OMNet: Learning Overlapping Mask for Partial-to-Partial Point Cloud Registration

Hao Xu[1,2] Shuaicheng Liu[1,2*] Guangfu Wang[2] Guanghui Liu[1*] Bing Zeng[1]

[1]University of Electronic Science and Technology of China
[2]Megvii Technology

Abstract

Point cloud registration is a key task in many computational fields. Previous correspondence matching based methods require the inputs to have distinctive geometric structures to fit a 3D rigid transformation according to point-wise sparse feature matches. However, the accuracy of transformation heavily relies on the quality of extracted features, which are prone to errors with respect to partiality and noise. In addition, they can not utilize the geometric knowledge of all the overlapping regions. On the other hand, previous global feature based approaches can utilize the entire point cloud for the registration, however they ignore the negative effect of non-overlapping points when aggregating global features. In this paper, we present OMNet, a global feature based iterative network for partial-to-partial point cloud registration. We learn overlapping masks to reject non-overlapping regions, which converts the partial-to-partial registration to the registration of the same shape. Moreover, the previously used data is sampled only once from the CAD models for each object, resulting in the same point clouds for the source and reference. We propose a more practical manner of data generation where a CAD model is sampled twice for the source and reference, avoiding the previously prevalent over-fitting issue. Experimental results show that our method achieves state-of-the-art performance compared to traditional and deep learning based methods. Code is available at https://github.com/megvii-research/OMNet.

1. Introduction

Point cloud registration is a fundamental task that has been widely used in various computational fields, e.g., augmented reality [2, 6, 4], 3D reconstruction [14, 19] and autonomous driving [38, 10]. It aims to predict a 3D rigid transformation aligning two point clouds, which may be potentially obscured by partiality and contaminated by noise.

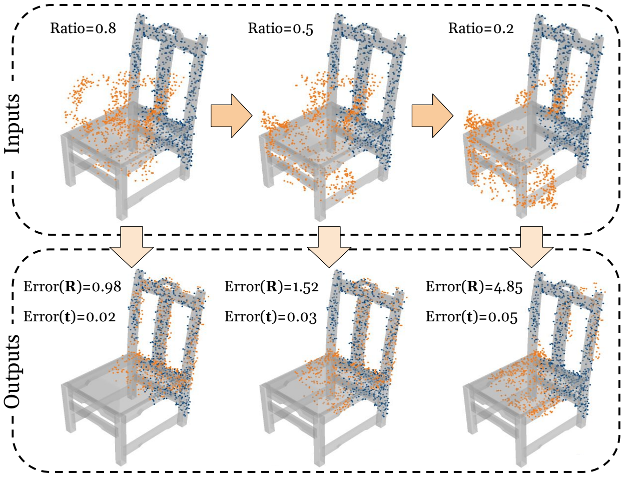

Figure 1. Our OMNet shows robustness to the various overlapping ratios of inputs. All inputs are transformed by the same 3D rigid transformation. $\mathrm{Error}(\mathbf{R})$ and $\mathrm{Error}(\mathbf{t})$ are isotropic errors.

Iterative Closest Point (ICP) [3] is a well-known algorithm for the registration problem, where 3D transformations are estimated iteratively by singular value decomposition (SVD) given the correspondences obtained by the closest point search. However, ICP easily converges to local minima because of the non-convexity problem. For this reason, many methods [23, 9, 28, 5, 20, 35] are proposed to improve the matching or search larger transformation space, and one prominent work is the Go-ICP [35], which uses a branch-and-bound algorithm to cross the local minima. Unfortunately, it is much slower than the original ICP. All these methods are sensitive to the initial positions.

Recently, several deep learning (DL) based approaches are proposed [32, 1, 33, 27, 36, 13, 17, 37] to handle the large rotation angles. Roughly, they could be divided into two categories: correspondence matching based methods and global feature based methods. Deep Closest Point (DCP) [32] determines the correspondences from learned features. DeepGMR [37] integrates Gaussian Mixture Model (GMM) to learn pose-invariant point-to-GMM

[*]Corresponding author

correspondences. However, they do not take the partiality of inputs into consideration. PRNet [33], RPMNet [36] and IDAM [17] are presented to mitigate this problem by using Gumbel–Softmax [15] with Sinkhorn normalization [31] or a convolutional neural network (CNN) to calculate matching matrix. However, these methods require the inputs to have distinctive local geometric structures to extract reliable sparse 3D feature points. As a result, they can not utilize the geometric knowledge of the entire overlapping point clouds. In contrast, global feature based methods overcome this issue by aggregating global features before estimating transformations, e.g., PointNetLK [1], PCRNet [27] and Feature-metric Registration (FMR) [13]. However, all of them ignore the negative effect of non-overlapping regions.

In this paper, we propose OMNet: an end-to-end iterative network that estimates 3D rigid transformations in a coarse-to-fine manner while preserving robustness to noise and partiality. To avoid the negative effect of non-overlapping points, we predict overlapping masks for the two inputs separately at each iteration. Given the accurate overlapping masks, the non-overlapping points are rejected during the aggregation of global features, which converts the partial-to-partial registration to the registration of the same shape. As such, regressing rigid transformation becomes easier given global features without interferes. This desensitizes the initial position of the inputs and enhances the robustness to noise and partiality. Fig. 1 shows the robustness of our method to the inputs with different overlapping ratios. Experiments show that our approach achieves state-of-the-art performance compared with previous algorithms.

Furthermore, ModelNet40 [34] dataset is adopted for the registration [32, 1, 33, 27, 36, 13, 17, 37], which has been originally applied to the task of classification and segmentation. Previous works follow the data processing of PointNet [21], which has two problems: (1) a CAD model is sampled only once during the point cloud generation, yielding the same source and the reference points, which often causes over-fitting issues; (2) ModelNet40 dataset involves some axisymmetrical categories where it is reasonable to obtain an arbitrary angle on the symmetrical axis. We propose a more suitable method to generate a pair of point clouds. Specifically, the source and reference point clouds are randomly sampled from the CAD model separately. Meanwhile, the data of axisymmetrical categories are removed. In summary, our main contributions are:

- We propose a global feature based registration network OMNet, which is robust to noise and different partial manners by learning masks to reject non-overlapping regions. Mask prediction and transformation estimation can be mutually reinforced during iteration.
- We expose the over-fitting issue and the axisymmetrical categories that existed in the ModelNet40 dataset when adopted to the registration task. In addition, we propose a more suitable method to generate data pairs for the registration task.
- We provide qualitative and quantitative comparisons with other works under clean, noisy and different partial datasets, showing state-of-the-art performance.

2. Related Works

Correspondence Matching based Methods. Most correspondence matching based methods solve the registration problem by alternating two steps: (1) set up correspondences between the source and reference point clouds; (2) compute the least-squares rigid transformation between the correspondences. ICP [3] estimates correspondences using spatial distances. Subsequent variants of ICP improve performance by detecting keypoints [11, 23] or weighting correspondences [12]. However, due to the non-convexity of the first step, they are often strapped into local minima. To address this, Go-ICP [35] uses a branch-and-bound strategy to search the transformation space at the cost of a much slower speed. Recently proposed Symmetric ICP [22] improves the original ICP by designing the objective function. Instead of using spatial distances, PFH [25] and FPFH [24] design rotation-invariant descriptors and establish correspondences from handcrafted features. To avoid computation of RANSAC [8] and nearest-neighbors, FGR [40] uses alternating optimization techniques to accelerate iterations.

More recent DL based method DCP [32] replaces the handcrafted feature descriptor with a CNN. DeepGMR [37] further estimates the points-to-components correspondences in the latent GMM. In summary, the main problem is that they require the inputs to have distinctive geometric structures, so as to promote sparse matched points. However, not all regions are distinctive, resulting in a limited number of matches or poor distributions. In addition, the transformation is calculated only from matched sparse points and their local neighbors, leaving the rest of the points untouched. In contrast, our work can use the predicted overlapping regions to aggregate global features.

Global Feature based Methods. Different from correspondence matching based methods, the previous global feature based methods compute rigid transformation from the entire point clouds (including overlapping and non-overlapping regions) of the two inputs without correspondences. PointNetLK [1] pioneers these methods, which adapts PointNet [21] with the Lucas &Kanade (LK) algorithm [18] into a recurrent neural network. PCRNet [27] improves the robustness against the noise by alternating the LK algorithm with a regression network. Furthermore, FMR [13] adds a decoder branch and optimizes the global feature distance of the inputs. However, they all ignore the negative effect of the non-overlapping points and fail to register partial-to-partial inputs. Our network can deal with partiality and shows robustness to different partial manners.

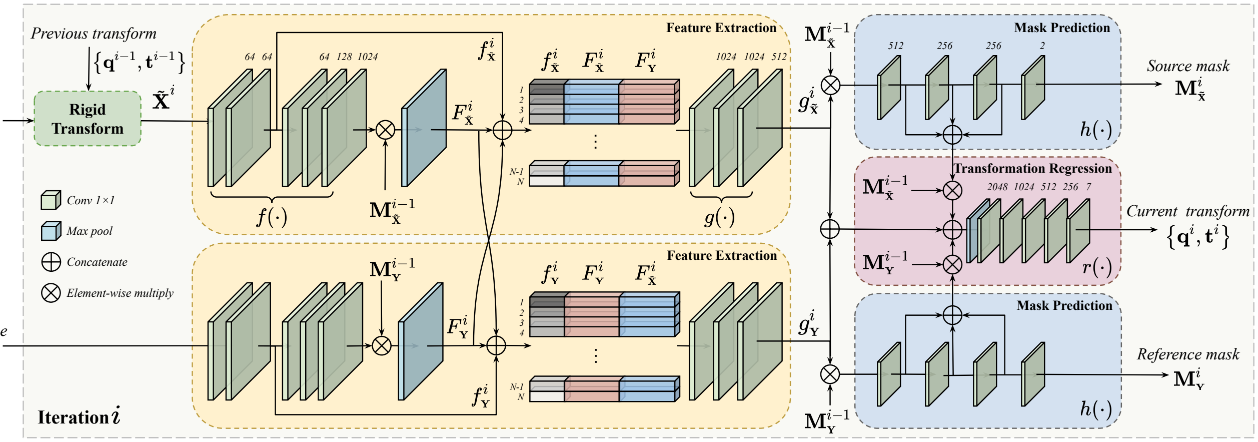
Previous transform
$\{\mathbf{q}^{i-1}, \mathbf{t}^{i-1}\}$
Source
$\mathbf{X}$
Rigid Transform
$\tilde{\mathbf{X}}^i$
Conv 1×1
Max pool
Concatenate
Element-wise multiply
Reference
$\mathbf{Y}$
Iteration i
Feature Extraction
64 64
64 128 1024
$f(\cdot)$
$\mathbf{M}^{i-1}_{\tilde{\mathbf{X}}}$
$F^i_{\tilde{\mathbf{X}}}$
$f^i_{\tilde{\mathbf{X}}}$
$f^i_{\tilde{\mathbf{X}}}$ $F^i_{\tilde{\mathbf{X}}}$ $F^i_{\mathbf{Y}}$
1 2 3 4
N-1 N
1024 1024 512
$g(\cdot)$
$g^i_{\tilde{\mathbf{X}}}$
Feature Extraction
$\mathbf{M}^{i-1}_{\mathbf{Y}}$
$F^i_{\mathbf{Y}}$
$f^i_{\mathbf{Y}}$
$f^i_{\mathbf{Y}}$ $F^i_{\mathbf{Y}}$ $F^i_{\tilde{\mathbf{X}}}$
$g^i_{\mathbf{Y}}$
Mask Prediction
512 256 256 2
Source mask
$\mathbf{M}^i_{\tilde{\mathbf{X}}}$
$h(\cdot)$
Transformation Regression
2048 1024 512 256 7
$\mathbf{M}^{i-1}_{\tilde{\mathbf{X}}}$
$\mathbf{M}^{i-1}_{\mathbf{Y}}$
Current transform
$\{\mathbf{q}^i, \mathbf{t}^i\}$
$r(\cdot)$
Mask Prediction
Reference mask
$\mathbf{M}^i_{\mathbf{Y}}$
$h(\cdot)$
$\mathbf{M}^{i-1}_{\mathbf{Y}}$

Figure 2. The overall architecture of our OMNet. During process of feature extraction, the global features $F_{\tilde{\mathbf{X}}}$ and $F_{\mathbf{Y}}$ are duplicated N times to concatenate with the point-wise features $f_{\tilde{\mathbf{X}}}$ and $f_{\mathbf{Y}}$, where N is the number of points in the inputs. The same background color denotes sharing weights. Superscripts denote the iteration count.

Partial-to-partial Registration Methods. Registration of partial-to-partial point clouds is presented as a more realistic problem by recent works [33, 36, 17]. In particular, PRNet [33] extends DCP as an iterative pipeline and tackles the partiality by detecting keypoints. Moreover, the learnable Gumble-Softmax [15] is used to control the smoothness of the matching matrix. RPMNet [36] further utilizes Sinkhorn normalization [31] to encourage the bijectivity of the matching matrix. However, they suffer from the same problem as the correspondence matching based methods, i.e., they can only use sparse points. In contrast, our method can utilize information from the entire overlapping points.

3. Method

Our pipeline is illustrated in Fig. 2. We represent the 3D transformation in the form of quaternion $\mathbf{q}$ and translation $\mathbf{t}$. At each iteration i, the source point cloud $\mathbf{X}$ is transformed by the rigid transformation $\left\{\mathbf{q}^{i-1}, \mathbf{t}^{i-1}\right\}$ estimated from the previous step into the transformed point cloud $\tilde{\mathbf{X}}^i$. Then, the global features of two point clouds are extracted by the feature extraction module (Sec. 3.1). Concurrently, the hybrid features from two point clouds are fused and fed to an overlapping mask prediction module (Sec. 3.2) to segment the overlapping region. Meanwhile, a transformation regression module (Sec. 3.3) takes the fused hybrid features as input and outputs the transformation $\left\{\mathbf{q}^i, \mathbf{t}^i\right\}$ for the next iteration. Finally, the loss functions are detailed in Sec. 3.4.

3.1. Global Feature Extraction

The feature extraction module aims to learn a function $f(\cdot)$, which can generate distinctive global features $F_{\mathbf{X}}$ and $F_{\mathbf{Y}}$ from the source point cloud $\mathbf{X}$ and the reference point cloud $\mathbf{Y}$ respectively. An important requirement is that the orientation and the spatial coordinates of the original input should be maintained, so that the rigid transformation can be estimated from the difference between the two global features. Inspired by PointNet [21], at each iteration, the global features of input $\tilde{\mathbf{X}}^i$ and $\mathbf{Y}$ are given by:

$$F^i_\beta = \max\{\mathbf{M}^{i-1}_\beta \cdot f\left(\beta\right)\}, \quad \beta \in \{\tilde{\mathbf{X}}^i, \mathbf{Y}\}, \tag{1}$$

where $f(\cdot)$ denotes a multi-layer perceptron network (MLP), which is fed with $\tilde{\mathbf{X}}^i$ and $\mathbf{Y}$ to generate point-wise features $f^i_{\tilde{\mathbf{X}}}$ and $f^i_{\mathbf{Y}}$. $\mathbf{M}^{i-1}_{\tilde{\mathbf{X}}}$ and $\mathbf{M}^{i-1}_{\mathbf{Y}}$ are the overlapping masks of $\tilde{\mathbf{X}}^i$ and $\mathbf{Y}$, which are generated by the previous step and detailed in Sec. 3.2. The point-wise features $f_{\tilde{\mathbf{X}}}$ and $f_{\mathbf{Y}}$ are aggregated by a max-pool operation $\max\{\cdot\}$, which can deal with an arbitrary number of orderless points.

3.2. Overlapping Mask Prediction

In partial-to-partial scenes, especially those including the noise, there exists non-overlapping regions between the input point clouds $\mathbf{X}$ and $\mathbf{Y}$. However, not only does it have no contributions to the registration procedure, but it also interferes to the global feature extraction, as shown in Fig. 3. RANSAC [8] is widely adopted in traditional methods to find the inliers when solving the most approximate matrix for the scene alignment. Following a similar idea, we propose a mask prediction module to segment the overlapping region automatically. Refer to PointNet [21], point segmentation only takes one point cloud as input and requires a combination of local and global knowledge. However, overlapping region prediction requires additional geometric information from both two input point clouds $\mathbf{X}$ and $\mathbf{Y}$. We can achieve this in a simple yet highly effective manner.

Specifically, at each iteration, the global features $F^i_{\tilde{\mathbf{X}}}$ and $F^i_{\mathbf{Y}}$ are fed back to point-wise features by concatenating with each of the point features $f^i_{\tilde{\mathbf{X}}}$ and $f^i_{\mathbf{Y}}$ accordingly. Then, a MLP $g(\cdot)$ is applied to fuse the above hybrid features, which are further used to segment overlapping regions and regress the rigid transformation. So we can obtain

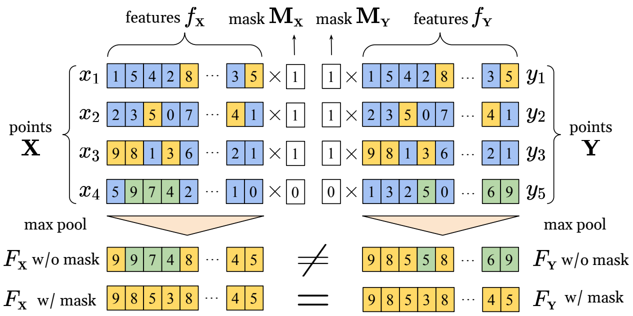
features $f_{\mathbf{X}}$
mask $\mathbf{M}_{\mathbf{X}}$
mask $\mathbf{M}_{\mathbf{Y}}$
features $f_{\mathbf{Y}}$
points $\mathbf{X}$
points $\mathbf{Y}$
x_1 1 5 4 2 8 ··· 3 5 × 1
1 × 1 5 4 2 8 ··· 3 5 y_1
x_2 2 3 5 0 7 ··· 4 1 × 1
1 × 2 3 5 0 7 ··· 4 1 y_2
x_3 9 8 1 3 6 ··· 2 1 × 1
1 × 9 8 1 3 6 ··· 2 1 y_3
x_4 5 9 7 4 2 ··· 1 0 × 0
0 × 1 3 2 5 0 ··· 6 9 y_5
max pool
max pool
$F_{\mathbf{X}}$ w/o mask 9 9 7 4 8 ··· 4 5
$\neq$
9 8 5 5 8 ··· 6 9 $F_{\mathbf{Y}}$ w/o mask
$F_{\mathbf{X}}$ w/ mask 9 8 5 3 8 ··· 4 5
$=$
9 8 5 3 8 ··· 4 5 $F_{\mathbf{Y}}$ w/ mask

Figure 3. We show 4 orderless points of each point cloud. The same subscript denotes the corresponding points. Yellow indicates the maximum of each channel in the features of overlapping points and green indicates the interference of non-overlapping points. The global features of $\mathbf{X}$ and $\mathbf{Y}$ are the same only when they are weighted by the masks $\mathbf{M}_{\mathbf{X}}$ and $\mathbf{M}_{\mathbf{Y}}$.

two overlapping masks $\mathbf{M}^i_{\tilde{\mathbf{X}}}$ and $\mathbf{M}^i_{\mathbf{Y}}$ as,

$$\mathbf{M}^i_{\tilde{\mathbf{X}}} = h\left(g\left(f^i_{\tilde{\mathbf{X}}} \oplus F^i_{\tilde{\mathbf{X}}} \oplus F^i_{\mathbf{Y}}\right) \cdot \mathbf{M}^{i-1}_{\tilde{\mathbf{X}}}\right), \tag{2}$$

$$\mathbf{M}^i_{\mathbf{Y}} = h\left(g\left(f^i_{\mathbf{Y}} \oplus F^i_{\mathbf{Y}} \oplus F^i_{\tilde{\mathbf{X}}}\right) \cdot \mathbf{M}^{i-1}_{\mathbf{Y}}\right), \tag{3}$$

where $h(\cdot)$ denotes the overlapping prediction network, which consists of several convolutional layers followed by a softmax layer. We define the fused point-wise features of the inputs $\mathbf{X}$ and $\mathbf{Y}$ produced by $g(\cdot)$ as $g_{\mathbf{X}}$ and $g_{\mathbf{Y}}$. $\oplus$ denotes the concatenation operation.

3.3. Rigid Transformation Regression

Given the point-wise features $g^i_{\tilde{\mathbf{X}}}$ and $g^i_{\mathbf{Y}}$ at each iteration i, we concatenate them with the features outputting from intermediate layers of the overlapping mask prediction module. Therefore, the features used to regress transformation can be enhanced by the classification information in the mask prediction branch. Meanwhile, the features used to predict the masks benefit from the geometric knowledge in the transformation branch. Then, the concatenated features are fed to the rigid transformation regression network, which produces a 7D vector, with the first 3 values of the 7D vector we use to represent the translation vector $\mathbf{t} \in \mathbb{R}^3$ and the last 4 values represent the 3D rotation in the form of quaternion [29] $\mathbf{q} \in \mathbb{R}^4, \mathbf{q}^T\mathbf{q} = 1$. $r(\cdot)$ represents the whole process in every iteration i, i.e.

$$\left\{\mathbf{q}^i, \mathbf{t}^i\right\} = r\left(\max\{g^i_{\tilde{\mathbf{X}}} \oplus h^i_{\tilde{\mathbf{X}}} \cdot \mathbf{M}^{i-1}_{\tilde{\mathbf{X}}} \oplus g^i_{\mathbf{Y}} \oplus h^i_{\mathbf{Y}} \cdot \mathbf{M}^{i-1}_{\mathbf{Y}}\}\right), \tag{4}$$

where $h^i_{\tilde{\mathbf{X}}}$ and $h^i_{\mathbf{Y}}$ are the concatenated features from the mask prediction branch. $\mathbf{M}^{i-1}_{\tilde{\mathbf{X}}}$ and $\mathbf{M}^{i-1}_{\mathbf{Y}}$ are used to eliminate the interference of the non-overlapping points.

After N iterations, we obtain the overall transformation between the two inputs by accumulating all the estimated transformations at each iteration.

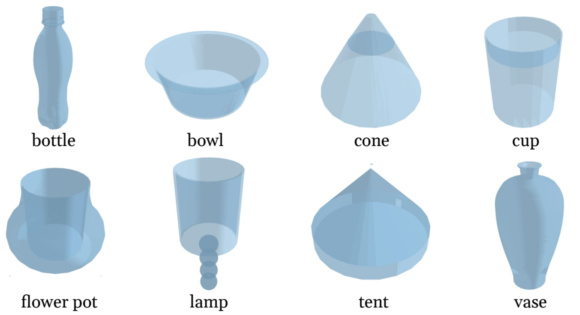

Figure 4. Example CAD models of 8 axisymmetrical categories.

3.4. Loss Functions

We simultaneously predict overlapping masks and estimate rigid transformations, so that two loss functions are proposed to supervise the above two procedures separately.

Mask Prediction Loss. The goal of mask prediction loss is to segment the overlapping region in the input point clouds $\mathbf{X}$ and $\mathbf{Y}$. To balance the contributions of positive and negative samples, the frequency weighted softmax cross-entropy loss is exploited at each iteration i, i.e.

$$\mathcal{L}_{mask} = -\alpha \mathbf{M}^i_g \log(\mathbf{M}^i_p) - (1-\alpha)(1-\mathbf{M}^i_g)\log(1-\mathbf{M}^i_p), \tag{5}$$

where $\mathbf{M}_p$ denotes the probability of points belonging to the overlapping region, and α is the overlapping ratio of the inputs. We define the assumed mask label $\mathbf{M}_g$ to represent the overlapping region of the two inputs, which is computed by setting fixed threshold (set to 0.1) for the closest point distances between the source that transformed by the ground-truth transformation and reference. Each element is

$$M_g = \begin{cases} 1 & \text{if point } \mathbf{x}_j \text{ corresponds to } \mathbf{y}_k \\ 0 & \text{otherwise} \end{cases}. \tag{6}$$

The current mask is estimated based on the previous mask, so the label needs to be recalculated for each iteration.

Transformation Regression Loss. Benefiting from the continuity of the quaternions, it is able to employ a fairly straightforward strategy for training, measuring the deviation of $\{\mathbf{q}, \mathbf{t}\}$ from ground truth for the generated point cloud pairs. So the transformation regression loss at iteration i is

$$\mathcal{L}_{reg} = |\mathbf{q}^i - \mathbf{q}_g| + \lambda \|\mathbf{t}^i - \mathbf{t}_g\|_2, \tag{7}$$

where subscript g denotes ground-truth. We notice that using the combination of ℓ^1 and ℓ^2 distance can marginally improve performance during the training and the inference. λ is empirically set to 4.0 in most of our experiments.

The overall loss is the sum of the two losses:

$$\mathcal{L}_{total} = \mathcal{L}_{mask} + \mathcal{L}_{reg}. \tag{8}$$

We compute the loss for every iteration, and they have equal contribution to the final loss during training.

	Method	RMSE(**R**)		MAE(**R**)		RMSE(**t**)		MAE(**t**)		Error(**R**)		Error(**t**)	
		OS	*TS*	*OS*	*TS*	*OS*	*TS*	*OS*	*TS*	*OS*	*TS*	*OS*	*TS*
(a) Unseen Shapes	ICP [3]	21.043	21.246	8.464	9.431	0.0913	0.0975	0.0467	0.0519	16.460	17.625	0.0921	0.1030
	Go-ICP [35]	13.458	11.296	3.176	3.480	0.0462	0.0571	0.0149	0.0206	6.163	7.138	0.0299	0.0407
	Symmetric ICP [22]	5.333	6.875	4.787	6.069	0.0572	0.0745	0.0517	0.0668	9.424	12.103	0.0992	0.1290
	FGR [40]	4.741	28.865	1.110	16.168	0.0269	0.1380	0.0070	0.0774	2.152	30.192	0.0136	0.1530
	PointNetLK [1]	16.429	14.888	7.467	7.603	0.0832	0.0842	0.0443	0.0464	14.324	14.742	0.0880	0.0920
	DCP [32]	4.291	5.786	3.006	3.872	0.0426	0.0602	0.0291	0.0388	5.871	7.903	0.0589	0.0794
	PRNet [33]	1.588	3.677	0.976	2.201	0.0146	0.0307	0.0101	0.0204	1.871	4.223	0.0201	0.0406
	FMR [13]	2.740	3.456	1.448	1.736	0.0250	0.0292	0.0112	0.0138	2.793	3.281	0.0218	0.0272
	IDAM [17]	4.744	7.456	1.346	4.387	0.0395	0.0604	0.0108	0.0352	2.610	8.577	0.0216	0.0698
	DeepGMR [37]	13.266	21.985	6.883	11.113	0.0748	0.0936	0.0476	0.0587	13.536	20.806	0.0937	0.1171
	Ours	0.898	1.045	0.325	0.507	0.0078	0.0084	0.0049	0.0056	0.639	0.991	0.0099	0.0112
(b) Unseen Categories	ICP [3]	17.236	18.458	8.610	9.335	0.0817	0.0915	0.0434	0.0505	16.824	18.194	0.0855	0.0993
	Go-ICP [35]	13.572	14.162	3.416	4.190	0.0448	0.0533	0.0152	0.0206	6.688	8.286	0.0299	0.0409
	Symmetric ICP [22]	6.599	7.415	5.962	6.552	0.0654	0.0759	0.0592	0.0684	11.713	13.113	0.1134	0.1315
	FGR [40]	6.390	29.838	1.240	16.361	0.0375	0.1470	0.0081	0.0818	2.204	31.153	0.0156	0.1630
	PointNetLK [1]	18.294	21.041	9.730	10.740	0.0917	0.1130	0.0526	0.0629	18.845	20.438	0.1042	0.1250
	DCP [32]	6.754	7.683	4.366	4.747	0.0612	0.0675	0.0403	0.0427	8.566	9.764	0.0807	0.0862
	PRNet [33]	2.712	6.506	1.372	3.472	0.0171	0.0388	0.0118	0.0257	2.607	6.789	0.0237	0.0510
	FMR [13]	5.041	5.119	2.304	2.349	0.0383	0.0296	0.0158	0.0147	4.525	4.553	0.0314	0.0292
	IDAM [17]	6.852	8.346	1.761	4.540	0.0540	0.0590	0.0138	0.0329	3.433	8.679	0.0275	0.0656
	DeepGMR [37]	18.890	23.472	9.322	12.863	0.0870	0.0987	0.0559	0.0658	17.513	24.425	0.1108	0.1298
	Ours	2.079	2.514	0.619	1.004	0.0177	0.0147	0.0077	0.0078	1.241	1.949	0.0154	0.0154
(c) Gaussian Noise	ICP [3]	19.945	21.265	8.546	9.918	0.0898	0.0966	0.0482	0.0541	16.599	18.540	0.0949	0.1070
	Go-ICP [35]	13.612	12.337	3.655	3.880	0.0489	0.0560	0.0174	0.0218	7.257	7.779	0.0348	0.0433
	Symmetric ICP [22]	5.208	6.769	4.703	5.991	0.0518	0.0680	0.0462	0.0609	9.174	11.895	0.0897	0.1178
	FGR [40]	22.347	34.035	10.309	19.188	0.1070	0.1601	0.0537	0.0942	19.934	35.775	0.1068	0.1850
	PointNetLK [1]	20.131	22.399	11.864	13.716	0.0972	0.1092	0.0516	0.0601	18.552	20.250	0.1032	0.1291
	DCP [32]	4.862	4.775	3.433	2.964	0.0486	0.0474	0.0340	0.0300	6.653	6.024	0.0690	0.0616
	PRNet [33]	1.911	3.197	1.213	2.047	0.0180	0.0294	0.0123	0.0195	2.284	3.932	0.0245	0.0392
	FMR [13]	2.898	3.551	1.747	2.178	0.0246	0.0273	0.0133	0.0155	3.398	4.200	0.0260	0.0307
	IDAM [17]	5.551	6.846	2.990	3.997	0.0486	0.0563	0.0241	0.0318	5.741	7.810	0.0480	0.0629
	DeepGMR [37]	17.693	20.433	8.578	10.964	0.0849	0.0944	0.0531	0.0593	16.504	20.830	0.1048	0.1183
	Ours	1.009	1.305	0.548	0.757	0.0089	0.0103	0.0061	0.0075	1.076	1.490	0.0123	0.0149

Table 1. Results on ModelNet40. For each metric, the left column *OS* denotes the results on the original once-sampled data, and the right column *TS* denotes the results on our twice-sampled data. Red indicates the best performance and blue indicates the second-best result.

4. Experiments

In this section, we first describe the pre-processing for the datasets and the implementation details of our method in Sec. 4.1. Concurrently, the experimental settings of competitors are presented in Sec. 4.2. Moreover, we show the results for different experiments to demonstrate the effectiveness and robustness of our method in Sec. 4.3 and Sec. 4.4. Finally, we perform ablation studies in Sec. 4.5.

4.1. Dataset and Implementation Details

ModelNet40. We use the ModelNet40 dataset to test the effectiveness following [1, 32, 27, 33, 13, 36, 17]. The ModelNet40 contains CAD models from 40 categories. Previous works use processed data from PointNet [21], which has two issues when adopted to registration task: (1) for each object, it only contains 2,048 points sampled from the CAD model. However, in realistic scenes, the points in $\mathbf{X}$ have no exact correspondences in $\mathbf{Y}$. Training on this data cause over-fitting issue even adding noise or resampling, which is demonstrated by the experiment shown in our supplementary; (2) it involves some axisymmetrical categories, including *bottle*, *bowl*, *cone*, *cup*, *flower pot*, *lamp*, *tent* and *vase*, Fig. 4 shows some examples. However, giving fixed ground-truths to axisymmetrical data is illogical, since it is possible to obtain arbitrary angles on the symmetrical axis for accurate registration. Fixing the label on symmetrical axis makes no sense. In this paper, we propose a proper manner to generate data. Specifically, we uniformly sample 2,048 points from each CAD model 40 times with different random seeds, then randomly choose 2 of them as $\mathbf{X}$ and $\mathbf{Y}$. It guarantees that we can obtain $C_{40}^{2} = 780$ combinations for each object. We denote the data that points are sampled only once from CAD models as **once-sampled (*OS*)** data, and refer our data as **twice-sampled (*TS*)** data. Moreover, we simply remove the axisymmetrical categories.

To evaluate the effectiveness and robustness of our network, we use the official train and test splits of the first 14 categories (*bottle*, *bowl*, *cone*, *cup*, *flower pot* and *lamp* are removed) for training and validation respectively, and the test split of the remaining 18 categories (*tent* and *vase* are removed) for test. This results in 4,196 training, 1,002 validation, and 1,146 test models. Following previous works [32, 33, 13, 36, 17], we randomly generate three Euler angle rotations within $[0°, 45°]$ and translations within $[-0.5, 0.5]$ on each axis as the rigid transformation.

Stanford 3D Scan. We use the Stanford 3D Scan dataset [7] to test the generalizability of our method. The dataset has 10 real scans. The partial manner in PRNet [33] is applied to generate partially overlapping point clouds.

7Scenes. 7Scenes [30] is a widely used registration benchmark where data is captured by a Kinect camera in indoor environments. Following [39, 13], multiple depth images are projected into point clouds, then fused through truncated signed distance function (TSDF). The dataset is divided into 293 and 60 scans for training and test. The partial manner in PRNet [33] is applied.

Implementation Details. Our network architecture is illustrated in Fig. 2. We run $N = 4$ iterations during training and test. Nevertheless, the $\{\mathbf{q}, \mathbf{t}\}$ gradients are stopped at the beginning of each iteration to stabilize the training. Since the masks predicted by the first iteration may be inaccurate at the beginning of training, some overlapping points may be misclassified and affect the sequent iterations, we apply the masks after the second iteration. We train our network with Adam [16] optimizer for 260k iterations. The initial learning rate is 0.0001 and is multiplied by 0.1 after 220k iterations. The batch size is set to 64.

4.2. Baseline Algorithms

We compare our method to traditional methods: ICP [3], Go-ICP [35], Symmetric ICP [22], FGR [40], as well as recent DL based works: PointNetLK [1], DCP [32], RPMNet [36], FMR [13], PRNet [33], IDAM [17] and DeepGMR [37]. We use implementations of ICP and FGR in Intel Open3D [41], Symmetric ICP in PCL [26] and the others released by their authors. Moreover, the test set is fixed by setting random seeds. Note that the normals used in FGR and RPMNet are calculated after data pre-processing, which is slightly different from the implementation in RPMNet. We use the supervised version of FMR.

Following [32, 36], we measure anisotropic errors: root mean squared error (RMSE) and mean absolute error (MAE) of rotation and translation, and isotropic errors:

$$\text{Error}(\mathbf{R}) = \angle\left(\mathbf{R}_g^{-1}\mathbf{R}_p\right), \ \text{Error}(\mathbf{t}) = \|\mathbf{t}_g - \mathbf{t}_p\|_2, \quad (9)$$

where $\mathbf{R}_g$ and $\mathbf{R}_p$ denote the ground-truth and predicted rotation matrices converted from the quaternions $\mathbf{q}_g$ and $\mathbf{q}_p$ respectively. All metrics should be zero if the rigid alignment is perfect. The angular metrics are in units of degrees.

4.3. Evaluation on ModelNet40

To evaluate the effectiveness of different methods, we conduct several experiments in this section. The data preprocessing settings of the first 3 experiments are the same as PRNet [33] and IDAM [17]. In addition, the last experiment shows the robustness of our method to different partial manners, which is used in RPMNet [36].

Unseen Shapes. In this experiment, we train models on training set of the first 14 categories and evaluate on validation set of the same categories without noise. Note that all points in $\mathbf{X}$ have **exact correspondences** in $\mathbf{Y}$ for the *OS* data. We partial $\mathbf{X}$ and $\mathbf{Y}$ by randomly placing a point in space and computing its 768 nearest neighbors respectively, which is the same as used in [33, 17]. All DL based methods are trained independently on both *OS* and *TS* data. Table 1(a) shows the results.

We can find that ICP [3] performs poorly because of the large difference in initial positions. Go-ICP [35] and FGR [40] achieve better performances, which are comparable to some DL based methods [1, 32, 13, 17]. Note that the large performance gap of FGR on two different data is caused by the calculation manner of normals. We use normals that are computed after data pre-processing, so that normals of $\mathbf{X}$ and $\mathbf{Y}$ are different in our *TS* data. In addition, the results of IDAM [17] are marginally worse than PRNet [33] because of the fixing manner of the test data, which is used in other DL based methods. Our method achieves very accurate registration and ranks first in all metrics. Example results on *TS* data are shown in Fig. 6(a).

Unseen Categories. We evaluate the performance on unseen categories without noise in this experiment. Models are trained on the first 14 categories and tested on the other 18 categories. The data pre-processing is the same as the first experiment. The results are summarized in Table 1(b). We can find that the performances of all DL based methods become marginally worse without training on the same categories. Nevertheless, traditional algorithms are not affected so much because of the handcrafted features. Our approach outperforms all the other methods. A qualitative comparison of the registration results can be found in Fig. 6(b).

Gaussian Noise. In this experiment, we add noises that sampled from $\mathcal{N}(0, 0.01^2)$ and clipped to $[-0.05, 0.05]$, then repeat the first experiment (unseen shapes). Table 1(c) shows the results. FGR is sensitive to noise, so that it performs much worse than the noise-free case. All DL based methods get worse with noises injected on the *OS* data. The performances of correspondences matching based methods (DCP, PRNet and IDAM) show an opposite tendency on the *TS* data compared to the global feature based methods (PointNetLK, FMR and ours), since the robustness of local feature descriptor is improved by the noise augmentation during training. Our method achieves the best performance. Example results are shown in Fig. 6(c).

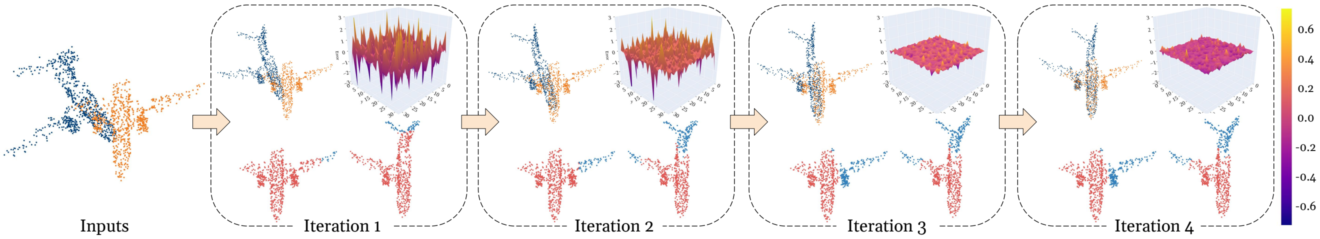

Figure 5. We show the registration result (top left), the difference between the global features of the inputs $\mathbf{X}$ and $\mathbf{Y}$ (top right), and the predicted masks (bottom) at each iteration. Red and blue indicate the predicted overlapping and non-overlapping regions respectively.

Method	RMSE($\mathbf{R}$)	MAE($\mathbf{R}$)	RMSE($\mathbf{t}$)	MAE($\mathbf{t}$)	Error($\mathbf{R}$)	Error($\mathbf{t}$)
ICP [3]	21.893	13.402	0.1963	0.1278	26.632	0.2679
Symmetric ICP [22]	12.576	10.987	0.1478	0.1203	21.807	0.2560
FGR [40]	46.213	30.116	0.3034	0.2141	58.968	0.4364
PointNetLK [1]	29.733	21.154	0.2670	0.1937	42.027	0.3964
DCP [32]	12.730	9.556	0.1072	0.0774	12.173	0.1586
RPMNet [36]	6.160	2.467	0.0618	0.0274	4.913	0.0589
FMR [13]	11.674	7.400	0.1364	0.0867	14.121	0.1870
Ours	4.356	1.924	0.0486	0.0223	3.834	0.0476

Table 2. Results on the twice-sampled (*TS*) unseen categories with Gaussian noise using the partial manner of RPMNet.

Different Partial Manners. We notice that the previous works [33, 36] use different partial manners. To evaluate the effectiveness on different partial data, we also test the performance of different algorithms on the test set used in [36]. We retrain all DL based methods and show the results of the most difficult situation (unseen categories with Gaussian noise) in Table 2. For details about the partial manners, please refer to our supplementary.

4.4. Evaluation on Real Data

To further evaluate the generalizability, we conduct experiments on the Stanford 3D Scan and 7Scenes datasets. Since the Stanford 3D Scan dataset only has 10 real scans, we directly use the model trained on the ModelNet40 without fine-tuning. Some qualitative examples are shown in Fig. 9. Furthermore, we evaluate our method on the 7Scenes indoor dataset. The point clouds are normalized into the unit sphere. Our model is trained on 6 categories (*Chess*, *Fires*, *Heads*, *Office*, *Pumpkin* and *Stairs*) and tested on the other category (*Redkitchen*). Fig. 10 shows some examples. For more results, please refer to our supplementary.

4.5. Ablation Studies

We perform ablation studies on the unseen shapes with noise *TS* data to show the effectiveness of our components and settings. As shown in Table 3, we denote our model that removed the following components as **B**aseline (B): **M**ask prediction module (M), **M**ask prediction **L**oss (ML), **F**usion layers (F) before the regression module, and **C**onnection (C) between mask prediction and regression modules. Besides, we only use top-k points based on the mask prediction probabilities to estimate rigid transformations. Moreover, we set different λ in the loss function.

Model	RMSE($\mathbf{R}$)	MAE($\mathbf{R}$)	RMSE($\mathbf{t}$)	MAE($\mathbf{t}$)	Error($\mathbf{R}$)	Error($\mathbf{t}$)
B	3.216	2.751	0.0267	0.0232	5.250	0.0463
B+M	3.437	2.943	0.0349	0.0301	5.550	0.0604
B+M+ML	1.655	1.417	0.0158	0.0138	2.681	0.0274
B+M+ML+F	1.453	0.892	0.0111	0.0087	1.722	0.0171
B+M+ML+F+C	**1.305**	**0.757**	**0.0103**	**0.0075**	**1.490**	**0.0149**
Top-k, k=500	1.364	1.168	0.0127	0.0109	2.255	0.0220
Top-k, k=300	1.399	1.203	0.0161	0.0141	2.282	0.0278
Top-k, k=100	1.483	1.270	0.0180	0.0157	2.458	0.0311
λ=2.0	1.356	0.900	0.0109	0.0077	1.721	0.0154
λ=0.5	1.397	0.986	0.0116	0.0085	1.890	0.0169
λ=0.1	1.416	1.068	0.0127	0.0095	2.038	0.0189

Table 3. Ablation studies of each component and different settings.

We can see that without being supervised by the mask prediction loss, it has no improvement based on the baseline, which shows that the mask prediction can not be trained unsupervised. Comparing the third to the fifth lines with the baseline, we can find that all the components improve the performance. Since we do not estimate the matching candidates among the overlapping points, the top-k points from the source and reference may not be correspondent and distributed in the point clouds centrally, so that the results of top-k models are worse than using the entire masks. Furthermore, we adjust the λ in the loss function. Since the data generation manner of [33, 36] constrain the translation within $[-0.5, 0.5]$ as we use the ℓ^2 loss for the translation, the translation loss is smaller than the quaternion, so that a large λ aims to form comparable terms.

5. Discussion

In this section, we conduct several experiments to better understand how various settings affect our algorithm.

5.1. Effects of Mask

To have a better intuition about the overlapping masks, we visualize the intermediate results in Fig. 5. We reshape the global feature vector of length 1,024 into a 32×32 square matrix and compute the error between the transformed source $\tilde{\mathbf{X}}$ and reference $\mathbf{Y}$. At the first few iterations, the global feature differences are large, and the inputs are poorly aligned given inaccurate overlapping masks. With continuous iterating, the global feature difference becomes extremely small, while the alignment and predicted overlapping masks are almost perfect.

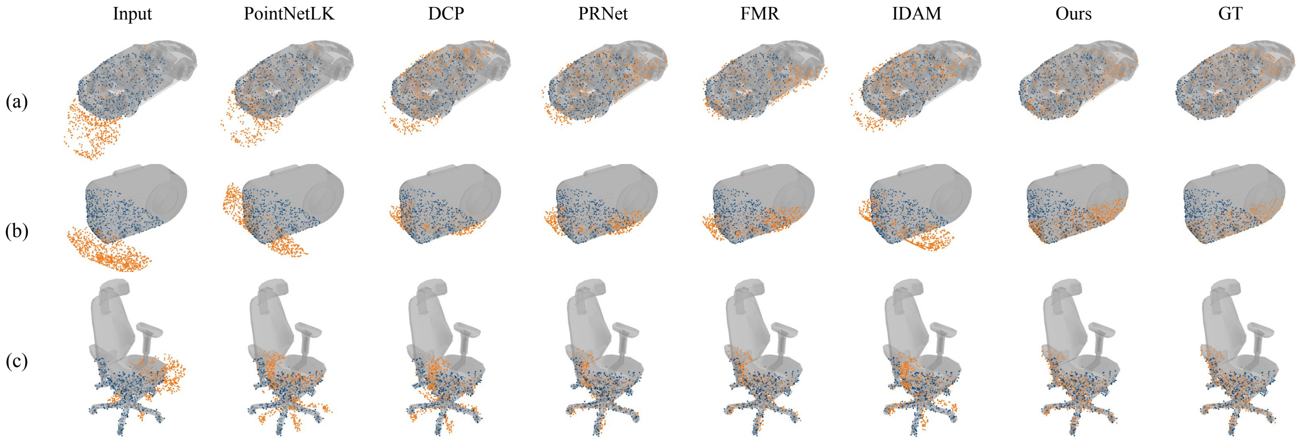

Figure 6. Example results on ModelNet40. (a) Unseen shapes, (b) Unseen categories, and (c) Unseen shapes with Gaussian noise.

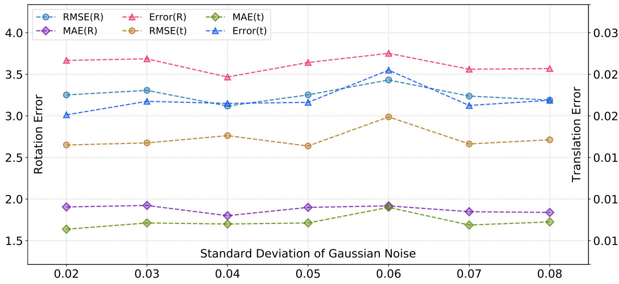

Figure 7. Errors of our method under different noise levels.

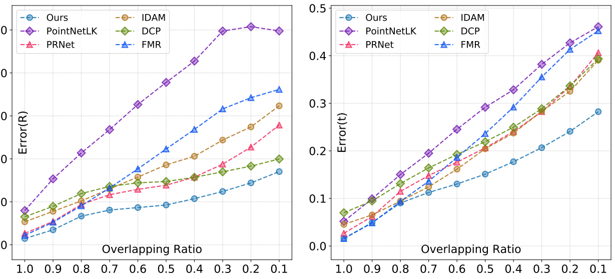

Figure 8. Isotropic errors of different overlapping ratios.

5.2. Robustness Against Noise

To further demonstrate the robustness of our method, we train and test our models on ModelNet40 under different noise levels, as shown in Fig. 7. We add noise sampled from $N(0, \sigma^2)$ and clipped to $[-0.05, 0.05]$. The data is the same as the third experiment in Sec 4.3. Our method achieves comparable performance under various noise levels.

5.3. Different Overlapping Ratio

We test the best models of all methods from the first experiment in Sec. 4.3 on the ModelNet40 *TS* validation set with the overlapping ratio decreasing from 1.0 to 0.1. We first partial $\mathbf{X}$, then randomly select two adjacent parts from overlapping and non-overlapping regions of $\mathbf{Y}$. Fig. 8 shows the results. Unfortunately, DeepGMR fails to obtain sensible results. Our method shows the best performance.

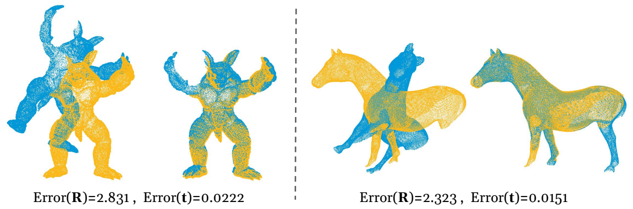
Error(**R**)=2.831 , Error(**t**)=0.0222
Error(**R**)=2.323 , Error(**t**)=0.0151

Figure 9. Example results on Stanford 3D Scan.

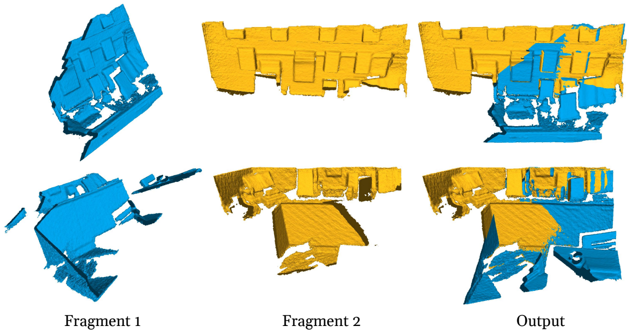

Figure 10. Example results on 7Scenes.

6. Conclusion

We have presented the OMNet, a novel method for partial-to-partial point cloud registration. Previous global feature based methods pay less attention to partiality. They treat the input points equally, which are easily disturbed by the non-overlapping regions. Our method learns the overlapping masks to reject non-overlapping points for robust registration. Besides, we propose a practical data generation manner to solve the over-fitting issue and remove the axisymmetrical categories in the ModelNet40 dataset. Experiments show the state-of-the-art performance of our method.

Acknowledgements. This work is supported by the National Key R&D Plan of the Ministry of Science and Technology (Project No.2020AAA0104400) and the National Natural Science Foundation of China (NSFC) under Grants No.62071097, No.61872067, No.62031009 and No.61720106004.

References

[1] Yasuhiro Aoki, Hunter Goforth, Rangaprasad Arun Srivatsan, and Simon Lucey. PointNetLK: Robust & efficient point cloud registration using pointnet. In *Proc. CVPR*, pages 7163–7172, 2019. 1, 2, 5, 6, 7

[2] Ronald T Azuma. A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, 6(4):355–385, 1997. 1

[3] Paul J. Besl and Neil D. McKay. A method for registration of 3d shapes. volume 14, pages 239–256, 1992. 1, 2, 5, 6, 7

[4] Mark Billinghurst, Adrian Clark, and Gun Lee. A survey of augmented reality. *Interaction*, 8(2-3):73–272, 2014. 1

[5] Sofien Bouaziz, Andrea Tagliasacchi, and Mark Pauly. Sparse iterative closest point. In *Computer Graphics forum*, volume 32, pages 113–123, 2013. 1

[6] Julie Carmigniani, Borko Furht, Marco Anisetti, Paolo Ceravolo, Ernesto Damiani, and Misa Ivkovic. Augmented reality technologies, systems and applications. *Multimedia Tools and Applications*, 51(1):341–377, 2011. 1

[7] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312, 1996. 6

[8] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2, 3

[9] Andrew W. Fitzgibbon. Robust registration of 2d and 3d point sets. *Image and Vision Computing*, 21(13-14):1145–1153, 2003. 1

[10] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proc. CVPR*, pages 3354–3361, 2012. 1

[11] Natasha Gelfand, Leslie Ikemoto, Szymon Rusinkiewicz, and Marc Levoy. Geometrically stable sampling for the icp algorithm. In *International Conference on 3D Digital Imaging and Modeling*, pages 260–267, 2003. 2

[12] Guy Godin, Marc Rioux, and Rejean Baribeau. Three-dimensional registration using range and intensity information. In *Videometrics III*, volume 2350, pages 279–290, 1994. 2

[13] Xiaoshui Huang, Guofeng Mei, and Jian Zhang. Feature-metric registration: A fast semi-supervised approach for robust point cloud registration without correspondences. In *Proc. CVPR*, pages 11366–11374, 2020. 1, 2, 5, 6, 7

[14] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Annual ACM Symposium on User Interface Software and Technology*, pages 559–568, 2011. 1

[15] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016. 2, 3

[16] P. Diederik Kingma and Lei Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2015. 6

[17] 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. In *Proc. ECCV*, pages 378–394, 2020. 1, 2, 3, 5, 6

[18] Bruce D. Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. IJCAI*, page 674–679, 1981. 2

[19] Michael Merickel. 3d reconstruction: the registration problem. *Computer vision, Graphics, and Image Processing*, 42(2):206–219, 1988. 1

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

[21] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proc. CVPR*, pages 652–660, 2017. 2, 3, 5

[22] Szymon Rusinkiewicz. A symmetric objective function for icp. *ACM Trans. Graphics*, 38(4):1–7, 2019. 2, 5, 6, 7

[23] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *International Conference on 3-D Digital Imaging and Modeling (3DIM)*, pages 145–152, 2001. 1, 2

[24] 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

[25] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *International Conference on Intelligent Robots and Systems*, pages 3384–3391, 2008. 2

[26] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *Proc. ICRA*, pages 1–4, 2011. 6

[27] Vinit Sarode, Xueqian Li, Hunter Goforth, Yasuhiro Aoki, Rangaprasad Arun Srivatsan, Simon Lucey, and Howie Choset. Pcrnet: Point cloud registration network using pointnet encoding. *arXiv preprint arXiv:1908.07906*, 2019. 1, 2, 5

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

[29] Ken Shoemake. Animating rotation with quaternion curves. In *Annual Conference on Computer Graphics and Interactive Techniques*, pages 245–254, 1985. 4

[30] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proc. CVPR*, pages 2930–2937, 2013. 6

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

[32] Yue Wang and Justin M. Solomon. Deep closest point: Learning representations for point cloud registration. In *Proc. ICCV*, pages 3523–3532, 2019. 1, 2, 5, 6, 7

[33] Yue Wang and Justin M. Solomon. Prnet: Self-supervised learning for partial-to-partial registration. In *Proc. NeurIPS*, pages 8814–8826, 2019. 1, 2, 3, 5, 6, 7

[34] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d ShapeNets: A deep representation for volumetric shapes. In *Proc. CVPR*, pages 1912–1920, 2015. 2

[35] Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-icp: Solving 3d registration efficiently and globally optimally. In *Proc. CVPR*, pages 1457–1464, 2013. 1, 2, 5, 6

[36] Zi Jian Yew and Gim Hee Lee. Rpm-net: Robust point matching using learned features. In *Proc. CVPR*, pages 11824–11833, 2020. 1, 2, 3, 5, 6, 7

[37] Wentao Yuan, Benjamin Eckart, Kihwan Kim, Varun Jampani, Dieter Fox, and Jan Kautz. Deepgmr: Learning latent gaussian mixture models for registration. In *Proc. ECCV*, pages 733–750, 2020. 1, 2, 5, 6

[38] Ekim Yurtsever, Jacob Lambert, Alexander Carballo, and Kazuya Takeda. A survey of autonomous driving: Common practices and emerging technologies. *IEEE Access*, 8:58443–58469, 2020. 1

[39] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *Proc. CVPR*, pages 1802–1811, 2017. 6

[40] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast global registration. In *Proc. ECCV*, pages 766–782, 2016. 2, 5, 6, 7

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