electronics

MDPI

Article

Package Positioning Based on Point Registration Network DCDNet-Att

Juan Zhu [1,*], Chunrui Yang [2], Guolyu Zhu [1], Xiaofeng Yue [1] and Qingming Zhao [3]

[1] School of Mechanical and Electrical Engineering, Changchun University of Technology, Changchun 130012, China; 2202201114@stu.ccut.edu.cn (G.Z.); yuexf@ccut.edu.cn (X.Y.)
[2] School of Mechanical and Electrical Engineering, Changchun University of Science and Technology, Changchun 130022, China; yyychunrui@163.com
[3] Jilin Jibang Automation Technology Co., Ltd., Changchun 130031, China; zhaoqingming@jb963.com
* Correspondence: zhujuan@ccut.edu.cn

Abstract: The application of robot technology in the automatic transportation process of packaging bags is becoming increasingly common. Point cloud registration is the key to applying industrial robots to automatic transportation systems. However, current point cloud registration models cannot effectively solve the registration of deformed targets like packaging bags. In this study, a new point cloud registration network, DCDNet-Att, is proposed, which uses a variable weight dynamic graph convolution module to extract point cloud features. A feature interaction module is used to extract common features between the source point cloud and the template point cloud. The same geometric features between the two pairs of point clouds are strengthened through a bottleneck module. A channel attention model is used to obtain the channel attention weights. The attention weight of each spatial position is calculated, and a rotation translation structure is used to sequentially obtain quaternions and translation vectors. A feature fitting loss function is used to constrain the parameters of the neural network model to have a larger receptive field. Compared with seven methods, including the ICP algorithm, GO-ICP algorithm, and FGR algorithm, the proposed method had rotation errors (MAE, RMSE, and Error of 1.458, 2.541, and 1.024 in the ModelNet40 dataset, respectively) and translation errors (MAE, RMSE, and Error of 0.0048, 0.0114, and 0.0174, respectively). When registering the ModelNet40 dataset with Gaussian noise, the rotation errors (MAE, RMSE, and Error) were 2.028, 3.437, and 2.478, respectively, and the translation errors (MAE, RMSE, and Error) were 0.0107, 0.0327, and 0.0285, respectively. The experimental results were superior to those of the other methods, and the model was effective at registering packaging bag point clouds.

Keywords: point cloud registration; deep learning; variable weight dynamic graph convolution; feature extraction

Academic Editor: Taeshik Shon

Received: 20 December 2024
Revised: 14 January 2025
Accepted: 16 January 2025
Published: 17 January 2025

Citation: Zhu, J.; Yang, C.; Zhu, G.; Yue, X.; Zhao, Q. Package Positioning Based on Point Registration Network DCDNet-Att. *Electronics* **2025**, *14*, 352. https://doi.org/10.3390/electronics14020352

1. Introduction

With the rapid development of industrial robot technology, many tasks which require traditional manual labor are gradually being replaced and completed by industrial robots. Packaging bags are widely used in various industries, and the transportation of packaging bags consumes large amounts of manpower and resources. It is particularly important to apply industrial robots to the automatic transportation systems of packaging bags [1]. The automatic positioning of packaging bags requires the use of a 3D point cloud camera to extract the point cloud data of the packaging bags, identify the rotation and translation

matrices of the targets in the point cloud relative to the template, and send the positioning results to the robot through coordinate transformation to achieve grasping. Therefore, point cloud registration technology is the key to automatic positioning of packaging bags.

PointNet is a widely used deep learning network model for processing point cloud data. It introduces symmetric functions to achieve permutation invariance for point clouds and uses T-Net to achieve rotation invariance for point clouds. There is a series of variant models based on PointNet, such as PointNet++ and PointCNN [2–5]. In 2018, Wang et al. proposed dynamic graph convolution (DGCNN), which enhances the representation ability of node features by considering the neighboring node features and edge features of nodes and rebuilding the graph structure in each iteration. In the same year, Wang et al. proposed deep closest point (DCP) [6–8], which uses DGCNN to extract the local features of point clouds and iterates on point clouds according to the calculation process of the iterative closest point (ICP) algorithm. In the pose regression stage, DCP uses a probability function to estimate the rotation matrix. The disadvantage of DCP is that it cannot handle non-overlapping points. In 2019, Wang et al. proposed Partial-to-Partial Registration Net (PRNet) [9], which added the Sinkhorn–Gumbel softmax algorithm to the DCP network and used self-supervised learning to directly extract geometric features from point clouds. PRNet uses a partial-to-partial approach to solve the problem of non-overlapping point clouds and successfully validates the rationality of applying deep learning to partial matching. In 2019, Sarode et al. proposed the Point Cloud Registration Network (PCRNet), which treats PointNet as a feature extractor and utilizes it to extract global features from source and target point clouds. After extracting the global features through PointNet, PCRNet directly fuses these features and generates a feature vector through a fully connected layer. The characteristic of PCRNet is that it achieves fast point cloud registration through this simple and efficient method, but it cannot handle non-overlapping point clouds [10,11]. In 2020, Yew et al. proposed the Robust Point Matching Network (RPMNet), which uses differentiable Sinkhorn normalization and annealing methods to enforce double random constraints. The model can effectively handle outliers [12,13]. In 2020, Yuan et al. proposed Deep Gaussian Mixture Registration (DeepGMR), based on the Gaussian mixture model. DeepGMR utilizes the pose-invariant correspondence between the original point cloud and the Gaussian mixture model (GMM) parameters to model point cloud registration as optimizing the KL divergence between two Gaussian mixture models [14–16]. In 2021, Zeng et al. proposed CorrNet3D. CorrNet3D uses DGCNN to extract point clouds and then max pooling to obtain global features. By approximating a doubly random matrix, it reorders the points in the source point cloud and finds their corresponding points in the template point cloud to achieve registration of non-rigid point clouds [17–19].

This study designs a point cloud registration network based on deep learning to solve the registration problem of packaging bag point clouds. The main contributions of this article are as follows:

(1) We improve the dynamic graph convolution module by introducing weight coefficients and enhancing global features such that the dynamic graph convolution module based on corresponding point matching can be applied in point cloud registration networks based on global features. We propose a point cloud feature interaction module to enable the network model to handle the registration problem of non-overlapping point clouds. A rotation translation separation structure is proposed which completely separates the calculation of rotation parameters and translation parameters which do not belong to the same vector space, making the point cloud registration network modular.

(2) In the feature fusion stage between the source point cloud and the template point cloud, the channel attention module is used to dynamically adjust the weights of

different feature dimensions of the point cloud for the model to better utilize the most useful features among them. Spatial attention is used to help the model focus on the overlapping area between two clouds, improving the efficiency and registration accuracy of the model.

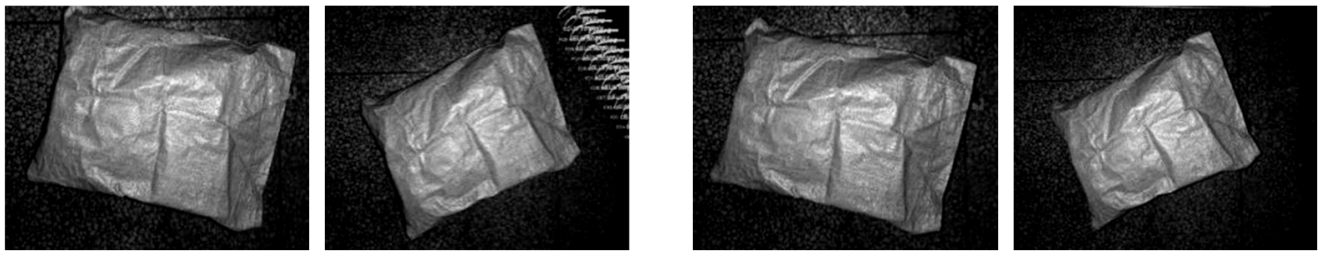
2. Construction of Packaging Bag Point Cloud Dataset and Preprocessing

2.1. Point Cloud Collection

To construct a packaging bag point cloud dataset, a point cloud data acquisition platform for packaging bags based on the Cognex A5120 (Cognex Vision Inspection System (Shanghai) Co., Ltd, Shanghai, China) structured light camera was built, as shown in Figure 1.

Figure 1. Point cloud data collection platform for packaging bags.

Packaging bags were placed in different positions and angles, and data were collected from the packaging bags in different poses. The collected data included left camera image data and right camera image data. Based on the left and right camera image data and camera parameters, point cloud data could be calculated. Figure 2 shows the collected left and right camera images.



(**a**) Left camera image data (**b**) Right camera image data

Figure 2. Image data.

Figure 3 shows the obtained point cloud data, where blue is the reference plane of the point cloud plane. The distance between the point clouds was consistent with the spectral color arrangement order; that is, the point clouds farther away from the blue plane tended to be closer to red.

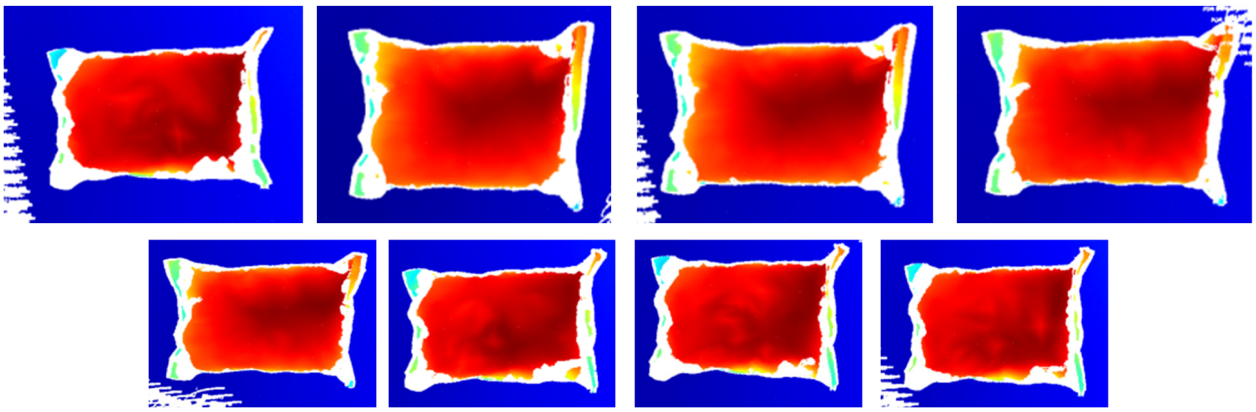


Figure 3. Point cloud of packaging bags.

2.2. Point Cloud Segmentation Based on RANSAC Algorithm

The Random Sampling Consistency (RANSAC) algorithm [20] is a classic point cloud processing algorithm widely used in the fields of point cloud segmentation and registration.

To remove the background, we used the RANSAC algorithm to segment the points from the background. The RANSAC algorithm estimates a mathematical model iteratively from a set of data containing a background. This algorithm divides point cloud data containing noise into inliers and outliers. Here, inliers are real points which exist in the target object, and outliers are the background.

2.3. Statistical Filtering to Remove Outliers

The statistical filter calculates the mean and standard deviation of the distance between each point in the point cloud and determines whether it is an outlier based on the standard deviation. Assuming that the point cloud data are $P = \{p_i | i = 0, 1, 2, 3, \ldots, n\} \subset R^3$, a point p_i is randomly selected. Suppose that the set of neighboring points of p_i is N, where $N = \{p_{i1}, p_{i2}, \cdots, p_{ik}\}$. We calculated the distance from p_i to each point in the set N and obtain the average distance $\overline{d}_i$.

Then, we calculated the distance mean μ and distance standard deviation σ of the entire point cloud using the following formula:

$$\mu = \frac{1}{mk}\sum_{i=1}^{m}\sum_{j=1}^{k} d_{ij}, \sigma = \sqrt{\frac{1}{mk}\sum_{i=1}^{m}\sum_{j=1}^{k}(d_{ij} - \mu)^2} \quad (1)$$

We then calculated the mean and standard deviation of the points within the point cloud and considered them outliers if they were not within the confidence interval 3σ.

The advantage of the statistical removal method is that it can accurately identify outliers through statistical features and does not require excessive parameter adjustment. However, this method is sensitive to the distribution and statistical characteristics of point cloud data and may be affected by data noise and the sampling density. Therefore, in practical applications, it is necessary to carefully select and adjust relevant parameters to achieve better denoising results [21].

2.4. Farthest Point Cloud Downsampling

Farthest point downsampling is based on the farthest point sampling method of point clouds, which is a commonly used sampling method for point clouds. Assuming that the point cloud data are $P = \{p_i | i = 0, 1, 2, 3, \ldots, n\} \subset R^3$, one can define an initial set $N = \varnothing$. We selected k points from the point cloud and then randomly selected a point p_0 from the point cloud as the initial point and put it into the set N. We then chose the point p_a which

was farthest from p_0 for the set N. At this time, the set was $N = \{p_0, p_a\}$, and we calculated the distances between the remaining points and p_0 and p_a separately. Suppose that the farthest point from p_0 is p_{c1} and the distance is d_1. The farthest point from p_a is p_{c2}, with the distance denoted by d_2. We compared d_1 with d_2. If $d_1 > d_2$, then p_{c1} would be placed in the set N. Otherwise, p_{c2} would be placed in the set N. We repeated this process until the required number of selected points was reached.

The advantage of farthest point downsampling is that it can preserve the main features of the point cloud without the need for pre-set parameters. In addition, this method performs relatively efficiently in processing large-scale point clouds [22].

The preprocessing of the packaging bag point cloud is shown in Figure 4.

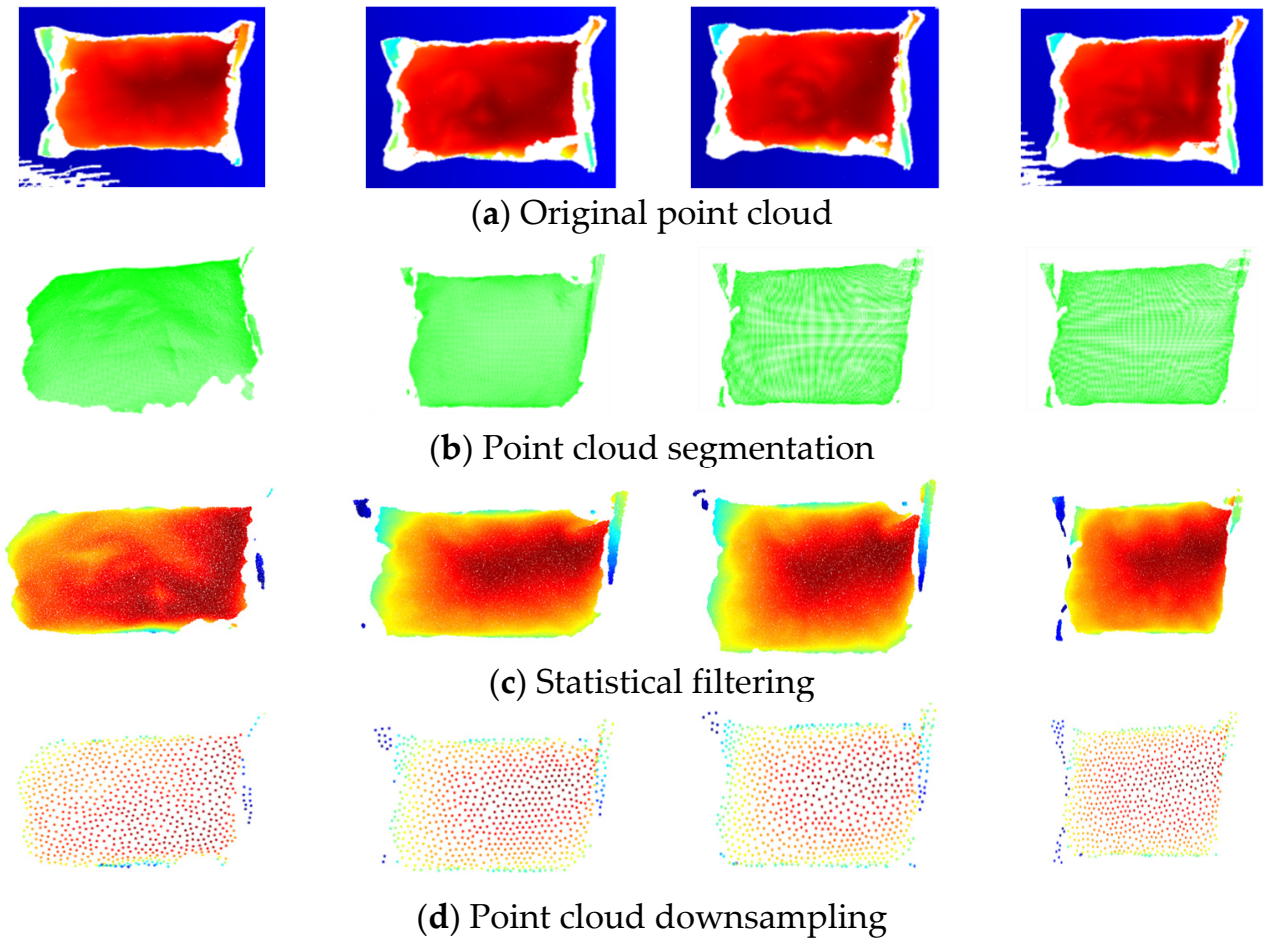


Figure 4. Preprocessing of packaging bag point cloud.

3. Methodology

The DCDNet-Att model structure is shown in Figure 5. The green, solid-lined blocks represent the rotate edgeConv block. The yellow, solid-lined blocks represent the translation edgeConv block. The gray cylinder represents the max pool function, $\oplus$ represents tensor concatenation, and $\otimes$ represents tensor multiplication. The green dashed box represents the rotate fully connected module. The yellow dashed box represents the translation fully connected module. The solid arrow represents the source path for point cloud data transfer. The dashed arrow represents the reference path for template point cloud data transfer, x represents the source point cloud, and y represents the template point cloud.

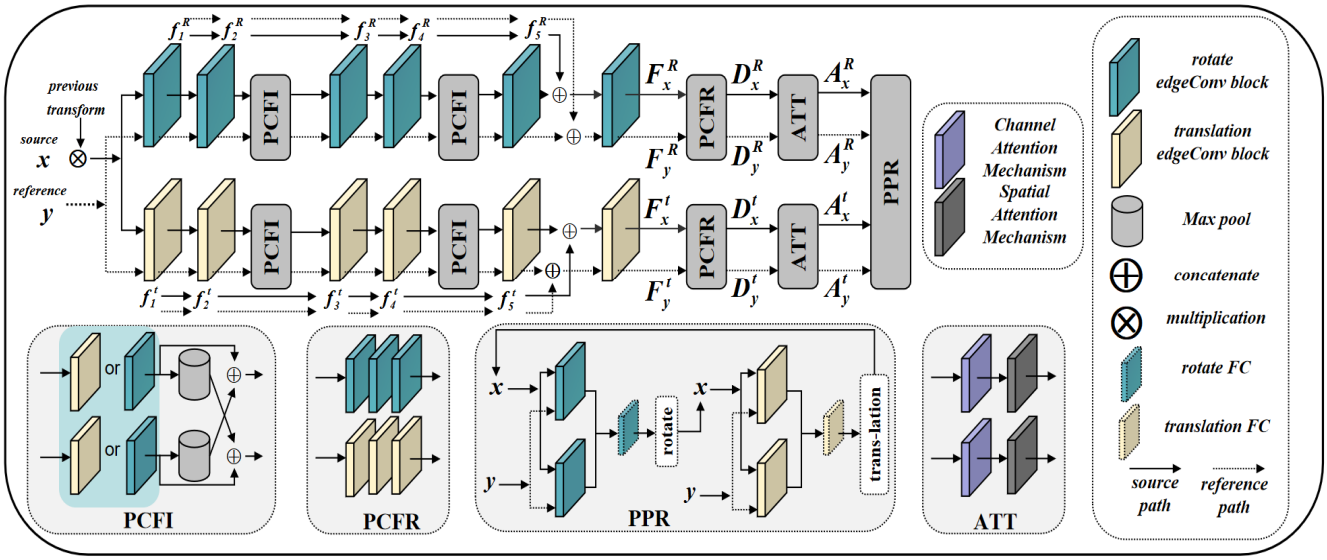


Figure 5. DCDNet-Att model structure.

Both the x and y point clouds were simultaneously input into the variable weight dynamic graph convolution module which did not share parameters. Because the quaternion and the translation vector do not belong to the same vector space, the proposed network uses a point cloud feature extraction module which does not share parameters.

After two rounds of rotation, for the x point cloud and y point cloud, rotated point cloud features $f_2^R x$ and $f_2^R y$ were obtained. The rotated point cloud features were transmitted to the point cloud feature interaction (PCFI) module for feature interaction. The point cloud features from the first feature interaction step were processed by feature extraction, feature interaction, and feature extraction. The point cloud features obtained are named $f_5^R x$ and $f_5^R y$. One can find f_x^R by performing tensor addition on five sets—$f_1^R x$, $f_2^R x$, $f_3^R x$, $f_4^R x$, and $f_5^R x$—and obtain f_y^R by performing tensor addition on five sets: $f_1^R y$, $f_2^R y$, $f_3^R y$, $f_4^R y$, and $f_5^R y$. The translation vector extracts point cloud features from f_x^R and f_y^R through the rotation variable weight dynamic graph convolution module to obtain F_x^R and F_y^R, respectively. It then combines F_x^R and F_y^R into a pair of datasets and uses the point cloud feature rise (PCFR) module. After processing by the PCFR module, D_x^R and D_y^R are obtained. By incorporating a channel attention mechanism and spatial attention mechanism, A_x^R and A_y^R are obtained. Tensor addition of A_x^R and A_y^R is performed to obtain the fusion features F_R of point clouds x and y. Finally, F_R is subjected to pose regression through a fully connected module to obtain the quaternion. Figure 5 shows the point cloud pose regression (PPR) module. After multiplying the quaternion q by point cloud x, the x and y point clouds are input again into the translation feature extraction module, and all of the above operations are repeated to obtain the translation vector. At this point, the point cloud alignment process ends. The translation vector obtained in the previous round is added to the point cloud, and all of the above processes are repeated until the iteration ends.

The calculation process of DCDNet-Att can be expressed as follows:

$$I_R^1 x, I_R^1 y = I\left(E_R\left(E_R\left(con(x + t^{-1}, y)\right)\right)\right) \tag{2}$$

$$I_R^2 x, I_R^2 y = I\left(E_R\left(E_R\left(con(I_R^1 x, I_R^1 y)\right)\right)\right) \tag{3}$$

$$D_x^R, D_y^R = R_R\left(E_R\left(con(I_R^2 x, I_R^2 y)\right)\right) \tag{4}$$

$$A_x^R, A_y^R = S_R\left(C_R\left(D_x^R, D_y^R\right)\right) \tag{5}$$

$$q = FC_R\left(con\left(A_x^R, A_y^R\right)\right) \tag{6}$$

$$I_t^1 x, I_t^1 y = I(E_t(E_t(con(q \cdot x, y)))) \tag{7}$$

$$I_t^2 x, I_t^2 y = I\left(E_t\left(E_t\left(con(I_t^1 x, I_t^1 y)\right)\right)\right) \tag{8}$$

$$D_x^t, D_y^t = R_t\left(E_t\left(con(I_t^2 x, I_t^2 y)\right)\right) \tag{9}$$

$$A_x^t, A_y^t = S_t\left(C_t\left(D_x^t, D_y^t\right)\right) \tag{10}$$

$$t = FC_t\left(con\left(A_x^t, A_y^t\right)\right) \tag{11}$$

where *con* represents the tensor concatenation, E_R represents the rotation feature extraction module, E_t represents the translation feature extraction module, I represents the point cloud feature interaction module, R_R represents the rotation bottleneck module, R_t represents the rotation bottleneck module, FC_R represents the rotation feature regression function, FC_t represents the translation feature regression function, which is a fully connected neural network, q represents quaternions, t represents the translation vector obtained in this round of the registration process, t^{-1} represents the translation vector obtained in the previous round of the registration process, t is zero, $S(\cdot)$ represents the spatial attention mechanism, and $C(\cdot)$ represents the channel attention mechanism.

The attention mechanism enables the model to dynamically focus its attention on different parts of the input data, thereby improving the performance and performance of the model.

3.1. PCFI, PCFR, and PPR Modules

The PCFI module encourages the model to focus on the same geometric features between the x and y point clouds by interacting with their global and local features. These features include the overall shape of the packaging bag point cloud and the outer surface, which has not undergone deformation.

The PCFR module consists of three conventional convolutional layers, with the convolutional layer dimensions being 1024, 2048, and 1024. The green and yellow parts in Figure 5 represent one-dimensional convolution modules, which are combined into a bottleneck module. The bottleneck module is the post-processing process of point clouds. After the point cloud is processed through the variable weight dynamic graph convolution module and feature interaction module, the model has a preliminary ability to focus on the overlapping areas of the point cloud. Through the bottleneck module, the point cloud is processed to enhance the overlapping point cloud features and weaken the non-overlapping point cloud features.

The network first calculates the rotation features of the point cloud and regresses the rotation features to obtain quaternions. The quaternions are multiplied by the source point cloud to obtain a preliminary transformed source point cloud. The transformed source and target point clouds are then input back into the neural network to extract translation features. Finally, the translation vector is obtained by regressing the translation features.

3.2. Channel Attention

The role of the channel attention mechanism is to fully integrate the source point cloud features with the template point cloud features. A channel attention mechanism is an attention mechanism used to handle the relationships between data channels. In deep learning, each channel of tensor data represents specific features or information, but not

all channels are equally important to the final task. A channel attention mechanism aims to dynamically adjust the weights of these channels such that the model can better utilize the most useful information, and by learning the weights of each channel, the model can automatically focus on the channels which are more meaningful to the task [23,24]:

(1) Calculating weights: Unlike spatial attention mechanisms, channel attention mechanisms do not require convolution processing of point cloud features; rather, they directly use channel attention functions to process point cloud features to obtain channel attention weights.
(2) Channel weighted summary: Based on the calculated channel attention weights, the point cloud features are weighted and summarized to obtain the final point cloud features.

The channel attention mechanism can be expressed as follows:

$$\theta_{ij} = \frac{\exp\left(\left(F^{B\times N\times C}\right)_i{}^T \otimes \left(F^{B\times N\times C}\right)_j\right)}{\sum_i^j \sum_i^j \exp\left(\left(F^{B\times N\times C}\right)_i{}^T \otimes \left(F^{B\times N\times C}\right)_j\right)} \tag{12}$$

$$C_j{}^{B\times N\times C} = \omega \sum_{i=1}^{j}\left(\theta_{\mathrm{ij}} \cdot \left(F^{B\times N\times C}\right)_{\mathrm{i}}\right) + \left(F^{B\times N\times C}\right)_j \tag{13}$$

where $F^{B\times N\times C}$ presents the point cloud features of dimensions $B \times N \times C$. Suppose that j is greater than i. Then, θ_{ij} is the channel attention coefficient between position i and j. Meanwhile, $C_j^{B\times N\times C}$ is the feature of the point cloud at position j. The initial value of ω is zero, and ω will be updated in the training process of the attention mechanism.

3.3. Spatial Attention Mechanism

In the process of point cloud processing, the spatial attention mechanism focuses not only on the geometric features of the input point cloud but also on the positional relationship of the point cloud in space. A spatial attention mechanism can make the model no longer simply extract features from point cloud data but help the model focus on information in certain specific areas, namely the overlapping areas between two point clouds, which can improve the efficiency and registration accuracy of the model [25].

The calculation process of the spatial attention mechanism used in this article is as follows:

(1) Extracting point cloud features: It extracts point cloud features through two-dimensional convolution and performs average pooling or global pooling on the point cloud features,
(2) Encoding point cloud spatial position information: It encodes the spatial position information of the pooled point cloud features so that the model can consider the positional relationships between features,
(3) Calculate weight: It calculates the attention weight of each spatial position to reflect its importance.
(4) Weighted summary: Based on the calculated spatial attention weights, the point cloud features are weighted and summarized to obtain the final point cloud features.

The spatial attention mechanism can be expressed as follows:

$$F_1^{B\times N\times C} = Conv_1\left(F^{B\times N\times C}\right) \tag{14}$$

$$F_2^{B\times N\times C} = Conv_2\left(F^{B\times N\times C}\right) \tag{15}$$

$$F_3^{B\times N\times C} = Conv_3\left(F^{B\times N\times C}\right) \tag{16}$$

$$\alpha_{ij} = \frac{\exp\left(\left(F_1^{B\times N\times C}\right)_i^T \otimes \left(F_2^{B\times N\times C}\right)_i\right)}{\sum_i^j \sum_i^j \exp\left(\left(F_1^{B\times N\times C}\right)_i^T \otimes \left(F_2^{B\times N\times C}\right)_i\right)} \tag{17}$$

$$S_j^{B\times N\times C} = \beta \sum_{i=1}^{j}\left(\alpha_{ij} \cdot \left(F_3^{B\times N\times C}\right)_i\right) + \left(F_1^{B\times N\times C}\right)_j \tag{18}$$

where $F^{B\times N\times C}$, $F_1^{B\times N\times C}$, $F_2^{B\times N\times C}$, and $F_3^{B\times N\times C}$ present the point cloud features of dimensions $B \times N \times C$, $Conv_1$, $Conv_2$, and $Conv_3$ present a two-dimensional convolution group, α_{ij} is the spatial attention coefficient between positions i and j, $S_j^{B\times N\times C}$ is the feature of the point cloud at position j, and β is a feature scaling factor. The initial value of β is zero, and β will be updated as the training process of attention mechanism is carried out.

4. Evaluation

The evaluation metrics used in this article include the root mean square error (RMSE), mean absolute error (MAE), and anisotropy error (Error), which can be expressed using the following formulae:

$$RMSE(R) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left\|[\theta_R, \lambda_R, \omega_R] - [\theta_T, \lambda_T, \omega_T]\right\|_2} \tag{19}$$

$$MAE(R) = \frac{1}{n}\sum_{i=1}^{n}\left|[\theta_R, \lambda_R, \omega_R] - [\theta_T, \lambda_T, \omega_T]\right| \tag{20}$$

$$Error(R) = \frac{1}{n}\sum_{i=1}^{n}\arccos\left(\frac{tr\left(Ra_R Ra_T{}^{-1}\right) - 1}{2}\right) \tag{21}$$

$$RMSE(t) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left\|[x_R, y_R, z_R] - [x_T, y_T, z_T]\right\|_2} \tag{22}$$

$$MAE(t) = \frac{1}{n}\sum_{i=1}^{n}\left|[x_R, y_R, z_R] - [x_T, y_T, z_T]\right| \tag{23}$$

$$Error(t) = \frac{1}{n}\sum_{i=1}^{n}\arccos\left(\frac{tr\left(ta_R ta_T{}^{-1}\right) - 1}{2}\right) \tag{24}$$

where θ, λ, and ω are the rotation angle of the point cloud around the X, Y, and Z axes, respectively, R represents the predicted value, T is the real value, and $tr(\cdot)$ represents the trace of the matrix.

5. Discussion

The training and optimization of DCDNet-Att were carried out on the AutoDL cloud GPU platform, with the deep learning model running on Ubuntu 22.04 and the deep learning framework being PyTorch 1.13.0+cu113. The CPU used for training the model was a 15 vCPU Intel (R) Xeon (R) Platinum 8358P CPU @ 2.60 GHz, and the GPU was an NVDIA A100-SXM4-80GB (80 GB). For the network model parameter settings, the initial learning rate was 0.0001, and the learning rate was multiplied by 0.1 every 50 epochs. The total number of iterations for the entire dataset was 300 epochs, the batch size was 64, and the random seed size was 1234. The training set had 7500 samples, the validation set had 1000 samples, and the test set had 1000 samples.

5.1. Ablation Experiment

In order to verify the effectiveness of the attention mechanism, this section conducted ablation experiments on DCDNet-Att and calculated the registration error between not adding an attention mechanism, adding a certain attention mechanism separately, and adding different attention mechanisms in different orders. The results are shown in Table 1.

Table 1. The impacts of the different attention mechanisms on DCDNet-Att.

Attention Mechanism	MAE (R)	RMSE (R)	Error (R)	MAE (t)	RMSE (t)	Error (t)
☒	1.738	3.516	2.115	0.0108	0.0297	0.0256
S	1.642	3.041	1.847	0.0091	0.0137	0.0201
C	1.678	2.889	1.485	0.0088	0.0175	0.0247
S + C	1.497	2.574	1.034	0.0057	0.0124	0.0187
C + S	**1.458**	**2.541**	**1.024**	**0.0048**	**0.0114**	**0.0174**

In Table 1, ☒ represents the basic network without adding an attention mechanism, S represents adding only a spatial attention mechanism in the feature fusion stage, C represents adding only a channel attention mechanism in the feature fusion stage, and S + C represents adding both a spatial attention mechanism and channel attention mechanism in the feature fusion stage, with the spatial attention mechanism at the front and the channel attention mechanism at the back. Similarly, C + S represents a channel attention mechanism at the front and a spatial attention mechanism at the back. From the table, it can be seen that compared with not adding an attention mechanism, any attention mechanism among S, C, S + C, and C + S can reduce the registration error of the model, proving that attention mechanisms significantly improved the network. The combination of channel attention and spatial attention mechanisms had the greatest improvement effect on the model.

5.2. Generalization Experiment

In order to test the generalization ability of DCDNet-Att, this section compares DCDNet-Att with other algorithm models using a dataset with unseen categories. Table 2 shows a comparison of the generalization experimental results based on the data in this article.

Table 2. Comparison of generalization experiment results for DCDNet-Att.

Model	MAE (R)	RMSE (R)	Error (R)	MAE (t)	RMSE (t)	Error (t)
ICP	22.821	11.217	22.987	0.1207	0.2356	0.2974
GO-ICP	71.395	10.572	73.734	0.1997	0.3974	0.4215
FGR	48.672	28.837	57.159	0.2077	0.3602	0.4873
PointNetLK	29.148	19.901	31.823	0.1874	0.2985	0.5144
DCP	15.523	10.305	8.967	0.0842	0.0803	0.1843
FMR	12.716	66.228	11.421	0.0905	0.1711	0.1941
DeepGMR	68.217	45.654	54.638	0.2158	0.3574	0.4132
DCDNet-Att	**1.575**	**2.798**	**1.501**	**0.0068**	**0.0151**	**0.0199**

5.3. Comparison Experiment with Different Iteration Times

The impact of different iterations on the registration error of DCDNet-Att is shown in Table 3 and Figure 6, which show the rotation MAE, RMSE, and Error data and a line graph for seven iterations, respectively.

The X axis represents the number of iterations. From the chart, it can be seen that for DCDNet-Att, after six iterations, the rotation MAE, RMSE, and Error values for point cloud registration were 1.458, 2.541, and 1.024, respectively, which were small and tended to stabilize.

Table 3. Comparison of rotation errors of DCDNet-Att with different iterations.

Times	MAE	RMSE	Error
1	7.542	8.013	8.867
2	5.278	6.475	5.012
3	3.741	4.742	3.984
4	2.795	3.582	2.011
5	1.871	3.047	1.523
6	**1.458**	**2.541**	**1.024**
7	1.461	2.616	1.027

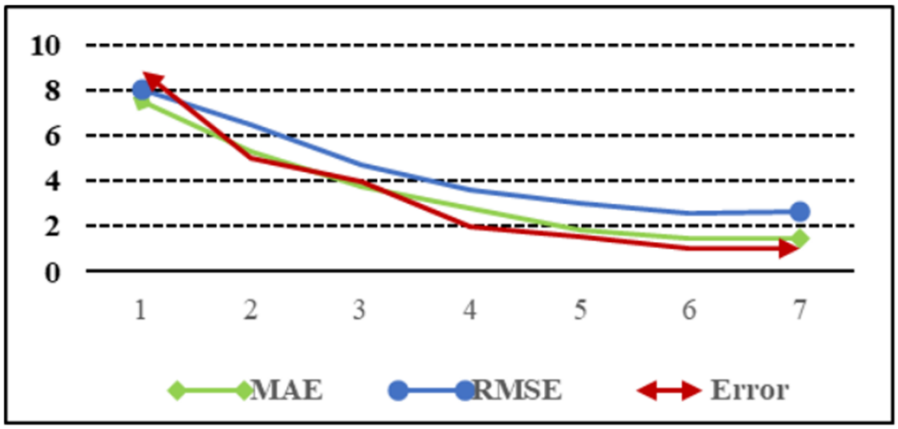


Figure 6. DCDNet-Att rotation error line chart with different iterations.

Table 4 and Figure 7 show the MAE, RMSE, and Error data and a line graph of the translation errors corresponding to the first seven iterations, respectively.

Table 4. Comparison of translation errors of DCDNet-Att with different iterations.

Times	MAE	RMSE	Error
1	0.0891	0.0805	0.1051
2	0.0758	0.0637	0.0714
3	0.0549	0.0455	0.0413
4	0.0247	0.0197	0.0183
5	0.0114	0.0145	0.0189
6	**0.0048**	**0.0114**	**0.0174**
7	0.0047	0.0121	0.0179

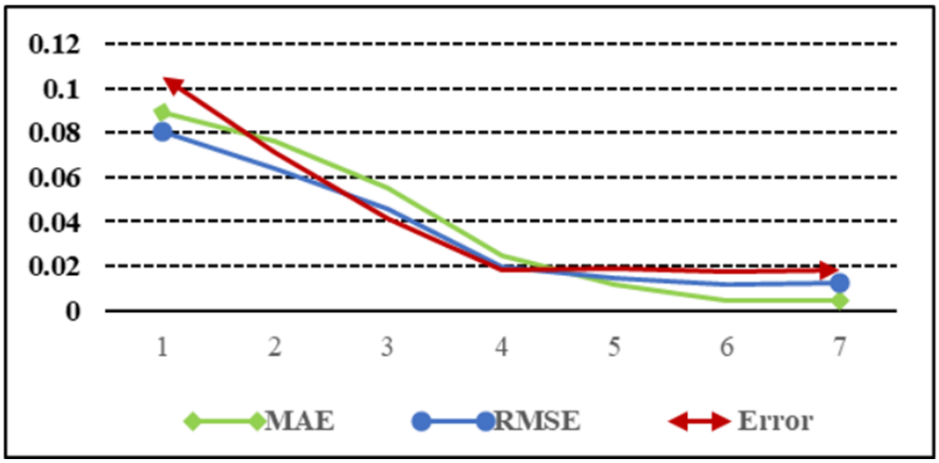


Figure 7. DCDNet-Att translation error line chart with different iterations.

The X axis represents the number of iterations. From the chart, it can be seen that for DCDNet-Att, after six iterations, the translation MAE, RMSE, and Error values of the point cloud registration were 0.0048, 0.0114, and 0.0174, respectively, and tended to stabilize. Therefore, for DCDNet-Att, six iterations could obtain the optimal results. Fewer iterations resulted in low network registration accuracy, while more iterations increased the computational complexity of the network and could not better improve the model's registration accuracy.

5.4. Registration Results for Noise-Free Point Clouds

The ModelNet40 [26] dataset was registered using the DCDNet-Att network, and some of the registration results are shown in Figure 8.

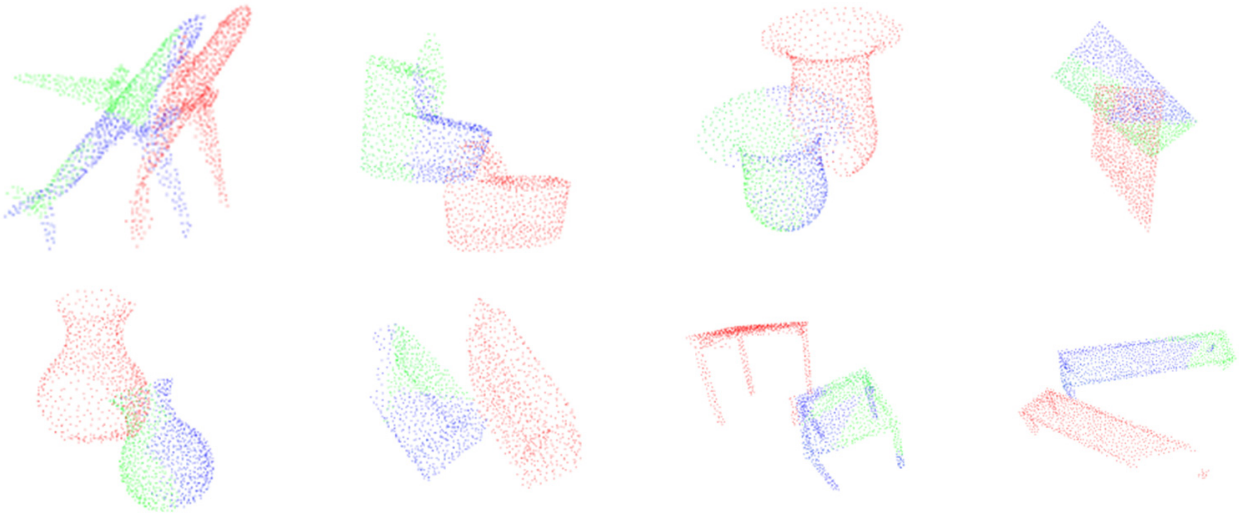


Figure 8. DCDNet-Att point cloud registration results of noiseless ModelNet40 dataset.

The red point cloud in Figure 8 is the template point cloud, the green point cloud is the source point cloud, and the blue point cloud is the registered template point cloud. It can be seen from the figure that DCDNet-Att had a good registration effect on the noise-free, partially overlapping point clouds.

We compared the registration performance of DCDNet-Att based on the data in this article with eight methods, including ICP, GO-ICP, and FGR. The results are shown in Table 5.

Table 5. Comparison of ModelNet40 point cloud registration error results.

Model	MAE (R)	RMSE (R)	Error (R)	MAE (t)	RMSE (t)	Error (t)
ICP	20.387	12.651	22.232	0.1191	0.1893	0.2597
GO-ICP	69.747	39.646	71.462	0.1807	0.3111	0.3996
FGR	46.161	27.475	55.685	0.1965	0.292	0.4068
PointNetLK	27.903	18.661	29.374	0.1623	0.2368	0.3454
DCP	13.387	9.971	7.835	0.0524	0.0695	0.1049
FMR	10.365	64.465	9.741	0.0726	0.1208	0.1634
DeepGMR	66.122	43.499	52.736	0.1917	0.2772	0.3986
DCDNet-ATT	**1.458**	**2.541**	**1.024**	**0.0048**	**0.0114**	**0.0174**

From the chart, it can be seen that compared with other methods, DCDNet-Att had significant advantages in both rotation accuracy and translation accuracy. DCDNet-Att performed well in the noise-free point cloud registration tasks.

5.5. Registration Results for Point Clouds with Gaussian Noise

To verify whether DCDNet-Att still had good robustness to noise during point cloud registration, Gaussian noise with a mean of 0.5 and a standard deviation of 0.01 was added to the ModelNet40 dataset for registration using this network. The partial registration results are shown in Figure 9.

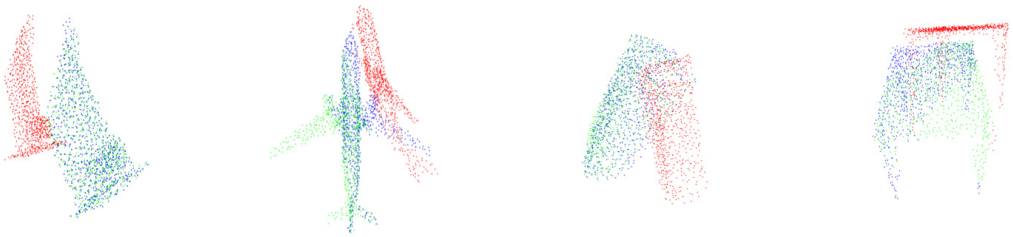


Figure 9. DCDNet-Att for Gaussian point cloud registration results.

In Figure 9, the red point cloud represents the number of template point clouds, the green point cloud represents the source point cloud, and the blue point cloud represents the registered template point cloud data. From the registration result graph, it can be seen that DCDNet-Att had good registration results for the Gaussian noise point clouds, with a mean of 0.5 and a standard deviation of 0.01.

Table 6 shows a comparison of the results for the registration errors between DCDNet-Att and seven algorithms, including ICP in the ModelNet40 point cloud dataset with Gaussian noise.

Table 6. DCDNet-Att Gaussian noise point cloud registration error results.

Model	MAE (R)	RMSE (R)	Error (R)	MAE (t)	RMSE (t)	Error (t)
ICP	22.724	14.462	23.132	0.2085	0.2576	0.3148
GO-ICP	8.385	42.738	74.814	0.2210	0.347	0.4951
FGR	49.913	30.879	58.952	0.2493	0.3521	0.6420
PointNetLK	30.207	20.306	31.440	0.2764	0.2952	0.6001
DCP	16.549	11.248	10.601	0.1024	0.0923	0.2459
FMR	11.661	67.573	11.795	0.1178	0.1715	0.2994
DeepGMR	68.127	45.912	56.978	0.2678	0.3568	0.5139
DCDNet-ATT	**2.028**	**3.437**	**2.478**	**0.0107**	**0.0327**	**0.0285**

From the chart, it can be seen that compared with the other seven algorithms, DCDNet-Att had rotation MAE, RMSE, and Error values of 2.028, 3.437, and 2.478, respectively, and translation values of 0.0107, 0.0327, and 0.0285, respectively, all of which were superior to the other algorithms. It can be seen that DCDNet-Att has robust anti-noise performance, and the registration rotation accuracy and translation accuracy were the best.

5.6. Registration Results of Packaging Bags

To verify the registration effect of DCDNet-Att on packaging bags, eight different packaging bag templates were used for the packaging bag point cloud registration experiments. The registration results are shown in Figure 10.

The red point cloud is the template point cloud, the blue point cloud is the point cloud to be registered, and the green point cloud is the registered point cloud data. From the registration results, it can be seen that although there was deformation in the point cloud of the packaging bags, which resulted in incomplete overlap between the target point cloud and the template point cloud, the network fully utilized the global and local features of the packaging bag for registration, and the registration results were good. Using registration data to express the position of the target can help accurately locate the packaging bag.

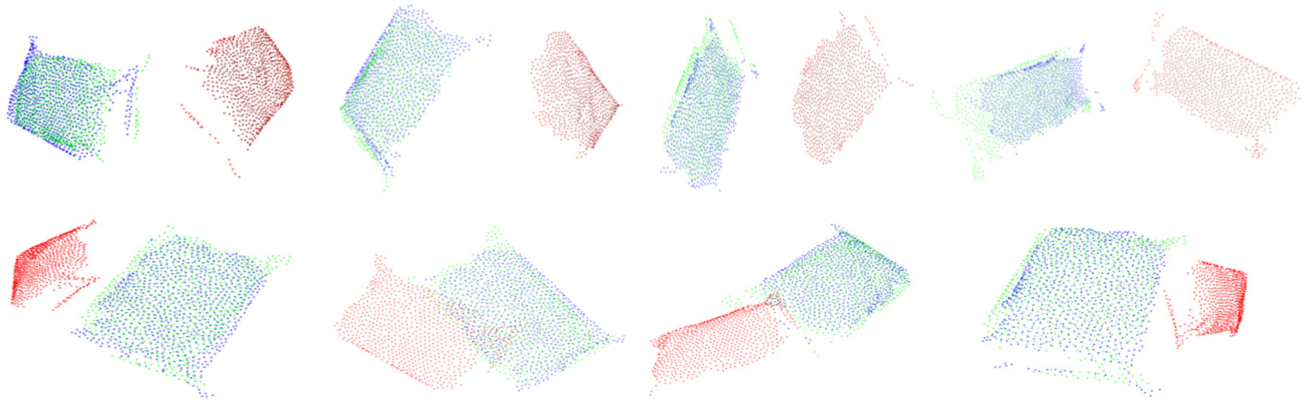


Figure 10. Schematic diagram of point cloud registration for packaging bags.

6. Conclusions

This article proposed a dual branch point cloud registration network (DCDNet-Att) based on variable weight dynamic graph convolution with a dual attention mechanism to solve the problem of packaging bag point cloud registration. The network uses a variable weight dynamic graph convolution module to extract point cloud features, a feature interaction module to extract common features between the source point cloud and the template point cloud, and a bottleneck module to further emphasize the same geometric features between the two point clouds. It uses a channel attention function to process point cloud features and obtain the channel attention weights, weighs and summarizes point cloud features, and then encodes the spatial position information of the point cloud features so that the model can consider the positional relationship between features, calculate the attention weight of each spatial position, and use a rotation translation separation structure to sequentially obtain quaternions and translation vectors. It uses a feature-fitting loss function to constrain the parameters of the neural network model to make the model have a larger receptive field. Experimental verification was conducted on the registration of packaging bag point clouds with publicly available dataset point clouds, and the results demonstrated the effectiveness of DCDNet-Att.

The packaging bag point cloud dataset need further enrichment. Due to experimental limitations, the collected packaging bag sizes, shapes, and quantities were limited, and the validation results of the model were limited. Increasing the data volume of the packaging bag point cloud dataset can better verify the generalization ability and registration effect of deep learning models on packaging bag point clouds.

Author Contributions: Data curation, writing—original draft, and methodology, J.Z.; software and formal analysis, C.Y.; data curation, G.Z.; resources and funding acquisition, X.Y.; validation and project administration, Q.Z. All authors have read and agreed to the published version of the manuscript.

Funding: The authors acknowledge the Department of Science and Technology of Jilin province (20220203091SF, 20240302033GX).

Data Availability Statement: The datasets used and analyzed during the current study are available from the corresponding author upon reasonable request.

Conflicts of Interest: Author Qingming Zhao was employed by the company Jilin Jibang Automation Technology Co., Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Liu, H. Application of Industrial Robots in Anchor Chain Automation Manufacturing. *Autom. Appl.* **2023**, *64*, 11–13.
2. Lee, J.H.; Park, S.M.; Kang, L.S. Methodology for Activity Unit Segmentation of Design 3D Models Using PointNet Deep Learning Technique. *KSCE J. Civ. Eng.* **2024**, *28*, 29–44. [CrossRef]
3. Haznedar, B.; Bayraktar, R.; Ozturk, A.E.; Arayici, Y. Implementing PointNet for point cloud segmentation in the heritage context. *Herit. Sci.* **2023**, *11*, 2. [CrossRef]
4. Fareed, N.; Flores, J.P.; Das, A.K. Analysis of UAS-LiDAR ground points classification in agricultural fields using traditional algorithms and PointCNN. *Remote Sens.* **2023**, *15*, 483. [CrossRef]
5. Chen, Z.Y.; Peng, S.W.; Zhu, H.D.; Zhao, R.; Zhou, X.; Hua, R. Research on Point Cloud Classification of Transmission Channels Based on Sample Weighted PointNet++. *Remote Sens. Technol. Appl.* **2022**, *36*, 1299–1305.
6. Chen, Y.; Xu, Y.; Xing, Y.; Liu, G. DGCNN network architecture with densely connected point pairs in multiscale local regions for ALS point cloud classification. *IEEE Geosci. Remote Sens. Lett.* **2021**, *19*, 6502405. [CrossRef]
7. Wang, Y.; Solomon, J.M. Deep closest point: Learning representations for point cloud registration. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3523–3532.
8. Gu, Z.Y.; Chen, C.; Zheng, J.J.; Sun, D.H. Dynamic Graph Convolutional Neural Network Traffic Flow Prediction Considering Spatiotemporal Similarity. *Control. Decis.-Mak.* **2023**, 3399–3408. [CrossRef]
9. Wang, Y.; Solomon, J.M. Prnet:Self-supervised learning for partial-to-partial registration. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 8814–8826.
10. Sarode, V.; Li, X.; Goforth, H.; Aoki, Y.; Srivatsan, R.A.; Lucey, S.; Choset, H. Pcrnet: Point cloud registration network using pointnet encoding. *arXiv* **2019**, arXiv:1908.07906.
11. Sun, H.; Jin, Y.Q.; Zhang, W.A.; Fu, M.L. Deep completion based on multi guided structure perception network. *Control Decis.-Mak.* **2024**, *39*, 401–410.
12. Yew, Z.J.; Lee, G.H. Rpm-net:Robust point matching using learned features. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 11824–11833.
13. Wu, Y.; Yuan, Y.Z.; Xiang, B.H. Overview of Computational Intelligence Methods in 3D Point Cloud Registration. *J. Image Graph.* **2023**, *28*, 2763–2787. [CrossRef]
14. Yuan, W.; Eckart, B.; Kim, K.; Jampani, V.; Fox, D.; Kautz, J. Deepgmr: Learning latent gaussian mixture models for registration. In Proceedings of the Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Proceedings, Part V 16. Springer International Publishing: Berlin/Heidelberg, Germany, 2020; pp. 733–750.
15. Liu, J.; Lin, S.J.; Liang, W.K.; Wang, Q.; Liu, M. Short term probabilistic prediction of photovoltaic output based on high-order Markov chain and Gaussian mixture model. *Power Grid Technol.* **2023**, *47*, 266–274.
16. Wu, J.; Duan, Y.Y.; Ma, X.H. Thermal infrared target tracking algorithm based on KL divergence and channel selection. *Infrared Technol.* **2023**, *45*, 33–39.
17. Zeng, Y.; Qian, Y.; Zhu, Z.; Hou, J.; Yuan, H.; He, Y. Corrnet3d: Unsupervised end-to-end learning of dense correspondence for 3d point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 6052–6061.
18. Li, W.; Huang, X.Y.; Feng, Y.R. Unsupervised Common Sense Question Answering Model Based on Course Learning. *Comput. Appl. Res.* **2023**, *40*, 1674–1678.
19. Zhang, L.J.; Wang, B.B.; Wang, W.; Wu, D.; Zhang, N. A registration method for non homologous low overlap point clouds in pedicle screw internal fixation surgery. *Chin. J. Lasers* **2023**, *50*, 0907108.
20. Fischler, M.A.; Bolles, R.C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* **1981**, *24*, 381–395. [CrossRef]
21. Li, X.; Lu, J.; Ding, H.; Zhou, J.T.; Chee, Y.M. PointCVaR: Risk-Optimized Outlier Removal for Robust 3D Point Cloud Classification. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024; Volume 38, pp. 21340–21348.
22. Carleton, W.C.; Groucutt, H.S. Sum things are not what they seem: Problems with point-wise interpretations and quantitative analyses of proxies based on aggregated radiocarbon dates. *Holocene* **2021**, *31*, 630–643. [CrossRef]
23. Liu, Q.; Zhai, J.W.; Zhong, S.; Zhang, Z.C.; Zhou, Q.; Zhang, P. A deep cyclic Q-network model based on visual attention mechanism. *J. Comput. Sci.* **2017**, *40*, 14.
24. Sun, Y.; Ding, W.P.; Wang, J.S.; Ju, H.; Zhang, C.; Yang, G.; Lin, C.T. RCAR-UNet: Retinal vessel segmentation network based on rough channel attention mechanism. *Comput. Res. Dev.* **2023**, *60*, 15.

25. Sasibhooshan, R.; Kumaraswamy, S.; Sasidharan, S. Image caption generation using visual attention prediction and contextual spatial relation extraction. *J. Big Data* **2023**, *10*, 18. [CrossRef]
26. Chen, L.; Wei, M. ELF-Net: Enriching local features network for 3D point cloud classification and semantic segmentation. *J. Intell. Fuzzy Syst.* **2021**, *41*, 3973–3983. [CrossRef]