electronics

Article

Two-Stage Point Cloud Registration Framework Based on Graph Neural Network and Attention

Xiaoqian Zhang [1,2,3], Junlin Li [1,2,*], Wei Zhang [1,2], Yansong Xu [1,2,3] and Feng Li [1,2,3]

[1] State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang 110016, China; zhangxiaoqian@sia.cn (X.Z.); zhangwei@sia.cn (W.Z.); xuyansong21@mails.ucas.ac.cn (Y.X.); lifeng212@mails.ucas.ac.cn (F.L.)
[2] Institutes for Robotics and Intelligent Manufacturing, Chinese Academy of Sciences, Shenyang 110169, China
[2] University of Chinese Academy of Sciences, Beijing 100049, China
* Correspondence: lijunlin@sia.cn

Abstract: In recent years, due to the wide application of 3D vision in the fields of autonomous driving, robot navigation, and the protection of cultural heritage, 3D point cloud registration has received much attention. However, most current methods are time-consuming and are very sensitive to noises and outliers, resulting in low registration accuracy. Therefore, we propose a two-stage framework based on graph neural network and attention—TSGANet, which is effective in registering low-overlapping point cloud pairs and is robust to variable noises as well as outliers. Our method decomposes rigid transformation estimation into two stages: global estimation and fine-tuning. At the global estimation stage, multilayer perceptrons are employed to estimate a seven-dimensional vector representing rigid transformation directly from the fusion of two initial point cloud features. For the fine-tuning stage, we extract contextual information through an attentional graph neural network consisting of attention and feature-enhancing modules. A mismatch-suppression mechanism is also proposed and applied to keep our method robust to partially visible data with noises and outliers. Experiments show that our method yields a state-of-the-art performance on the ModelNet40 dataset.

Keywords: point cloud registration; deep learning; graph neural network; attention

check for updates

Citation: Zhang, X.; Li, J.; Zhang, W.; Xu, Y.; Li, F. Two-Stage Point Cloud Registration Framework Based on Graph Neural Network and Attention. *Electronics* **2024**, *13*, 654. https://doi.org/10.3390/electronics13030654

Academic Editor: Stefanos Kollias

Received: 2 January 2024
Revised: 31 January 2024
Accepted: 1 February 2024
Published: 4 February 2024

1. Introduction

Point cloud registration is the process of aligning multiple three-dimensional point clouds to integrate data from different perspectives or time points, creating an accurate and complete 3D model. This technology is crucial for applications such as 3D reconstruction, robotics, autonomous driving, and object recognition. Point cloud registration enhances model accuracy, supports time-sensitive applications, and advances experiences in virtual reality and simulations. Understanding point cloud registration is essential for researchers and practitioners engaged in decision making and interactive applications based on three-dimensional data.

Point cloud registration methods can be categorized into traditional methods and deep learning-based methods. The traditional ICP (Iterative Closest Points) algorithm and its variants [1–3] align point clouds by iteratively reducing the distance between source and target point clouds, which are commonly used in industry. Point-to-plane ICP algorithms [2,4] approximate the distance between point clouds with the point-to-plane distance, which is a fine-registration algorithm that requires a reasonable initial transformation given by the measurement platform or feature matching. Similarly, point-to-point distance is used to approximate the distance between point clouds in point-to-point ICP algorithms [2]. Variants of ICP algorithms are summarized in six steps by [3]: (1) selecting point sets; (2) matching the point sets; (3) weighting the corresponding pairs; (4) eliminating abnormal corresponding pairs; (5) assigning an error metric based on the point pairs; and (6) minimizing the error metric. Li and Wang et al. [5] show a trade-off between computational efficiency and effectiveness when applying the ICP algorithms. Taking point-to-plane and point-to-point ICP algorithms as examples, although the former is more efficient than the latter, its effective range is smaller. In addition, point-to-plane ICP algorithms tend to oscillate and cannot converge to global minima when the input data is heavily affected by noise. Some traditional methods [6,7] first extract 3D interest points as well as descriptors and then utilize SVD (Singular Value Decomposition) combined with RANSAC (Random Sample Consensus) proposed by Fischler and Bolles [8] to estimate rigid transformation. The bottleneck of the RANSAC-based methods is the computational complexity. When the point cloud is denser and noisier, more comprehensive and detailed sampling is required to obtain a highly robust estimate, resulting in computational complexity increasing exponentially with the scale of the point cloud. Moreover, the effectiveness and accuracy of the traditional interest point-based methods depend heavily on the efficiency of 3D interest point-detection algorithms, such as the accuracy of 3D location and uniqueness of the description.

Learned registration methods can be classified as direct estimation-based methods [9,10] and point correspondence-based methods [11–14] according to different transformation estimation blocks. Direct estimation-based methods simply predict vectors representing rigid transformation by estimation block consisting of feature pooling layers and the subsequent MLPs (Multilayer Perceptrons), leading to the loss of much feature information. Point correspondence-based methods generate point correspondences from pointwise features and then solve for rigid transformation by the SVD method [15], making sufficient use of the extracted features. However, they need to prevent inference from mismatching point pairs. To better exploit the geometric information of point clouds, carefully constructed complicated inputs are also employed in some point correspondence-based methods [12,14].

Direct estimation-based methods extract a feature vector for the entire point cloud and employ multi-layer perceptrons to directly estimate the parameters representing the transformation. While these methods are fast, they tend to waste much valuable feature information, leading to suboptimal registration results. On the other hand, point correspondence-based methods estimate point correspondences according to the feature vectors of each point. These methods are suitable for iterative registration, but the process of estimating point correspondences can be time-consuming. Existing learned methods typically either solely rely on direct estimation or exclusively use point correspondence-based methods. In this paper, we introduce a two-stage framework based on graph neural network and attention—TSGANet. Our method combines two kinds of transformation estimation blocks, decomposing the registration process into two stages: global estimation and fine-tuning. At the global estimation stage, we exploit local geometric information from inputs and obtain a rough transformation by the direct transformation estimation block. For the fine-tuning stage, a point correspondence-based estimation block is applied to make up for the information loss in the previous global estimation stage. In all, the first stage provides a rough transformation for the second stage, and the second stage outputs the final fine-tuning results. This combination enables our method to save time while achieving greater registration results through several iterations. As shown in Figure 1, our method can accurately register point clouds after three iterations. An attentional graph neural network is proposed in the fine-tuning stage to better utilize and enhance the contextual information within a single point cloud and between different point clouds. Existing methods rarely incorporate a mismatch-suppression mechanism. To ensure robustness to noise and outliers, our approach introduces a mismatch-suppression module, which filters poor corresponding point pairs by learnable affinity threshold. In addition, unlike other existing methods, our approach does not require complex normal vectors and angular information; it solely relies on the three-dimensional coordinates of spatial points as input.

In this paper, Section 2 introduces some representations of point cloud registration; Section 3 provides a detailed description of the methods employed by TSGANet; Section 4 presents the experimental results, where we train our model on the ModelNet40 [16] dataset and test it on both ModelNet40 dataset and real data; Section 5 concludes the paper; and Section 6 discusses the limitations of this method and suggests future improvements.

Our main contributions are as follows:

1. We propose a simple and fast deep network for point cloud registration;
2. We decompose the registration process into global estimation and fine-tuning stages;
3. For the fine-tuning stage, an attentional graph neural network with attention as well as feature-enhancing modules and a mismatch-suppression mechanism is proposed, which has proved effective against partially visible data with noises and outliers;
4. Experiments show that our method is effective in registering low-overlapping point cloud pairs and robust to variable noises as well as outliers;
5. Our method achieves a state-of-the-art performance on the ModelNet40 dataset over various evaluation criteria and is computationally efficient.

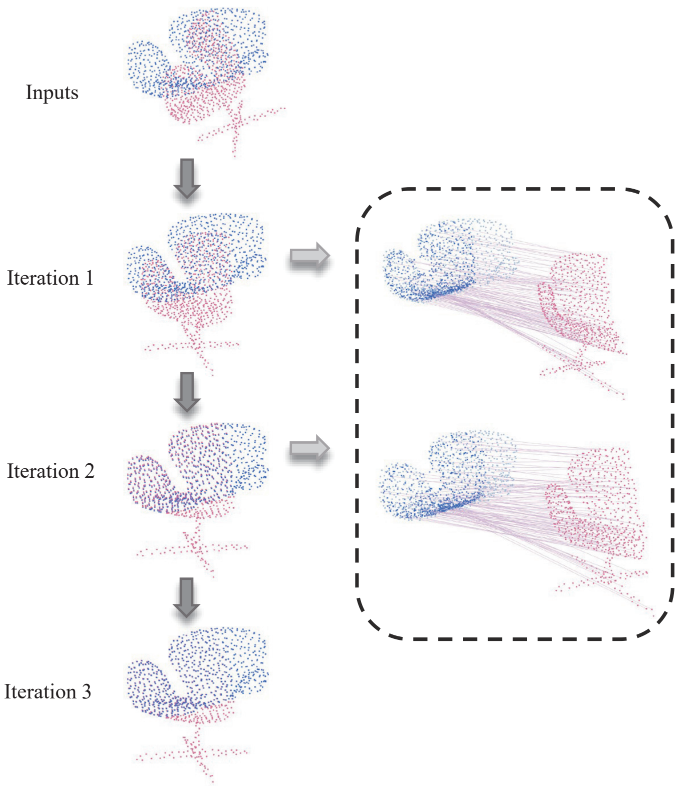


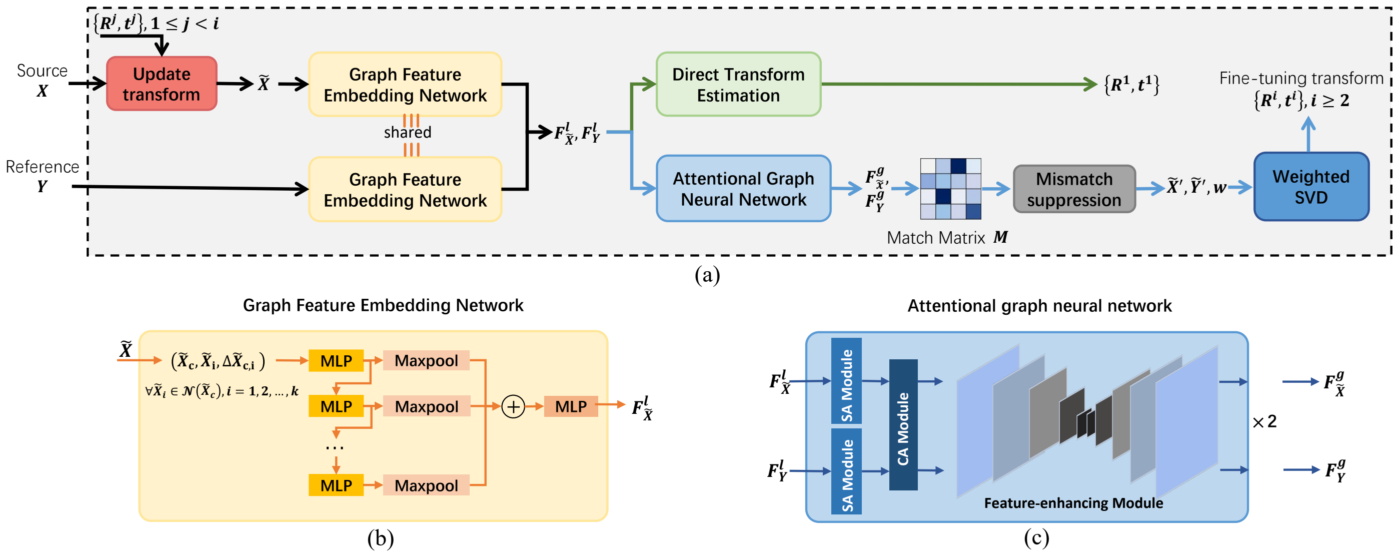
Figure 1. Our TSGANet accurately registers point clouds after three iterations. The first iteration estimates an initial rigid transformation roughly aligning point clouds, and the next two iterations infer the fine-tuning rigid transformation based on point correspondences.

2. Problem Formulation

Two given unaligned point clouds, X and Y, represent the source and target point clouds, respectively. The point clouds comprise spatial coordinates of points. Our goal is to recover the unknown rigid transformation $T = \{R, t\}$ between X and Y, where $R \in SO(3)$ denotes the rotation matrix and $t \in \mathbb{R}^3$ denotes the translation vector.

3. TSGANet

Figure 2 shows an illustration of our method. The global estimation stage only goes through one iteration. The fine-tuning stage is iterative and goes through several iterations. At the beginning of iteration i, the input point cloud X is transformed into $\tilde{X}$ using the rigid transformation estimated in all previous iterations. For the first iteration, $\tilde{X}$ is equal to X. Formulate the result of iteration i as T^i, then the overall transformation can be expressed as $T = T^i T^{i-1} \ldots T^i$.



$\{R^j, t^j\}, 1 \le j < i$
Source X
Update transform
$\tilde{X}$
Graph Feature Embedding Network
shared
Reference Y
Graph Feature Embedding Network
$F^l_{\tilde{X}}, F^l_Y$
Direct Transform Estimation
$\{R^1, t^1\}$
Fine-tuning transform $\{R^i, t^i\}, i \ge 2$
Attentional Graph Neural Network
$F^g_{\tilde{X}}, F^g_Y$
Match Matrix M
Mismatch suppression
$\tilde{X}', \tilde{Y}', w$
Weighted SVD
(a)
Graph Feature Embedding Network
$(\tilde{X}_c, \tilde{X}_i, \Delta\tilde{X}_{c,i})$
$\forall \tilde{X}_i \in \mathcal{N}(\tilde{X}_c), i = 1, 2, \ldots, k$
MLP
Maxpool
MLP
$F^l_{\tilde{X}}$
(b)
Attentional graph neural network
$F^l_{\tilde{X}}$
F^l_Y
SA Module
CA Module
Feature-enhancing Module
$\times 2$
$F^g_{\tilde{X}}$
F^g_Y
(c)

Figure 2. (**a**) Overview of our TSGANet, (**b**) graph feature-embedding network, (**c**) attentional graph neural network. The black lines in (**a**) point to the steps required at both stages, while the green lines point to the steps required only at the global estimation stage and the blue lines point to the steps required only at the fine-tuning stage.

Both the global estimation stage and the fine-tuning stage encode local features from input point clouds through the graph feature embedding network, but only the latter further utilizes the attentional multilayer graph neural network to extract contextual information.

3.1. Input of Model

When processing sparse and low-overlapping point clouds, it is beneficial to encode pose features from local geometry instead of from a single point [10]. Therefore, we incorporate the neighborhood information of each point into its input representation. For each point x_c in $\tilde{X}$, we construct the k-neighborhood $\mathcal{N}(x_c)$; then, the input corresponding to x_c is:

$$inp(x_c) = (x_c, x_{c,i}, x_c - x_{c,i}), \tag{1}$$

where $x_{c,i} \in \mathcal{N}(x_c)$. Similarly, the input of the Graph feature-embedding network corresponding to $\tilde{X}$ can be represented as:

$$inp(\tilde{X}) = (\tilde{X}, \mathcal{N}(\tilde{X})). \tag{2}$$

3.2. Global Estimation Stage

3.2.1. Graph Feature Embedding Network

We adopt the practice of stacking EdgeConv layers from DGCNN [17] to explicitly incorporate local geometric properties into the input representation of the model. However, we use a fixed graph structure in the coordinate space of the point cloud in each EdgeConv layer, rather than dynamically updating it in the feature space. The graph feature embedding network embeds feature vectors for each point in point clouds. The embedded features for input $\tilde{X}$ can be formulated as:

$$F^l_{\tilde{X}} = f(inp(\tilde{X})), \tag{3}$$

where f represents the forward propagation process of the feature embedding network. F^l_Y is calculated in the same way as $F^l_{\tilde{X}}$, and they share the weights of the network.

3.2.2. Direct Transformation Estimation

For computational efficiency, only the initial local features extracted by the graph feature embedding network are employed at the global estimation stage. We first concatenate initial features $F^l_{\tilde{X}}$ and F^l_Y of shape (N, C) over the second dimension to a new feature $F^l_{\tilde{X}Y}$ of shape $(N, 2C)$. The max pooling is then performed on $F^l_{\tilde{X}Y}$ over the first dimension to eliminate redundant features, generating a new fused feature of shape $(2C)$. The initial rigid transformation can be obtained by passing the fused feature map to the following MLPs. Note that what the MLPs predict for rotation is quaternion and we have to convert it into rotation matrix.

3.3. Fine-Tuning Stage

3.3.1. Attentional Multilayer Graph Neural Network

When a person is asked to match the features of two objects, they usually first observe single object separately; and then review between two objects [18]. This corresponds to an iterative process where the focus of attention is constantly changing. Therefore, we treat the initial feature map of a point cloud as a fully connected graph and employ an attentional multilayer graph neural network to aggregate information at different levels in this subnet. Inspired by [18], we alternately perform self-attention aggregation and cross-attention aggregation modules in this multilayer network. In addition, we also propose a feature-enhancing module.

The self-attention aggregation module is used to extract global information within a single point cloud first. Denote the output of the l-th layer of the multilayer graph neural network by ${}^lF^g_{\tilde{X}}$ and ${}^lF^g_Y$, respectively. Assuming that the $(l+1)$-th layer is a self-attention aggregation module, its output can be expressed as:

$$ {}^{l+1}F^g_{\tilde{X}_{SA}} = {}^lF^g_{\tilde{X}} + \mathrm{MLP}\left(A\left({}^lq^{\tilde{X}}, {}^lk^{\tilde{X}}, {}^lv^{\tilde{X}}\right)\right), \tag{4} $$

$$ {}^lq^{\tilde{X}} = {}^lW_q{}^lF^g_{\tilde{X}}, {}^lk^{\tilde{X}} = {}^lW_k{}^lF^g_{\tilde{X}}, {}^lv^{\tilde{X}} = {}^lW_v{}^lF^g_{\tilde{X}}. \tag{5} $$

q, k, and v are the query, key, and value calculated by different linear layers, respectively. A is the attention function that maps the query, key, and value to the output:

$$ A(q,k,v) = \frac{softmax\left(q^{\mathrm{T}}k\right)}{\sqrt{d}}v, \tag{6} $$

where d represents the length of each feature vector. ${}^{l+1}F^g_{Y_{SA}}$ is calculated in the same way as ${}^{l+1}F^g_{\tilde{X}_{SA}}$ and they share the network weights.

The cross-attention aggregation module is employed to make the features of the source and target point clouds communicate with each other, thereby enhancing related features and suppressing redundant features. Different from the self-attention aggregation module, the cross-attention aggregation module exchanges information between point clouds by changing the sources of the query and value:

$$ {}^{l+1}F^g_{\tilde{X}_{CA}} = {}^lF^g_{\tilde{X}} + \mathrm{MLP}\left(A\left({}^lq^{\tilde{X}}, {}^lk^{Y}, {}^lv^{Y}\right)\right), \tag{7} $$

$$ {}^{l+1}F^g_{Y_{CA}} = {}^lF^g_{Y} + \mathrm{MLP}\left(A\left({}^lq^{Y}, {}^lk^{\tilde{X}}, {}^lv^{\tilde{X}}\right)\right). \tag{8} $$

Similarly, ${}^{l+1}F^g_{Y_{CA}}$ shares network weights with ${}^{l+1}F^g_{\tilde{X}_{CA}}$. The final outputs of the attentional multilayer graph neural network are denoted as $F^g_{\tilde{X}}$ and F^g_Y.

The feature-enhancing module follows the cross-attention aggregation module. To further enhance relevant information, the module is composed of 10 Conv1d layers, whose output channels are $[256, 256, 128, 64, 32, 64, 128, 256, 256, 512]$, first reducing the channel of the attention output feature and then increasing the channel.

3.3.2. Point Correspondence-Based Transformation Estimation

After the first round of alignment, local geometric properties between corresponding point pairs in two input point clouds will be easier to extract. Information exchange in cross-attention modules also helps exploit correlations between point pairs. Therefore, when estimating rigid transformation at the fine-tuning stage, we do not fuse features; but calculate point correspondences. First, compare the affinity of each point pair between $\tilde{X}$ and Y by the L2 distance between features $F_{\tilde{X}}^{g}$ and F_{Y}^{g}:

$$affinity = -\left\| F_{\tilde{X}}^{g} - F_{Y}^{g} \right\|_2^2. \tag{9}$$

Then, a differentiable Sinkhorn layer [19] is employed to make the sum of each row and column of the affinity matrix close to 1.0, letting the estimated point correspondences be reasonable. After that, a soft corresponding matrix M is obtained.

3.3.3. Mismatch Suppression Mechanism

For partially visible data and data with outliers, not each point in the source point cloud has a corresponding point in the target point cloud. Thus, we propose a mismatch-suppression mechanism, which reduces the probability of mismatch by setting correlation thresholds. With matrix M, we calculate $M_{\tilde{X}}$ and M_Y:

$$M_{\tilde{X}} = max(M, \text{axis} = 1), \tag{10}$$

$$M_Y = max(M, \text{axis} = 0). \tag{11}$$

$M_{\tilde{X}}$ represents the maximum correlation value of each point in $\tilde{X}$ to points in Y, while M_Y represents the opposite. Then we select points $\tilde{X}'$ and Y' whose corresponding values in $M_{\tilde{X}}$ and M_Y are beyond the threshold t_X and t_Y:

$$\tilde{X}' = (\tilde{X}_i), M_{\tilde{X}i} \geq t_X, Y' = (Y_i), M_{Yi} \geq t_Y. \tag{12}$$

$$t_X = \alpha.medium(M_{\tilde{X}}), t_Y = \beta.medium(M_Y). \tag{13}$$

α and β are learnable parameters of the network, initialized to 1.0. The *medium* function can also be changed to the *average* function. $\tilde{X}'$ is the new generated source point cloud. The reference point cloud $\tilde{Y}'$ to source $\tilde{X}'$ is then obtained by:

$$\tilde{Y}' = M'Y', \tag{14}$$

where M' is the soft corresponding matrix between $\tilde{X}'$ and Y'. Weighted SVD is then employed to estimate the transformation. When solving weighted SVD, the weight of each corresponding pair is $w_i = max\left(M'_i\right)$. Then, we normalize the weights of the corresponding point pairs by a softmax layer.

3.4. Loss Function

The loss of our network is briefly defined by the error between the final recovered rigid transformation of the unaligned point clouds and the ground truth value. The error includes the rotation error and translation error:

$$Loss = MSE\left(R_{\text{est}}R_{\text{gt}}{}^{T} - I\right) + MSE(t_{\text{est}} - t_{\text{gt}}). \tag{15}$$

4. Experiments

4.1. Implementation Details

We determine the neighborhood of $k = 20$ for each point in the input representation through ablation studies in Section 4.6.2. The feature vector length is 512. Through extra experiments, we determine that the network undergoes three iterations during both training and testing, with one iteration in the global estimation stage and two in the fine-tuning stage. In the fine-tuning stage, the combination of attention and feature-enhancing modules appears twice. α and β are all initialized to 1.0. We train the network using the Adam optimizer with an initial learning rate of 0.0001, and the learning rate decreases by a factor of 0.5 every 100 epochs. The batch size during training is 16.

4.2. Datasets and Evaluation Metrics

Most experiments are conducted on the ModelNet40 [16] dataset, and the remains are on the real data. The real data will be introduced later in Section 4.5. The ModelNet40 dataset contains 12,311 CAD models from 40 categories. We use the processed data from the previous work [20], which include 2048 points randomly sampled from each CAD model, and the data of each point cloud are composed of 3D coordinates and normal vectors. Each category in the dataset contains train and test splits. We use the train splits of all categories as the training set and the test splits as the testing set. Point clouds are down-sampled to contain 1024 points in our method. For a target point cloud Y, we apply a random rigid transformation to the source point cloud X. For the random rigid transformation applied, we randomly sample three Euler angles in the range of $[0, 45]$ for rotation and three displacements in the range of $[-0.5, 0.5]$ for translation.

We use three evaluation metrics from RegTR [21]: (1) Relative Rotation Error (RRE); (2) Relative Translation Error (RTE); and (3) Chamfer distance (CD) between the registered scans. In addition to the above, there are also the registration recall (RR) and execution time. The registration recall refers to the fraction of point cloud pairs whose RRE and RTE are below 4 and 0.1, respectively.

We compare our TSGANet to ICP [1] and FGR [22], as well as recently learned registration methods: DCP-v2 [11], RPM-Net [12], RGM [13], RegTR [21], and OGMM [23]. The ICP and FGR methods are implemented by Open3D [24]. For learned methods, we train the models with implementation provided by the authors and make our best efforts to fine-tune them.

4.3. Low-Overlapping Data

To evaluate the effectiveness of our method on low-overlapping point cloud pairs, we conduct experiments on point clouds with 70% and 50% completeness. To generate partially visible point clouds, we follow the strategy Yew and Li [12] propose. For each point cloud, we first generate a 3D plane passing through the coordinate origin and then move the plane in the direction of its normal until a certain fraction of the points remain. Gaussian noise sampled from $\mathrm{N}(0, 0.01)$ and 10% outliers are also applied to these point clouds, respectively. For outliers, we randomly sample three values from $\mathrm{U}(-1.0, 1.0)$ as the 3D coordinates of the outliers. All learned methods are trained on 70% completeness data without noises and outliers and then tested on 70% and 50% completeness data with noises as well as outliers. Table 1 shows the experimental results of all algorithms on 70% completeness point clouds, and Table 2 shows the results on 50% completeness point clouds. Our method ranks first or second in many metrics, which proves our method is effective in registering low-overlapping point cloud pairs. Furthermore, as an iterative method, although our TSGANet is slightly slower than ICP, FGR, and noniterative DCP-v2, it is much faster than other iterative learned methods and consume almost the same time with OGMM. Qualitative comparisons of the registration results on 70% completeness point clouds are shown in Figure 3. Figure 4 shows the registration examples of our method on 50% completeness point clouds. The performance of various algorithms on clean data can be found in Appendix A.

Table 1. Performance of various algorithms on 70% completeness data with noises and outliers. Note that the singly bolded numbers indicate the first rank, while the numbers in bold and italics indicate the second rank.

Methods	**Data with Noises**					**Data with Outliers**			
	RRE	**RTE**	**CD**	**RR (%)**	**Time (ms)**	**RRE**	**RTE**	**RR (%)**	**Time (ms)**
ICP	24.02	0.1856	0.013	11.7	3	28.42	0.1881	9.2	3
FGR	53.17	0.1859	0.030	7.0	9	17.25	0.0505	64.5	27
DCP-v2	17.94	0.2132	0.019	1.1	9	20.76	0.2053	0.16	10
RPM-Net	***1.24***	***0.0121***	$\mathbf{7.2 \times 10^{-4}}$	***95.3***	37	4.36	0.0301	66.8	43
RGM	2.79	0.0219	1.7×10^{-3}	87.8	127	4.68	0.0337	73.3	149
RegTR	1.68	0.0137	$\boldsymbol{8.4 \times 10^{-4}}$	93.2	35	***2.69***	0.0237	83.8	42
OGMM	3.1543	**0.0084**	0.1834	87.4	26	2.7766	**0.0068**	***85.7***	26
Ours	**1.14**	0.0130	9.8×10^{-4}	**97.3**	27	**1.53**	***0.0128***	**96.0**	29

Table 2. Performance of various algorithms on 50% completeness data with noises and outliers. Note that the singly bolded numbers indicate the first rank, while the numbers in bold and italics indicate the second rank.

Methods	**Data with Noises**					**Data with Outliers**			
	RRE	**RTE**	**CD**	**RR (%)**	**Time (ms)**	**RRE**	**RTE**	**RR (%)**	**Time (ms)**
ICP	43.74	0.328	0.031	6.4	2	41.63	0.307	3.6	2
FGR	58.70	0.356	0.043	2.8	5	38.43	14.901	39.1	16
DCP-v2	32.75	0.565	0.259	0	6	29.69	0.561	0	7
RPM-Net	7.86	0.103	0.126	***61.4***	48	10.96	0.133	29.4	34
RGM	18.54	0.165	**0.086**	52.3	84	20.40	0.166	34.5	94
RegTR	5.09	***0.089***	***0.123***	58.8	28	***6.098***	***0.077***	***53.0***	31
OGMM	**3.078**	**0.0074**	0.184	**86.5**	26	**2.0082**	**0.0044**	**88.4**	27
Ours	***3.97***	0.103	0.184	55.4	18	6.100	0.082	49.1	21

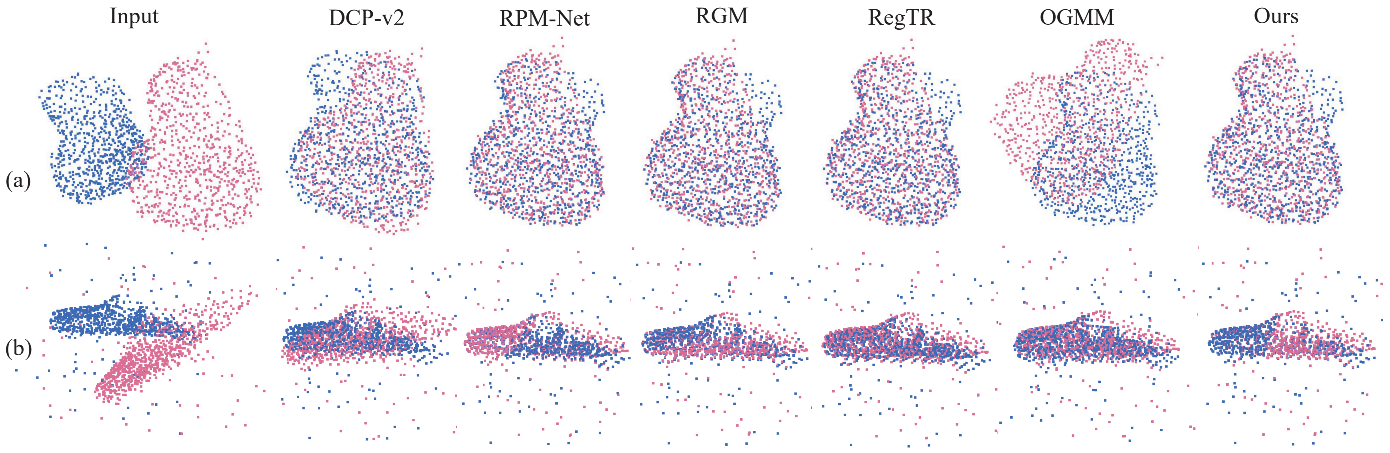


Figure 3. Qualitative registration examples on 70% completeness point clouds with (**a**) Gaussian noise sampled from $N(0, 0.01)$, (**b**) 10% outliers.

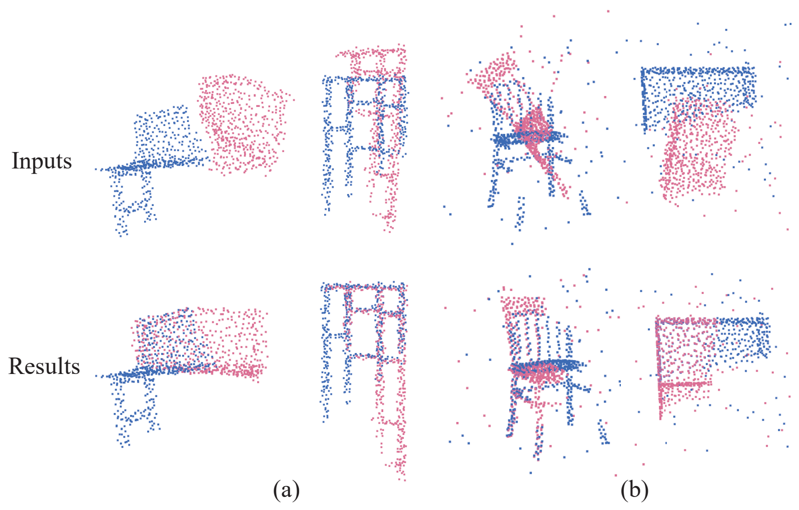


Figure 4. Registration examples of our method on 50% completeness point clouds with (**a**) Gaussian noise sampled from $N(0, 0.01)$, (**b**) 10% outliers.

4.4. Variable Noises and Outliers

In this experiment, we compare the robustness of various algorithms to variable noises and variable fractions of outliers. For experiments on noises, we apply Gaussian noise with standard deviations of 0.01, 0.02, 0.03, 0.04, and 0.05 to 70% completeness point clouds. Figure 5a,b illustrate the changes in RRE and RTE for all algorithms. It can be observed that as the noises increase, our method exhibits a slow growth in both RRE and RTE, but with a little deviation from the best-performing RPM-Net method. For experiments on outliers, we introduced outliers of fractions 0.1, 0.2, 0.3, 0.4, and 0.5 to 70% completeness point clouds. Figure 5c,d show that our method always ranks first. These experiments confirm the robustness of our approach to variable noises and outliers.

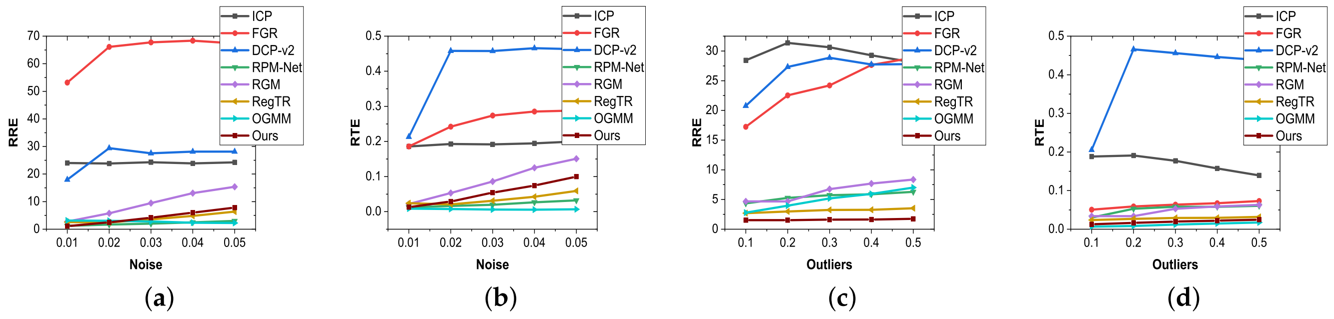


Figure 5. (**a**) RRE of various algorithms on partially visible data with variable noises. (**b**) RTE of various algorithms on partially visible data with variable noises. (**c**) RRE of various algorithms on partially visible data with a variable fraction of outliers. (**d**) RTE of various algorithms on partially visible data with a variable fraction of outliers.

To understand how the point correspondence-based registration methods work on data with outliers, we visualize the point correspondences obtained by our TSGANet and RPM-Net at the last iteration. The visualization is shown in Figure 6. The red and gray points represent the source and target point clouds, respectively, and the blue points represent the generated reference point cloud corresponding to the source point cloud. The red lines connect the corresponding point pairs between the source point cloud and the reference point cloud. The darker the color of the line is, the more credible the corresponding point pair is, and vice versa. It is observed that the lines connecting corresponding point pairs in Figure 6b are neater and darker than those in Figure 6a, which means our method is less disturbed by outliers than RPM-Net.

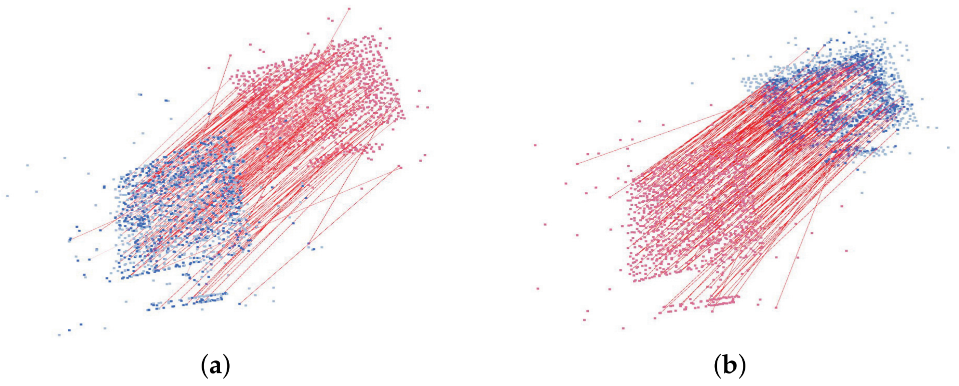


Figure 6. Visualizations of point correspondences at the iteration of (**a**) RPM-net and (**b**) our TSGANet. We compare our method with RPM-Net because they share some similarities in their computation processes. The source and target point clouds in RPM-Net and our method are opposite.

4.5. Real Data

In this experiment, we test the applicability of our method in scenarios closer to real-world applications. Since the research's primary future application involves the automated construction of infrastructure, and our goal is to estimate the pose of robotic arms' end-effectors during the construction process by aligning the point clouds of cooperating targets. We conduct experiments on the point clouds obtained from two mechanical parts using the Tango-S series 3D scanner. The mechanical parts and the registration results are both depicted in Figure 7. The experimental results demonstrate the favorable applicability of our method in the predefined application scenarios.

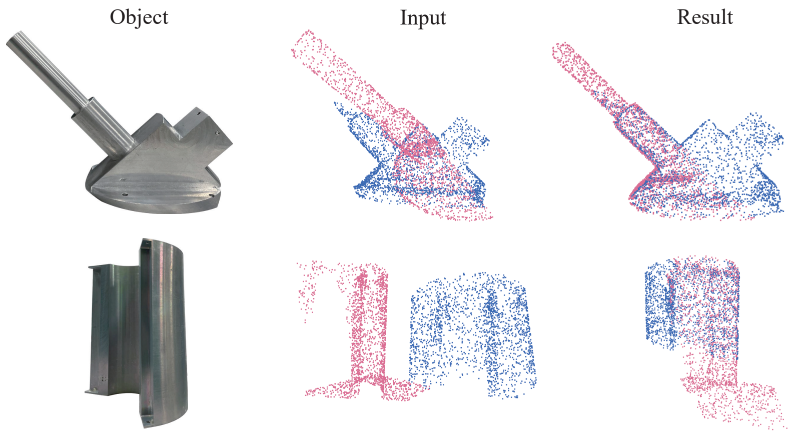


Figure 7. Mechanical parts and registration examples of our method on the point clouds of the mechanical parts.

4.6. Ablation Studies

4.6.1. Necessity of Each Module

We name the model with only the global estimation stage as TSGANet_v1, the model with only the fine-tuning stage as TSGANet_v2, and the model without the mismatch-suppression mechanism as TSGANet_v3. They are compared with TSGANet on 70% completeness point clouds with noises and outliers. The performance of all models is shown in Table 3. The model with only the fine-tuning stage performs a bit better than the model with no fine-tuning stage, which validates the effectiveness of calculating corresponding point pairs by features extracted by the attentional graph neural network. However, both of them are much worse than the complete TSGANet, proving the reasonability of combining the global estimation stage and fine-tuning stage. The model with no mechanism also performs significantly worse than the complete model on all data over all metrics, which proves the mismatch-suppression mechanism is indeed effective for partially visible data with noises and outliers.

Table 3. Performance of different combinations on 70% completeness point clouds with noises and outliers. Note that bold ranks first.

Models	**Data with Noises**			**Data with Outliers**	
	RRE	**RTE**	**CD**	**RRE**	**RTE**
TSGANet_v1	9.23	0.1902	0.023	15.65	0.1793
TSGANet_v2	2.96	0.0175	1.8×10^{-3}	3.50	0.0203
TSGANet_v3	1.79	0.0141	1.1×10^{-3}	1.95	0.0201
TSGANet	**1.14**	**0.0130**	$\mathbf{9.8 \times 10^{-4}}$	**1.53**	**0.0128**

4.6.2. Value of k

In this experiment, we investigate whether encoding neighborhood information and varying the value of k in k-nearest neighbors have impacts on the registration performance. Table 4 shows performance on 70% completeness point clouds with noises and outliers. It can be observed that encoding neighborhood information is beneficial for registration. Consider the performance of the model as the value of k varies, setting k to 20 is a reasonable choice.

Table 4. Performance of our method on 70% completeness point clouds with noises and outliers when the value of k in k-nearest neighbors varies. Note that, when $k = 0$, neighborhood information is not encoded.

k	**Data with Noises**			**Data with Outliers**	
	RRE	**RTE**	**CD**	**RRE**	**RTE**
0	15.4210	0.1977	0.024	14.1295	0.1497
5	4.3958	0.0717	4.6×10^{3}	1.0915	0.0125
10	1.7125	0.0241	1.3×10^{3}	1.2708	0.0124
20	1.1366	0.0130	9.8×10^{4}	1.5299	0.0128
30	1.3948	0.0149	1.0×10^{3}	1.7527	0.0142
40	1.6513	0.0168	1.1×10^{3}	1.9619	0.0163

5. Conclusions

We propose the point cloud registration framework TSGANet. Our approach decomposes the registration process into two stages: global estimation as well as fine-tuning. In the fine-tuning stage, we propose an attentional graph neural network with attention as well as feature-enhancing modules and a mismatch-suppression mechanism, which has proved effective against partially visible data with noises and outliers. Experiments show that our method is effective in registering low-overlapping point clouds and robust to variable noises as well as outliers. It achieves state-of-the-art performance on the ModelNet40 dataset over various evaluation criteria and is computationally efficient.

6. Limitations and Future Work

The limitation of TSGANet lies in its incapability to accommodate large-scale point clouds as input. Our model extracts point-wise features in the feature extraction phase and estimates correspondences between all points in the source and target point clouds during the fine-tuning stage. Constrained by computer memory and computational power, our model can handle only small-scale inputs. This study addresses the project requirements for registering small- to medium-sized point clouds of structures. Experiments show the excellent registration accuracy of our model. In practical applications, due to the relatively small volume and fewer details of structures, even if the real point clouds of the structures are downsampled to a fixed size, it will not result in a significant loss of geometric information. However, when applying the model to indoor or outdoor data, downsampling operations might lead to the failure of the registration process due to the discarding of too much detailed information. Therefore, future work will focus on refining the model to adapt to indoor and outdoor data. Additionally, we believe that leveraging deep learning-based multimodal data fusion will provide significant advantages for point cloud registration.

Author Contributions: Conceptualization, J.L. and X.Z.; Data curation, Y.X. and F.L.; Funding acquisition, J.L. and W.Z.; Investigation, J.L. and X.Z.; Methodology, X.Z.; Project administration, W.Z.; Writing—original draft, X.Z.; Writing—review and editing, X.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Key Deployment of Research Projects by the Chinese Academy of Sciences under Grant No. KGFZD-145-23-56-02. And The APC was funded by State Key Laboratory of Robotics, Shenyang Institute of Automation, Chinese Academy of Sciences.

Data Availability Statement: The data presented in this study are available in this article.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Clean Data

In this experiment, we evaluate the registration performance of various algorithms on clean data, which consist of partially visible data and complete data. For partially visible data, we train and test all algorithms on 70% completeness point clouds without noises and outliers. For complete data, we train and test all algorithms on complete point clouds without noises and outliers.

Table A1 shows the performance of various algorithms. Due to the claim of OGMM that it is effective for partially visible data, we did not experiment with it on complete data. The registration recall here refers to the fraction of point cloud pairs whose RRE and RTE are below 2 and 0.05, respectively. It is obvious that our method always ranks first or second in almost all metrics and requires significantly less time than other iterative-based learned methods.

Table A1. Performance of various algorithms on partially visible data and complete data without noises and outliers. Note that the singly bolded numbers indicate the first rank, while the numbers in bold and italics indicate the second rank.

Methods	**Partially Visible Data**					**Complete Data**				
	RRE	**RTE**	**CD**	**RR (%)**	**Time (ms)**	**RRE**	**RTE**	**CD**	**RR (%)**	**Time (ms)**
ICP	23.0985	0.1845	0.013	7.2	3	5.4000	0.0152	1.8×10^{-3}	67.6	3
FGR	11.8501	0.0410	5.3×10^{-3}	61.2	27	3.2779	4.9×10^{-3}	1.0×10^{-3}	82.1	36
DCP-v2	17.2519	0.2121	0.018	0.12	9	1.9726	4.8×10^{-3}	6.1×10^{-4}	61.4	14
RPM-Net	***1.0649***	0.0103	$\mathbf{1.6 \times 10^{-4}}$	***92.0***	38	0.6243	$\boldsymbol{\mathit{4.0 \times 10^{-4}}}$	1.3×10^{-4}	97.8	56
RGM	1.5093	***0.0095***	3.29×10^{-4}	92.7	126	**0.3908**	2.0×10^{-4}	8.0×10^{-7}	**99.8**	678
RegTR	1.3471	0.0121	3.32×10^{-4}	89.0	36	***0.5142***	1.0×10^{-3}	$\boldsymbol{\mathit{3.2 \times 10^{-7}}}$	89.4	57
OGMM	3.3694	0.0108	0.1824	86.4	26	——	——	——	——	——
Ours	**0.7103**	**0.0086**	$\boldsymbol{\mathit{2.6 \times 10^{-4}}}$	**96.6**	27	0.5274	$\mathbf{9.8 \times 10^{-5}}$	$\mathbf{1.4 \times 10^{-7}}$	***99.2***	42

References

1. Besl, P.J.; McKay, N.D. Method for registration of 3-D shapes. In Proceedings of the Sensor fusion IV: Control Paradigms and Data Structures, Boston, MA, USA, 14–15 November 1991; Volume 1611, pp. 586–606.
2. Chen, Y.; Medioni, G. Object modelling by registration of multiple range images. *Image Vis. Comput.* **1992**, *10*, 145–155. [CrossRef]
3. Rusinkiewicz, S.; Levoy, M. Efficient variants of the ICP algorithm. In Proceedings of the Third International Conference on 3-D Digital Imaging and Modeling, Quebec City, QC, Canada, 28 May–1 June 2001; pp. 145–152.
4. Low, K.L. Linear least-squares optimization for point-to-plane icp surface registration. *Chapel Hill Univ. N. C.* **2004**, *4*, 1–3.

5. Li, P.; Wang, R.; Wang, Y.; Tao, W. Evaluation of the ICP Algorithm in 3D Point Cloud Registration. *IEEE Access* **2020**, *8*, 68030–68048. [CrossRef]
6. Li, J.; Hu, Q.; Ai, M. Point Cloud Registration Based on One-Point RANSAC and Scale-Annealing Biweight Estimation. *IEEE Trans. Geosci. Remote Sens.* **2021**, *59*, 9716–9729. [CrossRef]
7. Theiler, P.W.; Wegner, J.D.; Schindler, K. Markerless point cloud registration with keypoint-based 4-points congruent sets. *ISPRS Ann. Photogramm. Remote. Sens. Spat. Inf. Sci.* **2013**, *2*, 283–288. [CrossRef]
8. 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]
9. 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.
10. Zhang, Z.; Chen, G.; Wang, X.; Shu, M. DDRNet: Fast point cloud registration network for large-scale scenes. *ISPRS J. Photogramm. Remote Sens.* **2021**, *175*, 184–198. [CrossRef]
11. Wang, Y.; Solomon, J. Deep Closest Point: Learning Representations for Point Cloud Registration. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Republic of Korea, 27 October–2 November 2019; pp. 3522–3531.
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 (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 11824–11833.
13. Fu, K.; Liu, S.; Luo, X.; Wang, M. Robust Point Cloud Registration Framework Based on Deep Graph Matching. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Nashville, TN, USA, 20–25 June 2021; pp. 8889–8898.
14. Qin, Z.; Yu, H.; Wang, C.; Guo, Y.; Peng, Y.; Ilic, S.; Hu, D.; Xu, K. GeoTransformer: Fast and Robust Point Cloud Registration with Geometric Transformer. *arXiv* **2023**, arXiv:2308.03768.
15. Arun, K.S.; Huang, T.S.; Blostein, S.D. Least-squares fitting of two 3-D point sets. *IEEE Trans. Pattern Anal. Mach. Intell.* **1987**, *PAMI-9*, 698–700. [CrossRef] [PubMed]
16. Wu, Z.; Song, S.; Khosla, A.; Yu, F.; Zhang, L.; Tang, X.; Xiao, J. 3D ShapeNets: A deep representation for volumetric shapes. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1912–1920.
17. Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S.E.; Bronstein, M.M.; Solomon, J.M. Dynamic graph cnn for learning on point clouds. *ACM Trans. Graph.* **2019**, *38*, 1–12. [CrossRef]
18. Sarlin, P.E.; DeTone, D.; Malisiewicz, T.; Rabinovich, A. SuperGlue: Learning Feature Matching With Graph Neural Networks. In Proceedings of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, USA, 13–19 June 2020; pp. 4937–4946.
19. Sinkhorn, R. A relationship between arbitrary positive matrices and doubly stochastic matrices. *Ann. Math. Stat.* **1964**, *35*, 876–879. [CrossRef]
20. Charles, R.Q.; Su, H.; Kaichun, M.; Guibas, L.J. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 77–85.
21. Yew, Z.J.; Lee, G.H. REGTR: End-to-End Point Cloud Correspondences With Transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 6677–6686.
22. Zhou, Q.Y.; Park, J.; Koltun, V. Fast global registration. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Berlin/Heidelberg, Germany, 2016; pp. 766–782.
23. Mei, G.; Poiesi, F.; Saltori, C.; Zhang, J.; Ricci, E.; Sebe, N. Overlap-guided Gaussian Mixture Models for Point Cloud Registration. In Proceedings of the 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 2–7 January 2023; pp. 4500–4509. [CrossRef]
24. Zhou, Q.Y.; Park, J.; Koltun, V. Open3D: A modern library for 3D data processing. *arXiv* **2018**, arXiv:1801.09847.