INDOOR SCENE REGISTRATION BASED ON SIAMESE NETWORK AND POINTNET

Z. Zhang[1], C. Wen[1], Y. Chen[1,*], W. Li[1], C. You[1], C. Wang[1], J. Li[1,2]

[1] Fujian Key Laboratory of Sensing and Computing for Smart Cities, Xiamen University, Xiamen,China
-(clwen,chenyiping,cwang)@xmu.edu.cn, (zhengzhang,ycbhh)@stu.xmu.edu.cn, weili_xs@163.com
[2] Department of Geography and Environmental Management, University of Waterloo, Waterloo, Canada - junli@xmu.edu.cn

KEY WORDS: Mobile LiDAR Data, Registration, Descriptor, Siamese Network, PointNet, Indoor Scene

ABSTRACT:

This paper presents a deep learning feature-based method for registration of indoor mobile LiDAR data. Our method is to input point cloud directly, which is more robust to noise than traditional algorithms. The proposed method involves three steps. We first extract the key points by Harris3D algorithm and get their local patches by our sampling method. Second, a Siamese network is trained to describe the patches as local descriptors. Finally, we obtain the final matching pairs depends on the distance which is between two descriptors, and then solve the transformation matrix. The accuracy of registration is within 6 cm when the overlap is greater than 35%. In order to improve the registration accuracy, the ICP algorithm is used to fine-tuning the registration results. And the final registration accuracy is within 3.5 cm. The experiments show that our method applied to the registration of indoor mobile LiDAR data robustly and accurately.

1. INTRODUCTION

In recent years, the high definition maps (HD maps) based on mobile LiDAR data draw more and more attention. Individual point cloud data is incomplete due to factors such as limited s-canning distance and occlusion. Complete and sufficient detailed HD maps can be obtained by registering multi-view data or multiple scanned data. Traditional registration methods are mainly implemented by hand-crafted descriptors. However, because of the big noise in the mobile LiDAR point cloud, such descriptors are difficult to be applied to the registration of the HD maps. Thus, developing a feature descriptor that can be used in the HD maps is necessary.

Registration is a basic and important technology in 3D reconstruction. Considering that feature descriptor is one of the important factors in registration, many research teams have proposed different kinds of hand-crafted 3D feature descriptors for 3D data registration, like Spin Images (Johnson and Hebert, 1999), A-COV (Zai et al., 2017), Rotational Projection Statistics (RoPS) (Guo et al., 2013) and local surface description (Tombari et al., 2010). However, Spin Image relies on the resolution of direction and is sensitive to noise. RoPS requires mesh data and cannot be directly applied to the point cloud. With the rapid development in the deep learning network, some feature descriptors based on networks are proposed. A data-driven model (3DMatch (Zeng et al., 2017)), learned a local volumetric patch descriptor for establishing correspondences between partial 3D data of RGB-D data. The descriptor of 3Dmatch is more robust than traditional descriptors. (Simonyan et al., 2012) achieved viewpoint invariant matching using sparse feature detectors, which was trained by descriptor learning from a deep learning network. This descriptor achieves good results. However, it only works on images. Therefore, it is meaningful to develop a deep learning feature descriptor for mobile LiDAR data. Deep learning networks proposed for processing point clouds, such as ShapeNet (Chang et al., 2015), PointNet (Qi et al., 2017a), OctNet (Riegler et al., 2017), PointNet++ (Qi et al., 2017b), PointSift (Jiang et al., 2018) are mainly used for classification or semantic labeling. Thus, there is a potential, with great challenges, to apply these networks for registration.

In this paper, we structure a deep learning feature descriptor, which is extracted by a Siamese network based on the PointNet for registration of mobile LiDAR data. Firstly, to generate local data for training network, we extract key points by The Harris3D algorithm and get local patches by sampling their neighborhood data. Secondly, with many pairs of patches labeling positive or negative manually based on whether they are around the same key points, we train a network to generate feature descriptor. Then, with Euclidean distance between descriptors, we find the matching features and solve the transformation relationship. Finally, fine registration is achieved by an Iterative Closest Point algorithm (ICP). Our contribution are mainly include two points: 1. For the input of PointNet network, a sampling method is proposed, which can be applied to point cloud data with uneven density distribution. 2. A descriptor based on PointNet twin network is constructed for point cloud registration.

2. METHOD

In this section, a local descriptor is proposed. The method in this research leverage on mobile LiDAR data, which includes data preprocessing, network architecture construction, and descriptor-based registration.

2.1 Data Preprocessing

The quality of data acquired by mobile laser scanning equipment is relatively low, and there are many noise points in the data. Therefore, we first use statistical filtering (Rusu and Cousins, 2011) to remove noise points and voxel filter (Rusu and Cousins, 2011) to sample data. Point clouds registration is to get a rigid body transformation for coordinate alignment of point clouds. The transformation can be obtained from the matching pairs of points between point clouds. In order to improve the efficiency of registration, the key points in point clouds are used instead of all points. Firstly, Harris3D method (Sipiran and Bustos, 2011) is

*Corresponding author

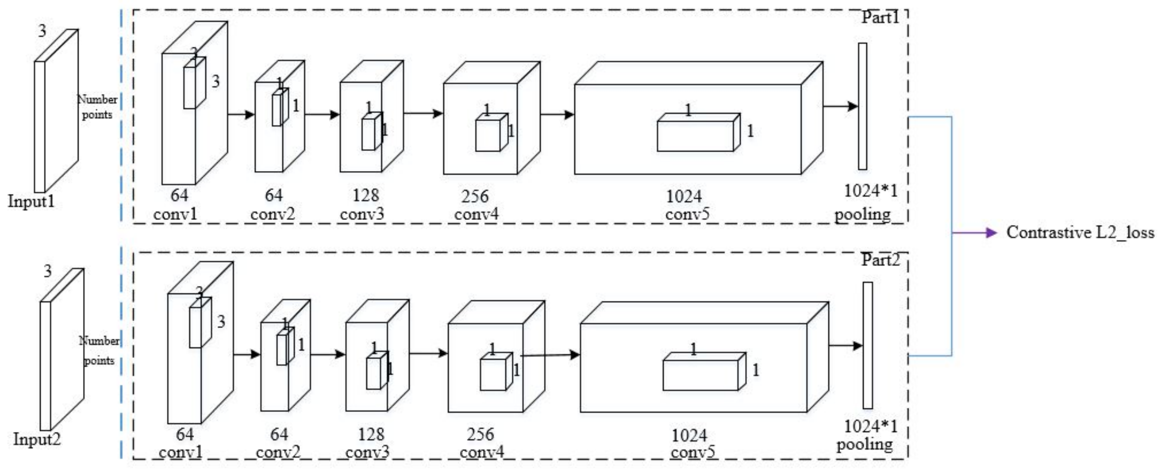


Figure 1. The Siamese network architecture

used to extract key points. Then, in order to improve the accuracy of key point matching, the key points are expressed by its neighborhood points. Because the descriptor generated by PointNet Siamese network is used in this paper, it is necessary to extract the neighborhood points of fixed points. The specific sampling algorithm is as follows:

Firstly, the key points of point clouds data are extracted by Harris3D algorithm.

Then, we sampled a patch representation for the local region surrounding each key point. The specific algorithm is as follows:

With each key point (O), we extract points (P) whose distance to O is less than r. The number of points in P is P_n. We divide P into eight subsets (P_1, P_2 ... P_8). The number of points in P_i is P_{in} (i=1...8). The distance (d_i) from points in P_i to the key point O satisfies:

$$\frac{r\cdot(i-1)}{8} < d_i \le \frac{r\cdot i}{8} \tag{1}$$

Point clouds are mainly distributed on the surface of objects and the form of local distribution trend surface. For extracting m points uniformly from P, the number of points extracted from each subset is proportional to the area it forms. The number of points (n_i) that should be extracted from P_i is as follows:

$$n_i = \frac{\pi\cdot i^2}{\pi\cdot 8^2}\cdot m - \frac{\pi\cdot(i-1)^2}{\pi\cdot 8^2}\cdot m = \frac{2\cdot i-1}{8^2}\cdot m \tag{2}$$

We use Algorithm 1 to accommodate point clouds with uneven density distribution.

Finally, the three-dimensional coordinates of the points in the sampled patch are normalized.

$$p_{nj} = \frac{(p_j - O)}{r} \tag{3}$$

Where p_{nj} is the normalized coordinate, p_j is the original coordinate. Final sampling result is shown in Figure. 2.

2.2 Network Architecture

With the neighborhood points of key points, descriptors can be generated. In the registration process, descriptors are used to

Algorithm 1 Adjustment

```
if $P_n < m$ then
    Data around this key point is too sparse, so we abandon the key point;
else
    Set the flag for each point in $P$ and initialize it to 0;
    $count$=0;
    % Sampling Fixed Points from Subsets
    for $i = 1 \rightarrow 8$ do
        if $n_i \le p_{in}$ then
            There are enough points in the subset. Randomly extract $n_i$ points in pi and set the flag of these points to 1;
            $count$=$count$+$n_i$;
        else
            There are not enough points in the subset. Extract all points in $P_i$ and set the flag of these points to 1;
            $count$=$count$+$P_{in}$;
        end if
    end for
    When the number of points in subsets is insufficient. Randomly extract ($m$-$count$) points in the points where the flag is 0;
end if
```

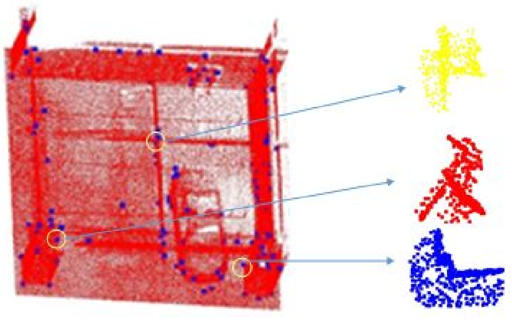


Figure 2. Data preprocessing. (a) Key points are extracted from the point cloud, (b) Local patches are extracted from the point cloud.

measure the similarity of key points, We use PointNet Siamese network to generate descriptors and evaluate the similarity of key points.

Our model is a Siamese network (Mou et al., 2017), which has two parts that are composed of PointNet. As shown in Figure. 1, Part 1 and Part 2 shared the parameters. Each part consists of five layers of convolution and one layer of pooling. The convolution layer is used to extract features, and the pooling layer is used to compress features and achieve rotation invariance.

Conv1 consists of the convolutional layer, relu layer, and bn layer. The size of the convolution kernel is 3x3 and the number of channels is 64. Conv2 consists of the convolutional layer, relu layer, and bn layer. The size of the convolution kernel is 1x1 and the number of channels is 64. Conv3 consists of the convolutional layer, relu layer, and bn layer. The size of the convolution kernel is 1x1 and the number of channels is 128. Conv4 consists of the convolutional layer, relu layer, and bn layer. The size of the convolution kernel is 1x1 and the number of channels is 256. Conv5 consists of the convolutional layer, relu layer, and bn layer. The size of the convolution kernel is 1x1 and the number of channels is 1024. The type of the pooling layer is max pooling and the pooling size is [Numberpoints,1].

The input of the network is the neighborhood data of two key points. Part1 and Part2 in the network generate two feature vectors based on the input neighborhood data. Then the normalized distance (d) of the two feature vectors is calculated. d is used to measure the similarity of key points. Positive samples are neighborhood data of two similar key points, and the ground truth of d is 0. Negative samples are neighborhood data of two different key points, and the ground truth of d is 1.

The contrastive loss of d is used in our model as follows:

$$Loss = \frac{1}{2N}\sum_{k=1}^{N}(yd^2 + (1-y)\cdot max(margin - d, 0)^2) \quad (4)$$

Where N is the number of samples, y is the label of d, $margin$ is the threshold of d(with $margin$ = 1 in this work).

2.3 Registration

Empirically, for source point cloud (A) and target point cloud (B), we can extract n and m key points respectively. Directly matching key points need to run $n * m$ times, and the time complexity is relatively high. So the nearest neighbor method is used to match descriptors. The specific process is as follows:

Firstly, the description set($F_a\{a_1, a_2, a_3, ... , a_n\}$ and $F_b\{b_1, b_2, b_3, ... , b_m\}$) of the source point cloud and the target point cloud are generated by used by the part1 of the Siamese network.

Then, the kd tree k_a andk_b are constructed from the set of two descriptors. For each feature descriptor in F_a, we search for its nearest feature descriptor in k_b. n pairs of closest feature descriptors($F_n\{ab_1, ab_2, ab_3, ... , ab_n\}$) are found. In the same way, for each feature descriptor in F_b, we find m pairs of closest feature descriptors($F_m\{ba_1, ba_2, ba_3, ... , ba_m\}$).

Finally,ab_i and ba_j are same matching pair when they have the same feature description. F_u (the union of F_n and F_m) and F_i (the intersection of F_n and F_m) can be get. As shown in Fig. 3 (a) is the matching pair in F_u and the accuracy is 31.5%. (b) is

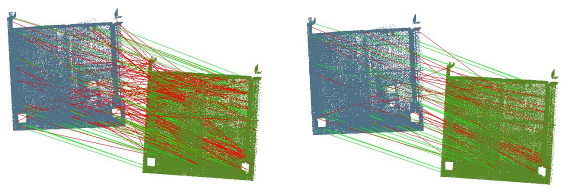


Figure 3. The matched point pairs. Red matching pairs are wrong matching pair and green matching pairs are right matching pair. (a) is the F_u matching pairs, the accuracy is 31.5%. (b) is the F_i matching pairs, the accuracy is 39.7%.

the matching pair in F_i and the accuracy is 39.7%. Therefore, F_i is selected as the final candidate matching pair.

Our goal was to find a rigid body tranformation (η) to minimize the distance of $\eta(A)$ and B. The rigid body transformation is include rotation transformation (R) and translation transformation(T). The distance of $\eta(A)$ and B is as follows:

$$J = \sum_{a_i \in A, b_i \in B} \parallel b_i - (Ra_i + T) \parallel \quad (5)$$

where a_i is the point in A. b_i is the nearest point in B to the point, a_i after the rigid body transformation. Setting threshold ρ for J, rough registration is completed when the J is less than threshold ρ.

Candidate matching pairs have correct pairs and wrong pairs, and can not be directly used to calculate rigid body transformation. Four matching pairs can calculate a set of rigid body transformation relations, so we use RANSAC method to randomly extract four matching pairs to calculate transformation relations until J is less than the threshold ρ. Finally, the ICP algorithm is used to fine the tuning registration results. We set the distance threshold α of corresponding points, the new distance threshold β of J_w and the number of iterations threshold c.

1. we set the weight w_i of each point in B. When the distance between b_i and a_i after rigid body transformation is less than α, w_i is 1, otherwise w_i is 0.

2. New distance function is constructed as follows:

$$J_w = \sum_{a_i \in A, b_i \in B} w_i \parallel b_i - (Ra_i + T) \parallel \quad (6)$$

J_w is solved by SVD method, new rigid body transformation and B are obtained.

3. if J_w is less than threshold β and the number of falls is less than c, the iteration stops, otherwise repeat 1 and 2.

3. EXPERIMENTAL RESULTS

In this paper, point clouds were obtained by a backpacked mobile mapping system (Gong et al., 2018). This system has two 16-line laser scanners (Velodyne VLP-16). Each laser scanner has sixteen laser-detector pairs individually aimed in 2 increments over the 30(-15 to +15) field of view. The experimental scene include 2 scenes. As shown in Figure. 4, the data of the two scenes have been rendered by lighting. Scene 1 is an underground garage and

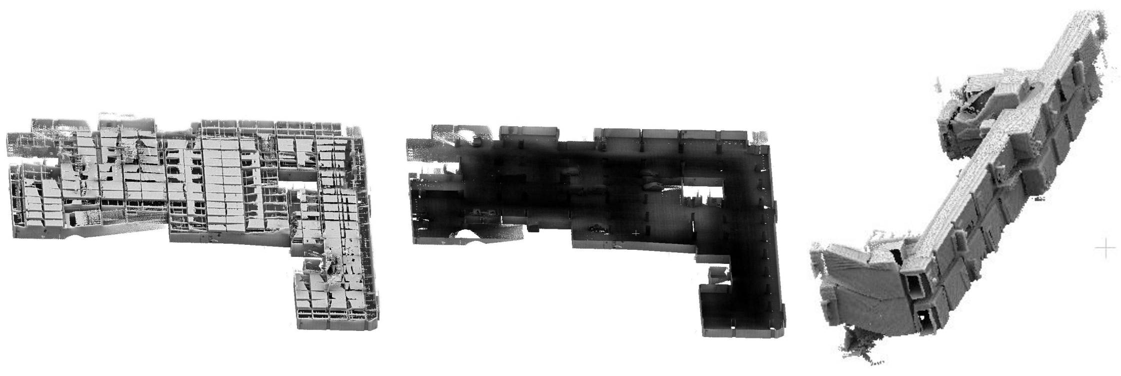

(a) (b) (c)

Figure 4. Experimental scenes. (a) Scene 1: underground garage. (b) The internal scene of the underground garage. (c) Scene 2:Two-story corridor.

scene 2 is a corridor. Scene 1 covers an area of 11874.29 m^2 and includes 16,186,035 points. Scene 2 covers an area of 360 m^2 and includes 4,924,335 points.

The training data includes 30,000 positive samples and 30,000 negative samples. The number of points in a patch is 512, and the distance threshold r=0.8m. The size of the input data for the network is 512×3×2. The network is built by tensorflow-GPU, and training took around 13 hours on a Nvidia Geforce GTX1080Ti, and it is running offline.

Experiments are divided into two parts. Firstly, we compared the descriptors learned by the network with the traditional descriptors(FPFH and SHOT).

The test data consists of 800 samples, which are randomly sampled from each scene by 400 samples (200 positive samples and 200 negative samples).

To compare the different features of descriptors, we normalize the features to a sphere with a radius of 1.

$$f_o = \frac{f_i}{||f_i||} \tag{7}$$

Where f_i is the original feature, f_o is the normalized feature. Each sample includes two patches, and it can generate two descriptors. The distance between the two descriptors is as follow:

$$d_f = ||f_{o1} - f_{o2}|| \tag{8}$$

d_{o1} and d_{o2} are normalized feature in a sample. d_f is the distance between two features.

Compared with negative samples, the patches of positive samples are more similar, so the d_f of positive samples is smaller than that of negative samples. All normalized descriptors are distributed on the sphere with radius 1, so the ground truth of d_f of positive samples is 0 and that of negative samples is 1. To compare the accuracy of descriptors, definition e is the difference between d_f and its ground truth. The smaller e is, the more accurate the descriptor is. The distribution of e on the test set is shown in Figure. 5. In Figure. 5. (a) (b) (c) is the result of scene 1, and (d) (e) (f) is the result of scene 2. (a) (d) is the e of the FPFH descriptor. (b) (e) is the e of SHOT descriptor. (c) (f) is the e of the descriptor proposed in this paper. The abscissa of each small figure is the ordinal number of the sample and the ordinate is the value of e.

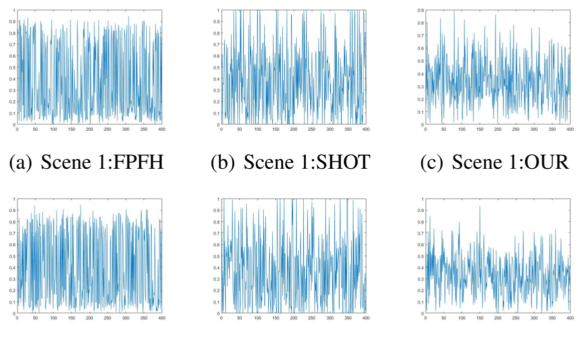

(a) Scene 1:FPFH (b) Scene 1:SHOT (c) Scene 1:OUR
(d) Scene 2:FPFH (e) Scene 2:SHOT (f) Scene 2:OUR

Figure 5. The e distribution of different descriptors on two test scenes.

		FPFH	SHOT	OUR
scene 1	MEAN	0.3837	0.3716	0.3385
	STD	0.3208	0.28815	0.1798
scene 2	MEAN	0.3627	0.3886	0.3332
	STD	0.3168	0.2819	0.1742

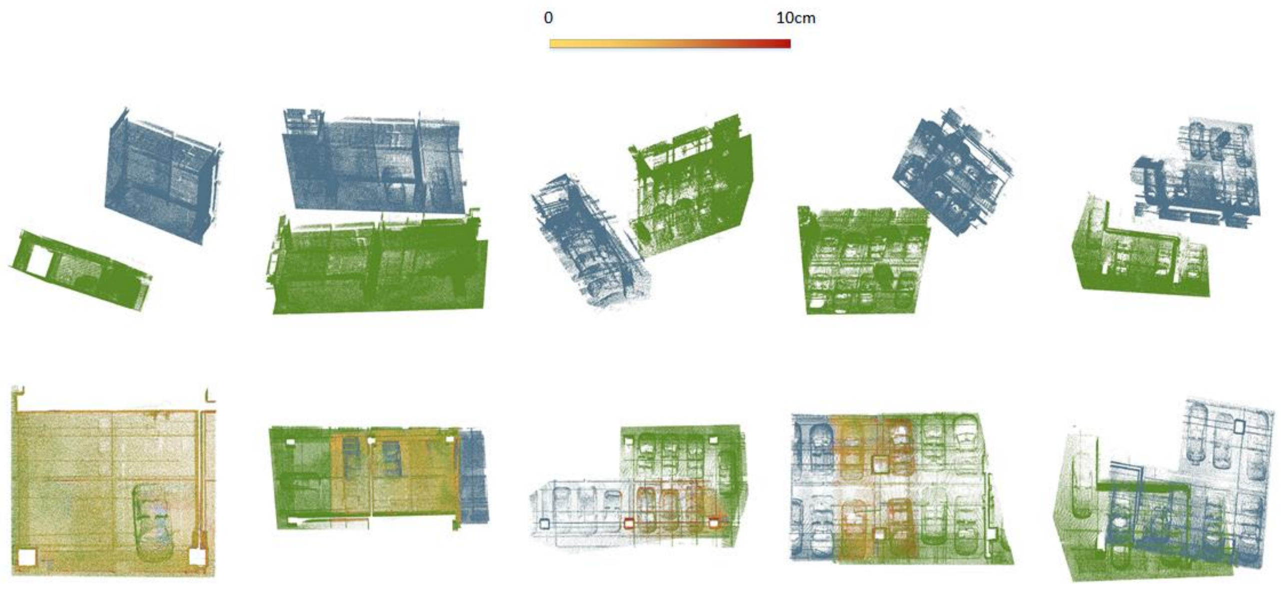
Table 1. Comparison of different descriptors

As shown in Table 1, MEAN is the average of e. STD represents the standard deviation of e.

From the Figure. 5 and TABLE 2, we can obtain that the MEAN and STD of our descriptors are lower than that of FPFH and SHOT. Our descriptors are more accurate and robust.

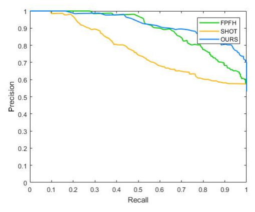
In this paper, we set a threshold for d_f. The samples whose d_f is greater than the threshold are positive samples and the others are negative samples. According to different thresholds, the accuracy and recall rate of different descriptors can be calculated., as shown in Figure. 7. The curve of our method has the largest integral area. Intuitively, the proposed method achieves the best accuracy.

Then, we discussed the effect of different overlapping rates on the results of the registration. The experimental data were collected twice (one week apart) from the underground garage. The registration errors are measured by the root mean square error ($RMSE$). M_{gt} is the known transformation, which perfectly aligns both point clouds Q and S. M_{our} is the transformation produced by our method. Then, the $RMSE$ of our method is

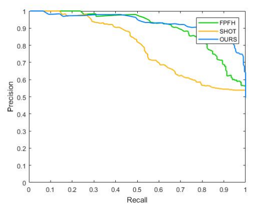


(a) (b) (c) (d) (e)

Figure 6. Different overlap scenes. The first line is the distributions of the original two point clouds. And the second line is the distributions of the point clouds after registration. (a) data overlap rate is 95%, (b) data overlap rate is 75%, (c) data overlap rate is 50%, (d) data overlap rate is 35%, (e) data overlap rate is 30%.



(a) Scene 1



(b) Scene 2

Figure 7. The PR curves of different descriptors on the two test scenes.

measured by

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|M_{gt} \cdot Q - M_{our} \cdot Q\|^2}{N}} \tag{9}$$

we set the distance threshold d_r between the corresponding points. If the distance between the corresponding points is less than the threshold d_r, the corresponding points are considered correct and the point pair is added to the calculation of $RMSE$. Otherwise the point is not added to the calculation of $RMSE$. In this work, d_r is 10 cm.

Ovelap(%)		95	75	50	35	30
$RMSE$(cm)	Before ICP	3.18	3.82	3.99	5.81	-
	After ICP	1.98	1.70	3.39	2.89	-

Table 2. Comparison of different descriptors

As shown in Figure. 6 and Table 2, Before ICP, the accuracy of registration is less than 4 cm when the overlap of the point clouds is higher than 50%. The accuracy of registration is less than 6 cm when the overlap of the point clouds is higher than 35%. After ICP, the accuracy of registration is within 3.5 cm when the overlap of the point clouds is higher than 35%. And the registration cannot be completed when the overlap rate is less than 30%.

4. CONCLUSION

A deep learning feature-based method for point clouds registration of mobile LiDAR data is presented in this paper. In order to adapt to the input of our network, we propose a novel sampling method. And we built a Siamese network to obtain the descriptor of the extracted feature points. The deep learning feature descriptors of the point clouds are used to register and the results show that $RMSE$ is less than 4 cm when the overlap is greater than 35%. In the future, we will experiment on more scenes to improve the robustness of the descriptor. Instead of RANSAC, we will selecte the final matching pair using a clustering method based on graph structure.

ACKNOWLEDGMENT

This work was supported by the National Natural Science Foundation of China (Grant No. 61601392 and No.41871380).

REFERENCES

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H. et al., 2015. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*.

Gong, Z., Wen, C., Wang, C. and Li, J., 2018. A target-free automatic self-calibration approach for multibeam laser scanners. *IEEE Transactions on Instrumentation and Measurement* 67(1), pp. 238–240.

Guo, Y., Sohel, F., Bennamoun, M., Lu, M. and Wan, J., 2013. Rotational projection statistics for 3d local surface description and object recognition. *International journal of computer vision* 105(1), pp. 63–86.

Jiang, M., Wu, Y. and Lu, C., 2018. Pointsift: A sift-like network module for 3d point cloud semantic segmentation. *arXiv preprint arXiv:1807.00652*.

Johnson, A. E. and Hebert, M., 1999. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on Pattern Analysis & Machine Intelligence* (5), pp. 433–449.

Mou, L., Schmitt, M., Wang, Y. and Zhu, X. X., 2017. A cnn for the identification of corresponding patches in sar and optical imagery of urban scenes. In: *Urban Remote Sensing Event (JURSE), 2017 Joint*, IEEE, pp. 1–4.

Qi, C. R., Su, H., Mo, K. and Guibas, L. J., 2017a. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* 1(2), pp. 4.

Qi, C. R., Yi, L., Su, H. and Guibas, L. J., 2017b. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: *Advances in Neural Information Processing Systems*, pp. 5099–5108.

Riegler, G., Ulusoy, A. O. and Geiger, A., 2017. Octnet: Learning deep 3d representations at high resolutions. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Vol. 3.

Rusu, R. B. and Cousins, S., 2011. Point cloud library (pcl). In: *2011 IEEE International Conference on Robotics and Automation*, pp. 1–4.

Simonyan, K., Vedaldi, A. and Zisserman, A., 2012. Descriptor learning using convex optimisation. In: *European Conference on Computer Vision*, Springer, pp. 243–256.

Sipiran, I. and Bustos, B., 2011. Harris 3d: a robust extension of the harris operator for interest point detection on 3d meshes. *Visual Computer* 27(11), pp. 963.

Tombari, F., Salti, S. and Di Stefano, L., 2010. Unique signatures of histograms for local surface description. In: *European conference on computer vision*, Springer, pp. 356–369.

Zai, D., Li, J., Guo, Y., Cheng, M., Huang, P., Cao, X. and Wang, C., 2017. Pairwise registration of tls point clouds using covariance descriptors and a non-cooperative game. *ISPRS Journal of Photogrammetry and Remote Sensing* 134, pp. 15–29.

Zeng, A., Song, S., Nießner, M., Fisher, M., Xiao, J. and Funkhouser, T., 2017. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In: *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, IEEE, pp. 199–208.