



PointpartNet: 3D point-cloud registration via deep part-based feature extraction

Shixun Yan, Sarthak Pathak & Kazunori Umeda

To cite this article: Shixun Yan, Sarthak Pathak & Kazunori Umeda (2022): PointpartNet: 3D point-cloud registration via deep part-based feature extraction, Advanced Robotics, DOI: [10.1080/01691864.2022.2084346](https://doi.org/10.1080/01691864.2022.2084346)

To link to this article: <https://doi.org/10.1080/01691864.2022.2084346>



Published online: 21 Jun 2022.



Submit your article to this journal [↗](#)



View related articles [↗](#)






View Crossmark data [↗](#)

FULL PAPER



PointpartNet: 3D point-cloud registration via deep part-based feature extraction

Shixun Yan ^a, Sarthak Pathak ^a and Kazunori Umeda ^b

^aCourse of Precision Engineering, Chuo University, Tokyo, Japan; ^bDepartment of Precision Mechanics, Chuo University, Tokyo, Japan

ABSTRACT

This paper proposes a deep learning model for point cloud registrations of different sizes. 3D point clouds play a very important role in various fields. They have been widely studied, and recently deep learning has also started to deal with point clouds. PointNet was the first deep learning model for point cloud classification and semantic segmentation. Since then, methods based on PointNet for tasks like point cloud registration have also been proposed. However, these methods are only suitable for identical or nearly identical point clouds. However, in practice, the sizes of point clouds vary depending on the capture distance, sensor type, the environment, and many other factors. Therefore, it is often the case that point clouds that need to be registered are of very different sizes. For example, point clouds captured in the same environment by an omnidirectional LiDAR and an RGB-D camera will have very different sizes. Conventional methods cannot cope with such situations. In this paper, we propose 'PointpartNet', a new deep neural network based on partial feature extraction. This network enables feature extraction of partial point clouds by partitioning the point clouds. It uses the features of partial point clouds to search for matching regions between point clouds of different sizes. This makes it capable of registering point clouds of different sizes. In qualitative experiments, we demonstrate its high robustness and accuracy for point cloud registration of different sizes in comparison to previous research.

ARTICLE HISTORY

Received 7 October 2021
Accepted 21 May 2022

KEYWORDS

Point cloud; deep learning; registration

1. Introduction

Distance image sensors have been researched and developed for a long time. They are very useful for tasks such as 3D measurement, Simultaneous Localization and Mapping (SLAM), robot navigation, and environmental awareness. However, all types of distance image sensors have their own limitations of range, accuracy, measurement conditions, etc. Therefore, the fusion of multiple distance image sensors is common in order to improve robustness [1,2]. Towards this aim, robust and accurate registration of point clouds obtained from different distance image sensors is extremely important.

In recent years, deep learning techniques have yielded excellent results in various domains. PointNet [3] is one of the earliest deep learning models for classification and semantic segmentation that deals directly with point clouds. It has a model structure that is independent of the number and order of input point clouds. After PointNet [3], several other models based on it were proposed [4–8].

Among them, several registration models based on PointNet [3] have been proposed [9,10]. One of the most representative models is PointNetLK [9]. In these models, the accuracy of registration tends to decrease when the

point clouds are noisy or have different sizes when point clouds are acquired with different range image sensor. In order to solve this problem, a model called MaskNet [11] was proposed. It is a model that searches for corresponding regions between point clouds. However, this model is not robust and accurate if the difference in the size of the point clouds is too large. In this research, we propose 'PointpartNet', a neural network that partitions the point cloud, extracts feature for each part, and calculates the matching likelihood score for each part. This score predicts which part of the larger full point cloud is most similar to the smaller part cloud. Using this result, another neural network is used to perform the actual registration. In addition, we divide registration into global registration and local registration to avoid falling into local minima.

2. Related work

Many registration methods have been proposed over the years [12–14]. Iterative Closest Point (ICP) [12] is an iterative point cloud registration method. This method is known to suffer from inferior performance in the presence of outliers and is strongly dependent on the initial

translation and rotation of the point cloud. However, due to its simplicity, this is one of the most widely used methods. Normal Distributions Transform (NDT) [13] is a method that divides the point cloud space in equal intervals using voxels and grids, and expresses and matches the point clouds using the Normal Distribution obtained for each. Therefore, the calculation cost can be reduced. However, most of these methods are not suitable for the registration of point clouds with vast size differences. NDT matching are used for the registration between large point clouds with vast size differences. However, there are some errors may be found on the NDT mapping algorithm [15,16]. NDT mapping itself is very correct when viewed from above, but when viewed from the side there is a persistent deformation in the vertical axis [17]. Recently, many point cloud-based deep learning models have been proposed, as discussed below:

- Classification/Semantic Segmentation: PointNet [3] is the first deep learning model for the classification and semantic segmentation of point clouds. PointCNN [4] is a deep learning model for the classification and semantic segmentation considering order of points in point clouds. X-Conv [4] is a method of selecting representative points from a set of points and convolutionally aggregating the information of neighbouring points. This model is more robust than PointNet [3].
- Registration: There are PointNet [3]-based methods such as DirectNet [18] and PointNetLK [9]. DirectNet [18] treats the point cloud features extracted by the neural network as input to the Multilayer Perceptron (MLP). It is a model that derives rigid transformations from them. However, this model can only perform global registration. PointNetLK [9] is a model that uses PointNet [3] to extract the features of the entire point cloud and then compute a rigid transformation matrix to make the features of the target point clouds similar to each other. However, this method has a prerequisite that two point clouds must be almost identical. When the point cloud sizes are vastly different, (e.g. if one point cloud is part of the other), the registration accuracy is reduced.
- Matching Region Search: A model called MaskNet [11] was proposed to calculate the matching region between one point cloud and the other. However, the point clouds are considered as a whole, without looking at sub-regions in this method. As a result, MaskNet [11], too, cannot deal with point clouds of greatly different sizes or with point clouds of greatly different postures, and matching often fails.

Registration method based on dictionary learning has also been proposed [19]. However, dictionary learning

will make the method specific to the point cloud being trained on, whereas our approach can work with different point clouds in training and testing.

3. Pointpartnet

In this research, we pre-suppose the following scenario for the two point clouds to be registered – 1. a **full** point cloud extracted by a wide range scanner, such as a LiDAR and 2. a more close-up subsection of it called **part**, with greater detail, scanned by a small range scanner such as an RGB-D camera. An example is given below in Figure 1. The basic concept of PointpartNet is to create independent point-wise sub-point clouds of **full** via nearest neighbor-based partitioning. By extracting the features of the sub-point clouds, we can find a precise matching region between the points. Following is an overview of the PointpartNet, consisting of two steps: (1) Part-based feature extraction and matching region search, and (2) Registration with the matching point cloud found in step 1. An overview of the PointpartNet architecture is shown in Figure 2. The architecture will be explained below.

The overview of the architecture:

- (1) Part-based feature extraction: In order to find a region in **full** that matches with **part**, we partition **full** and extract partial features.
- (2) Matching region search: In matching region search, we can find the position where **part** most matches with **full**, and thus global translation \mathbf{t}_G can be obtained.
- (3) Registration: Initially, we perform global registration via the neural network and compute the global rotation \mathbf{R}_G . Next, we use Singular Value Decomposition (SVD) to perform more precise local registration and compute local rotation \mathbf{R}_L and local translation \mathbf{t}_L .

3.1. Part-based feature extraction

Since point clouds are not always of the same density, we need to downsample them to the same density. After that, the larger full point cloud is called **full** = $\mathbf{f}_1, \dots, \mathbf{f}_g$, and the smaller part is called **part** = $\mathbf{p}_1, \dots, \mathbf{p}_n$, where points $\mathbf{p}_i = (x_i, y_i, z_i)$, $\mathbf{f}_j = (x_j, y_j, z_j)$ are 3D points. g is the number of points in **full**, and n is the number of points in **part**.

The main objective of this step is to partition the point cloud **full** in order to extract the part-based feature of each part of **full**. For each 3D point in **full**, we choose the n nearest neighbour points and create a set of g sub-point clouds $\mathbf{full}_{group} = \mathbf{full}_{group_i} \mid i = 1, \dots, g$.

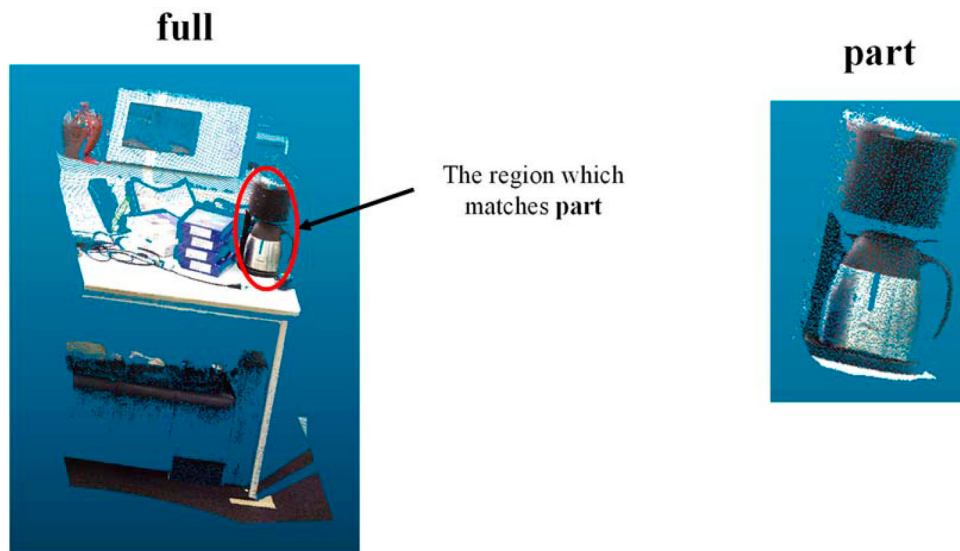


Figure 1. Point cloud sample.

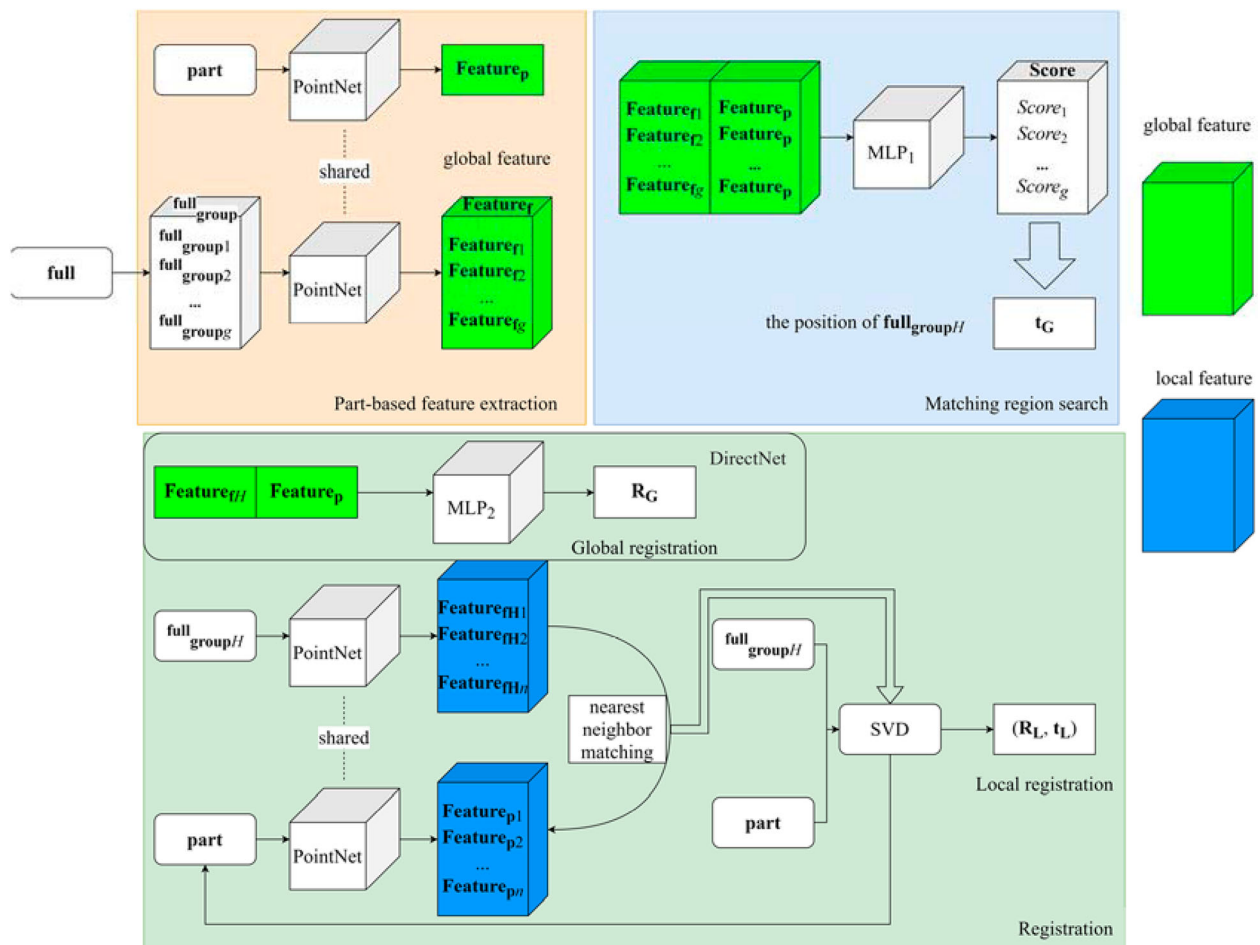


Figure 2. PointpartNet architecture.

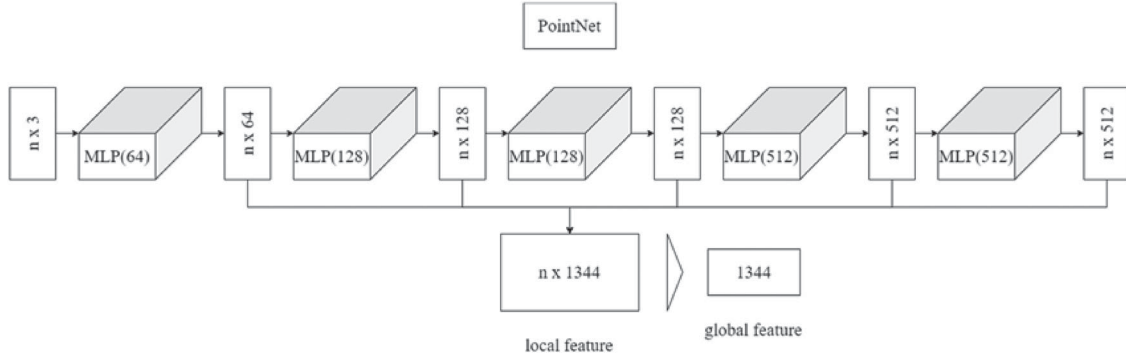


Figure 3. PointNet [3] architecture.

Then, we input **part** and **full_{group}** into PointNet [3] and extract their respective features $\mathbf{Feature}_p = \mathbf{F}_p \mid \mathbf{F}_p$, $\mathbf{Feature}_f = \mathbf{F}_{fi} \mid \mathbf{F}_{fi} \in \mathbb{R}^d, i = 1, \dots, g$. PointNet [3] is used to perform feature extraction for each input point cloud, as shown in Figure 3. In order to acquire part-based features independent of position, the point clouds are translated to make sure that their centroids of gravity coincide with the origin. Moreover, they are also normalized into a unit box [9] depending on **full** to increase robustness to scale. The MLP sizes used by PointNet [3] in this research are (64, 128, 128, 512, 512). The feature of each input point cloud is obtained by concatenating the results of each layer followed by pooling.

3.2. Matching region search

Following the feature extraction, we search for which regions of **part** and **full** are most similar. In essence, this step calculates the global translation \mathbf{t}_G . By extracting the global features of the partial point clouds **full_{group}** and **part** as inputs, we can search for which region in **full** matches **part**. We input $\mathbf{Feature}_p$ and $\mathbf{Feature}_f$ into MLP_1 to calculate the matching likelihoods $\mathbf{Score} = \text{Score}_i \mid i = 1, \dots, g$. The Score_i is set to 1 when the **full_{group_i}** is similar to **part** and 0 when it is not. The size of MLP_1 is (256, 128, 128, 1). Next, the highest score Score_H is selected, and the corresponding **full_{group_H}** is determined to be the region matching the **part**.

3.3. Registration

After the matching region **full_{group_H}** is selected, registration between point clouds is performed. The registration is divided into two parts, Global registration and Local registration. Global registration a process of fast and rough registration. Local registration is a process of registration with high accuracy. In the case of point cloud registration, if the initial orientation is too different, there

is a big possibility of falling into local minima. Therefore, we first estimate a rough pose by global registration, and then perform local registration to reduce the possibility of falling into the local minima. Finally, the equation for transforming point clouds using the obtained rigid transformation matrix is given in Equation (1):

$$\mathbf{full}_{\text{group}H} = \mathbf{R}\mathbf{part} + \mathbf{t} \quad (1)$$

where \mathbf{R} is the rotation matrix and \mathbf{t} is the translation matrix

3.3.1. Global registration

In matching region search, we estimated \mathbf{t}_G , the position of **full_{group_H}**. Thus, we perform global initial registration to obtain the global rotation \mathbf{R}_G . In this step, the procedure is to find the \mathbf{R}_G close to the true value as quickly as possible. Here, we use DirectNet [18] to perform global registration, which is marked in Figure 2. We use the global features (green box in Figure 2) of **full_{group_H}** and **part** as input and calculate the twist parameter ξ [9] by MLP_2 . The size of the MLP_2 is (512, 256, 128, 128, 64, 3). The rotation \mathbf{R}_G is represented by an exponential map as follows:

$$\mathbf{R}_G = \exp \left(\sum_i \xi_i \mathbf{T}_i \right) \xi = (\xi_1, \xi_2, \xi_3)^T \quad (2)$$

where \mathbf{T}_i is a generator of exponential map with twist parameters $\xi \in \mathbb{R}^3$.

3.3.2. Local registration

Local registration is conducted to register point clouds whose positions and orientations are close to each other, i.e. after global registration has been performed in the previous step. Here, as shown in Figure 2, the rigid transformation matrix is obtained repeatedly until convergence using SVD with the local features (blue box in Figure 2) obtained from PointNet [3]. We use the local features of **part** and the matched region **full_{group_H}** to find

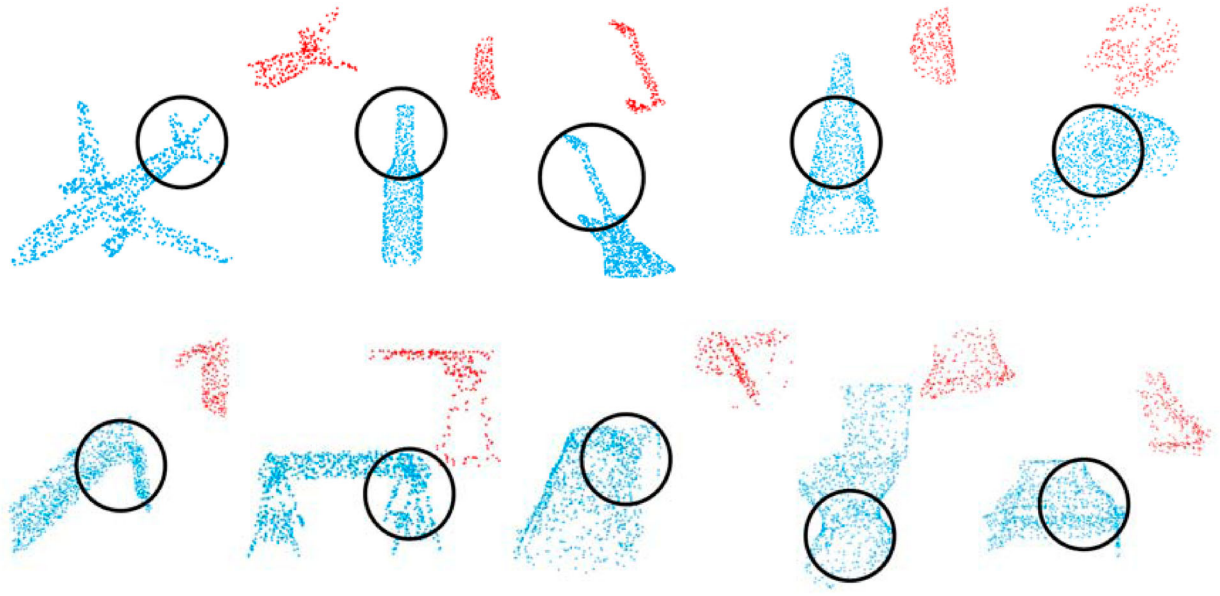


Figure 4. Point clouds sample: lower left: full, upper right: part.

the corresponding points and then use the 3D coordinates of **part** and $\mathbf{full}_{\text{group}H}$ as input to SVD to perform the local registration. The local feature is the feature of each point in the point cloud. Thus, a local feature has n number of features, where n is the number of **part** and $\mathbf{full}_{\text{group}H}$ points.

$$\mathbf{full}_{\text{group}H}' = \mathbf{full}_{\text{group}H} \left\{ I - \left(\frac{1}{n} \right) J \right\} \quad (3)$$

$$\mathbf{part}' = \mathbf{part} \left\{ I - \left(\frac{1}{n} \right) J \right\} \quad (4)$$

where I is the unit matrix, J is a matrix with all elements 1, n is the number of points, and $\mathbf{full}_{\text{group}H}'$ and \mathbf{part}' are the 3D point clouds of $\mathbf{full}_{\text{group}H}$ and **part** minus the translation component.

$$\mathbf{full}_{\text{group}H}' \mathbf{part}'^T = \mathbf{U} \Sigma \mathbf{V}^T \quad (5)$$

In this way, we decompose $\mathbf{full}_{\text{group}H}' \mathbf{part}'^T$ into singular values. where \mathbf{U} is the left singular vector matrix, Σ is the matrix with the singular values as diagonal components, and \mathbf{V} is the right singular vector matrix. Furthermore,

$$\mathbf{S} = \text{diag}(\| \mathbf{V} \mathbf{U}^T \|) \quad (6)$$

the rotation matrix \mathbf{R} can be obtained as follows with Equation (6):

$$\mathbf{R} = \mathbf{V} \mathbf{S} \mathbf{U}^T \quad (7)$$

Finally, the conversion formula for $\mathbf{full}_{\text{group}H}$ and **part** is given as follows:

$$\mathbf{full}_{\text{group}H} = \mathbf{R}_L (\mathbf{R}_G \mathbf{part} + \mathbf{t}_G) + \mathbf{t}_L \quad (8)$$

where \mathbf{R}_L is the rotation matrix and \mathbf{t}_L is the translation matrix obtained by local registration.

3.4. Training

For training, supervision is performed in two steps: matching region search and global registration. In addition, in order to improve the efficiency of the training process, the global registration used in the training is not $\mathbf{full}_{\text{group}H}$, but $\mathbf{full}_{\text{group}gt}$, which corresponds to the ground-truth of the matching region search, as input.

3.5. Loss function

The proposed model has two networks to train: matching region search, which involves choosing the highest score, and global registration. As a result, the loss function is divided into 2 parts: matching loss $loss_m$ and registration loss $loss_r$. $loss_m$ is the negative log-likelihood loss, which is often used in classification problems.

$$loss = loss_m + loss_r \quad (9)$$

$$loss_m = -\log \left(\frac{\exp(\text{Score}_{\text{label}})}{\sum_{i=1}^g \exp(\text{Score}_i)} \right) \quad (10)$$

where label is the index of the centre point \mathbf{f}_{gt} , the true value. $loss_r$ minimizes the difference between registration $\mathbf{G}_G = [\mathbf{R}_G, \mathbf{t}_G]$ and the ground truth transformation \mathbf{G}_{gt} . Thus, we use the Mean Square Error (MSE) to express it as follows.

$$loss_r = \| (\mathbf{G}_G)^{-1} \cdot \mathbf{G}_{gt} - \mathbf{I}_4 \|_F \quad (11)$$

4. Experiments

We conducted experiments on a public dataset as well as real-world data captured by our FARO laser scanner. First, we describe the experiments on the public dataset. Training was done using the ModelNet40 [20] dataset, which consists of 40 different Computer Aided Design (CAD) models. The model was trained on 20 of the training sets and tested on the other 20 sets to demonstrate the robustness of our model to untrained object types. Point cloud **full** is a resampled point cloud from ModelNet40 [20] data. The point cloud **part** is a randomly chosen quarter in the point cloud **full**, with noise added according to a normal distribution with mean 0 m and variance 0.05 m^2 . Point clouds sample are shown in Figure 4. The true value of the rigid transformation matrix used in training was randomly generated with rotation between $[0, 90]^\circ$ about arbitrarily chosen axes and random translation between $[0, 1.57] \text{ m}$. The true value of the rigid transformation matrix used for testing was randomly generated with rotation between $[0, 180]^\circ$ and translation between $[0, 3.14] \text{ m}$ to verify the robustness of our model to untrained positions and orientations. As a criterion for the experiment, the rotation estimation was considered to be successful when the rotation error was 10° or less, and the translation estimation was considered to be successful when the translation error was 0.1 m or less. Experimental conditions are shown in Table 1. The batch size used during training was 16 and the training epochs were 200. The number of points in the point cloud used for training were: **full** 256 points, **part** 64 points. Due to the limitation of GPU memory, we use downsampled point clouds to these sizes, and yet the accuracy is sufficient. The library used is python 3.6.10 and pytorch 1.7.0. We also compared our results with those of MaskNet [11]. The experiments were performed on a 3.60 GHz Intel i7-9700 K and a GeForce RTX2070 SUPER.

4.1. Comparison of matching region search

First, we conducted a matching comparison experiment to verify the usefulness of our matching region search model. The results are shown in Figures 5 and 6 and Table 2. As mentioned in section 3.2, the global position \mathbf{t}_G of the point cloud was estimated to evaluate the results of the matching region search. In this case, the conversion

Table 1. Experimental conditions.

	Batch size	The number of points in full	The number of points in part
Training	16	256	64
Testing	1	1024	256

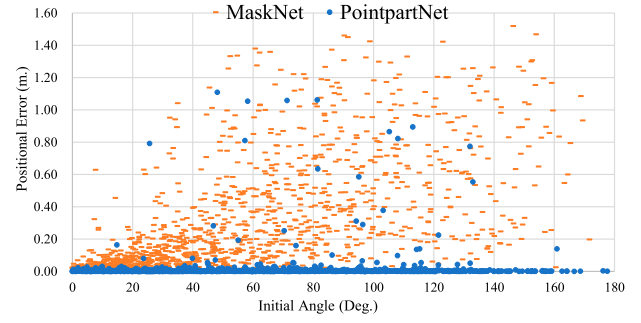
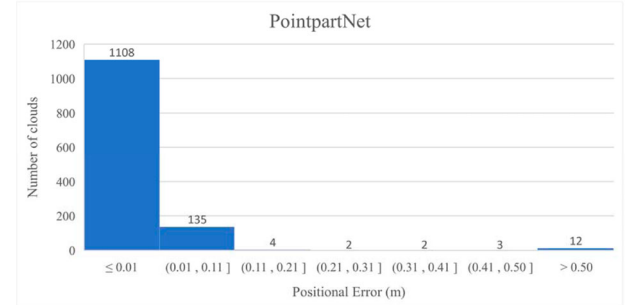
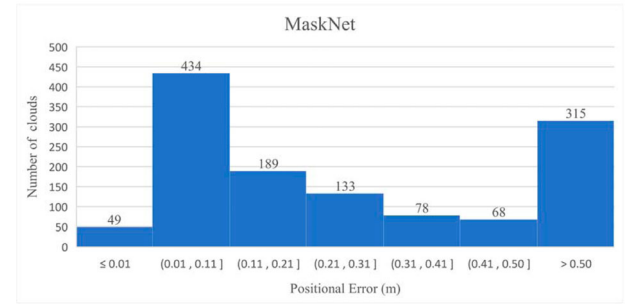


Figure 5. Matching results comparing PointpartNet and MaskNet [11].



(a) Histogram of positional error for PointpartNet



(b) Histogram of positional error for MaskNet[11]

Figure 6. Matching histogram comparing PointpartNet and MaskNet [11].

Table 2. Experimental results for each method.

	Success rate	Positional mean error (m)	Positional mean error on success (m)
MaskNet [11]	36.65%	0.326	0.043
PointpartNet	98.28%	0.013	0.003

formula for $\mathbf{full}_{\text{group}H}$ and part is given as follows

$$\mathbf{full}_{\text{group}H} \approx \mathbf{part} + \mathbf{t}_G \quad (12)$$

It is shown in Figures 5 and 6 that the position estimation error of MaskNet [11] increased rapidly when the initial rotation of exceeded 40° . Meanwhile, the results were stable across almost all initial rotation angles using the proposed method, PointpartNet. Figure 5 shows the

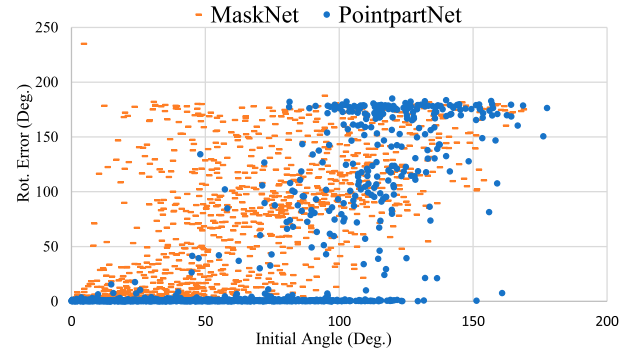
obtained error histograms of MaskNet [11] and the proposed method, PointpartNet. For MaskNet [11], the success rate of position estimation was 36.65%, and the overall average error of position estimation was 0.326 m. And when the position estimation was successful, the average error of position estimation was 0.043 m. In comparison, the overall results of the proposed PointpartNet network were more stable and robust. The success rate of position estimation was 98.28%, and the overall mean error of position estimation was 0.013 m. When the position estimation was successful, the average error of position estimation was 0.003 m. Thus, it can be seen that compared with MaskNet [11], the accuracy and success rate of position estimation of PointpartNet significantly improved and had high robustness to the initial rotation between the point clouds. The reason for the improved accuracy may be that PointpartNet partitions **full** and focuses on the sub-point clouds, while MaskNet [11] focuses only on each point of **full**, not sub-point clouds.

4.2. Comparison of registration

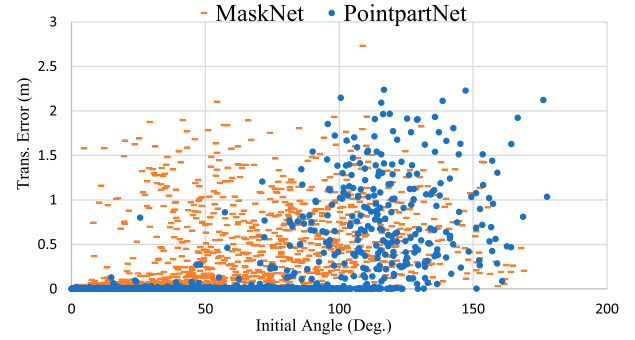
In addition, we also carried out experiments for point cloud registration. Since MaskNet [11] itself does not include registration, we used the registration of PointpartNet to obtain the rigid transformation matrix. The results are shown in Figures 7–9 and Table 3.

It is shown in Figure 7(a) that the rotation error of MaskNet [11] increased rapidly when the initial rotation exceeded 20°. The success rate of the rotation estimation was 29.38%, and the overall average rotation error was 68.95°. And when the rotation estimation was successful, the mean rotation error is 2.61°. Again, PointpartNet showed relatively more robust and accurate results as compared to MaskNet [11]. However, the rotation error tended to increase when the initial rotation exceeds 80°. The success rate of the rotation estimation was 80.25%, and the overall average rotation error was 26.40°. And when the rotation estimation was successful, the mean rotation error was 0.64°. Compared with MaskNet [11], PointpartNet significantly improved the accuracy and robustness of rotation estimation and proved to be more robust to initial rotation in rotation estimation.

In addition, it can be seen in Figure 7(b) that the translation error of MaskNet [11] increased rapidly when the initial rotation exceeded 20°. The success rate of the translation estimation was 38.47%, and the overall mean translation error was 0.417 m. And when the translation estimation was successful, the mean translation error was 0.031 m. PointpartNet was, again, relatively stable as compared to MaskNet [11], but the translation error also tended to increase when the initial rotation

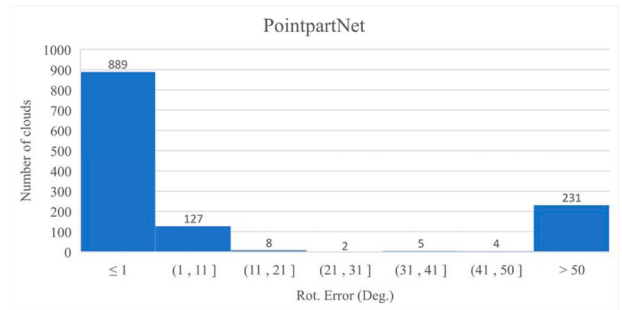


(a) Rotation error of PointpartNet and MaskNet[11]

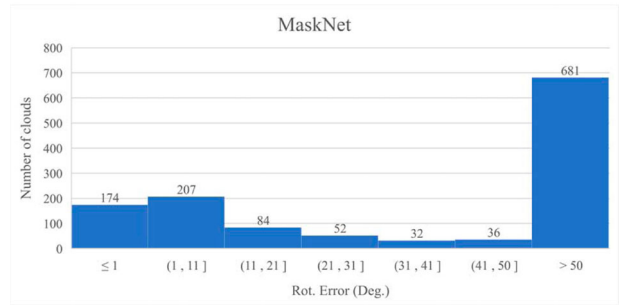


(b) Translation error of PointpartNet and MaskNet[11]

Figure 7. Registration results comparing PointpartNet and MaskNet [11].

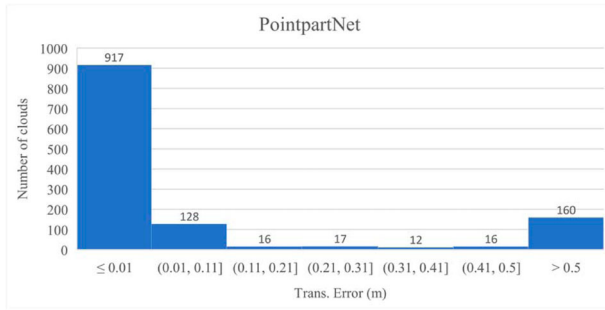


(a) Histogram of Rotation error for PointpartNet

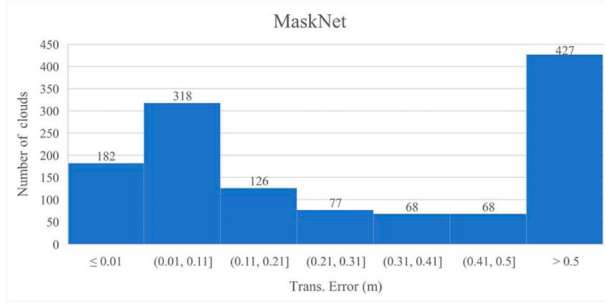


(b) Histogram of Rotation error for MaskNet[11]

Figure 8. Registration histogram comparing PointpartNet and MaskNet [11] (Rotation).



(a) Histogram of Translation error for PointpartNet



(b) Histogram of Translation error for MaskNet[11]

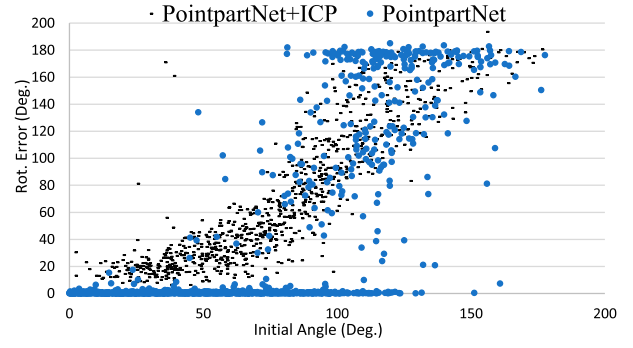
Figure 9. Registration histogram comparing PointpartNet and MaskNet [11] (Translation).**Table 3.** Experimental results for each method.

	Success rate	Rot. mean error (°)	Rot. mean error on success (°)
<i>(a) Rotation error comparison of PointpartNet and MaskNet [11]</i>			
MaskNet [11]	29.38%	68.95	2.61
PointpartNet	80.25%	26.40	0.64
<i>(b) Translation error comparison of PointpartNet and MaskNet [11]</i>			
MaskNet [11]	38.47%	0.417	0.031
PointpartNet	82.31%	0.160	0.006

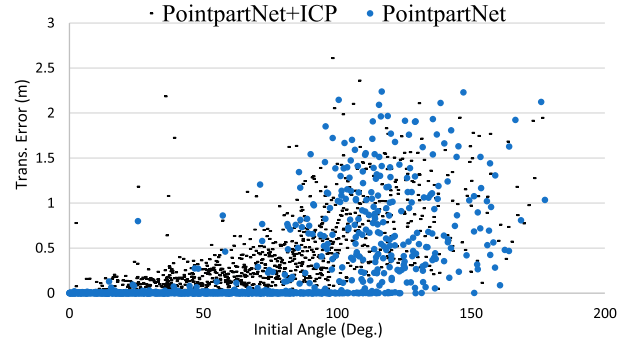
exceeds 80°. The success rate of translation estimation was 82.31%, and the overall mean translation error was 0.160 m. When the translation estimation was successful, the average translation error was 0.006 m. Similar to rotation estimation, PointpartNet proved to be much more accurate in translation estimation and more robust to initial rotation in rotation estimation as compared to MaskNet [11]. A sample of the registration is shown in Figure 10. The blue point cloud is **full**. The red point cloud is **part**.

The reason for the error could be that feature extraction using PointNet [3] is not sufficiently robust to orientation changes. Therefore, in future, it is necessary to consider an orientation-invariant feature extraction method. In addition, ModelNet40 [20] also contained objects with little feature, such as beds, laptops and tables. The registration of such objects is difficult and may fail.

We also conducted a comparison experiment by changing our local registration to ICP [12]. It can be seen



(a) Rotation error of PointpartNet and PointpartNet+ICP



(b) Translation error of PointpartNet and PointpartNet+ICP

Figure 10. Registration results comparing PointpartNet and PointpartNet+ICP.

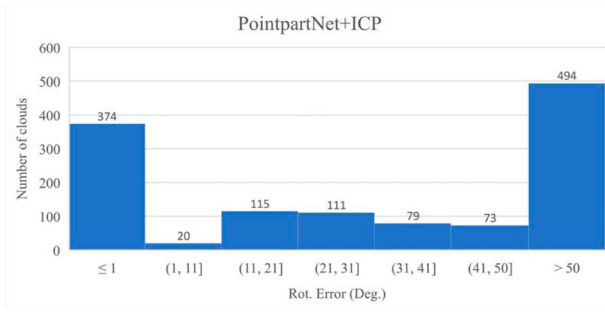
in Figures 11 and 12, It was shown that the accuracy of registration was reduced by changing local registration to ICP [12]. The reason for this is that ICP [12] requires a good initial position, but DirectNet [18] is not expected to provide the initial position required by ICP [12].

4.3. Processing time

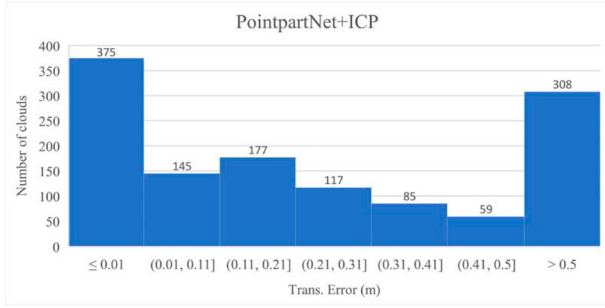
We estimated the processing time of each part of the registration process. The average processing time of each process is shown in Table 4. The processing time of part-based feature extraction was 0.306 s. The processing time for matching region search was 0.003 and 0.005 s for global registration. The processing time for local registration is 0.119 s.

4.4. Registration on real data

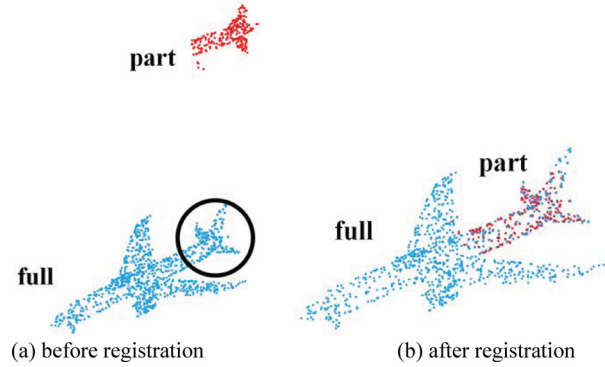
We used the network trained on ModelNet40 [20] and experimented with a point cloud taken from real range image sensor. The sensor we used was FARO Laser Scanner Focus^M70. The point cloud obtained by scanning a part of our laboratory with the sensor, shown in Figure 15(a), was taken as the **full** cloud. The **part** cloud was a set of about 500 point clouds cut randomly from



(a) Histogram of Rotation error for PointpartNet+ICP



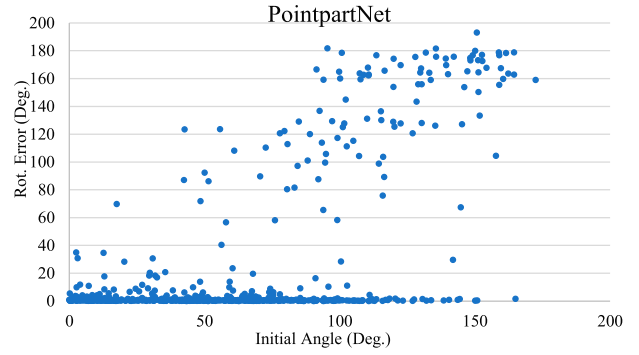
(b) Histogram of Translation error for PointpartNet+ICP

Figure 11. Registration histogram of PointpartNet+ICP.**Figure 12.** Point cloud registration sample.**Table 4.** Processing time of each process.

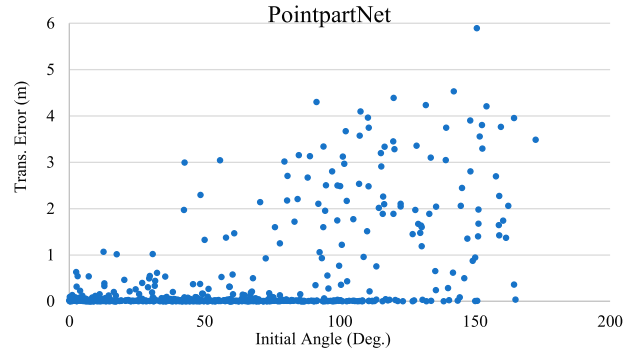
Part-based feature extraction	Matching region search	Global registration	Local registration
0.306 s	0.003 s	0.005 s	0.119 s

full. **full** and **part** have 1024 and 256 points, respectively. We added the same noise to **part** as done in experiments 4.1 and 4.2. The results of experiments are shown in Figures 13 and 14 and Table 5. A sample of the registration is shown in Figure 15.

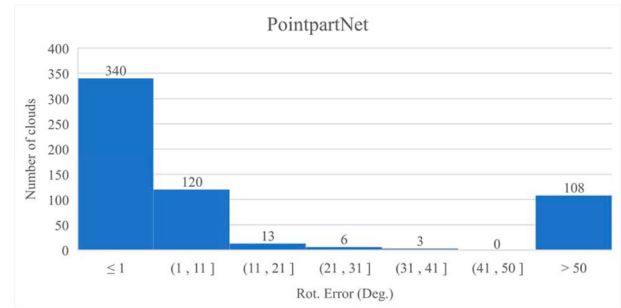
The success rate of the rotation estimation was 77.63%, and the overall average rotation error was 26.95°. And when the rotation estimation is successful, the mean rotation error is 1.13°. The success rate of translation estimation was 72.37%, and the overall mean translation error



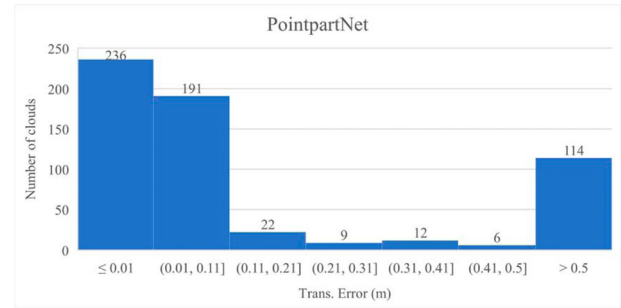
(a) Rotation error of PointpartNet on real data



(b) Translation error of PointpartNet on real data

Figure 13. Registration results of PointpartNet on real data.

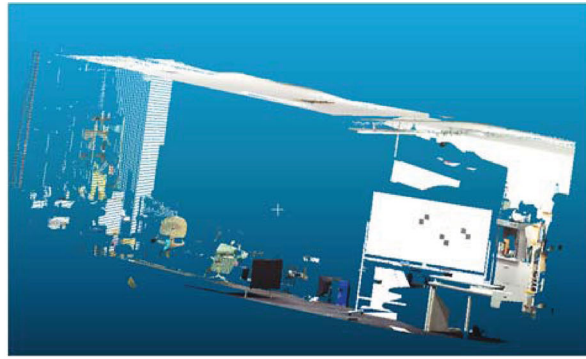
(a) Histogram of Rotation error for PointpartNet on real data



(b) Histogram of Translation error for PointpartNet

Figure 14. Histogram of PointpartNet on real data.

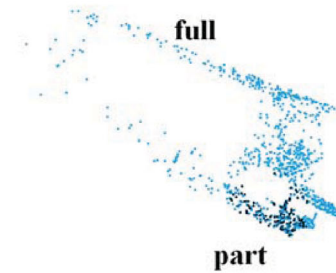
was 0.464 m. When the translation estimation was successful, the average translation error was 0.017 m. Thus, it can be seen that the proposed PointpartNet model



(a) color image of full



(b) before registration



(c) after registration

Figure 15. Point cloud registration sample on real data: (b) shows full and part separated to make it easier to see.**Table 5.** Experimental results for real data.

Real data	Success rate	Mean error	Mean error on success
Rot. (°)	77.63%	26.95	1.13
Trans. (m)	72.37%	0.464	0.017

performed reasonably well even with real-world data collected from a range sensor.

5. Conclusion

In this paper, we have proposed a deep-learning-based registration method for point clouds of different sizes with partial feature extraction. This model extracted point cloud features by partitioning them and searching for a suitable matching region. It succeeded in registering point clouds with different sizes, which was not successfully possible with previous models. Comparison experiments with previous methods proved that PointpartNet is more robust and accurate.

In the future, we aim to improve the accuracy in real environments and verify the usefulness of PointpartNet with different range image sensors. Besides that, we also aim to change the feature extraction to be less affected by initial orientation.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by JSPS KAKENHI [grant number 19H04191].

Notes on contributors

Shixun Yan received his Bachelors' degree from the School of Mechanical and Power Engineering, East China University of Science and Technology, China, in 2017. Following that he received his Masters' degree from the Department of Precision Mechanics, Graduate School of Science and Engineering, Chuo University, Japan, in 2020, and is currently a 3rd-year PhD student at the same. His main research interest is in topics involving point cloud registration and 3D measurement.

Sarthak Pathak received his B.T., and M.T. degrees in the Department of Engineering Design, Indian Institute of Technology Madras, India, in 2014. He received his Ph.D. in the Department of Precision Engineering, the University of Tokyo, Japan, in 2017. After working as a Postdoctoral Research Fellow and Project Assistant Professor at the same, he is currently an Assistant Professor in the Department of Precision Mechanics at Chuo University in Tokyo, Japan. His main research interest is in topics involving robot vision, specifically, localization,

3D reconstruction, and SLAM, especially using 360 degree cameras.

Kazunori Umeda received B.Eng., M.Eng., and Ph.D. degrees in precision machinery engineering from the University of Tokyo, Japan, in 1989, 1991 and 1994, respectively. He became a Lecturer of Precision Mechanics at Chuo University, Japan in 1994, and is currently a Professor since 2006. He was a visiting worker at National Research Council of Canada from 2003 to 2004. His research interests include robot vision, 3D vision, and human interface using vision. He is a member of RSJ, IEEE, JSPE, JSME, SICE, IEICE, etc.

ORCID

Shixun Yan  <http://orcid.org/0000-0002-8495-4754>

Sarthak Pathak  <http://orcid.org/0000-0002-5271-1782>

Kazunori Umeda  <http://orcid.org/0000-0002-4458-4648>

References

- [1] Zhao X. LiDAR-ToF-Binocular depth fusion using gradient priors. 2020 Chinese Control And Decision Conference (CCDC); 2020. p. 2024–2029.
- [2] Hambarde P, Dudhane A, Patil PW, et al. Depth estimation from single image and semantic prior. 2020 IEEE International Conference on Image Processing (ICIP); 2020. p. 1441–1445.
- [3] Qi CR, Su H, Mo K, et al. Pointnet: deep learning on point sets for 3d classification and segmentation. *Proc Comp Vision Pattern Recogn.* 2017; 77–85.
- [4] Li Y, Bu R, Sun M, et al. Pointcnn: convolution on x-transformed points. *Adv Neural Inf Process Syst.* 2018;31:820–830.
- [5] Qi CR, Yi L, Su H, et al. Pointnet++: deep hierarchical feature learning on point sets in a metric space. *Adv Neural Inf Process Syst.* 2017;30:5099–5108.
- [6] Cao H, Zhan R, Ma Y, et al. LFNNet: local rotation invariant coordinate frame for robust pointcloud analysis. *IEEE Signal Process Lett.* 2021;28:209–213.
- [7] Zhang Z, Hua B-S, Rosen DW, et al. Rotation invariant convolutions for 3D pointclouds deep learning. 2019 International Conference on 3D Vision (3DV); 2019. p. 204–213.
- [8] Rao Y, Lu J, Zhou J. Spherical fractal convolutional neural networks for pointcloud recognition. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019. p. 452–460.
- [9] Aoki Y, Goforth H, Srivatsan RA, et al. PointNetLK: Robust & efficient pointcloud registration using PointNet. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019. p. 7156–7165.
- [10] He Y, Lee C-H. An improved ICP registration algorithm by combining PointNet++ and ICP algorithm. 2020 6th International Conference on Control, Automation and Robotics (ICCAR); 2020. p. 741–745.
- [11] Sarode V, Dhagat A, Srivatsan RA, et al. MaskNet: a fully-convolutional network to estimate inlier points. 2020 International Conference on 3D Vision (3DV); 2020. p. 1029–1038.
- [12] Besl PJ, McKay ND. A method for registration of 3-D shapes. *IEEE Trans Pattern Anal Mach Intell.* 1992;14(2):239–256.
- [13] Biber P, Strasser W. The normal distributions transform: a new approach to laser scan matching. *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 3; 2003. p. 2743–2748.
- [14] Golyanik V, Ali SA, Stricker D. Gravitational approach for point set registration. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016. p. 5802–5810.
- [15] Caballo A. Characterization of multiple 3D LiDARs for localization and mapping performance using the NDT algorithm. *IEEE Intelligent Vehicles Symposium*; 2021. p. 327–334.
- [16] Zaganidis A. Semantic-assisted 3D Normal Distributions Transform for scan registration in environments with limited structure. *IEEE International Conference on Intelligent Robots and Systems*; 2017. p. 4064–4069.
- [17] Carballo A, Seiya S, Lambert J, et al. End-to-end autonomous mobile robot navigation with model-based system support. *J Robot Mechatr.* 2018;30(4):563–583.
- [18] Kurobe A, Sekikawa Y, Ishikawa K, et al. Corsnet: 3D Pointcloud registration by deep neural network. *IEEE Robot Automat Lett.* 2020;5(3):3960–3966.
- [19] Wang Y, Zheng Q, Heng PA. Online robust projective dictionary learning: shape modeling for MR-TRUS registration. *IEEE Trans Med Imaging.* 2018;37(4):1067–1078.
- [20] Wu Z, Song S, Khosla A, et al. 3d shapenets: a deep representation for volumetric shapes. *Proceedings of the IEEE conference on computer vision and pattern recognition*; 2015. p. 1912–1920.