KPSNET: KEYPOINT DETECTION AND FEATURE EXTRACTION FOR POINT CLOUD REGISTRATION

Anan Du, Xiaoshui Huang, Jian Zhang, Lingxiang Yao, Qiang Wu

University of Technology Sydney
School of Electrical and Data Engineering

ABSTRACT

This paper presents the KPSNet, a KeyPoint Siamese Network to simultaneously learn task-desirable keypoint detector and feature extractor. The keypoint detector is optimized to predict a score vector, which signifies the probability of each candidate being a keypoint. The feature extractor is optimized to learn robust features of keypoints by exploiting the correspondence between the keypoints generated from two inputs, respectively. For training, the KPSNet does not require to manually annotate keypoints and local patches pairwise. Instead, we design an alignment module to establish the correspondence between the two inputs and generate positive and negative samples on-the-fly. Therefore, our method can be easily extended to new scenes. We test the proposed method on the open-source benchmark and experiments show the validity of our method.

***Index Terms*—** 3D keypoints, point cloud registration, deep learning

1. INTRODUCTION

Keypoint detection plays a vital role in many computer vision tasks, such as image alignment, image stitching, 3D reconstruction, object recognition, indexing and retrieval. For 2D keypoint detection, the goal is to find same local features across images taken from different views or time. Many successful approaches have been proposed, such as SIFT and SURF. Similarly, 3D keypoint detection pursues a repeatable and distinctive 3D local representation from 3D data, e.g., point cloud and 3D mesh, which can be used to establish correspondences between 3D surfaces. Although more and more researchers work on 3D keypoint detection in recent years, it's still an open research topic. One of the biggest challenges is that it's difficult to define a keypoint in a 3D point cloud, as a point cloud consists of a set of discrete points with nonuniform densities. In addition, it's impossible for people to manually label the position of a keypoint in a 3D point cloud. Lack of annotated dataset limits researchers leveraging powerful supervision learning methods on 3D keypoint detection.

Lots of relevant research focuses on 3D local descriptors but does not address the detection of keypoint positions. So far, many 3D descriptors have been proposed [1]. [2, 3, 4] use histograms to estimate the similarity of keypoints by counting the number of points in each spatial bin or considering surface normal property. These methods are designed for specific applications and it's hard to extend to new scenes. [5, 6] learns local 3D features by using a siamese 3D convolutional neural network and has made significant progress. They collected pairs of matching and non-matching local 3D patches as training samples, and trained a 3D ConvNet-based descriptor.

For keypoint detector, it's common to use a hand-crafted saliency function by combining the domain knowledge, or uniformly select some local patches and taking their centres as keypoint. Differently, [7] proposed a descriptor-specific keypoint detector, which casts 3D keypoint detection as a classification problem in terms of whether the points can be matches by a pre-defined 3D descriptor. The performance of this method depends on the pre-defined 3D descriptor. However, learning a good 3D descriptor needs a large amount of annotated keypoints, and thus causes a chicken-egg problem.

In this paper, we explore whether one can simultaneously learn 3D keypoint detector and feature extractor in a unified framework. Inspired by [8], which learns an end-to-end framework for keypoint detection and its representation using depth images, we present a KeyPoint Siamese Network based framework (KPSNet) to simultaneously detect 3D keypoints and learn their feature representations directly from 3D point clouds. The KPSNet receives as input pairs of point clouds and their relative transformation matrix. Each branch of the KPSNet contains a keypoint detection sub-network to predict keypoints and a feature extraction sub-network to learn 3D features. These two branches are the same. We design an alignment module to establish correspondence between the two branches in order to jointly optimize the whole network. In other words, the alignment module generates pairs of matching and non-matching samples and labels them on-the-fly. Then we train the network to minimize the distance in feature space between matching pairs and maximize the distance between non-matching pairs. The key contributions of this work can be concluded as follows:

- We propose a unified framework to simultaneously learn a keypoint detector and a feature extractor for 3D

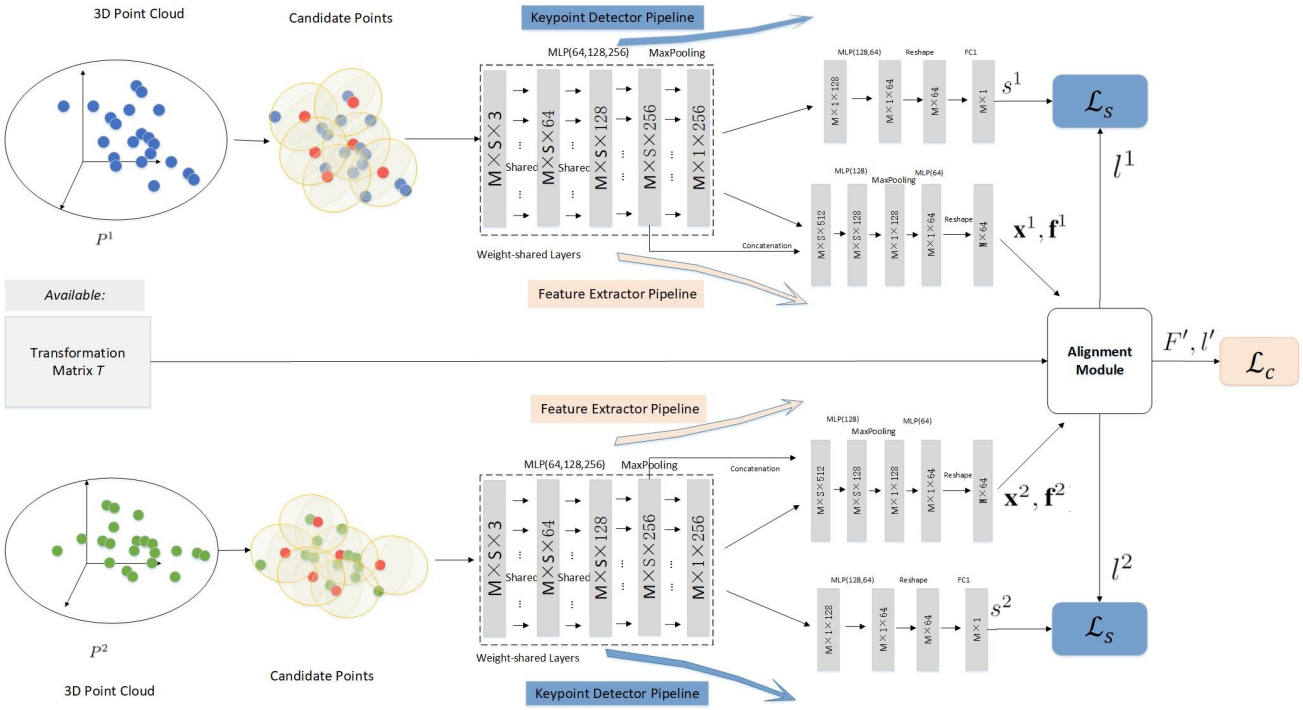

Fig. 1. Overview of our KPSNet. The KPSNet takes two point clouds and their relative transformation matrix as input and generate some candidates from each input point cloud. These candidates, along with their support regions, feed into a MLP module (MLP(64,128,256) denotes there are three convolution layers in this MLP module and the number of filters are 64,128,256 separately), followed by a max pooling layer. The keypoint detector branch generates predictions of each candidates, while the keypoint feature extractor branch generates features of each candidate. For each input point cloud, we pass the predictions and labels generated from the alignment module to the score loss. The features from the both two input point clouds are organized into pairs to the contrastive loss. Note that the weights between the two branches of the Siamese network are shared. For details on the notations please see section 2.2.

point cloud registration using point cloud data,

- We design an alignment module to establish the correspondence between the two inputs and generate pairs of positive and negative samples during training process. This avoids separately annotating keypoints and their matching relationships in 3D point clouds, which is time consuming and labor-intensive.

2. METHOD

2.1. Architecture

Fig.1 demonstrates the architecture of our proposed KPSNet. Each branch takes an entire point cloud as input and contains a keypoint detector and a feature extractor. It firstly generates some point clusters from the point cloud similar to [9, 10]. We consider the centroids of these point clusters as candidates of the keypoints and define the corresponding point clusters as their feature support regions. These candidates, along with their feature support regions, are fed into the keypoint detector and the feature extractor respectively. The keypoint detector contains two multi-layer perception (MLP) modules [11] and a max pooling layer. The output is a vector which signifies the probability of each candidate being a keypoint. The feature extractor contains three MLP modules with skip link feature concatenation. It outputs feature representations of the candidates. The keypoint detector and feature extractor share the first MLP module.

These two branches of our KPSNet share all structures and weights and linked by an alignment module. The alignment module establishes correspondence between the two sets of candidates using the known transformation matrix, thereby generating positive and negative pairs of samples. These pairs are passed to a contrastive loss to minimize feature distance between positive pairs and maximize the distance between negative pairs. In addition, the alignment module also generates a label for each candidate. The score loss use these labels to lead the keypoint detector to find as more keypoints as possible. Apparently, our proposed method does not require to separately annotate keypoints or local patches pairwise, which is cost-saving and easily extended to new scenes.

2.2. Training

The proposed KPSNet takes pairs of point clouds $\{P^1, P^2\}$ and their relative transformation matrix T as input. Each pair of point clouds must have some overlap. From each point cloud, the KPSNet generates a set of candidates, $K^m = \{(\mathbf{x}_n^m, s_n^m, \mathbf{f}_n^m)|n = 1, ..., N\}$, where $m \in \{1, 2\}$ corresponds to the pair of point clouds, N is the number of candidates, $\mathbf{x}_n^m = (x_n, y_n, z_n)$ are the 3D coordinates of the points, s_n^m is the output of the keypoint detector which signifies the probability of the candidate being a keypoint, and $\mathbf{f}_n^m$ is the corresponding feature vector.

Alignment Module Considering the point cloud registration task, we desire that a keypoint should be repeatable and distinctive. To be specific, if a keypoint is detected somewhere in one point cloud, then we desire that it can always be detected no matter what transformations apply to the coordinate system of the point cloud. In addition, for two different point clouds with different densities and coordinate systems, keypoints are expected to found in their overlapping region as more as possible. Under this scenario, we design the alignment module to establish the correspondences between the two sets of candidates and generate labels on-the-fly for training the whole model.

The alignment module receives the coordinates and feature vectors of the two sets of candidates,$\{\mathbf{x}^1, \mathbf{f}^1\}$ and $\{\mathbf{x}^2, \mathbf{f}^2\}$. It firstly transforms $\mathbf{x}^1$ and $\mathbf{x}^2$ to the same coordinate system using the given transformation matrix T and calculates the Euclidean distance between any two points from $\mathbf{x}^1, \mathbf{x}^2$ respectively. Then for each candidate $\mathbf{x}_i^1 (i = 1, ..., n)$, we find the closest candidate $\mathbf{x}_j^2 (j = 1, ..., n)$ based on the Euclidean distance and form the n^{th} pair of features $F'_n = (\mathbf{f}_i^1, \mathbf{f}_j^2)$. If the distance is less than a small threshold τ, $d(\mathbf{x}_i^1, \mathbf{x}_j^2) < \tau$, we consider F'_n as a positive pair, denote as $l'_n = 1$. At the same time, we take $\mathbf{x}_i^1$ and $\mathbf{x}_j^2$ as the positive samples of P^1 and P^2 respectively for training the keypoint detection sub-network and denoted as $l_i^1 = l_j^2 = 1$. Otherwise, we label F'_n, $\mathbf{x}_i^1$ and $\mathbf{x}_j^2$ as negative, denote as $l'_n = l_i^1 = l_j^2 = 0$. In this way, we obtain the required labels for training the keypoint detector and the feature extractor.

Joint Optimization We are interested in simultaneously learning a keypoint detector and a feature extractor for point cloud registration. As mentioned above, the desirable keypoints should have properties like repeatability, view-invariant and distinctive. Towards this end, we introduce the following multitask loss similar to [8]:

$$\begin{aligned}\mathcal{L}\left(\{K^1, K^2\}\right) =& \alpha \mathcal{L}_c\left(F', l'\right) + \\ & \beta \mathcal{L}_s^1\left(s^1, l^1\right) + \beta \mathcal{L}_s^2\left(s^2, l^2\right)\end{aligned} \tag{1}$$

where $F' = \{F'_n|n = 1, ..., N\}$ is the set of feature pairs generate by the alignment module, $l' = \{l'_n|n = 1, ..., N\}$ is the set of labels of F', $s^1 = \{s_n^1|n = 1, ..., N\}$, $s^2 = \{s_n^2|n = 1, ..., N\}$ are outputs of the keypoint detector and l^1, l^2 are labels of s^1, s^2, respectively. α, β are weighted factors. L_c is a modified contrastive loss that optimizes over the feature representation of pairs of the keypoints. Its basic principle is minimizing the feature distance of positive pairs and maximizing the feature distance of negative pairs. The contrastive loss is defined as:

$$\begin{aligned}\mathcal{L}_c\left(F', l'\right) = & \frac{\sum_{n=1}^{N} l'_n \left\|\mathbf{f}_n^1 - \mathbf{f}_n^2\right\|^2}{2N_{pos}} + \\ & \frac{\sum_{n=1}^{N} \left(1 - l'_n\right) \max\left(0, \delta - \left\|\mathbf{f}_n^1 - \mathbf{f}_n^2\right\|\right)^2}{2N_{neg}}\end{aligned} \tag{2}$$

where δ is the margin, N_{pos}, N_{neg} are number of positive and negative pairs respectively and $N = N_pos + N_neg$. Here, we conveniently let $\mathbf{f}_n^1, \mathbf{f}_n^2$ as the n^{th} feature pair. Considering the imbalance between the class sizes of positive and negative pairs, we normalize the contribution of each class to the loss by their number proportion.

$\mathcal{L}_s^m$ is the score loss of the prediction from the keypoint detection sub-network. As it's hard to define how a keypoint should be like, in this paper, we take task-desirable keypoints as the objective. In other words, we train the keypoint detection sub-network to select points for which we can find positive correspondence between two point clouds taken from different positions. So we only consider the positive points and and want to detect as many as positive keypoints as we can. The score loss is defined as:

$$\mathcal{L}_s^m\left(s^m, l^m\right) = -\frac{\gamma \sum_{n=1}^{N} l_n^m \log s_n^m + 1}{N_{pos} + 1} \tag{3}$$

where $m \in \{1, 2\}$ corresponds to the pair of input point clouds, γ is a regularization parameter.

3. EXPERIMENTS

3.1. Dataset and Setup

We use several open-source RGB-D reconstruction datasets provided by [5] for training and use the geometric registration benchmark for evaluation. These datasets consist of SUN3D[12, 13], 7-Scenes[14], RGB-D Scenes v2[15], BundleFusion[16] and Analysis by Synthesis[17]. We split these datasets into training and testing as the same as [5]. In order to train our model, we firstly create 3D point clouds from the training split of the RGB-D reconstruction datasets using the 3dmatch-toolbox. Then we prepare our training data by creating pairs of samples following the method of 3DMath geometric registration benchmark, along with their transformation matrixes. In total, we collect 22.8K pairs of training samples.

In our experiment, the threshold τ used in the alignment module is set to 0.05m. The support region of the candidate is set to a ball with radius 0.2m. The weighted factors α is set

Fig. 2. Visualization of estimated transformations. The first two columns show two diffident point clouds to be registered. The last columns show the results after registration.

Table 1. Our evaluations on the 3D-match benchmark

	Spin Images		SHOT		FPFH		USC		KPSNet(ours)	
	recall	prec.	recall	prec.	recall	prec.	recall	prec.	recall	prec.
Red Kitchen	0.27	0.49	0.21	0.44	0.36	0.52	0.52	0.60	0.58	**0.72**
Home 1	0.56	0.14	0.37	0.13	0.56	0.16	0.35	0.16	0.54	**0.23**
Home 2	0.35	0.10	0.30	0.11	0.43	0.13	0.47	**0.24**	0.40	0.19
Hotel 1	0.37	0.29	0.28	0.29	0.29	0.36	0.53	0.46	0.50	**0.53**
Hotel 2	0.33	0.12	0.24	0.11	0.36	0.14	0.20	0.17	0.40	**0.23**
Hotel 3	0.32	0.16	0.42	0.12	0.61	**0.21**	0.38	0.14	0.38	0.17
Study Room	0.21	0.07	0.14	0.07	0.31	0.11	0.46	**0.17**	0.52	0.15
MIT Lab	0.29	0.06	0.22	0.09	0.31	0.09	0.49	**0.19**	0.36	0.13
Average	0.34	0.18	0.27	0.17	0.40	0.21	0.43	0.27	0.46	**0.30**

to 0.5 and β is set to 0.2. The margin δ and the regularization parameter γ are both set to 0.5.

In order to validate our method, we test our KPSNet on the geometric registration benchmark. We predict the coordinates and feature representations of keypoints from each testing point cloud using our model. Then we perform nearest neighbor matching on them and use RANSAC on these nearest neighbor matches to estimate a rigid transformation between every two point clouds using 3dmatch-toolbox. The number of RANSAC iterations is limited to 5,000. No subsequent refinement, e.g. using ICP is performed.

3.2. Results

Fig.2 shows the visualization for our proposed KPSNet under several challenging scenarios with less variation in geometry. It can be intuitively seen that the main parts of each 3D point cloud pair have been perfectly matched.

We also evaluate our method against the baselines of Spin Images[2], SHOT[18], FPFH[3], USC[4]. Experiment results are shown in Table.1. It can be concluded that our proposed KPSNet outperforms all the hand-crafted methods in average recall and averge precision. Generally, the precision increases with the number of keypoints. Even though we only use 1024 keypoints and 5000 RANSAC iterations in our evaluation, our method still obtains a competitive performance. It is for sure that with more keypoints and more iterations, our proposed KPSNet can achieve a more remarkable performance

4. CONCLUSIONS

In this paper, we presented KPSNet, a unified framework to simultaneously learn a task-desirable keypoint detector and feature extractor for point cloud registration. We designed a novel strategy to generate required labels during training by using indirect ground truth of point cloud registration task, which avoid the costly manual keypoint annotation. To train our proposed network, we introduced a multitask losss function and jointly optimized the keypoint detector and feature extractor. Experiment results on public datasets show that our approach is rather competitive than other 3D keypoint feature extraction methods for 3D point cloud registration task. Next step, we further improve the novel methodology by adding more context information and use it into more outdoor scenes.

5. REFERENCES

[1] Gil Elbaz, Tamar Avraham, and Anath Fischer, "3d point cloud registration for localization using a deep neural network auto-encoder," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4631–4640.

[2] Andrew E Johnson and Martial Hebert, "Using spin images for efficient object recognition in cluttered 3d scenes," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, , no. 5, pp. 433–449, 1999.

[3] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*. Citeseer, 2009, pp. 3212–3217.

[4] Federico Tombari, Samuele Salti, and Luigi Di Stefano, "Unique shape context for 3d data description," in *Proceedings of the ACM workshop on 3D object retrieval*. ACM, 2010, pp. 57–62.

[5] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser, "3DMatch: Learning local geometric descriptors from RGB-D reconstructions," *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 199–208, 2017.

[6] Xuelun Shen, Cheng Wang, Chenglu Wen, Weiquan Liu, Xiaotian Sun, and Jonathan Li, "Discriminative learning of point cloud feature descriptors based on siamese network," in *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2018, pp. 4519–4522.

[7] Alessio Tonioni, Samuele Salti, Federico Tombari, Luigi Di Stefano, and Riccardo Spezialetti, "Learning to Detect Good 3D Keypoints," *International Journal of Computer Vision*, vol. 126, no. 1, pp. 1–20, 2018.

[8] Georgios Georgakis, Srikrishna Karanam, Ziyan Wu, Jan Ernst, Jana Kosecka, and Siemens Corporate Technology, "End-to-end learning of keypoint detector and descriptor for pose invariant 3D matching," in *Cvpr*, 2018, pp. 1965–1973.

[9] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems*, 2017, pp. 5099–5108.

[10] Zi Jian Yew and Gim Hee Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 607–623.

[11] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, vol. 1, no. 2, pp. 4, 2017.

[12] Jianxiong Xiao, Andrew Owens, and Antonio Torralba, "Sun3d: A database of big spaces reconstructed using sfm and object labels," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1625–1632.

[13] Maciej Halber and Thomas Funkhouser, "Fine-to-coarse global registration of rgb-d scans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, vol. 3.

[14] Samuele Salti, Federico Tombari, and Luigi Di Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, pp. 251–264, 2014.

[15] Kevin Lai, Liefeng Bo, and Dieter Fox, "Unsupervised feature learning for 3d scene labeling," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 3050–3057.

[16] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt, "Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 76a, 2017.

[17] Julien Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin, "Learning to navigate the energy landscape," in *3D Vision (3DV), 2016 Fourth International Conference on*. IEEE, 2016, pp. 323–332.

[18] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon, "Scene coordinate regression forests for camera relocalization in rgb-d images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2930–2937.