Robust Point Cloud Registration Framework Based on Deep Graph Matching

Kexue Fu Shaolei Liu Xiaoyuan Luo Manning Wang*
Digital Medical Research Center, School of Basic Medical Science, Fudan University
Shanghai Key Lab of Medical Image Computing and Computer Assisted Intervention
{kxfu18, liushaolei, xyluo19, mnwang}@fudan.edu.cn

Abstract

3D point cloud registration is a fundamental problem in computer vision and robotics. Recently, learning-based point cloud registration methods have made great progress. However, these methods are sensitive to outliers, which lead to more incorrect correspondences. In this paper, we propose a novel deep graph matching-based framework for point cloud registration. Specifically, we first transform point clouds into graphs and extract deep features for each point. Then, we develop a module based on deep graph matching to calculate a soft correspondence matrix. By using graph matching, not only the local geometry of each point but also its structure and topology in a larger range are considered in establishing correspondences, so that more correct correspondences are found. We train the network with a loss directly defined on the correspondences, and in the test stage the soft correspondences are transformed into hard one-to-one correspondences so that registration can be performed by singular value decomposition. Furthermore, we introduce a transformer-based method to generate edges for graph construction, which further improves the quality of the correspondences. Extensive experiments on registering clean, noisy, partial-to-partial and unseen category point clouds show that the proposed method achieves state-of-the-art performance. The code will be made publicly available at https://github.com/fukexue/RGM.

1. Introduction

Rigid point cloud registration is a task that finds a rigid transformation to align two point clouds, and it has long been a fundamental task in computer vision and robotics, with many important applications, such as autopilot [21, 19, 36], surgical navigation [44] and SLAM [13, 9]. There are two interlocked subproblems in point cloud registration: finding the transformation to align the two point clouds and finding the correspondences between the points [17]. Although when the solution to one subproblem is known, the other subproblem can be easily solved, it is difficult to solve both subproblems together. Point cloud registration becomes even harder when there are outliers, which are the points with no correspondences in the other point cloud. Outliers may come from the imperfectness of the sensors used to collect the point clouds or situations in which the two point clouds to be registered are not fully overlapped.

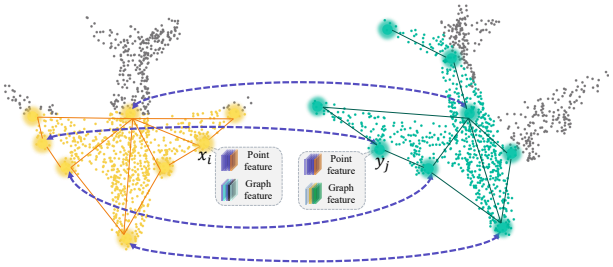


Figure 1. The idea of point cloud registration based on graph matching. Dashed lines represent correspondences. Point features and graph features are the features extracted directly through points and the features extracted based on graphs, respectively. The two points x_i and y_j have similar point features because they have similar local geometries, but they have different graph features because the graph topologies around them are different, so they are not mismatched when graph-based matching is used.

Iterative closest point (ICP) [3, 31] is arguably the most widely used method for rigid point cloud registration, which starts from an initial transformation and alternately updates the correspondences and transformation. One major limitation of ICP is that it can only converge to a local optimum near the initialization, and its convergence basin is fairly small, especially when there are noise and outliers. A series of global registration methods based on branch-and-bound (BnB) [42, 5, 6] have been proposed to alleviate the need for initialization by obtaining the global optimal solution, but the time-consuming BnB limits their practical applications. Another method for mitigating the need for ini-

*Corresponding author

tialization is by keypoint extraction and matching [28, 29]. Based on the correspondences established by matching key points, RANSAC-like methods can be explored for registration [28, 29]. However, the speed and accuracy of this type of method are sensitive to outliers and repetitive geometry [30]. Several recent methods integrate deep neural networks for establishing correspondences and a differentiable singular value decomposition (SVD) algorithm for calculating the transformation to build an end-to-end trainable network for point cloud registration, such as DCP [38], RPM-Net [43] and IDAM [18], and they do not need transformation initialization. These methods explore deep features to establish correspondences but the discrimination ability of the features extracted from point clouds is poor, as shown in Figure 1, which leads to a large proportion of incorrect correspondences and consequently devastates the registration accuracy.

In this paper, we propose a robust point cloud registration framework that utilizes deep graph matching to better handle outliers, and we denote it as RGM. By constructing graphs from point clouds to be registered and capturing the high-order structure of the graphs, RGM can find robust and accurate point-to-point correspondences to better solve the point cloud registration problem. To the best of our knowledge, this is the first time that deep graph matching has been applied to point cloud registration. RGM contains an end-to-end deep neural network, the first part of which is a feature extractor that extracts deep local features for each point by using its neighboring points. Instead of matching these local point features directly, we construct a graph for each of the two point clouds and embed [37] both the graph nodes (local features for each point) and graph structure (second-order or high-order structure) into node feature space. Then, we introduce an module consisting of an affinity layer, instance normalization and Sinkhorn to predict soft correspondences from the node features of the two graphs, and we denote it as AIS module. By using graph matching in the AIS module, not only the local geometry of each node but also its structure and topology in a larger range are considered in establishing correspondences so that more correct correspondences are found. In training, the binary cross-entropy loss between the predicted soft correspondences and the ground-truth correspondences are adopted, which directly promotes the network to learn better point-to-point correspondences. In testing, we use the linear assignment problem (LAP) solver [15] based on the Hungarian algorithm [16] to transform soft correspondences into one-to-one hard correspondences, and then SVD is employed to calculate the transformation from the hard correspondences. Similar to existing methods such as RPM-Net and ICP, we iteratively optimize the registration results.

Our main contributions are as follows:

- We propose using deep graph matching to solve the point cloud registration problem for the first time. Instead of only using the features of each point, graph matching can leverage the features of other nodes and the structural information of graphs when establishing correspondences so that it can better address the problem of outliers.

- We introduce the AIS module to establish reliable correspondences between nodes of two given graphs. The AIS module calculates an affinity matrix between any two nodes based on the embedded features, and by analyzing the affinity matrix globally and utilizing the Sinkhorn algorithm, it can effectively reduce the proportion of incorrect correspondences.

- We propose using a transformer to generate soft graph edges. In registering partial-to-partial point clouds, better correspondences can be established for the overlapping parts by utilizing the attention and co-attention mechanism in the transformer.

- Our method achieves state-of-the-art performance on clean, noisy, partial-to-partial datasets and unseen categories datasets.

2. Related Work

2.1. Traditional Registration Method

A large proportion of traditional methods need an initial transformation and find a locally optimal solution near the initialization, in which ICP [3, 31] is an early and representative method. ICP starts with an initial transformation and iteratively alternates between solving two trivial subproblems: finding the closest points as correspondences under current transformation and computing optimal transformation by SVD from found correspondences. Many variants have been proposed to improve ICP [26, 24, 27]. Nevertheless, ICP and its variants can only converge to a local optimum, and their success heavily relies on a good initialization. To improve the robustness to noise and outliers and enlarge the convergence basin, some methods transform point clouds into probability distributions and reformulate point cloud registration as matching two probability distributions, such as GMM [14] and HGMR [12]. These methods do not need to alternately solve correspondences and transformation, but their objective functions are nonconvex, so they still need a good initialization to avoid converging to a bad local optimum. Recently, a series of globally optimal methods based on BnB have been proposed, such as Go-ICP [42], GOGMA [5], GOSMA [6], and GoTS [20],but they are very slow and only practical in some limited scenarios. Another line of work avoids transformation initialization by establishing correspondences. They usually first extract keypoints from the original point clouds and construct

feature descriptors for them and then establish potential correspondences through feature matching [28, 29]. After that, RANSAC-like algorithms can be used to find the correct correspondences for registration. Different from RANSAC-like methods, FGR [49] optimizes a correspondence-based objective function by a graduated nonconvex strategy and achieves state-of-the-art performance in correspondence-based point cloud registration. However, correspondence-based methods are sensitive to duplicate structures and partial-to-partial point clouds because a large proportion of the potential correspondences will be incorrect in these scenarios. Specifically, the lack of good initialization, a large proportion of outliers and time constraints are still big challenges for traditional point cloud registration methods.

2.2. Learning-based Registration Method

The developments of deep learning on point clouds allow researchers to make good use of existing research, such as PointNet [25], and DGCNN [40], to extract point cloud features for downstream tasks. These studies have stimulated the interest of using deep learning in point cloud registration. One of the earliest works is PointNetLK [1], which calculates global feature descriptors of the two point clouds through PointNet and iteratively uses the IC-LK algorithm [2, 22] to minimize the distance between the two global feature descriptors to achieve registration. PCRNet [30] replaces the IC-LK algorithm in PointNetLK with a deep neural network. DCP [38] utilizes transformer [10, 35] to compute soft correspondences between two point clouds and utilizes a differentiable SVD algorithm to calculate the transformation. Although these methods have the advantages of being fast and some of them do not need transformation initialization, they cannot effectively handle partial-to-partial point cloud registration. PRNet [39] proposes a keypoint detector and uses the keypoint-to-keypoint correspondences in a self-supervised way to solve the partial-to-partial point cloud registration. DeepGMR [45] extracts pose-invariant correspondences between raw point clouds and Gaussian mixture model (GMM) parameters, and then recovers the transformation from the matched Gaussian mixture models. IDAM [18] integrates the iterative distance-aware similarity convolution module into the matching process, which can overcome the shortcomings of using inner products to obtain pointwise similarity. RPM-Net [43] proposes a network to predict optimal annealing parameters and uses annealing and Sinkhorn [32] to obtain soft correspondences from local features. Soft correspondences can increase robustness, but they lead to the decrease of registration accuracy, which is shown in our clean experiment. Although these methods can handle partial-to-partial point cloud registration to some extent, there is still room for improvement in their accuracy and robustness. The difference between our method and the existing learning-based methods is that we construct graphs from the original point clouds and merge structural information of the graphs into node features so that the nodes can be better matched.

Graph matching has been widely studied in computer vision and pattern recognition [11, 33, 48, 8]. Recently, learning-based graph matching has attracted considerable research interest [23, 46, 37], but, to the best of our knowledge, there is no research on using learning-based graph matching to solve the point cloud registration problem.

3. Problem Formulation

3D rigid point cloud registration refers to estimating a rigid transformation $\{\mathbf{R}, \mathbf{t}\}$ to align a source point cloud $\mathbf{X} = \left\{x_i \in \mathbf{R}^3 | i = 1, \cdots, N\right\}$ and a target point cloud $\mathbf{Y} = \left\{y_j \in \mathbf{R}^3 | j = 1, \cdots, M\right\}$, where $\mathbf{R} \in \mathbf{SO}(3)$, $\mathbf{t} \in \mathbf{R}^3$. N and M represent the number of points in $\mathbf{X}$ and $\mathbf{Y}$, respectively. The correspondences between points in $\mathbf{X}$ and $\mathbf{Y}$ are represented by matrix $\mathbf{C} = \{0, 1\}^{N \times M}$. If x_i and y_j are a pair of corresponding points, $\mathbf{C}_{i,j}$ is 1; otherwise, it is 0. We first consider the simple case where there are strict one-to-one correspondences between points in $\mathbf{X}$ and $\mathbf{Y}$, in which, $N = M$. The rigid point cloud registration problem can be formulated as minimizing the following objective function:

$$\boldsymbol{e}(\mathbf{C}, \mathbf{R}, \mathbf{t}) = \sum_{i}^{N} \sum_{j}^{M} \mathbf{C}_{i,j} \left\| \mathbf{R}x_i + \mathbf{t} - y_j \right\|_2^2, \quad (1)$$

subject to $\sum_j^M \mathbf{C}_{i,j} = 1, \forall i$, $\sum_i^N \mathbf{C}_{i,j} = 1, \forall j$, $\mathbf{C}_{i,j} \in \{0, 1\}^{N \times M}, \forall i, j$. In the more difficult case where there are no one-to-one correspondences, the equality constraints no longer hold, and they become inequality constraints. We can introduce slack variables in $\mathbf{C}$ as in [43] to convert inequality constraints back into equality constraints. The row constraints are converted as follows, and the column constraints are similarly converted:

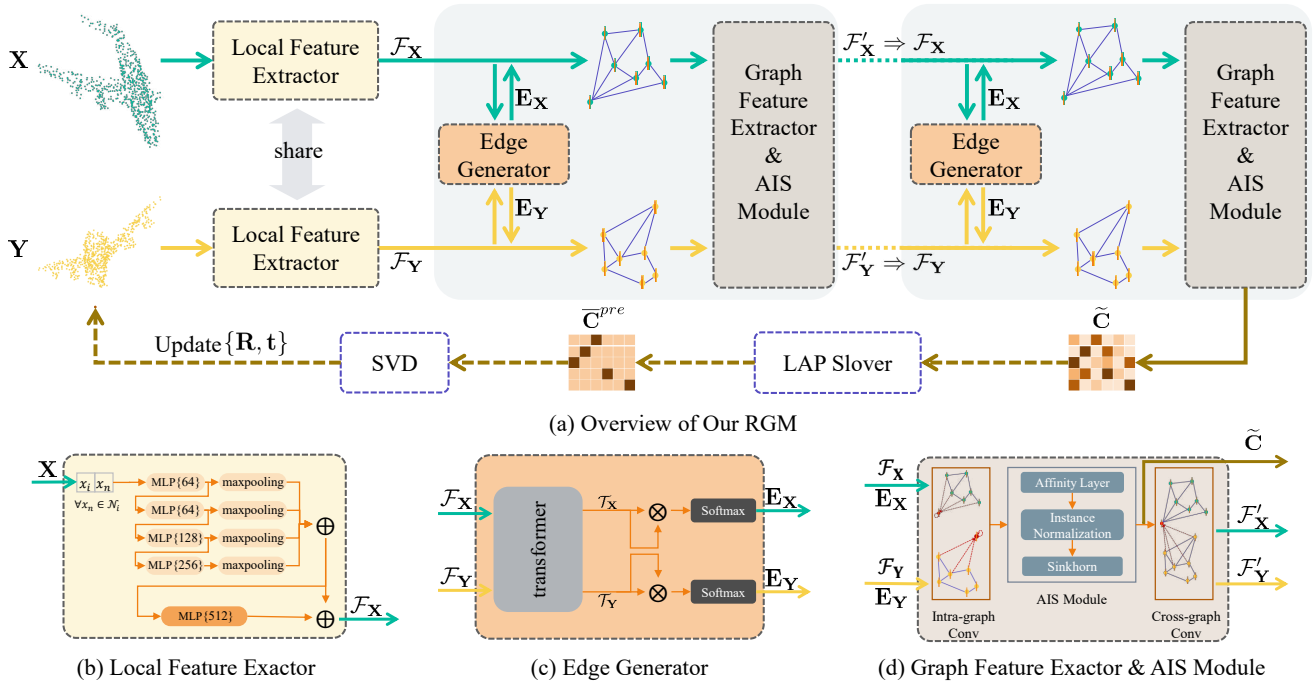
$$\sum_{j}^{M} \mathbf{C}_{i,j} \leq 1, \forall i \rightarrow \sum_{j}^{M+1} \mathbf{C}_{i,j} = 1, \forall i \leq N. \quad (2)$$

Please note that $\mathbf{C}$ becomes a $(N + 1) \times (M + 1)$ matrix after introducing one row and one column slack variables, and the sums of the added row and column are not restricted to be one.

In this paper, we use an end-to-end neural network to predict $\mathbf{C}$. Once we know the correspondences, the rigid transformation can be obtained by SVD.

4. RGM

Figure 2 (a) shows the overall pipeline of RGM. RGM consists of four components: local feature extractor, edge



(a) Overview of Our RGM

(b) Local Feature Exactor

(c) Edge Generator

(d) Graph Feature Exactor & AIS Module

Figure 2. The pipeline of the proposed 3D rigid point cloud registration framework, RGM, where $\bigoplus$ represents concatenate features and $\bigotimes$ represents matrix multiplication. The solid lines are the data flow of both training and testing, and the dotted lines are the data flow that exists only in testing.

generator, graph feature extractor & AIS module and LAP-SVD. During training, we use the shared local feature extractor to extract local features for each point in $\mathbf{X}$ and $\mathbf{Y}$, and take these local features as the node features $\mathcal{F}$ of the initial graph. Next, the edge generator generates edges and builds the source graph and target graph, and the graphs are inputted into the graph feature extractor, which processes the two graphs and outputs new node features $\mathcal{F}'$ and uses them to update $\mathcal{F}$. The AIS module predicts the soft correspondence matrix $\widetilde{\mathbf{C}}$ between nodes of the two graphs. By using blocks composed of three modules, the edge generator, graph feature extractor and AIS module, with the same structure but different weights L times, we can obtain node features $\mathcal{F}$ with better discrimination capability and a more accurate soft correspondence matrix $\widetilde{\mathbf{C}}$. Finally, the training loss is the cross-entropy between $\widetilde{\mathbf{C}}$ and the ground truth correspondences. During test, two point clouds are first inputted into the network to obtain the soft correspondence matrix $\widetilde{\mathbf{C}}$. Then, the soft correspondences are converted to hard correspondences using the LAP solver, and the transformation is solved by SVD. We also update the transformation iteratively, similar to ICP. The details of each component are explained in the following subsections.

4.1. Local Feature Extractor

To establish the correspondence matrix between two point clouds, it is necessary to embed the source point cloud $\mathbf{X}$ and the target point cloud $\mathbf{Y}$ into a common feature space. We only use the coordinates of the points to build a low-dimensional local feature descriptor $\mathcal{P}$ for each point. The local feature descriptor $\mathcal{P}_{x_i}$ of x_i is:

$$\mathcal{P}_{x_i} = \{(x_i, x_n) \mid \forall x_n \in \mathcal{K}_i\}, \tag{3}$$

where, $\mathcal{K}_i$ represents the K-nearest neighboring points of x_i.

Low-dimensional local feature descriptors are mapped to high-dimensional local feature spaces through nonlinear functions f_θ:$\mathbf{R}^{K\times 6} \rightarrow \mathbf{R}^V$, where V is the dimensionality of the final high-dimensional local feature. The implementation of f_θ is shown in Figure 2 (b), where θ represents the parameter of the nonlinear function, which consists of shared multilayer perceptron (MLP), maxpooling and concatenation. We use the high-dimensional local features as the node features $\mathcal{F}$ of the initial graph. The node feature $\mathcal{F}_{x_i}$ of x_i can be expressed as follows:

$$\mathcal{F}_{x_i} = f_\theta\left(\mathcal{P}_{x_i}\right). \tag{4}$$

Inspired by the idea of the Siamese network [4], the two point clouds share the same local feature exactor. When the two point clouds become closer, the local features also become similar, so this structure is suitable for iterative registration.

If only the local features are used to predict the correspondences between point clouds, it is easy to obtain incorrect correspondences, especially when there are outliers.

The reason is that the local features do not contain the structural information of the point cloud on a larger scale (self-correlation) and the association between the two point clouds (cross-correlation). Inspired by Wang's research on deep graph matching [37], we construct graphs from point clouds and use deep graph matching to establish better correspondences. Section 4.2 describes how to build graphs from point clouds, and section 4.3 introduces how to predict the correspondences by using deep graph matching and the AIS module.

4.2. Edge Generator Based on Transformer

The graphs built from $\mathbf{X}$ and $\mathbf{Y}$ are denoted as source graph $\mathcal{G}_s = \{\mathbf{X}, \mathbf{E}_\mathbf{X}\}$ and target graph $\mathcal{G}_t = \{\mathbf{Y}, \mathbf{E}_\mathbf{Y}\}$, respectively. The graph nodes are the original points, and the graph edges are represented by the adjacency matrix $\mathbf{E}$. The node features of $\mathcal{G}_s$ and $\mathcal{G}_t$ are denoted by $\mathcal{F}_{x_i}$ and $\mathcal{F}_{y_j}$, respectively. There are trivial methods to generate the edges, such as full connection, nearest neighbor connection and Delaunay triangulation but the features of graphs cannot be effectively aggregated, as shown in Figure 4 (d). Inspired by the success of BERT [10] in NLP, we introduce a transformer [35] module to dynamically learn the soft edges of any two nodes within a point cloud. The transformer-based edge generator is illustrated in Figure 2 (c). The transformer consists of several stacked encoder-decoder layers. The encoder uses a self-attention layer and shared MLP to encode node features, and the decoder associates and encodes features based on the co-attention mechanism. The transformer takes node features $\mathcal{F}_\mathbf{X}, \mathcal{F}_\mathbf{Y}$ as input and encodes them into embedding features $\mathcal{T}_\mathbf{X}, \mathcal{T}_\mathbf{Y}$. Soft edge adjacency matrices are obtained by applying a softmax function on the inner product of the embedding features as follows:

$$\mathcal{T}_\mathbf{X}, \mathcal{T}_\mathbf{Y} = f_{\text{transformer}}(\mathcal{F}_\mathbf{X}, \mathcal{F}_\mathbf{Y}), \tag{5}$$

$$\mathbf{E}_\mathbf{X} = \text{softmax}(\langle (\mathcal{T}_\mathbf{X})^T, \mathcal{T}_\mathbf{X} \rangle), \tag{6}$$

$$\mathbf{E}_\mathbf{Y} = \text{softmax}(\langle (\mathcal{T}_\mathbf{Y})^T, \mathcal{T}_\mathbf{Y} \rangle). \tag{7}$$

4.3. Graph Feature Extractor and AIS Module

This part is shown in Figure 2 (d), which consists of three consecutive steps as follows: First, we use intra-graph conv to explore the self-correlation of node features, where features are aggregated from nodes along edges within each graph. The message passing scheme between nodes is the same as PCA-GM [37]. A node self-correlation feature $\mathcal{F}_{x_i}^{corr}$ of $\mathcal{G}_s$ is computed by intra-graph convolution as follows:

$$\mathcal{F}_{x_i}^{corr} = \sum_{j=1}^{N} \breve{\mathbf{E}}_{i,j} * f_{adj}(\mathcal{F}_{x_j}) + f_{self}(\mathcal{F}_{x_i}), \tag{8}$$

and likewise for $\mathcal{G}_t$. Here, $\breve{\mathbf{E}}$ is the row normalized adjacency matrix calculated from $\mathbf{E}$, and f_{adj} and f_{self} are message passing functions, which are implemented by fully connected layers and ReLU.

Second, the AIS module is used to calculate a soft correspondence matrix. The AIS module consists of an affinity layer, instance normalization and Sinkhorn. An affinity matrix $\mathbf{A}$ between the two graphs is computed as follows:

$$\mathbf{A}_{i,j} = (\mathcal{F}_{x_i}^{corr})^T \mathbf{W} (\mathcal{F}_{y_j}^{corr}), \tag{9}$$

where $\mathbf{W}$ is the learnable parameter in the affinity layer. If $\mathcal{F}_{x_i}^{corr}$,$\mathcal{F}_{y_j}^{corr} \in \mathbf{R}^Q$, then $\mathbf{W} \in \mathbf{R}^{Q \times Q}$.

Before using Sinkhorn to compute the soft correspondence matrix $\widetilde{\mathbf{C}}$, we need to transform $\mathbf{A}$ into a matrix with positive elements within the finite values. There are two approaches to do so, and the naïve approach is to use softmax for rows or columns. The problem with this approach is that it processes each row or column and does not consider the matrix as a whole, which may result in the problem that a smaller value in $\mathbf{A}$ is transformed into a larger value in the transformed matrix[1]. To avoid this situation, we do not use softmax but use instance normalization [34] to transform $\mathbf{A}$. Instance normalization considers all the elements globally and uses an exponential function to ensure that all elements are positive. For handling outliers, we add an additional row and an additional column of ones to the transformed matrix and then input it into Sinkhorn [32] to calculate the soft correspondence matrix $\widetilde{\mathbf{C}}$ by the iterative process of alternating row and column normalizations.

Finally, we enhance the node features by exploring cross-correlation through cross-graph conv. Cross-graph conv is similar to intra-graph conv, except that features are aggregated from the node features of the other graph with edges replaced by $\widetilde{\mathbf{C}}$. The more similar the node pairs between the two graphs are, the higher the corresponding weight of $\widetilde{\mathbf{C}}$ will be. We obtain a new node feature $\mathcal{F}'_{x_i}$ of node x_i with a self-correlation feature and cross-correlation feature as follows:

$$\mathcal{F}'_{x_i} = f_{\text{cross}}(\mathcal{F}_{x_i}^{\text{corr}}, \sum_{j=1}^{\text{M}} \widetilde{\mathbf{C}}_{i,j} * \mathcal{F}_{y_j}^{\text{corr}}), \tag{10}$$

and likewise for $\mathcal{G}_t$. Here, f_{cross} consists of a feature concatenate and a fully connected layer, and it is shared for $\mathcal{G}_s$ and $\mathcal{G}_t$.

4.4. LAP Slover and SVD

To compute the hard correspondence matrix $\overline{\mathbf{C}}^{pre}$, which is binary, we sum the elements of each row and each column of $\widetilde{\mathbf{C}}$ and take out the rows and columns with a sum greater than 0.5, and apply a LAP solver based on Hungarian algorithm[16] on the resulting matrix to obtain a binary matrix. Then, the elements of the binary matrix are assigned

[1] visualization is detailed in Supplementary Material

to a zero matrix with the shape of $\widetilde{\mathbf{C}}$ according to their position in $\widetilde{\mathbf{C}}$, and the result is the we need hard correspondence matrix $\overline{\mathbf{C}}^{pre}$. Finally, we take $\overline{\mathbf{C}}^{pre}$ as input to predict the transformation $\{\hat{\mathbf{R}}, \hat{\mathbf{t}}\}$ by SVD.

4.5. Loss

Our loss function takes the ground truth correspondences directly as supervision, which is different from previous studies [38, 43, 45] that define loss on transformation parameters. Cross-entropy loss between soft correspondence matrix $\widetilde{\mathbf{C}}$ and ground-truth correspondence matrix $\overline{\mathbf{C}}^{gt}$ is adopted to train our model. The formula is as follows:

$$\text{loss} = -\sum_{i}^{N}\sum_{j}^{M}(\overline{\mathbf{C}}_{i,j}^{gt}\log\widetilde{\mathbf{C}}_{i,j} + (1-\overline{\mathbf{C}}_{i,j}^{gt})\log(1-\widetilde{\mathbf{C}}_{i,j})). \tag{11}$$

Since our loss function is only related to the soft correspondence matrix $\widetilde{\mathbf{C}}$, the calculations in section 4.4 do not need to be differentiable.

4.6. Implementation Details

Our local feature extractor considers a neighborhood of K = 20, and outputs final high-dimensional local features with the dimension V=1024. We set $L = 2$ in this study. We train the network using the SGD optimizer with an initial learning rate of 1e-3. This network is implemented using PyTorch. For more details of implementation please see the supplementary material.

5. Experiments

5.1. Datasets and Evaluation Metrics

All experiments are conducted on the ModelNet40 [41] dataset, which includes 12,311 meshed CAD models from 40 categories. We randomly sample 2,048 points from the mesh faces and rescale points into a unit sphere. Each category consists of official train/test splits. To select models for evaluation, we take 80% and 20% of the official train split as the training set and validation set, respectively, and the official test split for testing. For each object in the dataset, we randomly sample 1,024 points as the source point cloud $\mathbf{X}$, and then apply a random transformation on $\mathbf{X}$ to obtain the target point cloud $\mathbf{Y}$ and shuffle the point order. For the transformation applied, we randomly sample three Euler angles in the range of $[0, 45]^\circ$ for rotation and three displacements in the range of $[-0.5, 0.5]$ along each axis for translation. Unless otherwise noted, these settings are used by default in all experiments.

We use six evaluation metrics, and the first four are calculated from the estimated transformation parameters. They are the mean isotropic errors (MIE) of $\mathbf{R}$ and $\mathbf{t}$ proposed in RPM-Net [43], and the mean absolute errors (MAE) of $\mathbf{R}$ and $\mathbf{t}$ used in DCP [38], which are anisotropic. All rotation-related metrics are in units of degrees.

In addition, we propose a new metric, clip chamfer distance (CCD), which measures how close the two point clouds are brought to each other, and it is calculated as follows:

$$\begin{aligned}\text{CCD}(\widehat{\mathbf{X}}, \mathbf{Y}) = &\sum_{\hat{x}_i\in\mathbf{X}}\min(\min_{y_j\in\mathbf{Y}}(\|\hat{x}_i - y_j\|_2^2), d) \\ &+ \sum_{y_j\in\mathbf{Y}}\min(\min_{\hat{x}_i\in\mathbf{X}}(\|\hat{x}_i - y_j\|_2^2), d),\end{aligned} \tag{12}$$

where $\hat{\mathbf{X}}$ is the transformed source point cloud after registration and $\hat{x}_i$ is the ith point. To avoid the influence of outliers in partial-to-partial registration, the point pair whose distance is larger than 0.1 is not included in the calculation. This is implemented by seting the threshold $d = 0.1$.

Finally, we also reported the recall with MAE($\mathbf{R}$)$< 1^\circ$ and MAE($\mathbf{t}$)< 0.1. The best results are marked in bold font in tables.

5.2. Comparing Methods

We compare our method to ICP [3], fast global registration (FGR) [49], as well as three latest learning-based methods, RPM-Net [43], IDAM [18] and DeepGMR [45]. Other early learning-based methods, such as DCP and PointNetLK, are not directly compared, because experiments in [43, 18, 45] have already shown that these new methods have better performance. Our method performs two iterations during the test. We adopt the ICP and FGR implemented by Intel Open3D [50]. For IDAM and DeepGMR, we use the code provided by the authors and train the models according to the author's settings. For RPM-Net, we need to estimate the normal except in the clean experiment and use the code provided by the author. The number of iterations of RPM-Net was set to 5 according to the author's article. ICP uses the identity matrix as initialization, and none of the other methods need transformation initialization. All networks are retrained because no trained model is available.

5.3. Clean Point Cloud

We first evaluate the registration performance on clean point clouds and follow the sampling and transformation settings in section 5.1. The ground-truth correspondences are obtained by the strict correspondences between $\mathbf{X}$ and $\mathbf{Y}$. All models are trained and evaluated on clean data, and Table 1 shows the performance of our method and its peers. Our method achieves the best performance and greatly outperforms the strongest learning-based method. In addition, the success rate of RGM reaches 100%, and most of its error metrics are close to 0, which cannot be achieved by other existing methods. Although DeepGMR also achieves

method	MIE(R)	MIE(t)	MAE(R)	MAE(t)	CCD	Recall
ICP	3.079	0.02442	6.4467	0.05446	0.03009	74.19%
FGR	0.006	0.00005	0.0099	0.00010	0.00019	99.96%
RPM-Net	0.109	0.00050	0.2464	0.00112	0.00089	98.14%
IDAM	0.731	0.01244	1.3536	0.02605	0.04470	75.81%
DeepGMR	0.001	0.00001	0.0156	0.00002	0.00003	**100.00%**
RGM	**<0.001**	**<0.00001**	**0.0096**	**<0.00001**	**<0.00001**	**100.00%**

Table 1. Performance on clean point clouds

method	MIE(R)	MIE(t)	MAE(R)	MAE(t)	CCD	Recall
ICP	3.127	0.02256	6.5030	0.04944	0.05387	77.59%
FGR	5.405	0.03386	10.0079	0.07080	0.06918	30.75%
RPM-Net	0.305	0.00253	0.5773	0.00532	0.04257	96.68%
IDAM	1.818	0.01416	3.4916	0.02915	0.05436	49.59%
DeepGMR	1.178	0.00716	2.2736	0.01498	0.05029	56.52%
RGM	**0.080**	**0.00069**	**0.1496**	**0.00141**	**0.04185**	**99.51%**

Table 2. Performance on point clouds with Gaussian noise

method	MIE(R)	MIE(t)	MAE(R)	MAE(t)	CCD	Recall
ICP	12.456	0.12465	24.8777	0.26685	0.11511	6.56%
FGR	23.185	0.14560	42.4292	0.30214	0.12118	5.23%
RPM-Net	0.864	0.00834	1.6985	0.01763	0.08457	80.59%
IDAM	8.905	0.09192	16.9724	0.19209	0.12393	0.81%
DeepGMR	43.683	0.22479	70.9143	0.45705	0.14401	0.08%
RGM	**0.492**	**0.00414**	**0.9298**	**0.00874**	**0.08238**	**93.31%**

Table 3. Performance on partial-to-partial point clouds

method	MIE(R)	MIE(t)	MAE(R)	MAE(t)	CCD	Recall
ICP	13.326	0.13033	26.6447	0.27774	0.11879	6.71%
FGR	23.950	0.14067	41.9631	0.29106	0.12370	5.13%
RPM-Net	1.041	0.01067	1.9826	0.02276	0.08704	75.59%
IDAM	10.158	0.10063	19.3249	0.20729	0.12921	0.95%
DeepGMR	44.363	0.22039	71.0677	0.44632	0.14728	0.24%
RGM	**0.837**	**0.00674**	**1.5457**	**0.01418**	**0.08469**	**84.28%**

Table 4. Performance on unseen categories point clouds

a 100% success rate, its errors are larger than RGM. Some qualitative comparisons are shown in Figure 3 (a).

5.4. Gaussian Noise

To evaluate the robustness to noise, Gaussian noise sampled from $\mathcal{N}(0,\ 0.01)$ and clipped to $[-0.05,\ 0.05]$ is independently added to each coordinate of the points in clean point clouds. These noises might destroy the original correspondences, so we need to rebuild them for training models that need ground truth correspondences. First, we compute the point pair distance between $\mathbf{Y}$ and $\mathbf{X}'$, which is obtained by applying the ground truth transformation to $\mathbf{X}$. Then, if $x'_i \in \mathbf{X}'$ and $y_j \in \mathbf{Y}$ satisfy Eq. 13, they are regarded as a corresponding point pair and no longer appear in the next round calculation. Finally, we find corresponding point pairs again according to Eq. 13 from the remaining points. To avoid long-distance point pairs being selected as a correspondence, we only consider the point pairs whose distance is less than 0.1. The reason why we find the corresponding point pair again from the remaining points is that the distance between the two points may not be the smallest but the second smallest, so they are not found in the first round.

$$\min_{x'_n \in \mathbf{X}'}(\|x'_n - y_j\|_2^2) = \|x'_i - y_j\|_2^2 = \min_{y_m \in \mathbf{Y}}(\|x'_i - y_m\|_2^2). \tag{13}$$

All models are trained and evaluated on the noise data. The results are shown in Table 2. It is obvious that our method is much more accurate than the latest learning-based methods and the traditional methods, and the recall of our method is close to 100%. Some qualitative comparisons are shown in Figure 3 (b).

5.5. Partial-to-Partial

Partial-to-partial is the most challenging case for point cloud registration, and it is important because it occurs frequently in real-world applications. To generate partial-to-partial point cloud pairs, we follow the protocol in RPM-Net [43], which is closer to real-world applications. For each point cloud, we create a random plane passing through the origin independently, translate it along its normal, and retain 70% of the points. All models are trained and evaluated on partial-to-partial data and the results are illustrated in Table 3. Our method is obviously more accurate than the other methods, and its success rate is higher than 90%. RPM-Net is the second best method, but its error is still twice as large as ours. Some qualitative comparisons are shown in Figure 3 (c). For the inference time of our method and the comparison methods, please refer to the supplementary material.

5.6. Unseen Categories

To test each method's generalization capability on unseen shape categories, we take the official train and test splits for the first 20 categories as the training and validation sets, respectively, and test on the official test splits of the last 20 categories. Other experimental settings are the same as those in the partial-to-partial experiment. The experimental results are summarized in Table 4. We find that the performance of traditional methods does not change significantly. The generalization capability of RPM-Net is also good, but it is obvious that our method works better. The other learning-based methods do not generalize well to unseen categories. Some qualitative comparisons are shown in Figure 3 (d).

5.7. Ablation Studies

In this section, we present the results of the ablation study to analyze the effectiveness of two key components. All ablation studies are performed on the partial-to-partial dataset. We analyze the two key components as follows:

To demonstrate the effectiveness of the AIS module, we design a variant to replace the AIS module, and the resulting method is denoted as RGMVar1. The variant computes the distance matrix $\mathbf{D}$ between the nodes of the two graphs

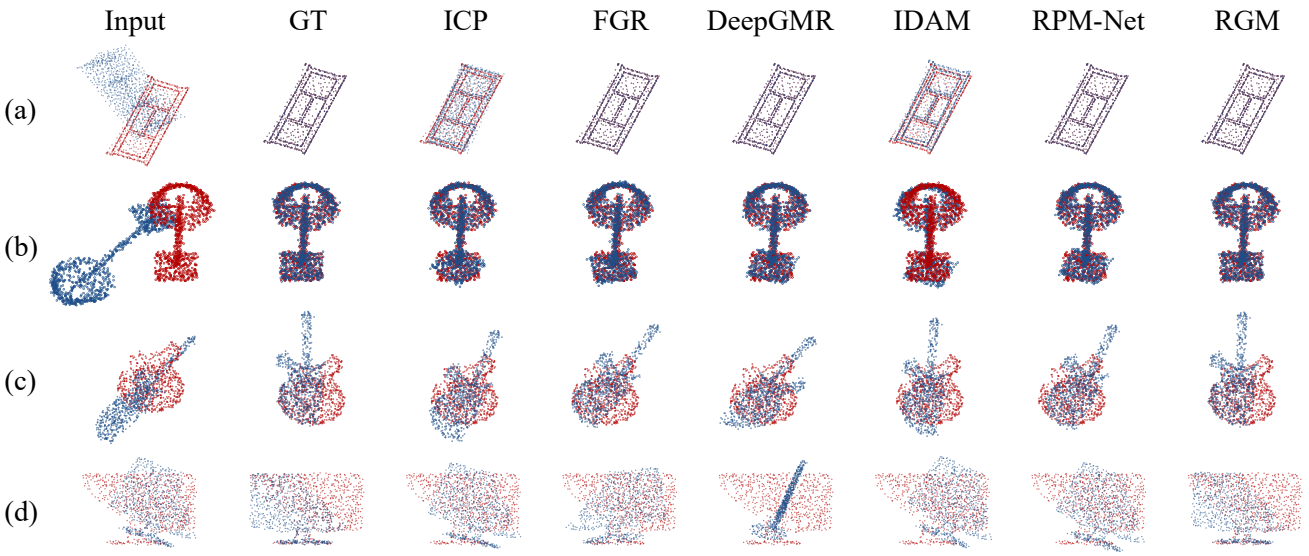


Figure 3. Qualitative registration results on ModelNet40, (a) clean, (b) noise, (c) partial-to-partial, and (d) unseen categories.

variants	MIE(R)	MIE(t)	MAE(R)	MAE(t)	CCD	Recall
RGMVar1	10.746	0.07014	19.2722	0.14255	0.11304	18.33%
RGMVar2	1.554	0.01454	2.9051	0.03101	0.08632	74.17%
RGMVar3	1.197	0.01083	2.2612	0.02236	0.08605	75.59%
RGM	**0.837**	**0.00674**	**1.5457**	**0.01418**	**0.08469**	**84.28%**

Table 5. Ablation studies

by computing the L2 norm of node features, transforms $\mathbf{D}$ into a positive matrix within the finite values by the formula $e^{-(\mathbf{D}_{i,j}\ -\ 0.5)}$, and uses Sinkhorn to calculate the soft correspondences. The results are listed in the first row of Table 5. We find that the registration accuracy becomes very poor by using the AIS variant, and this result shows that the proposed AIS module can effectively improve the registration performance. This is because the AIS module generates more correct matching than its variant, and an illustrative example of the hard correspondences generated by AIS and its variant is shown in Figure 4 (b) and (c).

To understand the importance of our edge generator, we design two variants those use full connection edges and sparse connection edges instead of building edges by a transformer, and the resulting methods are denoted as RGMVar2 and RGMVar3 respectively. The results are shown in the second and third rows of Table 5, and they are also inferior to the performance by using a transformer to generate edges. An example of the hard correspondences generated by this method is shown in Figure 4 (d) and (e).

5.8. Other Experiments

For experiments on ShapeNet[7] and 3DMatch[47], computational efficiency, visualizing the learned graph and so on, please see Supplementary Material.

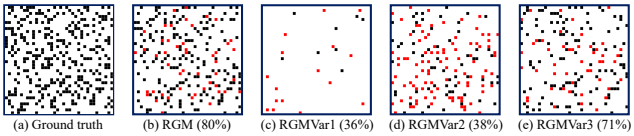


Figure 4. An illustrative case of the ground-truth correspondences and the hard correspondences generated by RGM and its variants. Black and red blocks represent the correct and incorrect correspondences, respectively. The number in brackets is the proportion of correct correspondences. Please note that there are 717 points in the two partial point clouds to be registered, and this is a subsampled figure with 36×36 blocks. Much more correct correspondences are generated by RGM.

6. Conclusion

We introduce deep graph matching to solve the point cloud registration problem for the first time and propose a novel deep learning framework RGM that achieves state-of-the-art performance. We propose the AIS module to establish accurate correspondences between the graph nodes to greatly improve registration performance. In addition, the transformer-based edge generator provides a new idea for building graph edges in addition to full connection, nearest neighbor connection and Delaunay triangulation. We think that the deep graph matching approach has the potential to be used in other registration problems, including 2D-3D registration and deformable registration.

Acknowledgment

This work was supported by the National Natural Science Foundation of China under Grant 62076070.

References

[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. Pointnetlk: Robust & efficient point cloud registration using pointnet. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7163–7172, 2019. 3

[2] Simon Baker and Iain Matthews. Lucas-kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. 3

[3] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 1, 2, 6

[4] Jane Bromley, Isabelle Guyon, Yann LeCun, Eduard Säckinger, and Roopak Shah. Signature verification using a "siamese" time delay neural network. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 737–744, 1994. 4

[5] Dylan Campbell and Lars Petersson. Gogma: Globally-optimal gaussian mixture alignment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5685–5694, 2016. 1, 2

[6] Dylan Campbell, Lars Petersson, Laurent Kneip, Hongdong Li, and Stephen Gould. The alignment of the spheres: Globally-optimal spherical mixture alignment for camera pose estimation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11796–11806, 2019. 1, 2

[7] Angel X. Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, and Hao and Su. Shapenet: An information-rich 3d model repository. *Computer ence*, 2015. 8

[8] Minsu Cho, Karteek Alahari, and Jean Ponce. Learning graphs to match. In *IEEE International Conference on Computer Vision (ICCV)*, pages 25–32, 2013. 3

[9] Jean-Emmanuel Deschaud. Imls-slam: scan-to-model matching based on 3d data. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 2480–2485, 2018. 1

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 3, 5

[11] Olivier Duchenne, Armand Joulin, and Jean Ponce. A graph-matching kernel for object categorization. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1792–1799, 2011. 3

[12] Benjamin Eckart, Kihwan Kim, and Jan Kautz. Hgmr: Hierarchical gaussian mixtures for adaptive 3d registration. In *European Conference on Computer Vision (ECCV)*, pages 705–721, 2018. 2

[13] Lei Han, Lan Xu, Dmytro Bobkov, Eckehard Steinbach, and Lu Fang. Real-time global registration for globally consistent rgb-d slam. *IEEE Transactions on Robotics*, 35(2):498–508, 2019. 1

[14] Bing Jian and Baba C Vemuri. Robust point set registration using gaussian mixture models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1633–1645, 2010. 2

[15] Roy Jonker and Anton Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4):325–340, 1987. 2

[16] Harold W Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. 2, 5

[17] Hongdong Li and Richard Hartley. The 3d-3d registration problem revisited. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1–8, 2007. 1

[18] Jiahao Li, Changhao Zhang, Ziyao Xu, Hangning Zhou, and Chi Zhang. Iterative distance-aware similarity matrix convolution with mutual-supervised point elimination for efficient point cloud registration. *European Conference on Computer Vision (ECCV)*, 2019. 2, 3, 6

[19] Ying Li, Lingfei Ma, Zilong Zhong, Fei Liu, Michael A Chapman, Dongpu Cao, and Jonathan Li. Deep learning for lidar point clouds in autonomous driving: a review. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–21, 2020. 1

[20] Yinlong Liu, Chen Wang, Zhijian Song, and Manning Wang. Efficient global point cloud registration by matching rotation invariant features through translation search. In *European Conference on Computer Vision (ECCV)*, pages 448–463, 2018. 2

[21] Weixin Lu, Yao Zhou, Guowei Wan, Shenhua Hou, and Shiyu Song. L3-net: Towards learning based lidar localization for autonomous driving. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6389–6398, 2019. 1

[22] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of Imaging Understanding Workshop*, pages 121–130, 1981. 3

[23] Alex Nowak, Soledad Villar, Afonso S Bandeira, and Joan Bruna. Revised note on learning quadratic assignment with graph neural networks. In *IEEE Data Science Workshop (DSW)*, pages 1–5, 2018. 3

[24] François Pomerleau, Francis Colas, and Roland Siegwart. A review of point cloud registration algorithms for mobile robotics. *Now Foundations and Trends in Robotics*, 4(1):1–104, 2015. 2

[25] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017. 3

[26] Szymon Rusinkiewicz. A symmetric objective function for icp. *ACM Transactions on Graphics (TOG)*, 38(4):1–7, 2019. 2

[27] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001. 2

[28] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast point feature histograms (fpfh) for 3d registration. In

IEEE International Conference on Robotics and Automation (ICRA), pages 3212–3217, 2009. 2, 3

[29] Samuele Salti, Federico Tombari, and Luigi Di Stefano. Shot: Unique signatures of histograms for surface and texture description. *Computer Vision and Image Understanding*, 125:251–264, 2014. 2, 3

[30] Vinit Sarode, Xueqian Li, Hunter Goforth, Yasuhiro Aoki, Rangaprasad Arun Srivatsan, Simon Lucey, and Howie Choset. Pcrnet: Point cloud registration network using pointnet encoding. In *IEEE International Conference on Computer Vision (ICCV)*, 2019. 2, 3

[31] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: Science and Systems*, volume 2, page 435, 2009. 1, 2

[32] Richard Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The Annals of Mathematical Statistics*, 35(2):876–879, 1964. 3, 5

[33] Nikolai Ufer and Bjorn Ommer. Deep semantic feature matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6914–6923, 2017. 3

[34] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv preprint arXiv:1607.08022*, 2016. 5

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5998–6008, 2017. 3, 5

[36] Guowei Wan, Xiaolong Yang, Renlan Cai, Hao Li, Yao Zhou, Hao Wang, and Shiyu Song. Robust and precise vehicle localization based on multi-sensor fusion in diverse city scenes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4670–4677. IEEE, 2018. 1

[37] Runzhong Wang, Junchi Yan, and Xiaokang Yang. Learning combinatorial embedding networks for deep graph matching. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3056–3065, 2019. 2, 3, 5

[38] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *IEEE International Conference on Computer Vision (ICCV)*, pages 3523–3532, 2019. 2, 3, 6

[39] Yue Wang and Justin M Solomon. Prnet: Self-supervised learning for partial-to-partial registration. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 8814–8826, 2019. 3

[40] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics*, 38(5):1–12, 2019. 3

[41] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 6

[42] Jiaolong Yang, Hongdong Li, Dylan Campbell, and Yunde Jia. Go-icp: A globally optimal solution to 3d icp point-set registration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(11):2241–2254, 2015. 1, 2

[43] Zi Jian Yew and Gim Hee Lee. Rpm-net: Robust point matching using learned features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11824–11833, 2020. 2, 3, 6, 7

[44] Hakje Yoo, Ahnryul Choi, and Joung Hwan Mun. Acquisition of point cloud in ct image space to improve accuracy of surface registration: Application to neurosurgical navigation system. *Journal of Mechanical Science and Technology*, 34(6):2667–2677, 2020. 1

[45] Wentao Yuan, Ben Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. Deepgmr: Learning latent gaussian mixture models for registration. In *European Conference on Computer Vision (ECCV)*, 2020. 3, 6

[46] Andrei Zanfir and Cristian Sminchisescu. Deep learning of graph matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2684–2693, 2018. 3

[47] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 8

[48] Feng Zhou and Fernando De la Torre. Factorized graph matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 127–134, 2012. 3

[49] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *European Conference on Computer Vision (ECCV)*, pages 766–782, 2016. 3, 6

[50] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3d: A modern library for 3d data processing. *arXiv preprint arXiv:1801.09847*, 2018. 6