGlue extremely relies on RANSAC for post-processing, when directly recovering the pose by the weighted eight-point algorithm, T-Net outperforms SuperGlue 8.15% and 12.87% on 5° and 10° threshold under the obviously fewer network parameters and computational cost.

5. Conclusion

In this work, we propose T-Net, a new end-to-end trainable model, for learning two-view correspondences and geometry. Our work mainly contains two contributions: (i) T-Structure architecture, which iteratively learns the correspondence features and predicts the final weight based on all the features from each sub-network. (ii) PCSE module, which is able to capture the contextual information from not only global but also channel-wise aspects. Extensive experiments demonstrate that T-Net achieves significant improvements over existing approaches on both camera pose estimation and outlier rejection task.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grants 62072223, and supported by the Natural Science Foundation of Fujian Province under Grant 2020J01131199.

References

[1] Paszke Adam, Gross Sam, Chintala Soumith, Chanan Gregory, Yang Edward, D Zachary, Lin Zeming, Desmaison Alban, Antiga Luca, and Lerer Adam. Automatic differentiation in pytorch. In *NIPS*, 2017.

[2] Daniel Barath and Jiří Matas. Graph-cut ransac. In *CVPR*, pages 6733–6741, 2018.

[3] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, 2016.

[4] Ondrej Chum, Tomas Werner, and Jiri Matas. Two-view geometry estimation unaffected by a dominant plane. In *CVPR*, pages 772–779, 2005.

[5] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPR*, pages 224–236, 2018.

[6] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint detection and description of local features. In *CVPR*, 2019.

[7] Martin A Fischler and Robert C Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Comm. ACM*, 24(6):381–395, 1981.

[8] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[9] Jared Heinly, Johannes L Schonberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the world* in six days*(as captured by the yahoo 100 million image dataset). In *CVPR*, pages 3287–3295, 2015.

[10] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenet v3. In *ICCV*, pages 1314–1324, 2019.

[11] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *CVPR*, pages 7132–7141, 2018.

[12] Yuhe Jin, Dmytro Mishkin, Anastasiia Mishchuk, Jiri Matas, Pascal Fua, Kwang Moo Yi, and Eduard Trulls. Image matching across wide baselines: From paper to practice. *arXiv preprint arXiv:2003.01587*, 2020.

[13] Xiang Li, Wenhai Wang, Xiaolin Hu, and Jian Yang. Selective kernel networks. In *CVPR*, pages 510–519, 2019.

[14] David G Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[15] Jiayi Ma, Xingyu Jiang, Aoxiang Fan, Junjun Jiang, and Junchi Yan. Image matching from handcrafted to deep features: A survey. *International Journal of Computer Vision*, 129(1):23–79, 2021.

[16] Kwang Moo Yi, Eduard Trulls, Yuki Ono, Vincent Lepetit, Mathieu Salzmann, and Pascal Fua. Learning to find good correspondences. In *CVPR*, pages 2666–2674, 2018.

[17] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[18] Hyeonwoo Noh, Andre Araujo, Jack Sim, Tobias Weyand, and Bohyung Han. Large-scale image retrieval with attentive deep local features. In *CVPR*, pages 3456–3465, 2017.

[19] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net: learning local features from images. In *INPS*, pages 6234–6244, 2018.

[20] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NIPS*, pages 5099–5108, 2017.

[21] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. Usac: A universal framework for random sample consensus. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):2022–2038, 2012.

[22] René Ranftl and Vladlen Koltun. Deep fundamental matrix estimation. In *ECCV*, pages 284–299, 2018.

[23] Jerome Revaud, Philippe Weinzaepfel, César Roberto de Souza, and Martin Humenberger. R2D2: repeatable and reliable detector and descriptor. In *NIPS*, 2019.

[24] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, pages 4938–4947, 2020.

[25] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, pages 4104–4113, 2016.

[26] Weiwei Sun, Wei Jiang, Eduard Trulls, Andrea Tagliasacchi, and Kwang Moo Yi. Acne: Attentive context normalization for robust permutation-equivariant learning. In *CVPR*, pages 11286–11295, 2020.

[27] Mingxing Tan, Bo Chen, Ruoming Pang, Vijay Vasudevan, Mark Sandler, Andrew Howard, and Quoc V Le. Mnasnet: Platform-aware neural architecture search for mobile. In *CVPR*, pages 2820–2828, 2019.

[28] Bart Thomee, David A Shamma, Gerald Friedland, Benjamin Elizalde, Karl Ni, Douglas Poland, Damian Borth, and Li-Jia Li. Yfcc100m: The new data in multimedia research. *arXiv preprint arXiv:1503.01817*, 2015.

[29] Giorgos Tolias, Yannis Avrithis, and Hervé Jégou. Image search with selective match kernels: aggregation across single and multiple images. *International Journal of Computer Vision*, 116(3):247–261, 2016.

[30] Philip HS Torr and Andrew Zisserman. Mlesac: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156, 2000.

[31] Qilong Wang, Banggu Wu, Pengfei Zhu, Peihua Li, Wangmeng Zuo, and Qinghua Hu. Eca-net: Efficient channel attention for deep convolutional neural networks. In *CVPR*, 2020.

[32] Changchang Wu. Towards linear-time incremental structure from motion. In *3DV*, pages 127–134, 2013.

[33] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. Sun3d: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, pages 1625–1632, 2013.

[34] Kwang Moo Yi, Eduard Trulls, Vincent Lepetit, and Pascal Fua. Lift: Learned invariant feature transform. In *ECCV*, pages 467–483, 2016.

[35] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, Will Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In *NIPS*, pages 4800–4810, 2018.

[36] Jiahui Zhang, Dawei Sun, Zixin Luo, Anbang Yao, Lei Zhou, Tianwei Shen, Yurong Chen, Long Quan, and Hongen Liao. Learning two-view correspondences and geometry using order-aware network. In *ICCV*, pages 5845–5854, 2019.

[37] Zhen Zhang and Wee Sun Lee. Deep graphical feature learning for the feature matching problem. In *ICCV*, pages 5087–5096, 2019.