



Article

NrtNet: An Unsupervised Method for 3D Non-Rigid Point Cloud Registration Based on Transformer

Xiaobo Hu ¹, Dejun Zhang ^{1,*}, Jinzhi Chen ¹, Yiqi Wu ¹ and Yilin Chen ²

¹ School of Computer Science, China University of Geosciences, Wuhan 430074, China; huxiaobo@cug.edu.cn (X.H.); chenjinzhi@cug.edu.cn (J.C.); wuyq@cug.edu.cn (Y.W.)

² Hubei Key Laboratory of Intelligent Robot (Wuhan Institute of Technology), Wuhan 430205, China; yilinchen@wit.edu.cn

* Correspondence: zhangdejun@cug.edu.cn

Abstract: Self-attention networks have revolutionized the field of natural language processing and have also made impressive progress in image analysis tasks. Corrnnet3D proposes the idea of first obtaining the point cloud correspondence in point cloud registration. Inspired by these successes, we propose an unsupervised network for non-rigid point cloud registration, namely NrtNet, which is the first network using a transformer for unsupervised large deformation non-rigid point cloud registration. Specifically, NrtNet consists of a feature extraction module, a correspondence matrix generation module, and a reconstruction module. Feeding a pair of point clouds, our model first learns the point-by-point features and feeds them to the transformer-based correspondence matrix generation module, which utilizes the transformer to learn the correspondence probability between pairs of point sets, and then the correspondence probability matrix conducts normalization to obtain the correct point set corresponding matrix. We then permute the point clouds and learn the relative drift of the point pairs to reconstruct the point clouds for registration. Extensive experiments on synthetic and real datasets of non-rigid 3D shapes show that NrtNet outperforms state-of-the-art methods, including methods that use grids as input and methods that directly compute point drift.



Citation: Hu, X.; Zhang, D.; Chen, J.; Wu, Y.; Chen, Y. NrtNet: An Unsupervised Method for 3D Non-Rigid Point Cloud Registration Based on Transformer. *Sensors* **2022**, *22*, 5128. <https://doi.org/10.3390/s22145128>

Academic Editor: Marcin Woźniak

Received: 9 May 2022

Accepted: 6 July 2022

Published: 8 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: NrtNet; self-attentive; transformer; unsupervised; non-rigid point cloud; registration

1. Introduction

The 3D object has better flexibility, and with the continuous development of 3D sensing technology in recent years, the 3D point cloud has been widely used in various fields, such as virtual reality [1], autonomous driving [2], and augmented reality [3]. Since LIDAR scanned point clouds do not correspond with each other, this great inconveniences downstream tasks of point cloud classification [4,5], segmentation [6,7], registration [8,9], and reconstruction [10,11].

Non-rigid point cloud registration can be divided into similar registration [12,13] and affine registration [14,15]. Similar registration is mostly based on ICP to improve the registration of point clouds by changing the optimization objective function and increasing the correspondence, while affine registration ensures that the parallelism between the lines remains unchanged during the transformation process. Corrnnet3D [16] proposes an alignment idea of finding the correspondence between the point clouds first, and then reconstructing the point clouds, which gives us inspiration. However, most of these methods require large-scale labeled data. Labeled data requires a lot of time and cost, which also promotes the development of unsupervised methods. In our work, we focus on unsupervised large deformation non-rigid point cloud registration, which means that only 3D point cloud data is required as input.

Figure 1 illustrates our idea that if we can align two sets of point clouds *A* and *B*, the registration process between the point clouds becomes easy. We permute the point clouds by a transformer because the transformer is better at handling natural language

correspondences [17,18]. We design a reconstruction module to reconstruct $A_{re_order} \in \mathbb{R}^{n \times 3}$ to B , which is more meaningful than reconstructing directly from A to B .

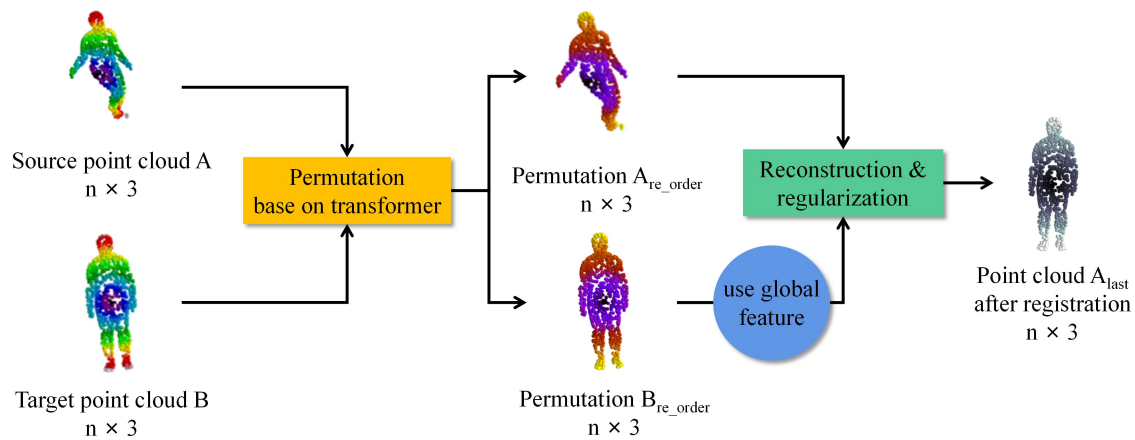


Figure 1. The idea of our proposed NrtNet. The point cloud is first rearranged using transformer, and then the permuted point cloud is reconstructed to achieve the registration.

Based on the above idea, we propose an unsupervised transformer-based registration network (NrtNet) for large deformation non-rigid point clouds. We propose a transformer-based permutation process. Specifically, this permutation process uses the encoder and decoder of the transformer to generate a point set correspondence matrix, which represents the correspondence between the source point cloud A and target point cloud B . During the training process, the global features of the target point cloud and the permutation source point cloud A_{re_order} are fed to the reconstruction module to obtain the reconstructed point cloud. The reconstruction module drives the learning of the correspondence matrix and the relative drift by optimizing the reconstruction error and additional regularization terms to achieve registration.

In general, our main contributions are:

- We propose a transformer-based point cloud correspondence learning framework for learning dense correspondences between point clouds, and we are the first to introduce a transformer into the field of non-rigid point cloud registration.
- Our network eliminates the reliance on ground truth and achieves unsupervised learning of non-rigid point cloud registration in an end-to-end manner, and has a better registration effect for different objects.
- Experiments demonstrate that NrtNet has significant advantages in non-rigid point cloud registration. In particular, it is superior to methods that directly compute the drift of coherent points between point clouds and methods that use a grid as input.

2. Related Work

In this section, we introduce the application of point clouds in deep learning, the study of non-rigid point cloud registration, and the development of transformer-based deep learning.

2.1. Deep Learning on Point Cloud

Compared with well-developed image-based deep learning methods, point cloud-based deep learning methods are more challenging and still in the developing stage due to the irregularity and disorder of point clouds. Three-dimensional data can be displayed in various forms, such as 2D multi-views, unstructured point clouds, voxelized volumes, etc. Voxelization methods convert 3D data into regular volume occupancy voxels, resulting in structured volumes that are well suited for 3D CNNs. Early point cloud tasks used end-to-end 3D convolutional networks [19–21]. Due to the sparse volume of 3D data and expensive 3D convolutions, voxelized representations are limited by resolution, and [22,23]

effectively solve the voxelized resolution problem. Qi et al. [24] projected 3D data into multiple 2D views and used the popular 2D CNN to process it.

PointNet [25] learns features directly from the point cloud, maps the point cloud to higher dimensions before aggregation, and takes symmetry operations in higher dimensions. Mapping to higher dimensions generates redundant information, which can be captured by maximization operations to avoid geometric information loss. PointNet only uses MLP and max-pooling and does not have the ability to capture local structural defects, which PointNet++ [26] improves upon. DGCNN [27] designs an EdgeConv that can efficiently extract features of local shapes of point clouds while still maintaining alignment invariance. Later, researchers investigated the use of merged features to represent the overall features and pointwise features [26] or more sophisticated RNN-based methods to extract features [28,29]. MortonNet [30] extracts more effective features based on learning an ordered sequence of point clouds. FoldingNet [31] learns to deform predefined 2D regular meshes into 3D shapes, AtlasNet [32] and 3D-Coded [33] are also based on the deformation of their networks, and they use fixed template deformations to reconstruct the point cloud or mesh.

2.2. Non-Rigid Point Cloud Registration

The development of registration optimization algorithms has attracted the attention of many researchers, and these algorithms are used to refine geometric transformations during iterations. The Iterative Closest Point (ICP) algorithm [34] is a classic case in rigid registration. The ICP initializes the estimation of the rigidity function and then iteratively selects the corresponding point to revise the transformation. However, ICP is not able to handle non-rigid point cloud variations efficiently due to the influence of initial values. Non-rigid point cloud registration can be divided into parametric registration and non-parametric registration by target transformation. The TPS-RSM algorithm [35] in parametric registration estimates the parameters of the non-rigid transformation with the penalty of the second derivative.

For classical nonparametric methods, coherent point drift (CPD) [36] introduces a process of fitting a Gaussian mixed likelihood that aligns the source point set with the target point set. Ma et al. [37] proposed the importance of exploiting local and global structures in non-rigid point set registration. CPD-Net [38] uses deep neural networks to fit functions that can adapt to geometric transformations of varying complexity. DispVoxNets [39] converts point clouds to voxels for nonlinear deformation in a supervised manner. PR-Net [40] introduces point-set shape features that determine the correlation between the source and target point set to predict the transformation, allowing source and target point sets to be statistically aligned. CorrNet3D [27] uses a new efficient de-smoothing module to optimize the point set pairs with better results. Ma et al. [41] used a robust transformation estimation method based on streamwise regularization for non-rigid point set registration, and the spatial transformation between two point sets is estimated by iteratively recovering the point correspondence. However, all extant methods do not perform well for point cloud registration with large deformations, and most of them rely on ground truth. These methods also work poorly for data with non-corresponding point sets. Our method eliminates the reliance on ground truth and has better registration results for large deformations and non-corresponding data sets.

2.3. Deep Learning Based on Transformer

CNN is a standard network model in computer vision [42], with the introduction of AlexNet [43], CNN began to become the dominant network model. Transformer and Self-Attention models revolutionized natural language processing [44,45], and some studies used Self-attention and Transformer to replace some or all of the spatial convolutional layers in the popular ResNet [46]. The encoder-decoder design in Transformer has recently been applied to object detection and instance segmentation tasks [47], and ViT [48] directly applies transformer to non-overlapping medium-sized image blocks for image classification.

AiR [49] is the first transformer-based image registration method. Point Transformer [50] is the first to introduce a transformer into the 3D point cloud domain, proposes a highly expressive point transform layer, and uses transformer to construct a high-performance point transform network for point cloud classification and dense prediction. Point Cloud Transformer [51] proposes a new transformer-based point cloud learning framework PCT, and uses implicit Laplace operators and normalized refinement to offset attention. Our method uses the transformer to derive correspondences between points to improve the effectiveness of the final registration.

3. Framework

3.1. Overview

As shown in Figure 2, NrtNet is composed of three modules: the feature extraction module, the transformer module, and the point cloud reconstruction module. Firstly, in order to get the point cloud features, the source point cloud $A \in \mathbb{R}^{n \times 3}$ and the target point cloud $B \in \mathbb{R}^{n \times 3}$ are fed into the point cloud feature extraction module to generate point cloud features $F_a \in \mathbb{R}^{n \times d}$ and $F_b \in \mathbb{R}^{n \times d}$, where d is the feature dimension, the pointwise feature of the point cloud is obtained by setting d to the same dimension as the number of point clouds. After that, the pointwise features F_a and F_b are fed into the transformer module, which finds the correspondence between the source and target point clouds. The point set correspondence matrix $P \in \mathbb{R}^{n \times n}$ is obtained to represent the point set correspondence, the parameter $p_{ij} = 1$ of P represents the i -th point a_i of the source point cloud and the j -th point b_j of the target point cloud. The transformer module is composed of a transformer encoder and a transformer decoder. The source point cloud A is permuted using P to obtain $A_{re_order} \in \mathbb{R}^{n \times 3}$. Finally, the global features of the target point cloud $V_b \in \mathbb{R}^d$ and the permuted source point cloud A_{re_order} are fed into the reconstruction module to obtain A_{last} , which is similar to the target point cloud B . The global features V_b are aggregated from the pointwise features. As in most papers, we optimize our model parameters by minimizing the similarity between the reconstructed point cloud and the target point cloud. To better learn the correspondence between point sets, we regularize the point cloud correspondence matrix and then minimize it to obtain the optimal point set correspondence. We can express it as follows:

$$M^{optimal} = argmin \left(\|B - A_{last}\|_F^2 + \|A - B_{last}\|_F^2 + G(P) \right), \quad (1)$$

where $A_{last} \in \mathbb{R}^{n \times 3}$ and $B_{last} \in \mathbb{R}^{n \times 3}$ are the point clouds after the registration, and $\|\cdot\|_F$ represents the Frobenius parametric matrix. $G(P)$ is a regularization operation on the corresponding matrix.

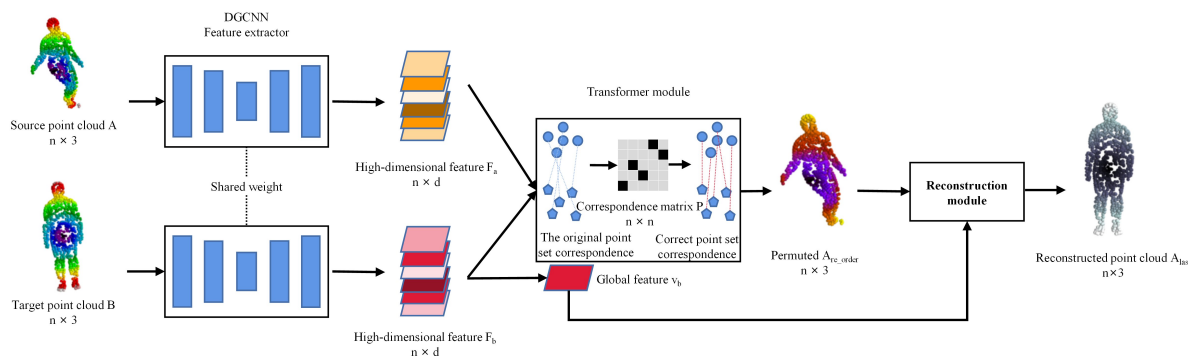


Figure 2. NrtNet is an unsupervised, end-to-end network for non-rigid point cloud registration. The source point cloud $A \in \mathbb{R}^{n \times 3}$ and the target point cloud $B \in \mathbb{R}^{n \times 3}$ are fed into the feature extraction module and the transformer module to generate the point set correspondence matrix $P \in \mathbb{R}^{n \times n}$. Then, the permuted point cloud is fed into the reconstruction module to generate the exact same point cloud A_{last} as B , which achieves the purpose of registration.

3.2. Feature Extraction Module

For the feature extraction module, instead of using the traditional PointNet and PointNet++, we use a DGCNN with shared parameters to map points A and B to high-dimensional features. DGCNN uses edge convolution, EdgeConv to dynamically build graph structures on each layer of the network, using each point as a centroid to characterize its edge with each neighboring point feature, and then aggregates these features to obtain a new representation of that point. Firstly, DGCNN defines the edge feature representation as:

$$e_{ij} = h(x_i, x_i - x_j) \quad (2)$$

where h is that the edge convolution operation considers both the global information x_i , and the local neighborhood information $x_i - x_j$, and $x_i \in \mathbb{R}^{1 \times d}$ is the feature extracted by the i -th point fed into the edge convolution. Then, aggregating the edge features to obtain the feature e_{ij}^{l+1} over the l -th layer is expressed as:

$$e_{ij}^{l+1} = \chi_{x_j \in \Omega_i} h((x_i, x_i - x_j)) \quad (3)$$

where χ indicates that the aggregation operation consists of the MLP and max-pooling, Ω denotes the set of point-set pairs formed between the remaining points centered at point x_i and the center point. After the multi-layer edge convolution, MLP and max-pooling operations, we can extract the pointwise features $F_a \in \mathbb{R}^{n \times d}$ and $F_b \in \mathbb{R}^{n \times d}$. The pointwise features are fed into a max-avg-pooling layer to get the global feature $V_a \in \mathbb{R}^d$ and $V_b \in \mathbb{R}^d$, which prepares for the later reconstruction.

3.3. Transformer Module

Because of the effectiveness of the transformer for word correspondence in NLP, we use the transformer to correspond to the point set. As shown in Figure 3, the transformer module consists of three parts: the transformer encoder, the transformer decoder, and a smooth module. The transformer module inputs the features of the source and target point clouds to learn the point set correspondence matrix $P \in \mathbb{R}^{n \times n}$, which can explicitly represent the correspondence between any two points in A and B . The $p_{ij} = 1$ in the matrix represents the i -th point a_i of the source point cloud A and the j -th point b_j of the target point cloud B are corresponding. This matrix is an inverted matrix. There are only two cases of correspondence between source and target point clouds, so this matrix should have only 0 or 1. The points in the source point cloud should correspond to the points in the target point cloud one by one, and each row and column of the matrix should have only one 1. The transformer module uses the transformer to find the similarity between point clouds, i.e., the probability matrix of the point clouds P_{rand} , which represents the probability of correspondence between the point clouds. Finally, a smoothing process is applied to this probability matrix to obtain an exact inverted binary matrix P .

The transformer encoder that references Point Transformer [50] is shown in Figure 4. The feature $f_i^l \in \mathbb{R}^{1 \times d}$ of the i -th point is fed into the standard scalar dot product attention layer. The standard scalar dot product attention layer is expressed as:

$$y_i = \sum_{f_j^a \in F_a} \text{softmax} \left(\gamma \left(\varphi(f_i^a)^T - \psi(f_j^a) \right) + \delta \right) \alpha(f_j^a) \quad (4)$$

where φ, ψ, α is the feature transform layer MLP, δ is a position encoder. γ is a mapping function. γ as a vector to represent the global features of the point cloud. The mapping function γ consists of an MLP, two linear layers, and a Relu activation function. Attention vectors are generated for later feature aggregation, feature transformation of φ minus ψ to obtain the vector relationship between them. Finally, the transformation features y_i are obtained by a softmax regularization function.

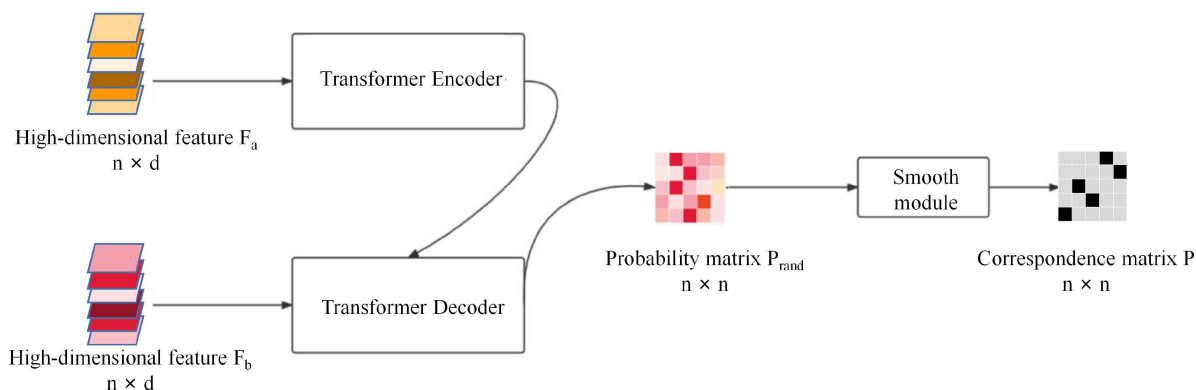


Figure 3. Transformer module. The probability matrix P_{rand} can be obtained by feeding the high-dimensional features of the point cloud $F_a \in \mathbb{R}^{n \times d}$ and $F_b \in \mathbb{R}^{n \times d}$ into the transformer encoder and transformer decoder, respectively. Then, the probability matrix P_{rand} is fed into the smooth module to obtain the inverted exact correspondence matrix P .

Due to the disordered nature of point clouds and their irregular embedding in the entire vector space, self-supervision is performed using the position of the point cloud itself. The positional encoding δ is added to the transformed feature α . In this way, the transformation feature is expressed as:

$$y_i = \sum_{f_i^a \in F_a(i)} softmax(\gamma(\varphi(f_i^a)^T - \psi(f_j^a)) + \delta) \cdot (\alpha(f_j^a) + \delta), \tag{5}$$

where $F_a(i) \in F_a$ is the feature of k neighboring points around the sought point $f_i^a \in \mathbb{R}^{1 \times d}$. Self-attention is applied to each data point in the local domain. In 3D point cloud alignment, the 3D point cloud itself comes with position information, and the trainable parametric position encoder can be expressed as:

$$\delta = \theta(p_j - p_i), \tag{6}$$

where p_i represent the i -th point, p_j represents the j -th point around the i -th point, and θ has the same structure as γ . This position encoder has good effect enhancement for both attention generation and feature transformation.

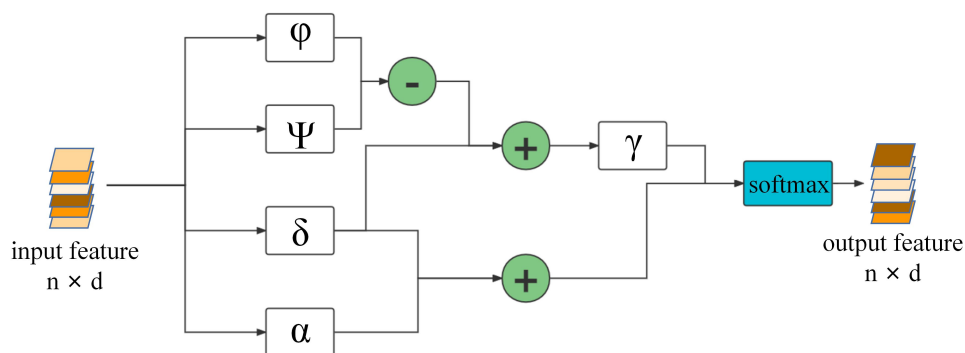


Figure 4. Transformer Encoder. φ, ψ is a linear layer, α is an mlp and they are all feature transform layers, δ is a linear layer which is a position encoder and γ is a mapping function.

As shown in Figure 5, the transformed features are fed into the transformer decoder and smoothing module to generate the point set correspondence matrix P . First, the

global features need to be obtained by point-by-point features. The global feature can be expressed as:

$$f_{aa} = \sum_{i=1}^d f_{trans_i}^a, f_{bb} = \sum_{j=1}^d f_{trans_j}^b \quad (7)$$

each transformed feature $f_{trans_i}^a$ is summed to obtain a one-dimensional global feature f_{aa} , so that we can find the global features $F_{aa} \in \mathbb{R}^{n \times 1}$ and $F_{bb} \in \mathbb{R}^{n \times 1}$ of the source and target point clouds. To obtain the probability matrix, we first obtain the distances of F_{aa} and F_{bb} . The distance formula can be expressed as:

$$P_{dis} = (F_{aa} \cdot I)^T + F_{bb} \cdot I - 2(F_{aa} \cdot F_{bb})^T, \quad (8)$$

where I is a $1 \times n$ unit column vector. Equation (8) returns an $n \times n$ point set corresponding distance matrix. The larger the distance, the smaller the probability they correspond to. The probability matrix P_{rand} corresponding to its point set is obtained by inverting the distance matrix. The probability matrix can be expressed as:

$$P_{rand} = \frac{1}{P_{dis}}, \quad (9)$$

since there are only two cases for the correspondence of point sets, the probability matrix cannot effectively represent the correspondence between point sets. It can only represent the corresponding probability between point sets. We need to smooth this matrix, and we refer to Corrnnet3D. Each row of the probability matrix should follow a normal distribution with mean μ_i and variance σ_i , i.e., $p_{rand_{ij}} \sim N(\mu_i, \sigma_i^2)$. In order to better filter the incorrect point set correspondence, we normalize this normal distribution $z_{ij} = (p_{rand_{ij}} - \mu_i) / \sigma_i$, z_{ij} obeys the standard normal distribution $z_{ij} \sim N(0, 1)$. Finally, we select the corresponding point set according to the threshold τ . The number corresponding to the correct point set is z_{num} . For the points close to the middle, it should find a larger number of corresponding points, and z_{num} should also obey the normal distribution. It obeys the three-sigma rule. The probability of the value in $[\mu_{z_{num}} - 3\sigma_{z_{num}}, \mu_{z_{num}} + 3\sigma_{z_{num}}]$ is 0.9973, which is almost 1. The softmax operation on z_{num} can be calculated to obtain the correct point set corresponding matrix P .

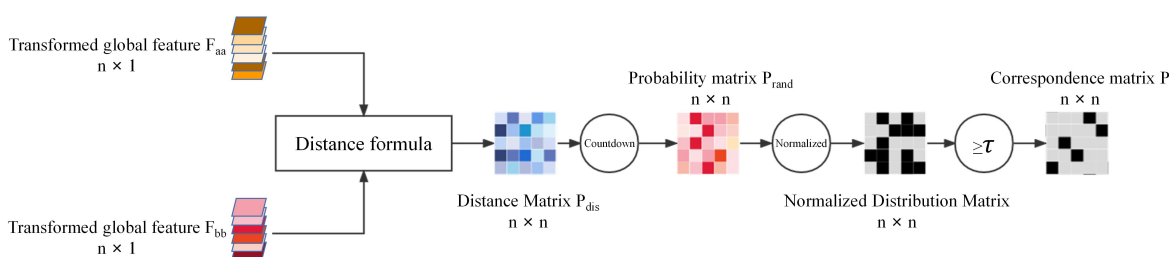


Figure 5. Transformer decoder and the smooth module. These two modules convert the transformed features obtained from the transformer encoder into an exact point set correspondence matrix.

3.4. The Reconstruction Module

When the correct correspondence labels between point cloud A and B are given, the shape feature relationship between them can be learned well, and thus it is easy to learn the amount of drift between point sets. FoldingNet [31] and AtlasNet [32] reconstruct the global features by stitching point on top of the 2D grid. CPD-Net [38] learns point-to-point drift by concatenating point and global features. As shown in Figure 6, the reconstruction module based on point correspondence is proposed.

Point clouds A and B are permuted by the point set correspondence matrix P . A and B after permuting can be expressed as:

$$A_{re_order} = P^T A, B_{re_order} = PB \quad (10)$$

The permuted source point cloud A_{re_order} correspond to the point of B one by one, so that the large deformation registration can be learned, which CPD-Net cannot learn. The relative drifts between A_{re_order} and B are learned by using the global features. As shown in Figure 6, A_{re_order} and the global feature $V_b \in \mathbb{R}^d$ are concatenated, and then the drift of each point is learned through three MLPs. The reconstructed point cloud A_{last} is the source point cloud A plus the drift. The module is able to efficiently learn the drift between points for the purpose of registration. The reconstruction module learns a displacement field function to estimate the geometric transformations and is able to predict the geometric transformations of the alignment between positional objects.

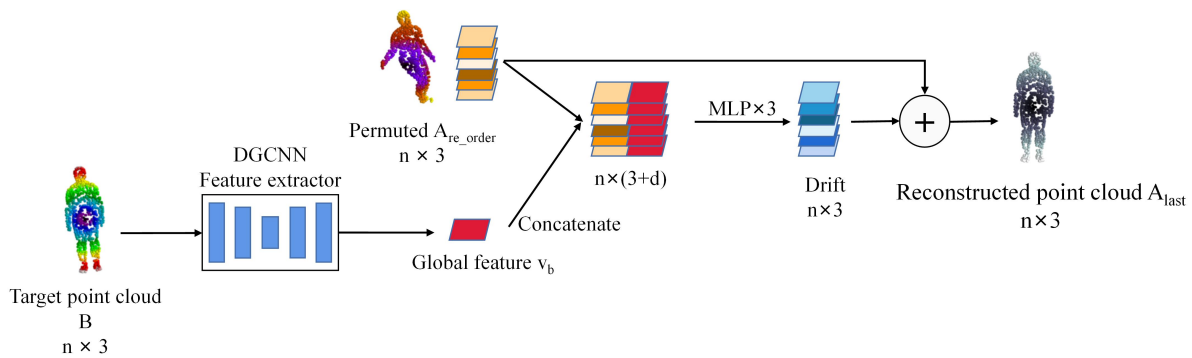


Figure 6. The reconstruction module concatenates the global features v_b of the target point cloud to each permuted point cloud A_{re_order} , and feeds them into the MLP to reconstruct the point cloud A_{last} .

3.5. Unsupervised Loss Function

The source point cloud A_{last} should be similar to the target point cloud B after registration. The Euclidean distance loss between B_{last} and A is added to the standard loss, which can better learn the relationship between A and B . According to the similarity of the source and target point clouds after deformation, the distance loss is expressed as:

$$\mathcal{L}_{dis} = \|B - A_{last}\|^2 + \|A - B_{last}\|^2 \quad (11)$$

Since the points in A and B should be in one-to-one correspondence, their correspondence matrix should be an inverted matrix. The transpose of the inverted matrix and its own dot product should be infinitely close to the unit matrix. Based on this property, the matrix optimization loss formula is expressed as:

$$\mathcal{L}_{mat} = \|P^T P - I_n\|^2, \quad (12)$$

where P is the correspondence matrix. I_n is an $n \times n$ unit matrix.

There are similar local features between the target point cloud B and the source point cloud after the permutation A_{re_order} , and similarly the source point cloud A and the target point cloud after the permutation B_{re_order} also have similar local features. Based on this property, the proximity similarity loss is expressed as:

$$\mathcal{L}_{pro} = \sum_{i=1}^n \left(\sum_{k \in \Omega_i^a} \frac{\|b_{re_i} - b_{re_j}\|_2^2}{\|a_i - a_k\|_2^2} + \sum_{l \in \Omega_i^b} \frac{\|a_{re_i} - a_{re_j}\|_2^2}{\|b_i - b_l\|_2^2} \right), \quad (13)$$

where Ω_i^k represents the set of k indexes around the i -th point in A , $b_{re_i} \subset B_{re_order}$ and $a_{re_i} \subset A_{re_order}$ are the points after rearrangement.

Finally, we aggregate these losses and the final loss is expressed as:

$$\mathcal{L} = \mathcal{L}_{dis} + \lambda \mathcal{L}_{mat} + \eta \mathcal{L}_{pro}, \quad (14)$$

where λ and $\eta > 0$ are superparameters to regulate the balance between several losses.

4. Experiment

In this section, the experimental results of NrtNet's non-rigid point cloud registration are presented. Details of the dataset and laboratory parameters used for training and testing are described in Section 4.1, and a brief introduction to the experimental evaluation method. In Section 4.2, a comparison of rigid point cloud registration results from different networks is discussed. In Section 4.3, the experimental results of non-rigid body methods in rigid registration are discussed. In Section 4.4, the registration results of NrtNet on small deformation datasets are presented. In Section 4.5, the effects of different losses on the experiments are compared. In Section 4.6, we show the registration effect of NrtNet on real scan data.

4.1. Experimental Setup

Dataset. We use the 200k sampled dataset from Surreal [33] as the unsupervised training datasets, and divide these 200k datasets into 100 random pairs for registration training. We used the 300 pairs dataset from Shrec [52] as the test dataset. We downsampled Shrec's dataset to 1024 grids and took the grid vertices as input to keep the variables constant. In order to compare the robustness of different datasets, we used the dataset of Bednarik, J et al. [53] including small deformation datasets of paper, tshirt, sweater, and cloth to learn for different data to ensure the reliability of NrtNet.

Evaluation. We reviewed a large amount of information on whether CPD-Net [38], DispVoxNets [39], or other articles such as PR-Net [40] have most of the evaluations as direct comparison of CD loss or subjective comparisons of the experimental result plots after registration. Almost none of them had registration again by finding correspondence for point pairs like we do, so we refer to Corrnnet3D's [16] evaluation method to evaluate the goodness of the model based on whether the point set corresponds to each other or not. The point correspondence rate is expressed as:

$$P_{rate} = \frac{1}{n} \|P \circ P_{gt}\|_1, \quad (15)$$

where \circ is the Hadamard product and $\|\cdot\|_1$ is the parametric matrix. P_{gt} is the ground truth of the point set corresponding to the matrix. We set the percentage of correct correspondence under different tolerances to compare the pros and cons of the method. The point correspondence rate under different fault tolerance is expressed as:

$$P_{corr} = \frac{r}{\max\{\|a_i - a_j\|_2 \mid \forall i, j\}}, \quad (16)$$

where r is the error tolerance radius.

Experimental parameters and configuration. We set the superparameter $\lambda = 0.1$ and $\eta = 0.01$. Our method was implemented in pytorch and our evaluation system was trained and tested on an NVIDIA GTX 1080 GPU. The learning-rate was 1×10^{-4} , batchsize was two, and we trained 50 epochs on the large Surreal dataset [33] and 500 epochs on the small deformed dataset [53].

4.2. Experimental Evaluation of Non-Rigid Point Cloud Registration

NrtNet was compared with unsupervised FlowNet3D [54], unsupervised Corrnnet3D [16], and unsupervised CPD-Net [38]. Figure 7 and Table 1 show a quantitative comparison of different methods, it can be seen that our method consistently outperforms other unsupervised methods. In particular, we have more significant performance advantages when comparing FlowNet3D and CPD-Net, and we also have some performance improvements when comparing Corrnnet3D. The point set correspondence rate of CPD-Net is low, and the registration effect is poor for large deformation datasets. The point set correspondence rate of CPD-Net is low, and the registration effect is poor for large deformation datasets. Although FlowNet3D has a high correspondence rate, its registration effect is very dependent on the dataset, and some test datasets have a good registration effect, while some test datasets have a poor registration effect. Only NrtNet and the recently published Corrnnet3D have better registration results. Because NrtNet uses a transformer that is better than Corrnnet3D in point correspondence, it can still achieve better registration results for some datasets with larger deformations.

Table 1. Point set correspondence rates of different methods under different fault tolerance rates in non-rigid point cloud registration.

Method	0% Error Tolerance	10% Error Tolerance	20% Error Tolerance
CPD-Net	0.3311	6.8212	24.8963
FlowNet3D	1.2133	19.7614	41.3494
Corrnnet3D	2.0494	25.68	48.8636
NrtNet	2.6889	30.0429	51.8758

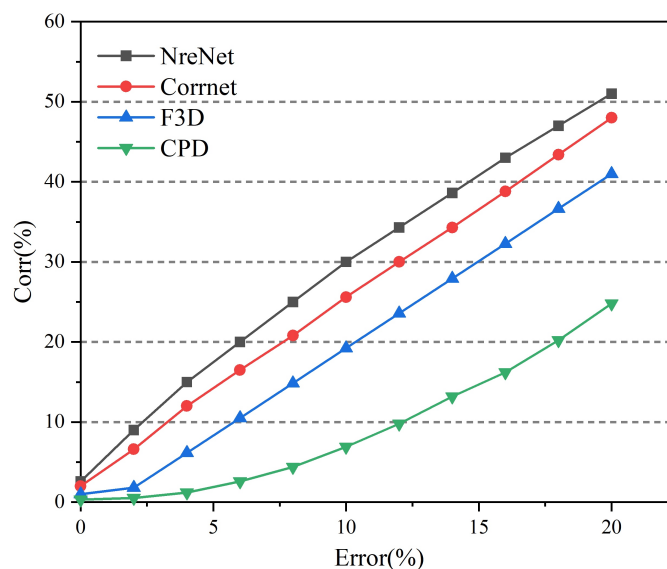


Figure 7. Quantitative comparison of point set correspondence rates for non-rigid registration under different methods.

Figure 8 shows the qualitative comparison results. NrtNet suffers less from unsupervised large-deformation non-rigid point cloud registration and can generate a point cloud with accurate correspondence. In contrast, CPD-Net and FlowNet3D are affected by large deformation, which makes their correspondence deviate and cannot achieve effective registration when the target point cloud varies greatly from the source point cloud. NrtNet learns the point set correspondence between the target and source point cloud, and thus can effectively make the registration effect better. Our network can further enhance the robustness to the degree of deformation by learning the specific type.

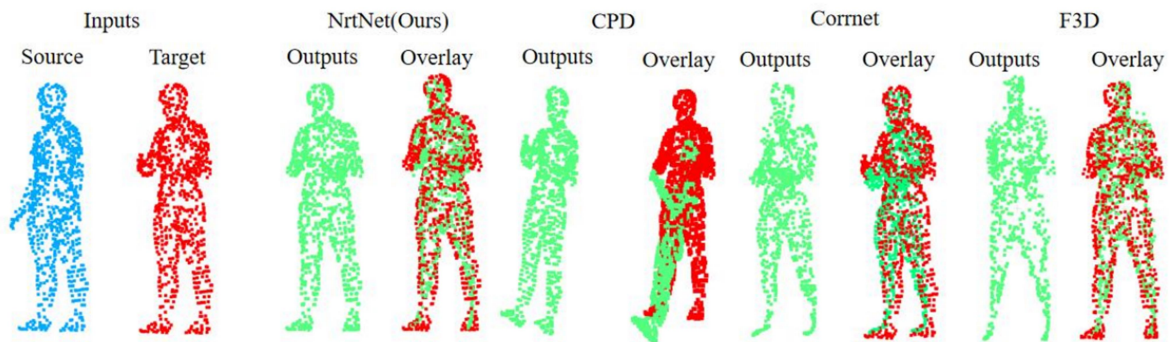


Figure 8. In a qualitative comparison between Nrtnet and other methods in a large deformed human pose, the experiments show the effectiveness of different methods for non-rigid point cloud registration.

4.3. Experimental Evaluation of Rigid Point Cloud Registration

We use the non-rigid registration method to register the rigid point cloud, and compare the effect of our method with FlowNet3D, Corrnnet3D, and CPD-Net on the rigid point cloud registration. Figure 9 and Table 2 show our method and other methods compared with different fault tolerances. It can be seen from the table that our method has the best results under the same fault tolerance, Corrnnet3D has a great improvement for FlowNet3D, and our method also has improvement for Corrnnet3D. Compared with non-rigid point cloud registration, the unsupervised registration effect of CPD-Net in rigid point cloud registration has little improvement, while our method NrtNet has better registration effect and non-rigid point cloud registration in rigid point cloud registration.

Table 2. Point set correspondence rates of different methods under different fault tolerance rates in rigid point cloud registration.

Method	0% Error Tolerance	10% Error Tolerance	20% Error Tolerance
CPD-Net	0.1769	9.6191	24.286
FlowNet3D	10.8945	71.5576	90.368
Corrnnet3D	12.4062	80.895	95.61
NrtNet	12.7793	85.6191	96.247

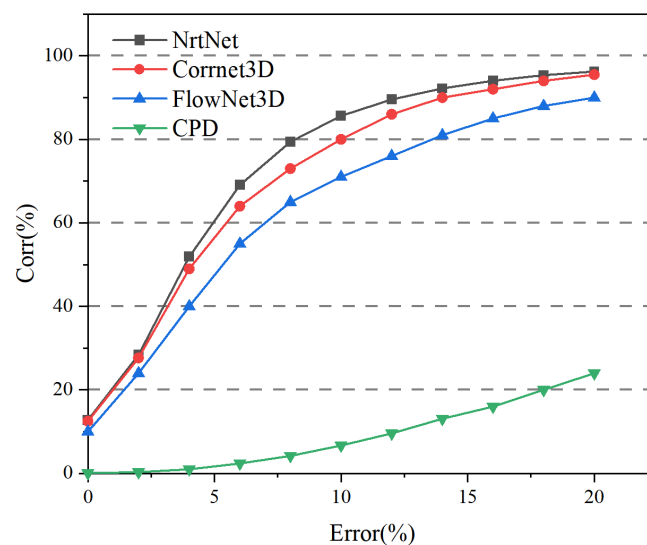


Figure 9. Quantitative comparison of point set correspondence rates for rigid registration under different methods.

4.4. Comparison between Different Datasets

NrtNet was tested on the dataset of Bednarik, J et al. [53] for learning and registration to test the stability on different datasets. The dataset was divided into a training set and a test set in a ratio of 8:2. As shown in Figure 10, NrtNet has better registration for small deformation datasets, not only for learning the deformation part efficiently, but also for rigid transformations of deformed point clouds. NrtNet not only has a good registration effect on large deformation datasets, but also has good registration effects on small deformation datasets compared with existing methods. This makes the registration more efficient to first obtain the point set correspondence through the transformer.

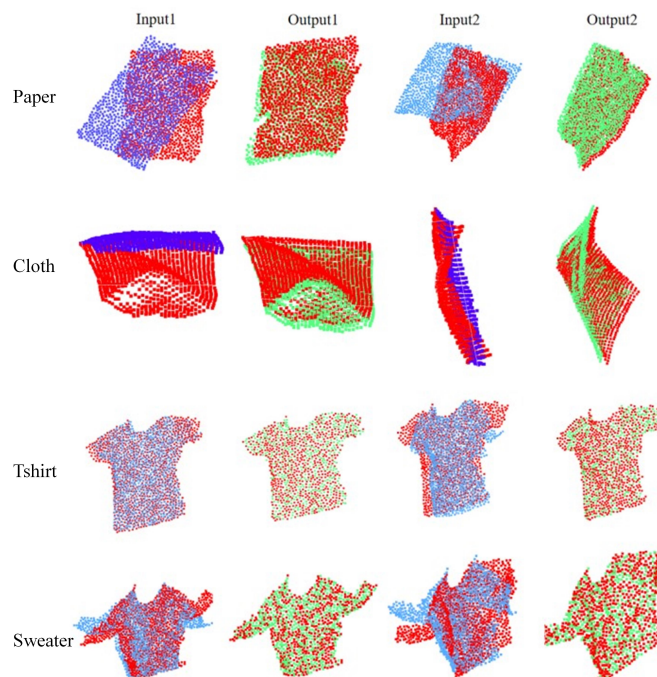


Figure 10. Registration performance of NrtNet in small deformation dataset paper, cloth, sweater, and t-shirt.

4.5. Comparison of Different Losses

In the experiment, we compared the difference between the Euclidean distance \mathcal{L}_{dis} and the CD Loss, and we also showed the improvement of the Euclidean distance and the cd distance by adding optimization losses $\mathcal{L}_{mat} + \mathcal{L}_{pro}$. Figure 11 and Table 3 show the point-set correspondence rate at different losses, and it can be seen that the loss of NrtNet achieves the best results with the same fault tolerance. It can be seen that the improvement of $\mathcal{L}_{mat} + \mathcal{L}_{pro}$ to \mathcal{L}_{dis} is very obvious by comparing \mathcal{L}_{dis} and $\mathcal{L}_{dis} + \mathcal{L}_{mat} + \mathcal{L}_{pro}$. When CDloss and \mathcal{L}_{dis} are compared separately, CDloss has a certain improvement. When CDloss and $CDloss + \mathcal{L}_{mat} + \mathcal{L}_{pro}$ are compared separately, $\mathcal{L}_{mat} + \mathcal{L}_{pro}$ have little effect on CDloss, and the increase in correspondence rate is minimal. Experiments show that the loss of NrtNet can achieve the best experimental results.

Table 3. Correspondence rates of point sets for different losses under different error tolerance rates.

Loss	0% Error Tolerance	10% Error Tolerance	20% Error Tolerance
\mathcal{L}_{dis}	0.0214	2.4287	16.081
CDloss	0.0879	6.3662	24.9873
$CDloss + \mathcal{L}_{mat} + \mathcal{L}_{pro}$	0.1045	6.7686	30.7441
$\mathcal{L}_{dis} + \mathcal{L}_{mat} + \mathcal{L}_{pro}$	12.7793	85.6191	96.247

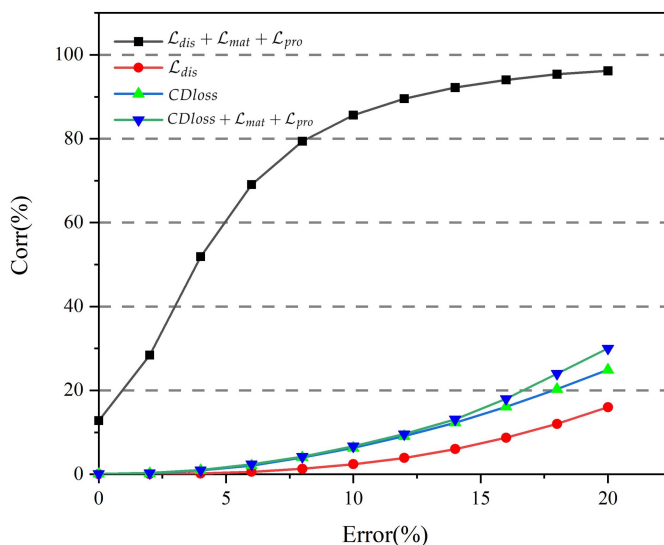


Figure 11. The point set correspondence rate under different losses. The experiments qualitatively compare the correspondence differences between NrtNet’s losses and ordinary losses.

4.6. Real Scan Data

This section shows the effect of NrtNet registration on real data. The experiments used Shrec’s human real scan dataset [52], and since the experiments were conducted without ground truth, it is hard to qualitatively evaluate the effects of the experiments. Figure 12 shows the final registration results of the experiments for a rational analysis of the results. NrtNet is able to effectively align the point cloud actions and shapes, and NrtNet is able to align any data without ground truth. As shown in Figure 12, The same color represents the correspondence of point sets, and NrtNet has better results for the correspondence of point set pairs. Although there are registration errors in some details, the experimental results are already much better than traditional non-rigid registration networks. The results are able to have better registration results for each movement.

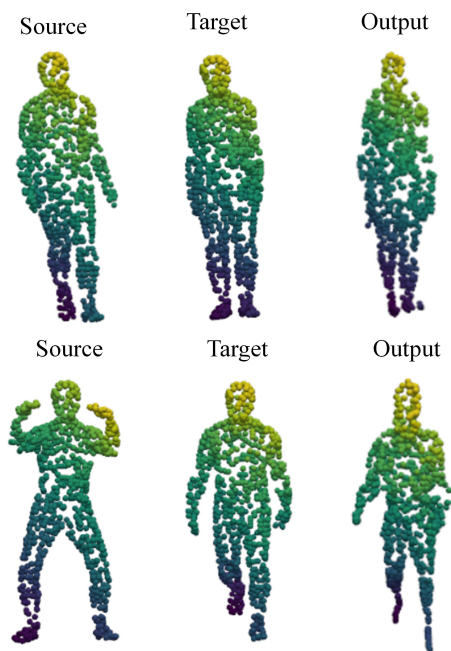


Figure 12. The registration effect of NrtNet on the real scan data.

5. Conclusions

We propose NrtNet, an unsupervised transformer-based registration architecture, which can learn the correspondence between pairs of large deformed point sets to effectively improve registration performance. NrtNet is much better than FlowNet3D in large deformation point cloud registration, and also significantly outperforms the state-of-the-art Corrnet3D. This shows that NrtNet can be used for most large deformation registration applications. We also show registration results on real scan data in the absence of ground truth, and still have good registration results. NrtNet has taken a long term step in large deformation non-rigid point cloud registration and eliminates the reliance on ground truth to conduct non-rigid point cloud registration.

In future work, NrtNet can be extended to voxels for non-rigid point cloud registration. Our correspondence may be inappropriate for the correspondence between points that are far apart. For this, we will sort the point cloud in future experiments and then use the transformer to do the point set correspondence, which corresponds to the word in NLP. Similarly, we believe that the registration effect can be improved to a certain extent after doing so. We believe that NrtNet can bring some help to other large scene point cloud registration, as well as human motion analysis and animal and plant growth analysis. Meanwhile, the model size of NrtNet can be further optimized to reduce training time.

Author Contributions: X.H. conceived of and designed the algorithm and the experiments; X.H. and D.Z. analyzed the data; X.H. wrote the manuscript; D.Z. supervised the research; J.C., Y.W. and Y.C. provided suggestions for the proposed method and its evaluation and assisted in the preparation of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Nature Science Foundation of China (grant No. 61802355) and Hubei Key Laboratory of Intelligent Robot (HBIR 202105).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Our experiments employ open datasets in [33,52,53].

Acknowledgments: We would like to thank the anonymous reviewers for their constructive and valuable suggestions on the earlier drafts of this manuscript.

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

References

1. Apostolopoulos, J.G.; Chou, P.A.; Culbertson, B.; Kalker, T.; Trott, M.D.; Wee, S. The road to immersive communication. *Proc. IEEE* **2012**, *100*, 974–990. [[CrossRef](#)]
2. Raviteja, T.; Vedaraj, I.R. An introduction of autonomous vehicles and a brief survey. *J. Crit. Rev.* **2020**, *7*, 196–202.
3. Silva, R.; Oliveira, J.C.; Giraldo, G.A. Introduction to augmented reality. *Natl. Lab. Sci. Comput.* **2003**, *11*, 1–11.
4. Hackel, T.; Savinov, N.; Ladicky, L.; Wegner, J.D.; Schindler, K.; Pollefeys, M. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv* **2017**, arXiv:1704.03847.
5. Reisi, A.R.; Moradi, M.H.; Jamasb, S. Classification and comparison of maximum power point tracking techniques for photovoltaic system: A review. *Renew. Sustain. Energy Rev.* **2013**, *19*, 433–443. [[CrossRef](#)]
6. Rabbani, T.; Van Den Heuvel, F.; Vosselmann, G. Segmentation of point clouds using smoothness constraint. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2006**, *36*, 248–253.
7. Brox, T.; Malik, J. Object segmentation by long term analysis of point trajectories. In Proceedings of the European Conference on Computer Vision, Heraklion, Greece, 5–11 September 2010; Springer: Berlin/Heidelberg, Germany, 2010; pp. 282–295.
8. Myronenko, A.; Song, X. Point set registration: Coherent point drift. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 2262–2275. [[CrossRef](#)]
9. Pomerleau, F.; Colas, F.; Siegwart, R. A review of point cloud registration algorithms for mobile robotics. *Found. Trends Robot.* **2015**, *4*, 1–104. [[CrossRef](#)]
10. Tang, P.; Huber, D.; Akinici, B.; Lipman, R.; Lytle, A. Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Autom. Constr.* **2010**, *19*, 829–843. [[CrossRef](#)]
11. Vosselman, G.; Dijkman, S. 3D building model reconstruction from point clouds and ground plans. *Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2001**, *34*, 37–44.

12. Zha, H.; Ikuta, M.; Hasegawa, T. Registration of range images with different scanning resolutions. In Proceedings of the SMC 2000 Conference Proceedings, 2000 IEEE International Conference on Systems, Man and Cybernetics. 'Cybernetics Evolving to Systems, Humans, Organizations, and Their Complex Interactions', Nashville, TN, USA, 8–11 October 2000; IEEE: Piscataway, NJ, USA, 2000; Volume 2, pp. 1495–1500; cat. no. 0.
13. Zinßer, T.; Schmidt, J.; Niemann, H. Point set registration with integrated scale estimation. In Proceedings of the International Conference on Pattern Recognition and Image Processing, Estoril, Portugal, 7–9 June 2005; pp. 116–119.
14. Amberg, B.; Romdhani, S.; Vetter, T. Optimal step nonrigid ICP algorithms for surface registration. In Proceedings of the 2007 IEEE Conference on Computer Vision and Pattern Recognition, Minneapolis, Minnesota, 17–22 June 2007; IEEE: Piscataway, NJ, USA, 2007; pp. 1–8.
15. Wang, C.; Shu, Q.; Yang, Y.; Yuan, F. Point cloud registration in multidirectional affine transformation. *IEEE Photonics J.* **2018**, *10*, 1–15. [[CrossRef](#)]
16. 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.
17. Vyas, A.; Katharopoulos, A.; Fleuret, F. Fast transformers with clustered attention. *Adv. Neural Inf. Process. Syst.* **2020**, *33*, 21665–21674.
18. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Berlin/Heidelberg, Germany, 2020; pp. 213–229.
19. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5828–5839.
20. Huang, J.; You, S. Point cloud labeling using 3d convolutional neural network. In Proceedings of the 2016 23rd International Conference on Pattern Recognition (ICPR), Cancun, Mexico, 4–8 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 2670–2675.
21. Maturana, D.; Scherer, S. Voxnet: A 3d convolutional neural network for real-time object recognition. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; IEEE: Piscataway, NJ, USA, 2015; pp. 922–928.
22. Engelcke, M.; Rao, D.; Wang, D.Z.; Tong, C.H.; Posner, I. Vote3deep: Fast object detection in 3d point clouds using efficient convolutional neural networks. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1355–1361.
23. Graham, B. Spatially-sparse convolutional neural networks. *arXiv* **2014**, arXiv:1409.6070.
24. Qi, C.R.; Su, H.; Nießner, M.; Dai, A.; Yan, M.; Guibas, L.J. Volumetric and multi-view cnns for object classification on 3d data. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 5648–5656.
25. Qi, C.R.; Su, H.; Mo, K.; Guibas, L.J. Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 652–660.
26. Qi, C.R.; Yi, L.; Su, H.; Guibas, L.J. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–10.
27. Phan, A.V.; Le Nguyen, M.; Nguyen, Y.L.H.; Bui, L.T. Dgcnn: A convolutional neural network over large-scale labeled graphs. *Neural Netw.* **2018**, *108*, 533–543. [[CrossRef](#)]
28. Huang, Q.; Wang, W.; Neumann, U. Recurrent slice networks for 3d segmentation of point clouds. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 2626–2635.
29. Ye, X.; Li, J.; Huang, H.; Du, L.; Zhang, X. 3d recurrent neural networks with context fusion for point cloud semantic segmentation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 403–417.
30. Thabet, A.; Alwassel, H.; Ghanem, B. Mortonnet: Self-supervised learning of local features in 3d point clouds. *arXiv* **2019**, arXiv:1904.00230.
31. Yang, Y.; Feng, C.; Shen, Y.; Tian, D. Foldingnet: Point cloud auto-encoder via deep grid deformation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 206–215.
32. Vakalopoulou, M.; Chassagnon, G.; Bus, N.; Marini, R.; Zacharaki, E.I.; Revel, M.P.; Paragios, N. Atlasnet: Multi-atlas non-linear deep networks for medical image segmentation. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Granada, Spain, 16–20 September 2018; Springer: Berlin/Heidelberg, Germany, 2018; pp. 658–666.
33. Groueix, T.; Fisher, M.; Kim, V.G.; Russell, B.C.; Aubry, M. 3d-coded: 3d correspondences by deep deformation. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 230–246.
34. 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, 12–15 November 1991; SPIE: Bellingham, WA, USA, 1992; Volume 1611, pp. 586–606.
35. Chui, H.; Rangarajan, A. A new algorithm for non-rigid point matching. In Proceedings of the Proceedings IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2000 (Cat. No. PR00662), Hilton Head Island, SC, USA, 13–15 June 2000; IEEE: Piscataway, NJ, USA, 2000; Volume 2, pp. 44–51.

36. Myronenko, A.; Song, X.; Carreira-Perpinan, M. Non-rigid point set registration: Coherent point drift. *Adv. Neural Inf. Process. Syst.* **2006**, *19*, 1–8.
37. Ma, J.; Zhao, J.; Yuille, A.L. Non-rigid point set registration by preserving global and local structures. *IEEE Trans. Image Process.* **2015**, *25*, 53–64.
38. Wang, L.; Li, X.; Chen, J.; Fang, Y. Coherent point drift networks: Unsupervised learning of non-rigid point set registration. *arXiv* **2019**, arXiv:1906.03039.
39. Shimada, S.; Golyanik, V.; Tretschk, E.; Stricker, D.; Theobalt, C. Dispvoxnets: Non-rigid point set alignment with supervised learning proxies. In Proceedings of the 2019 International Conference on 3D Vision (3DV), Quebec City, QC, Canada, 16–19 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 27–36.
40. Wang, L.; Chen, J.; Li, X.; Fang, Y. Non-rigid point set registration networks. *arXiv* **2019**, arXiv:1904.01428.
41. Ma, J.; Wu, J.; Zhao, J.; Jiang, J.; Zhou, H.; Sheng, Q.Z. Nonrigid point set registration with robust transformation learning under manifold regularization. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *30*, 3584–3597. [[CrossRef](#)]
42. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [[CrossRef](#)]
43. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* **2012**, *25*, 1–9. [[CrossRef](#)]
44. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 1–11.
45. Wu, F.; Fan, A.; Baeviski, A.; Dauphin, Y.N.; Auli, M. Pay less attention with lightweight and dynamic convolutions. *arXiv* **2019**, arXiv:1901.10430.
46. Tchapmi, L.; Choy, C.; Armeni, I.; Gwak, J.; Savarese, S. Segcloud: Semantic segmentation of 3d point clouds. In Proceedings of the 2017 International Conference on 3D Vision (3DV), Qingdao, China, 10–12 October 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 537–547.
47. Dovrat, O.; Lang, I.; Avidan, S. Learning to sample. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 2760–2769.
48. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929.
49. Wang, Z.; Delingette, H. Attention for Image Registration (AiR): An unsupervised Transformer approach. *arXiv* **2021**, arXiv:2105.02282.
50. Zhao, H.; Jiang, L.; Jia, J.; Torr, P.H.; Koltun, V. Point transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 16259–16268.
51. Guo, M.H.; Cai, J.X.; Liu, Z.N.; Mu, T.J.; Martin, R.R.; Hu, S.M. Pct: Point cloud transformer. *Comput. Vis. Media* **2021**, *7*, 187–199. [[CrossRef](#)]
52. Donati, N.; Sharma, A.; Ovsjanikov, M. Deep geometric functional maps: Robust feature learning for shape correspondence. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 8592–8601.
53. Bednarik, J.; Fua, P.; Salzmann, M. Learning to reconstruct texture-less deformable surfaces from a single view. In Proceedings of the 2018 International Conference on 3D Vision (3DV), Verona, Italy, 5–8 September 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 606–615.
54. Liu, X.; Qi, C.R.; Guibas, L.J. Flownet3d: Learning scene flow in 3d point clouds. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 529–537.