CROSS-ATTENTION-BASED FEATURE EXTRACTION NETWORK FOR 3D POINT CLOUD REGISTRATION

Shiyi Guo[1,2]*, Yujie Fu*[1,2]*, Zhengda Qian*[1,2]*, Zheng Rong*[1,*] *and Yihong Wu*[1,2,*]

[1]National Laboratory of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, China
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, China
{shiyi.guo, yujie.fu, zhengda.qian, zheng.rong, yhwu}@nlpr.ia.ac.cn

ABSTRACT

In recent years, the feature-based point cloud registration methods have attracted more attention. However, most existing methods focus on extracting features with strong anti-interference ability from a single point cloud while neglecting the differences within point cloud pairs. In this paper, unlike these methods treating each point cloud independently, we instead consider the information between point cloud pairs when extracting features. Specifically, we propose a cross-attention-based network for modeling the correlation between a pair of point clouds, where a 3D cross-attention mechanism is proposed and combined with 3D convolution elegantly for feature extraction. The extracted features achieve better robustness under various conditions, such as rotation and translation changes. Then accurate point cloud registration is achieved by matching these features. Experimental results on 3DMatch dataset show that the proposed method achieves state-of-the-art performance on feature matching and point cloud registration tasks compared with the previous feature-based methods.

***Index Terms*—** Point cloud registration, Feature extraction and matching, Cross-attention, 3D convolution network

1. INTRODUCTION

The goal of point cloud registration is to find an optimal transformation to match two given partially overlapped point clouds with unknown point correspondences. It is an extremely important task in many applications of computer vision and mobile robots, such as simultaneous localization and mapping (SLAM) [1] and 3D reconstruction [2]. Recently, with the rapid development of 3D local features [3, 4, 5, 6, 7], feature-based point cloud registration has attracted more attention. The state-of-the-art feature-based pipelines for point cloud registration consist of local feature extraction, feature matching, and outlier rejection. From these we can find that the fundamental task is to build feature correspondence through feature extraction and matching.

The typical procedure of feature extraction and matching methods is: first a descriptor is generated with varying degrees of invariant to rotation, translation and density [4, 3] of each point or keypoint in the point cloud, and then these descriptors are matched between point clouds by comparing descriptors exhaustively.

Previous algorithms learn descriptors for each point cloud without considering knowledge of the other point cloud. Therefore, in order to make the algorithm more robust, the extracted descriptors need to be strongly invariant to various changes, such as rotation and density changes. However, with the increasing invariance of descriptors, their discriminability decreases, which limits the accuracy of feature matching in point cloud. Instead, in this paper, we abandon the previous way and calculate the descriptors considering information on both point clouds, which allows the descriptors to be changed based on the differences between the point clouds.

Motivated by [8] in 2D image matching, we propose a novel network (CAFeat3D) to learn the descriptors considering information on both point clouds and enhance the discriminability of the learned descriptors under the circumstance of matching these two point clouds. Considering that D3Feat [3] utilizes KPConv [9], a convolution operation on 3D point cloud, as backbone network and achieves state-of-the-art performance in 3D feature extraction, we also adopts KPConv to build our backbone. Firstly, a U-Net like network developed for single point cloud feature extraction is implemented. Secondly, an identical encoder module of previous U-Net like network and an cross-attention-based module proposed by us are embedded in the previous U-Net like network, which can effectively provide the information of another point cloud for the original U-Net like network. Finally, circle loss [10] and detector loss in [3] are utilized to train our network. Compared with D3Feat [3], the proposed CAFeat3D achieves better performance with almost the same number of parameters.

To summarize, our contributions are as follows:

*Yihong Wu and Zheng Rong are the corresponding authors.

This work was supported in part by the National Natural Science Foundation of China under Grants 61836015 and 62002359, and in part by the Beijing Advanced Discipline Fund under Grant 115200S001.

- We propose a novel network to learn the features considering information on both point clouds in point cloud pairs. To our best knowledge, our work is the first attempt to realize feature matching between point clouds by considering differences within the point cloud pair.
- An cross-attention-based module is proposed to leverage differences between a point cloud pair.
- Quantitative experimental results of both feature matching and point cloud registration on public datasets show that our framework presents the superior performance over the previous methods.

The rest of this paper is organized as follows. The related work is reviewed in Section 2 and the details of CAFeat3D are described in Section 3. Experiments are presented in Section 4, followed by conclusion in Section 5.

2. RELATED WORK

2.1. Point Cloud Registration

Traditional algorithms of point cloud registration have been reviewed in [11]. In traditional algorithms, Iterative Closest Point (ICP) [12] is one of the most popular algorithms for its good performance. However, ICP-based methods is prone to fall into wrong local minimum when without good initialization. Recently, learning-based registration algorithms have attracted more attention. The learning-based 3D features, such as 3DMatch [6], FCGF [4] and D3Feat [3], have strong feature description ability, which make feature-based point cloud registration more robust. In addition, end-to-end networks for registration, such as PointNetLK [13] and RP-Net [14], have been proposed. However, their robustness needs to be improved in complex scenes [15].

2.2. 3D Features

3D features include 3D local descriptors and keypoints, which are extremely important in point cloud registration.

For 3D keypoint detection, most existing methods are hand-crafted. These alogrithms like Harris3D [16] and ISS [17] utilize local geometric properties of point clouds to detect keypoints. Recently, to make the learning-based detector, Li et al. [18] proposed USIP, an unsupervised network, to learn keypoint detector, which can detect keypoints with high repeatability and accurate localization from 3D point clouds.

For 3D local descriptors, early methods are mainly hand-crafted such as FPFH [19], which describes the local geometry around a point in point cloud by using surface normal. However, the hand-craft descriptors usually lack higher robustness. To solve this problem, learning-based approaches attract more attention recently. In [7, 6, 5, 20], different representations of 3D data have been proposed to learn local descriptors, which need point cloud patches around keypoints as input and output descriptors corresponding to these keypoints. However, the features generated by these methods have low resolution. In [4], a fully convolutional network [21] based on convolutional neural network in [22] is implemented to obtain dense feature description corresponding to all points in a point cloud. But it cannot detect keypoints.

There are also some works of joint learning keypoints and descriptors. D3Feat [3] is the most representative one, which uses a full convolutional network by KPConv [9] and gives a density-invariant keypoint selection strategy and detection loss function. It achieves state-of-the-art performance.

Different from all the above feature extraction methods, we aim to learn the descriptors considering information on both point clouds in the point cloud pair and enhance the discriminability of the learned descriptors when matching these two point clouds. We construct CAFeat3D by KPConv [9], which is also used in D3Feat [3]. The number of parameters in CAFeat3D is almost the same as that in D3Feat [3]. Our algorithm realizes better feature matching performance than the above algorithms between a point cloud pair.

3. METHODOLOGY

To learn the descriptors considering information on both point clouds and realize robust feature matching between a point cloud pair, we design a deep neural network (CAFeat3D), which is shown in Fig.1. The input pair of point clouds, P_1 and P_2, are passed through two weight sharing encoder modules to obtain self-features of P_1 and P_2. Our encoder is a 5-layer convolutional network which uses KPConv [9]. Then the self-features of P_1 and P_2 from the same layer (denoted as SF_1 and SF_2) in encoder are passed through our proposed cross-attention module to obtain conditioned features. After this, the SF_1 and its corresponding conditioned features are concatenated with the fused features generated by corresponding decoder layer. The features after concatenating are passed from the next layer in decoder to obtain new fused features. The output of decoder module is the final fused features of P_1. The final fused features of P_2 can be obtained by swapping P_1 and P_2. In the following, we describe how to generate self-features in Section 3.1, and the construction of the proposed cross-attention module in Section 3.2. How to obtain the fused features is detailed in Section 3.3. Finally, how our whole network is trained is described in Section 3.4.

3.1. Self-features

Given two point clouds of same scene, $P_1 \in \mathbb{R}^{N_1 \times 3}$ and $P_2 \in \mathbb{R}^{N_2 \times 3}$, we obtain multiple self-features of P_1 and P_2 generated by each layers in encoder, which are at different point cloud density.

Our encoder is a 5-layer convolutional network. Each layer contains three blocks and the first one is strided excepts

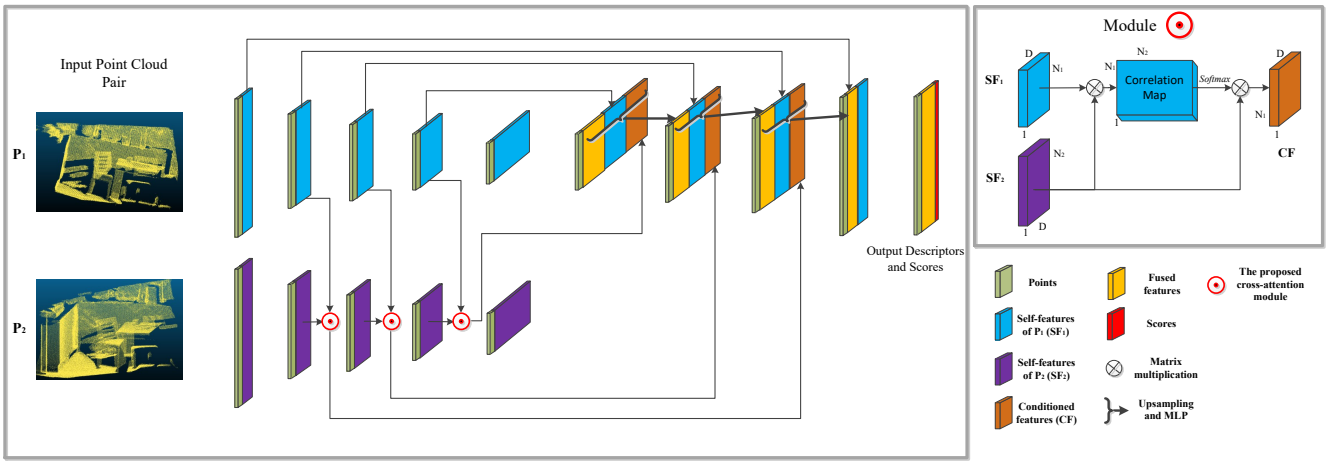

Fig. 1. The network architecture of CAFeat3D which can obtain final fused features from a input point cloud pair (Left). The input pair of point clouds, P_1 and P_2, are passed through two weight sharing encoder modules to obtain self-features of P_1 and P_2, which are shown in blue and purple. Our encoder is a 5-layer convolutional network and each block is a ResNet block using KPConv [9]. Then the self-features of P_1 and P_2 from the same layer (denoted as SF_1 and SF_2) in encoder are passed through the proposed cross-attention module (Upper right) to obtain conditioned features. After this, the SF_1 and its corresponding conditioned features are concatenated with the fused features generated by corresponding decoder layer. These are passed from the next layer in decoder to obtain new fused features. Each layer in decoder contains a nearest upsampleblock and a MLP. The feature blocks corresponding to different colors and main symbols in the Fig.1 are marked in the lower right corner. **Best viewed in color with 200% zoom in.**

for the first layer like KP-CNN [9]. Each block in encoder module is a Resnet block using KPConv [9], which defines 3D convolution on point clouds by utilizing kernel points that carry weights to simulate the kernel pixels in 2D convolution. Each block is followed by BN (batch normalization) and ReLU. KPConv used in [3] is briefly described as below.

Given a point cloud $P \in \mathbb{R}^{N\times 3}$ and a set of features $F \in \mathbb{R}^{N\times D}$ which is corresponding to P, we denote the i-th point in P and its feature in F as x_i and f_i respectively. The convolution by kernel g at point x is defined as

$$(F * g) = \frac{1}{|N_x|} \sum_{x_i \in N_x} (\sum_{k=1}^{K} h(x_i - x, \hat{x}_k) W_k) f_i, \qquad (1)$$

where N_x is the radius neighborhood of x, x_i is the supporting point in N_x, h is the function related to the kernel point $\hat{x}_k$ and the supporting point x_i, K is equal to the number of kernel points and W_k is the weight matrix of $\hat{x}_k$. We refer reader to [9, 3] for more information.

The self-features of P_1 will be injected into the decoder. The self-features of P_1 and P_2 generated from the same layer are injected into our proposed cross-attention module, which is detailed in the next section.

3.2. Cross-attention Module

We hope to fuse features from both point clouds and obtain conditioned features in order to enhance the discriminability of the learned descriptors under the circumstance of matching these two point clouds. However, due to the changes of viewpoint, density and the disorder of point clouds, given a point and its feature in one point cloud, the corresponding feature in another point cloud is usually not at the same location as the given point.

Based on this, given a set of self-features, h, and a feature at location i of h, h_i, we propose an attenion mechanism to dig out the corresponding feature of h_i, $\hat{h}_i$, from another set of self-features, t. For each position i in h, a feature $\hat{h}_i$ is obtained by a weighted sum over all features in t:

$$\hat{h}_i = \sum_j A_{ij} * t_j, \qquad (2)$$

in which A is calculated by

$$A_{ij} = \frac{\exp(\gamma h_i^T t_j)}{\sum_k \exp(\gamma h_i^T t_k)}, \qquad (3)$$

and γ is a constant value.

For self-features $F_1 \in \mathbb{R}^{N_1\times D}$ of $P_1 \in \mathbb{R}^{N_1\times 3}$ and self-features $F_2 \in \mathbb{R}^{N_2\times D}$ of $P_2 \in \mathbb{R}^{N_2\times 3}$, we apply this attention mechanism to obtain conditioned features of P_1, as shown in the upper right of Fig.1. The detailed steps are as follows. First, F_1 and F_2 are L2-normalized to unit length and denoted as $\bar{F}_1$ and $\bar{F}_2$ respectively. Second we get A by

$$A = softmax(\gamma * \bar{F}_1 \times \bar{F}_2^T), \qquad (4)$$

in which γ can adjust the weight of different features in F_2. The larger γ represents the more similar features have higher weight. A is a correlation map [23] processed by a softmax operation. Finally, we obtain conditioned features of P_1 through F_2 and A:

$$F_1^a = A \times \bar{F}_2. \tag{5}$$

Given the self-features of input point cloud P_1 and P_2, features from the same layer in the two weight-sharing encoders are put into the cross-attention module to obtain conditioned features corresponding to different density for P_1.

3.3. Fused Features

Given the self-features and conditioned features, fused features are generated from decoder module. For input pair of point cloud, $P_1 \in \mathbb{R}^{N_1 \times 3}$ and $P_2 \in \mathbb{R}^{N_2 \times 3}$, the decoder module is used to get the final point-wise features for the first input point cloud P_1 and the output of our decoder is a dense feature map $F_{p1} \in \mathbb{F}^{N_1 \times d}$, where d is the dimension of feature vector. The feature upsampling strategy in decoder is nearest upsampling. Skip links are implemented between intermediate layers of above encoder and the decoder and between cross-attention module and intermediate layers of decoder. Each layer of the decoder has three input sources: self-features, conditioned features and fused features from the previous layer. In each layer, there is a feature fusion module consisting of an upsampling operation, MLP, BN and Leaky ReLU. Those features are concatenated and put into the feature fusion module. Then upsampling of point cloud features is realized and new fused features are obtained.

3.4. Training and Loss Functions

Our network jointly learns descriptors and keypoints, so our loss function consists of two parts: descriptor loss item and keypoint detector loss item. We first describe the descriptor loss item and then the keypoint detector loss item.

Descriptor loss We adapt the circle loss function [10] to train the parameters of our network.

Given a pair of point clouds P_1 and P_2, and a set of pairs of corresponding 3D points. Suppose (A^i, B^i) is a correspondence pair with their corresponding normalized descriptors pairs (d_{A^i}, d_{B^i}). The distance between a positive pair is:

$$d_p(i) = \left\| d_{A^i} - d_{B^i} \right\|_2. \tag{6}$$

The distance between a negative pair is:

$$d_n(i,j) = \left\| d_{A^i} - d_{B^j} \right\|_2 s.t. (i \neq j). \tag{7}$$

The circle loss item is defined as follows:

$$L_{desc} = \log\left[1 + \sum_i \sum_{j, j \neq i} \exp(\lambda \alpha_n^{i,j}(M_n - d_n(i,j))) \times \sum_i \exp(\lambda \alpha_p^i (d_p(i) - M_p))\right], \tag{8}$$

in which $\alpha_n^{i,j}$ and α_p^i are non-negative weighting factors and can be calculated by $d_n^{i,j}$ and d_p^i [10]. In Eq.(8), λ is a scale factor, M_n is the margin for negative pairs and M_p is the margin for positive pairs.

Keypoint detector loss We adopt the detector loss in [3] to raise keypoint detection scores of the points which are easy to be correctly matched. The detector loss item is defined as

$$L_{det} = \frac{1}{n} \sum_i \left[(d_p(i) - min(d_n(i,j)))(s_{A^i} + s_{B^i}) \right], \tag{9}$$

in which s_{A^i} and s_{B^i} are the keypoint detection scores of point A^i and B^i. The keypoint detection scores of point p can evaluate how salient a point is when it is compared with its neighborhood and its own distinction. The point with high keypoint detection score is suitable as keypoint.

The loss function of our network is constructed as follows:

$$L_{total} = L_{desc} + \beta L_{det}, \tag{10}$$

in which β is a constant value and can adjust the component of L_{det}.

4. EXPERIMENTS

In this section, we firstly introduce the implementation details in Section 4.1. Secondly, we evaluate the feature matching performance of our model in Section 4.2. Thirdly, the point cloud registration evaluation using feature correspondence is described in Section 4.3.

4.1. Implementation Details

CAFeat3D is implemented in Pytorch and two TITAN RTX GPU cards with 24G memory. We evaluate CAFeat3D on 3DMatch dataset [6]. We follow the same protocols [6] in 3DMatch dataset to obtain the training and test data. The test data includes 8 scenes and about 2000 point cloud pairs. We use all the point cloud pairs which are more than 30% overlap when training. Grid subsampling is used to control the density of point cloud. During training, we apply data augmentation same as D3Feat [3] to improve the robustness of our network and use a batch size of 1. For Eq.(4), the γ is set as 80. We only put the features which output from the last four layers of the encoder into the cross-attention module to obtain the conditioned features for saving video memory. The positive margin M_p, negative margin M_n, scale factor λ in circle loss are set as 0.1, 1.4 and 10 respectively. The β in Eq.(10) is set as 1. We optimize the network using SGD optimizer with learning rate of 0.01 and the momentum is set to 0.98.

4.2. Feature Matching Evaluation

Following FCGF [4], we use *Feature Matching Recall* proposed in [20], the percentage of registration whose inlier ratio

Table 1. The comparison of *Feature Matching Recall* on the 3DMatch dataset.

Methods	Feature Matching Recall (%)
Shot [24]	23.8
FPFH [19]	35.9
3DMatch [6]	59.6
PPFNet [20]	62.3
PPF-FoldNet [25]	71.8
PerfectMatch [5]	94.7
FCGF [4]	95.2
D3Feat [3]	95.8
Ours	**96.1**

is larger than a threshold $\mu_1 = 5\%$, to evaluate the performance of feature matching. In the following experiments, a match is regarded as an inlier if the distance between the points is smaller than $\mu_2 = 0.1m$ under ground truth transformation.

We compare our algorithm with state-of-the-arts on 3DMatch dataset. We report the *Feature Matching Recall* in Table 1. The number of keypoints is set to 5000. From Table 1, we can see our algorithm has the highest *Feature Matching Recall*. This benefits from that our method takes into account the information between point clouds and obtains descriptors that achieve higher precision matching.

Moreover, we demonstrate the robustness of our algorithm by varying inlier ratio threshold (μ_1), which is originally defined as $\mu_1 = 5\%$ in *Feature Matching Recall*. The experimental results are shown in Fig.2, from which we can see our algorithm performs well under different μ_1. In more strict inlier ratio threshold from 15% to 20%, our algorithm is significantly better than other methods, which indicates the discriminability of the features generated by our algorithm between two point clouds to be matched is enhanced.

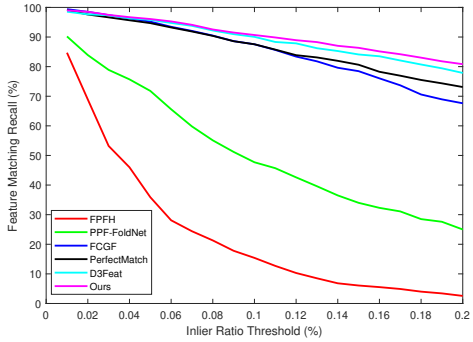

Fig. 2. *Feature Matching Recall* in relationship to inlier ratio threshold of different algorithms. **Best viewed in color with 200% zoom in.**

Table 2. *Feature Matching Recall* on the 3DMatch dataset under different numbers of features. (%)

Features number	5000	2500	1000	500	250
PerfectMatch [5]	94.7	94.2	92.6	90.1	82.9
FCGF [4]	95.2	95.5	94.6	93.0	89.9
D3Feat [3]	95.8	95.6	94.6	94.3	93.3
Ours	**96.1**	**96.0**	**95.7**	**94.7**	**93.8**

Table 3. *Registration Recall* on the 3DMatch dataset under different numbers of features. (%)

Features number	250	500	1000
PerfectMatch [5]	50.9	64.8	73.4
FCGF [4]	73.0	81.0	**85.8**
D3Feat [3]	79.3	82.5	84.9
Ours	**82.5**	**83.8**	85.7

Since the number of feature points is very important for feature matching task, we further report the results when reducing the feature points number in one point cloud from 5000 to 2500, 1000, 500 and 250 and the experimental results are shown in Table 2. Due to lack of keypoint detection, the performances of FCGF and PerfectMatch drop rapidly with the decrease of the number of feature points. The performance of D3Feat and ours are less affected by the decrease of the number of feature points for effective keypoints detection mechanism. Compared with D3Feat, our method always has better performance under different numbers of keypoints with almost the same number of parameters and 3D convolution operation, which further demonstrates the superiority of our algorithm and the effectiveness of our cross-attention module.

4.3. Using Feature Correspondence for Point Cloud Registration

Following 3DMatch [6], we use *Registration Recall* to measure the quality of features within point cloud registration system. *Registration Recall* is equal to the percentage of correctly registered point cloud pairs [6]. More specifically, a registration is correct if the RMSE of the ground truth correspondences under the calculated transformation matrix is lower than a threshold τ. In our implementation, we use RANSAC based on feature matching to estimate the transformation matrix between two point clouds and τ is set as $0.2m$. The maximum number of iterations for random sample selection is 5000.

We evaluate *Registration Recall* of PerfectMatch [5], FCGF [4], D3Feat [3] and ours under different numbers, 250, 500 and 1000, of features. The experimental results are shown

in Table 3, from which we can see our algorithm achieves the best performance or close to the best performance under different numbers of features. It can be seen from the second column of Table 3 that our method can even has a good point cloud registration effect when the number of feature points is small, which shows enough accurate correspondences can be constructed to realize the successful registration between a pairs of point clouds. To some extent, this also shows our extracted features have strong discrimination and sufficient invariance between point cloud pairs.

5. CONCLUSION

In this paper, to enhance the discriminability of the learned features under the circumstance of matching two point clouds, we propose a novel network to learn the descriptors considering information on both point clouds. Specifically, we first develop a U-Net like network for single point cloud feature extraction as our fundamental network. Then an identical encoder module and an cross-attention module proposed by us are embedded in the U-Net like network. The cross-attention module can effectively provide the information of another point cloud for the original U-Net like network. Extensive experimental results on 3DMatch large-scale dataset demonstrate the effectiveness of the proposed network on feature matching and point cloud registration.

6. REFERENCES

[1] Wolfgang Hess, Damon Kohler, Holger Rapp, and Daniel Andor, "Real-time loop closure in 2d lidar slam," in *ICRA*. IEEE, 2016, pp. 1271–1278.

[2] Biao Li, Qixing Xie, Shaoyi Du, Wenting Cui, Runzhao Yao, Yue Gao, and Nanning Zheng, "Tag-reg: Iterative accurate global registration algorithm," in *ICME*. IEEE, 2021, pp. 1–6.

[3] Xuyang Bai, Zixin Luo, Lei Zhou, Hongbo Fu, Long Quan, and Chiew-Lan Tai, "D3feat: Joint learning of dense detection and description of 3d local features," in *CVPR*, 2020, pp. 6359–6367.

[4] Christopher Choy, Jaesik Park, and Vladlen Koltun, "Fully convolutional geometric features," in *ICCV*, 2019, pp. 8958–8966.

[5] Zan Gojcic, Caifa Zhou, Jan D Wegner, and Andreas Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *CVPR*, 2019, pp. 5545–5554.

[6] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser, "3dmatch: Learning local geometric descriptors from rgb-d reconstructions," in *CVPR*, 2017, pp. 1802–1811.

[7] Lei Zhou, Siyu Zhu, Zixin Luo, Tianwei Shen, Runze Zhang, Mingmin Zhen, Tian Fang, and Long Quan, "Learning and matching multi-view descriptors for registration of point clouds," in *ECCV*, 2018, pp. 505–522.

[8] Olivia Wiles, Sebastien Ehrhardt, and Andrew Zisserman, "Co-attention for conditioned image matching," in *CVPR*, 2021, pp. 15920–15929.

[9] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas, "Kpconv: Flexible and deformable convolution for point clouds," in *ICCV*, 2019, pp. 6411–6420.

[10] Yifan Sun, Changmao Cheng, Yuhan Zhang, Chi Zhang, Liang Zheng, Zhongdao Wang, and Yichen Wei, "Circle loss: A unified perspective of pair similarity optimization," in *CVPR*, 2020, pp. 6398–6407.

[11] François Pomerleau, Francis Colas, and Roland Siegwart, "A review of point cloud registration algorithms for mobile robotics," *Foundations and Trends in Robotics*, vol. 4, no. 1, pp. 1–104, 2015.

[12] Paul J Besl and Neil D McKay, "Method for registration of 3-d shapes," in *Sensor fusion IV: control paradigms and data structures*. International Society for Optics and Photonics, 1992, vol. 1611, pp. 586–606.

[13] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey, "Pointnetlk: Robust & efficient point cloud registration using pointnet," in *CVPR*, 2019, pp. 7163–7172.

[14] Runnan Chen, Penghao Zhou, Wenzhe Wang, Nenglun Chen, Pai Peng, Xing Sun, and Wenping Wang, "Pr-net: Preference reasoning for personalized video highlight detection," in *ICCV*, 2021, pp. 7980–7989.

[15] Christopher Choy, Wei Dong, and Vladlen Koltun, "Deep global registration," in *CVPR*, 2020, pp. 2514–2523.

[16] Ivan Sipiran and Benjamin Bustos, "Harris 3d: a robust extension of the harris operator for interest point detection on 3d meshes," *The Visual Computer*, vol. 27, no. 11, pp. 963, 2011.

[17] Yu Zhong, "Intrinsic shape signatures: A shape descriptor for 3d object recognition," in *ICCV Workshops*. IEEE, 2009, pp. 689–696.

[18] Jiaxin Li and Gim Hee Lee, "Usip: Unsupervised stable interest point detection from 3d point clouds," in *ICCV*, 2019, pp. 361–370.

[19] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *ICRA*. IEEE, 2009, pp. 3212–3217.

[20] Haowen Deng, Tolga Birdal, and Slobodan Ilic, "Ppfnet: Global context aware local features for robust 3d point matching," in *CVPR*, 2018, pp. 195–205.

[21] Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic segmentation," in *CVPR*, 2015, pp. 3431–3440.

[22] Christopher Choy, JunYoung Gwak, and Silvio Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *CVPR*, 2019, pp. 3075–3084.

[23] Ignacio Rocco, Relja Arandjelovic, and Josef Sivic, "Convolutional neural network architecture for geometric matching," in *CVPR*, 2017, pp. 6148–6157.

[24] Federico Tombari, Samuele Salti, and Luigi Di Stefano, "Unique signatures of histograms for local surface description," in *ECCV*. Springer, 2010, pp. 356–369.

[25] Haowen Deng, Tolga Birdal, and Slobodan Ilic, "Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors," in *ECCV*, 2018, pp. 602–618.