Geometric Transformer for Fast and Robust Point Cloud Registration
Supplementary Material

A. Network Architecture Details

A.1. Geometric Structure Embedding

First, we provide the detailed computation for our geometric structure embedding. The geometric structure embedding encodes distances in superpoint pairs and angles in superpoint triplets. Due to the continuity of the sinusoidal embedding function [16], we use it instead of learned embedding vectors to compute the pair-wise distance embedding and the triplet-wise angular embedding.

Given the distance $\rho_{i,j} = \|\hat{\mathbf{p}}_i - \hat{\mathbf{p}}_j\|_2$ between $\hat{\mathbf{p}}_i$ and $\hat{\mathbf{p}}_j$, the pair-wise distance embedding $\mathbf{r}^D_{i,j}$ is computed as:

$$\begin{cases} r^D_{i,j,2k} = \sin(\frac{d_{i,j}/\sigma_d}{10000^{2k/d_t}}) \\ r^D_{i,j,2k+1} = \cos(\frac{d_{i,j}/\sigma_d}{10000^{2k/d_t}}) \end{cases}, \tag{1}$$

where d_t is the feature dimension, and σ_d is a temperature which controls the sensitivity to distance variations.

The triplet-wise angular embedding can be computed in the same way. Given the angle $\alpha^k_{i,j}$, the triplet-wise angular embedding $\mathbf{r}^A_{i,j,k}$ is computed as:

$$\begin{cases} r^A_{i,j,k,2x} = \sin(\frac{\alpha^k_{i,j}/\sigma_a}{10000^{2x/d_t}}) \\ r^A_{i,j,k,2x+1} = \cos(\frac{\alpha^k_{i,j}/\sigma_a}{10000^{2x/d_t}}) \end{cases}, \tag{2}$$

where σ_a is another temperature to control the sensitivity to angular variantions.

A.2. Point Matching Module

For completeness, we then provide the details of the optimal transport layer [13] in the point matching module. For each superpoint correspondence $(\hat{\mathbf{p}}_{x_i}, \hat{\mathbf{q}}_{y_i})$, its local point correspondences are extracted from their local patches $\mathcal{G}^{\mathcal{P}}_{x_i}$ and $\mathcal{G}^{\mathcal{Q}}_{y_i}$. We first compute a cost matrix $\mathbf{C}_i \in \mathbb{R}^{n_i \times m_i}$ using the feature matrices of the two patches:

$$\mathbf{C}_i = \mathbf{F}^{\mathcal{P}}_{x_i} (\mathbf{F}^{\mathcal{Q}}_{y_i})^T / \sqrt{\tilde{d}}, \tag{3}$$

where $n_i = |\mathcal{G}^{\mathcal{P}}_{x_i}|$, $m_i = |\mathcal{G}^{\mathcal{Q}}_{y_i}|$. The cost matrix $\mathbf{C}_i$ is then augmented to $\bar{\mathbf{C}}_i \in \mathcal{R}^{(n_i+1)\times(m_i+1)}$ by appending a new row and a new column filled with a learnable dustbin parameter α as in [13]. The point matching problem can then be formulated as an optimal transport problem which maximizes $\sum_{j,k} \bar{\mathbf{C}}_i \cdot \bar{\mathbf{Z}}_i$, where $\bar{\mathbf{Z}}_i \in \mathcal{R}^{(n_i+1)\times(m_i+1)}$ is the soft assignment matrix satisfying:

$$\sum_{k=1}^{m_i+1} \bar{z}^i_{j,k} = \begin{cases} 1, & 1 \le j \le n_i \\ m_i, & j = n_i + 1 \end{cases}, \tag{4}$$

$$\sum_{j=1}^{n_i+1} \bar{z}^i_{j,k} = \begin{cases} 1, & 1 \le k \le m_i \\ n_i, & k = m_i + 1 \end{cases}. \tag{5}$$

Here $\bar{\mathbf{Z}}_i$ can be solved by the differentiable Sinkhorn algorithm [14] with doubly-normalization iterations:

$${}^{(t)}u^i_j = \log \alpha^i_j - \log \sum_{k=1}^{m_i+1} \exp(\bar{c}^i_{j,k} + {}^{(t-1)}v^i_k), \tag{6}$$

$${}^{(t)}v^i_k = \log \beta^i_k - \log \sum_{j=1}^{n_i+1} \exp(\bar{c}^i_{j,k} + {}^{(t)}u^i_j), \tag{7}$$

where

$$\alpha^i_j = \begin{cases} \frac{1}{n_i+m_i}, & 1 \le j \le n_i \\ \frac{m_i}{n_i+m_i}, & j = n_i + 1 \end{cases}, \tag{8}$$

$$\beta^i_k = \begin{cases} \frac{1}{n_i+m_i}, & 1 \le k \le m_i \\ \frac{n_i}{n_i+m_i}, & k = m_i + 1 \end{cases}. \tag{9}$$

The algorithm starts with ${}^{(0)}\mathbf{u} = \mathbf{0}^{n_i+1}$ and ${}^{(0)}\mathbf{v} = \mathbf{0}^{m_i+1}$. The assignment matrix $\bar{\mathbf{Z}}_i$ is then computed as:

$$\bar{z}^i_{j,k} = \exp(\bar{c}^i_{j,k} + u^i_j + v^i_k) \cdot (n_i + m_i). \tag{10}$$

We run $t_0 = 100$ Sinkhorn iterations following [13]. $\bar{\mathbf{Z}}_i$ is then recovered to $\mathbf{Z}_i \in \mathbb{R}^{n_i \times m_i}$ by dropping the last row and the last column, which is used as the confidence matrix of the candidate matches. The local point correspondences are extracted by mutual top-k selection on $\mathbf{Z}_i$. We ignore the matches whose confidence scores are too small (*i.e.*, $z^i_{x_j,y_j} < 0.05$). The hyper-parameter k controls the number of point correspondences as described in Appx. A.3. At last, the final global dense point correspondences are generated by combining the local point correspondences from all superpoint correspondences together.

Stage	3DMatch	KITTI
Backbone		
1	KPConv($1 \rightarrow 64$) ResBlock($64 \rightarrow 128$)	KPConv($1 \rightarrow 64$) ResBlock($64 \rightarrow 128$)
2	ResBlock($64 \rightarrow 128$, strided) ResBlock($128 \rightarrow 256$) ResBlock($256 \rightarrow 256$)	ResBlock($64 \rightarrow 128$, strided) ResBlock($128 \rightarrow 256$) ResBlock($256 \rightarrow 256$)
3	ResBlock($256 \rightarrow 256$, strided) ResBlock($256 \rightarrow 512$) ResBlock($512 \rightarrow 512$)	ResBlock($256 \rightarrow 256$, strided) ResBlock($256 \rightarrow 512$) ResBlock($512 \rightarrow 512$)
4	ResBlock($512 \rightarrow 512$, strided) ResBlock($512 \rightarrow 1024$) ResBlock($1024 \rightarrow 1024$)	ResBlock($512 \rightarrow 512$, strided) ResBlock($512 \rightarrow 1024$) ResBlock($1024 \rightarrow 1024$)
5	-	ResBlock($1024 \rightarrow 1024$, strided) ResBlock($1024 \rightarrow 2048$) ResBlock($2048 \rightarrow 2048$)
6	-	NearestUpsampling UnaryConv($3072 \rightarrow 1024$)
7	NearestUpsampling UnaryConv($1536 \rightarrow 512$)	NearestUpsampling UnaryConv($1536 \rightarrow 512$)
8	NearestUpsampling UnaryConv($768 \rightarrow 256$)	NearestUpsampling UnaryConv($768 \rightarrow 256$)
Superpoint Matching Module		
1	Linear($1024 \rightarrow 256$)	Linear($2048 \rightarrow 128$)
2	GeometricSelfAttention(256, 4) FeatureCrossAttention(256, 4)	GeometricSelfAttention(128, 4) FeatureCrossAttention(128, 4)
3	GeometricSelfAttention(256, 4) FeatureCrossAttention(256, 4)	GeometricSelfAttention(128, 4) FeatureCrossAttention(128, 4)
4	GeometricSelfAttention(256, 4) FeatureCrossAttention(256, 4)	GeometricSelfAttention(128, 4) FeatureCrossAttention(128, 4)
5	Linear($256 \rightarrow 256$)	Linear($128 \rightarrow 256$)

Table 1. Network architecture for 3DMatch and KITTI.

A.3. Network Configurations

Backbone. We use a KPConv-FPN backbone for feature extraction. The grid subsampling scheme [15] is used to downsample the point clouds. Before being fed into the backbone, the input point clouds are first downsampled with a voxel size of 2.5cm on 3DMatch and 30cm on KITTI. The voxel size is then doubled in each downsampling operation. We use a 4-stage backbone for 3DMatch and a 5-stage backbone for KITTI because the point clouds in KITTI are much larger than those in 3DMatch. The configurations of KPConv are the same as in [7]. And we use group normalization [17] with 8 groups after the KPConv layers. The detailed network configurations are shown in Tab. 1.

Superpoint Matching Module. At the beginning of the superpoint matching module, a linear projection is used to compress the feature dimension. For 3DMatch, the feature dimension is 256. For KITTI, we halve the feature dimension to 128 to reduce memory footprint. We then interleave the geometric self-attention module and the feature-based

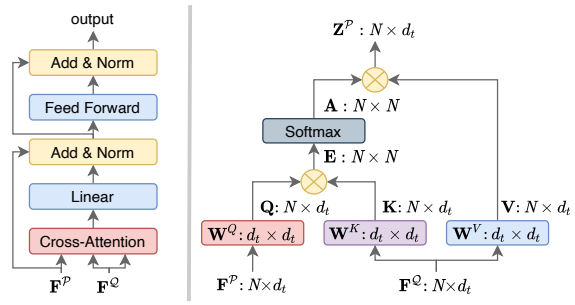


Figure 1. Left: The structure of feature-based cross-attention module. Right: The computation graph of cross-attention.

cross-attention module for $N_t = 3$ times:

$$^{(1)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{self}} = \text{GeometricSelfAtt}(\hat{\mathcal{P}}, \hat{\mathbf{F}}^{\mathcal{P}}\mathbf{W}_{\text{in}}), \tag{11}$$

$$^{(1)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{self}} = \text{GeometricSelfAtt}(\hat{\mathcal{Q}}, \hat{\mathbf{F}}^{\mathcal{Q}}\mathbf{W}_{\text{in}}), \tag{12}$$

$$^{(t)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{self}} = \text{GeometricSelfAtt}(\hat{\mathcal{P}}, {}^{(t-1)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{cross}}), \tag{13}$$

$$^{(t)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{self}} = \text{GeometricSelfAtt}(\hat{\mathcal{Q}}, {}^{(t-1)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{cross}}), \tag{14}$$

$$^{(t)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{cross}} = \text{FeatureCrossAtt}({}^{(t)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{self}}, {}^{(t)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{self}}), \tag{15}$$

$$^{(t)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{cross}} = \text{FeatureCrossAtt}({}^{(t)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{self}}, {}^{(t)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{cross}}). \tag{16}$$

All attention modules have 4 attention heads. In the geometric structure embedding, we use $\sigma_d = 0.2$m on 3DMatch and $\sigma_d = 4.8$m on KITTI (*i.e.*, the voxel size in the coarsest resolution level), while $\sigma_a = 15°$ on both datasets. The computation of the feature-based cross-attention for $\hat{\mathcal{P}}$ is shown in Fig. 1. Afterwards, we use another linear projection to project the features to 256-d, *i.e.*, the final $\hat{\mathbf{H}}^{\mathcal{P}}$ and $\hat{\mathbf{H}}^{\mathcal{Q}}$:

$$\hat{\mathbf{H}}^{\mathcal{P}} = {}^{(N_t)}\hat{\mathbf{F}}^{\mathcal{P}}_{\text{cross}}\mathbf{W}_{\text{out}}, \tag{17}$$

$$\hat{\mathbf{H}}^{\mathcal{Q}} = {}^{(N_t)}\hat{\mathbf{F}}^{\mathcal{Q}}_{\text{cross}}\mathbf{W}_{\text{out}}. \tag{18}$$

Local-to-Global Registration. In the local-to-global registration, we only use the superpoint correspondences with at least 3 local point correspondences to compute the transformation candidates. To select the best transformation, the acceptance radius is $\tau_a = 10$cm on 3DMatch and $\tau_a = 60$cm on KITTI. At last, we iteratively recompute the transformation with the surviving inlier matches for $N_r = 5$ times, which is similar with the post-refinement process in [1]. However, we do not change the weights of the correspondences during the refinement. The impact of the number of iterations in the refinement is studied in Appx. D.3.

Implementation details. We implement and evaluate our GeoTransformer with PyTorch [12] on a Xeon Glod 5218 CPU and an NVIDIA RTX 3090 GPU. The network is trained with Adam optimizer [8] for 40 epochs on 3DMatch and 80 epochs on KITTI. The batch size is 1 and the weight decay is 10^{-6}. The learning rate starts from 10^{-4} and decays exponentially by 0.05 every epoch on 3DMatch and every 4 epochs on KITTI. We use the matching radius of

$\tau = 5$cm for 3DMatch and $\tau = 60$cm for KITTI (*i.e.*, the voxel size in the resolution level of $\tilde{\mathcal{P}}$ and $\tilde{\mathcal{Q}}$) to determine overlapping during the generation of both superpoint-level and point-level ground-truth matches. The same data augmentation as in [7] is adopted. We randomly sample $N_g = 128$ ground-truth superpoint matches during training, and use $N_c = 256$ putative ones during testing.

Correspondences sampling strategy. For 3DMatch, we vary the hyper-parameter k in the mutual top-k selection of the point matching module to control the number of the point correspondences for GeoTransformer, *i.e.*, $k = 1$ for 250/500/1000 matches, $k = 2$ for 2500 matches, and $k = 3$ for 5000 matches. And we use top-k selection to sample a certain number of the correspondences instead of random sampling as in [5, 7, 18], which makes our correspondences deterministic. For the registration with LGR (Tab. 2(bottom) of our main paper), we use $k = 3$ to generate around 6000 correspondences for each point cloud pair. For the baselines, we report the results from their original papers or official models in Tab. 1 of our main paper.

For the registration with weighted SVD (Tab. 2(middle) of our main paper), the correspondences of the baselines are extracted in the following manner: we first sample 5000 keypoints and generate the correspondences with mutual nearest neighbor selection in the feature space, and then the top 250 correspondences with the smallest feature distances are used to compute the transformation. The weights of the correspondences are computed as $w_i = \exp(-\|\mathbf{f}^{\tilde{\mathcal{P}}}_{x_i} - \mathbf{f}^{\tilde{\mathcal{Q}}}_{y_i}\|_2^2)$, where $\mathbf{f}^{\mathcal{P}}_{x_i}$ and $\mathbf{f}^{\mathcal{Q}}_{y_i}$ are the respective descriptors of the correspondences. In the sampling strategies that we have tried, this scheme achieves the best registration results.

For KITTI, we use $k = 2$ and select the top 5000 point correspondences following [2, 7]. All other hyperparameters are the same as those in 3DMatch.

B. Metrics

Following common practice [2, 5, 7], we use different metrics for 3DMatch and KITTI. On 3DMatch, we report *Inlier Ratio*, *Feature Matching Recall* and *Registration Recall*. We also report *Patch Inlier Ratio* to evaluate the superpoint (patch) correspondences. On KITTI, we report *Relative Rotation Error*, *Relative Translation Error* and *Registration Recall*.

B.1. 3DMatch/3DLoMatch

Inlier Ratio (IR) is the fraction of inlier matches among all putative point matches. A match is considered as an inlier if the distance between the two points is smaller than $\tau_1 = 10$cm under the ground-truth transformation $\bar{\mathbf{T}}_{\mathbf{P}\to\mathbf{Q}}$:

$$\mathrm{IR} = \frac{1}{|\mathcal{C}|} \sum_{(\mathbf{p}_{x_i}, \mathbf{q}_{y_i}) \in \mathcal{C}} [\![\|\bar{\mathbf{T}}_{\mathbf{P}\to\mathbf{Q}}(\mathbf{p}_{x_i}) - \mathbf{q}_{y_i}\|_2 < \tau_1]\!], \quad (19)$$

where $[\![\cdot]\!]$ is the Iversion bracket.

Feature Matching Recall (FMR) is the fraction of point cloud pairs whose IR is above $\tau_2 = 0.05$. FMR measures the potential success during the registration:

$$\mathrm{FMR} = \frac{1}{M} \sum_{i=1}^{M} [\![\mathrm{IR}_i > \tau_2]\!], \quad (20)$$

where M is the number of all point cloud pairs.

Registration Recall (RR) is the fraction of correctly registered point cloud pairs. Two point clouds are correctly registered if their transformation error is smaller than 0.2m. The transformation error is computed as the root mean square error of the ground-truth correspondences $\mathcal{C}^*$ after applying the estimated transformation $\mathbf{T}_{\mathbf{P}\to\mathbf{Q}}$:

$$\mathrm{RMSE} = \sqrt{\frac{1}{|\mathcal{C}^*|} \sum_{(\mathbf{p}^*_{x_i}, \mathbf{q}^*_{y_i}) \in \mathcal{C}^*} \|\mathbf{T}_{\mathbf{P}\to\mathbf{Q}}(\mathbf{p}^*_{x_i}) - \mathbf{q}^*_{y_i}\|_2^2}, \quad (21)$$

$$\mathrm{RR} = \frac{1}{M} \sum_{i=1}^{M} [\![\mathrm{RMSE}_i < 0.2\mathrm{m}]\!]. \quad (22)$$

Patch Inlier Ratio (PIR) is the fraction of superpoint (patch) matches with actual overlap under the ground-truth transformation. It reflects the quality of the putative superpoint (patch) correspondences:

$$\mathrm{PIR} = \frac{1}{|\hat{\mathcal{C}}|} \sum_{(\hat{\mathbf{p}}_{x_i}, \hat{\mathbf{q}}_{y_i}) \in \hat{\mathcal{C}}} [\![\exists \tilde{\mathbf{p}} \in \mathcal{G}^{\mathcal{P}}_{x_i}, \tilde{\mathbf{q}} \in \mathcal{G}^{\mathcal{Q}}_{y_i} \text{ s.t. } \|\tilde{\mathbf{p}} - \tilde{\mathbf{q}}\|_2 < \tau]\!], \quad (23)$$

where the matching radius is $\tau = 5$cm as stated in A.3.

B.2. KITTI

Relative Rotation Error (RRE) is the geodesic distance in degrees between estimated and ground-truth rotation matrices. It measures the differences between the predicted and the ground-truth rotation matrices.

$$\mathrm{RRE} = \arccos\left(\frac{\mathrm{trace}(\mathbf{R}^T \cdot \bar{\mathbf{R}} - 1)}{2}\right). \quad (24)$$

Relative Translation Error (RTE) is the Euclidean distance between estimated and ground-truth translation vectors. It measures the differences between the predicted and the ground-truth translation vectors.

$$\mathrm{RTE} = \|\mathbf{t} - \bar{\mathbf{t}}\|_2. \quad (25)$$

Registration Recall (RR) on KITTI is defined as the fraction of the point cloud pairs whose RRE and RTE are both below certain thresholds (*i.e.*, $\mathrm{RRE} < 5^\circ$ and $\mathrm{RTE} < 2$m).

$$\mathrm{RR} = \frac{1}{M} \sum_{i=1}^{M} [\![\mathrm{RRE}_i < 5^\circ \wedge \mathrm{RTE}_i < 2\mathrm{m}]\!]. \quad (26)$$

Following [2, 5, 7, 10, 18], we compute the mean RRE and the mean RTE only for the correctly registered point cloud pairs in KITTI.

C. Analysis of Cross-Entropy Loss

In this section, we first give an analysis that adopting the cross-entropy loss in multi-label classification problem could suppress the classes with high confidence scores. Given the input vector $\mathbf{y} \in \mathbb{R}^n$ and the label vector $\mathbf{g} \in \{0,1\}^n$, the confidence vector $\mathbf{z}$ is computed by adopting a softmax on $\mathbf{y}$:

$$z_i = \frac{\exp(y_i)}{\sum_{j=1}^{n} \exp(y_j)}. \tag{27}$$

The cross-entropy loss is computed as:

$$\mathcal{L} = -\sum_{i=1}^{n} g_i \log(z_i). \tag{28}$$

And the gradient vector $\mathbf{d}$ of $\mathbf{y}$ is computed as:

$$d_i = \frac{\partial \mathcal{L}}{\partial y_i} = (\sum_{j=1}^{n} g_j) z_i - g_i. \tag{29}$$

The zero point of d_i w.r.t. z_i is $c_i = g_i / \sum_{j=1}^{n} g_j$. If there are multiple positive classes, we have $0 < c_i < 1$ for each positive class as $\sum_{j=1}^{n} g_j > 1$. Hence y_i will be increased if $z_i < c_i$ ($d_i < 0$), and be reduced if $z_i > c_i$ ($d_i > 0$). This indicates that the cross-entropy loss will suppress the positive classes with higher confidence scores during training.

Now we go back to context of superpoint matching. To supervise the superpoint matching, CoFiNet [18] adopts a cross-entropy loss with an optimal transport layer, which formulates the superpoint matching as a multi-class classification problem for each superpoint. The ground-truth superpoint correspondences are determined by whether their neighboring point patches overlap. In practice, one patch usually overlaps with multiple patches in the other point cloud, so superpoint matching is a multi-class classification problem. According to the analysis above, the positive matches with higher confidence scores will be suppressed by the cross-entropy loss, which hinders the model from extracting reliable superpoint correspondences. CoFiNet [18] further designs a reweighting method which gives better zero points for the gradients, but the problem cannot be solved completely. On the contrary, our overlap-aware circle loss supervises the superpoint matching in a metric learning manner, which avoids this issue.

D. Additional Experiments

In this section, we conduct more experiments to evaluate our method. In Appx. D.1, we provide more detailed comparison on 3DMatch and 3DLoMatch. In Appx. D.2, we compare our method with recent deep robust estimators. In Appx. D.3, we conduct more ablation studies to better understand our design choices.

Overlap	Vanilla Self-attention			Geometric Self-attention		
	PIR(%)	IR(%)	RR(%)	PIR(%)	IR(%)	RR(%)
90%−100%	0.974	0.829	1.000	0.989	0.894	1.000
80%−90%	0.948	0.787	1.000	0.969	0.859	1.000
70%−80%	0.902	0.731	0.931	0.935	0.815	0.931
60%−70%	0.884	0.686	0.933	0.939	0.783	0.946
50%−60%	0.843	0.644	0.957	0.913	0.750	0.970
40%−50%	0.787	0.579	0.935	0.867	0.689	0.944
30%−40%	0.716	0.523	0.917	0.818	0.644	0.940
20%−30%	0.560	0.406	0.781	0.666	0.518	0.839
10%−20%	0.377	0.274	0.639	0.466	0.372	0.705

Table 2. Comparison of the models with the vanilla self-attention and the geometric self-attention under different overlap ratios. The results are reported on the union of 3DMatch and 3DLoMatch.

D.1. Detailed Results on 3DMatch

Registration results with different overlaps. We first compare the performance of the models with vanilla self-attention and our geometric self-attention under different overlap ratios on 3DMatch and 3DLoMatch. As shown in Tab. 2, our method consistently outperforms the vanilla self-attention counterpart on all the metrics in all levels of overlap ratio. The gains are greater when the overlap ratio is below 30%, demonstrating our method is more robust in low-overlap scenarios.

Scene-wise registration results. We present the scene-wise registration results on 3DMatch and 3DLoMatch in Tab. 4. Following [2, 5, 7], we report mean median RRE and RTE for the successfully registered point cloud pairs. For the registration recall, our method outperforms the baselines in most scenes on 3DMatch, especially the hard scenes such as `Home_2` and `Lab`. And it surpasses the baselines by a large margin in all scenes on 3DLoMatch. Moreover, our GeoTransformer also achieves consistently superior results on the rotation and translation errors.

D.2. Comparison with Deep Robust Estimators

We further compare with recent deep robust estimators: 3DRegNet [11], DGR [4], PointDSC [1], DHVR [9] and PCAM [3] on 3DMatch and KITTI. Following common practice, we report RTE, RRE and RR on both benchmarks. Here RR is defined as in Appx. B.2. The RTE threshold is 30cm on 3DMatch and 60cm on KITTI, while the RRE threshold is 15° on 3DMatch and 5° on KITTI. As shown in Tab. 3, our method outperforms all the baselines on both benchmarks. Although different correspondence extractors are used, these results can already demonstrate the superiority of GeoTransformer. It is noteworthy that our LGR is parameter-free and does not require training a specific network, which contributes to faster registration speed (0.013s

Model	RTE(cm)	RRE(°)	RR(%)
3DMatch			
FCGF+3DRegNet [11]	8.13	2.74	77.8
FCGF+DGR [4]	7.36	2.33	86.5
FCGF+PointDSC [1]	6.55	2.06	93.3
FCGF+DHVR [9]	6.61	2.08	91.4
PCAM [3]	∼7	2.16	92.4
GeoTransformer (*ours*, LGR)	**5.69**	**1.98**	**95.0**
3DLoMatch			
FCGF+PointDSC [1]	10.50	3.82	56.2
FCGF+DHVR [9]	11.76	3.88	55.6
GeoTransformer (*ours*, LGR)	**8.55**	**2.98**	**77.5**
KITTI			
FCGF+DGR [4]	21.7	0.34	96.9
FCGF+PointDSC [1]	20.9	0.33	98.2
FCGF+DHVR [9]	19.8	0.29	99.1
PCAM [3]	∼8	0.33	97.2
GeoTransformer (*ours*, LGR)	**6.5**	**0.24**	**99.5**

Table 3. Comparison with deep robust estimators on 3DMatch and KITTI. The RTE of PCAM is rounded to centimeter in the original paper [3].

vs. 0.08s [1] in our experiments).

D.3. Additional Ablation Studies

Transformation invariance. We first evaluate the transformation invariance of different positional embeddings in Tab. 5. For each model, we randomly apply arbitrary rotations to the *superpoints* when computing the superpoint embeddings. Among all the variants, enlarged rotations severely degrade the performance of (a) self-attention with absolute coordinate embedding, which indicates the lack of transformation variance in it. Surprisingly, the performance of (b) self-attention with relative coordinate embedding is quite stable. However, after masking the relative coorinate embedding out, we find that the results of this model still remain the same, which means the relative coordinate embedding contributes little to the final performance during testing. In constrast, our (c) geometric self-attention shows strong invariance to rigid transformation.

Geometric structure embedding. Next, we study the design of geometric structure embedding. We first vary the number of nearest neighbors for computing the triplet-wise angular embedding. As shown in Tab. 6(top), increasing the neighbors slightly improves the registration recall, but also requires more computation. To better balance accuracy and speed, we select $k=3$ in our experiments unless otherwise noted. We then replace max pooling with average pooling when aggregating the triplet-wise angular embedding in Eq. (5) of our main paper. From Tab. 6(middle), the results of two pooling methods are very close and max pooling performs slightly better than average pooling.

Dual-normalization. We then investigate the effectiveness of the dual-normalization operation in the superpoint matching module. As shown in Tab. 6(bottom), it slightly improves the accuracy of the superpoint correspondences in low-overlap scenarios. As there is less overlapping context when the overlapping area is small, it is much easier to extract outlier matches between the less geometrically discriminative patches. The dual-normalization operation can mitigate this issue and slightly improves the performance.

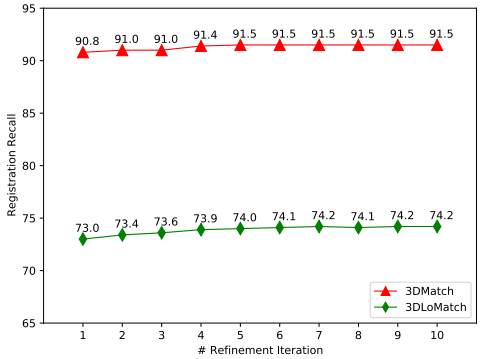


Figure 2. Ablation experiments on the pose refinement.

Pose refinement. At last, we evaluate the impact of the pose refinement in LGR. As shown in Fig. 2, the registration recall consistently improves with more iterations and gets saturated after about 5 iterations. To better balance accuracy and speed, we choose 5 iterations in the experiments.

E. Limitations

GeoTransformer relies on uniformly downsampled superpoints to hierarchically extract correspondences. However, there could be numerous superpoints if the input point clouds cover a large area, which will cause huge memory usage and computational cost. In this case, we might need to carefully select the downsampling rate to balance performance and efficiency.

Besides, it is inflexible to uniformly sample superpoints (patches). In practice, it is common that a single object is decomposed into multiple patches, which could be easily registered as a whole. So we think that it is a very promising topic to integrate point cloud registration with semantic scene understanding tasks (*e.g.*, semantic segmentation and object detection), which converts scene registration into object registration. We will leave this for future work.

F. Qualitative Results

We provide more qualitative results on 3DLoMatch in Fig. 3. The registration results from Predator [7] and CoFiNet [18] are also shown for comparison. Here Predator and CoFiNet use RANSAC-50k for registration, while LGR is used in GeoTransformer. Our method performs quite well in these low-overlap cases. It is noteworthy that our method can distinguish similar objects at different positions (see the comparison of CoFiNet and GeoTransformer in the 4^{th} and

Model	3DMatch									3DLoMatch								
	Kitchen	Home_1	Home_2	Hotel_1	Hotel_2	Hotel_3	Study	Lab	Mean	Kitchen	Home_1	Home_2	Hotel_1	Hotel_2	Hotel_3	Study	Lab	Mean
Registration Recall (%) ↑																		
3DSN [6]	90.6	90.6	65.4	89.6	82.1	80.8	68.4	60.0	78.4	51.4	25.9	44.1	41.1	30.7	36.6	14.0	20.3	33.0
FCGF [5]	98.0	94.3	68.6	96.7	91.0	84.6	76.1	71.1	85.1	60.8	42.2	53.6	53.1	38.0	26.8	16.1	30.4	40.1
D3Feat [2]	96.0	86.8	67.3	90.7	88.5	80.8	78.2	64.4	81.6	49.7	37.2	47.3	47.8	36.5	31.7	15.7	31.9	37.2
Predator [7]	97.6	97.2	74.8	**98.9**	**96.2**	**88.5**	85.9	73.3	89.0	71.5	58.2	60.8	77.5	64.2	61.0	45.8	39.1	59.8
CoFiNet [18]	96.4	**99.1**	73.6	95.6	91.0	84.6	**89.7**	84.4	89.3	76.7	66.7	64.0	81.3	65.0	63.4	53.4	69.6	67.5
P2PNet (*ours*)	**98.9**	97.2	**81.1**	**98.9**	89.7	**88.5**	88.9	**88.9**	**91.5**	**85.9**	**73.5**	**72.5**	**89.5**	**73.2**	**66.7**	**55.3**	**75.7**	**74.0**
Relative Rotation Error (°) ↓																		
3DSN [6]	1.926	1.843	2.324	2.041	1.952	2.908	2.296	2.301	2.199	3.020	3.898	3.427	3.196	3.217	3.328	4.325	3.814	3.528
FCGF [5]	**1.767**	1.849	2.210	1.867	1.667	2.417	2.024	1.792	1.949	2.904	3.229	3.277	2.768	2.801	2.822	3.372	4.006	3.147
D3Feat [2]	2.016	2.029	2.425	1.990	1.967	2.400	2.346	2.115	2.161	3.226	3.492	3.373	3.330	3.165	2.972	3.708	3.619	3.361
Predator [7]	1.861	1.806	2.473	2.045	1.600	2.458	2.067	1.926	2.029	3.079	2.637	3.220	2.694	2.907	3.390	3.046	3.412	3.048
CoFiNet [18]	1.910	1.835	2.316	1.767	1.753	1.639	2.527	2.345	2.011	3.213	3.119	3.711	2.842	2.897	3.194	4.126	3.138	3.280
P2PNet (*ours*)	1.797	**1.353**	**1.797**	**1.528**	**1.328**	**1.571**	**1.952**	**1.678**	**1.625**	**2.356**	**2.305**	**2.541**	**2.455**	**2.490**	**2.504**	**3.010**	**2.716**	**2.547**
Relative Translation Error (m) ↓																		
3DSN [6]	0.059	0.070	0.079	0.065	0.074	0.062	0.093	0.065	0.071	0.082	0.098	0.096	0.101	0.080	0.089	0.158	0.120	0.103
FCGF [5]	0.053	0.056	0.071	0.062	0.061	0.055	0.082	0.090	0.066	0.084	0.097	0.076	0.101	0.084	0.077	0.144	0.140	0.100
D3Feat [2]	0.055	0.065	0.080	0.064	0.078	0.049	0.083	0.064	0.067	0.088	0.101	0.086	0.099	0.092	0.075	0.146	0.135	0.103
Predator [7]	0.048	0.055	0.070	0.073	0.060	0.065	0.080	0.063	0.064	0.081	0.080	0.084	0.099	0.096	0.077	**0.101**	0.130	0.093
CoFiNet [18]	0.047	0.059	0.063	0.063	0.058	**0.044**	0.087	0.075	0.062	0.080	0.078	0.078	0.099	0.086	0.077	0.131	0.123	0.094
P2PNet (*ours*)	**0.042**	**0.046**	**0.059**	**0.055**	**0.046**	0.050	**0.073**	**0.053**	**0.053**	**0.062**	**0.070**	**0.071**	**0.080**	**0.075**	**0.049**	0.107	**0.083**	**0.074**

Table 4. Scene-wise registration results on 3DMatch and 3DLoMatch.

Model	3DMatch		3DLoMatch	
	original	rotated	original	rotated
(a) self-attention w/ ACE	89.3	87.2 -2.1	69.3	67.4 -1.9
(b) self-attention w/ RCE	88.5	88.5 same	68.7	68.7 same
(c) geometric self-attention	91.5	91.4 -0.1	74.0	73.8 -0.2

Table 5. Ablation experiments with rotated superpoints.

Model	3DMatch				3DLoMatch			
	PIR	FMR	IR	RR	PIR	FMR	IR	RR
(a) distance only	84.9	98.0	69.1	90.7	50.6	85.8	40.3	72.1
(b) $k = 1$ angles	86.5	97.9	70.6	91.0	54.6	87.1	42.7	73.1
(c) $k = 2$ angles	86.1	97.9	70.4	91.3	55.0	88.2	43.5	73.5
(d) $k = 3$ angles	86.1	97.7	70.3	91.5	54.9	88.1	43.3	74.0
(e) $k = 4$ angles	86.6	98.0	70.7	91.7	55.1	88.4	43.5	74.2
(f) max pooling	86.1	97.7	70.3	91.5	54.9	88.1	43.3	74.0
(g) average pooling	86.3	98.0	70.2	91.3	54.6	87.3	42.8	74.0
(h) w/ dual-normalization	86.1	97.7	70.3	91.5	54.9	88.1	43.3	74.0
(i) w/o dual-normalization	86.2	97.7	70.3	91.0	53.5	87.9	42.8	73.8

Table 6. Additional ablation experiments.

6^{th} rows) thanks to the transformation invariance obtained from the geometric self-attention. Fig. 4 visualizes the registration results of GeoTransformer in the bird's-eye view on KITTI. It can be observed that our method attains very accurate registration even without using RANSAC.

We also provide some failure cases of our method on 3DLoMatch in Fig. 5. Generally, if the overlaping region between two point clouds is small and geometrically indiscriminative (*e.g.*, wall, ceiling and floor) or the non-overlapping region is relatively complicated, the registration could fail. A commonality of these cases is that they cannot provide adequate geometric cues to detect overlapping area and extract reliable correspondences. A possible solution could combine the information from multiple point clouds. We will leave this for future work.

References

[1] Xuyang Bai, Zixin Luo, Lei Zhou, Hongkai Chen, Lei Li, Zeyu Hu, Hongbo Fu, and Chiew-Lan Tai. Pointdsc: Robust point cloud registration using deep spatial consistency. In *CVPR*, pages 15859–15869, 2021. 2, 4, 5

[2] Xuyang Bai, Zixin Luo, Lei Zhou, Hongbo Fu, Long Quan, and Chiew-Lan Tai. D3feat: Joint learning of dense detection and description of 3d local features. In *CVPR*, pages 6359–6367, 2020. 3, 4, 6

[3] Anh-Quan Cao, Gilles Puy, Alexandre Boulch, and Renaud Marlet. Pcam: Product of cross-attention matrices for rigid registration of point clouds. In *ICCV*, pages 13229–13238, 2021. 4, 5

[4] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration. In *CVPR*, pages 2514–2523, 2020. 4, 5

[5] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *CVPR*, pages 8958–8966, 2019. 3, 4, 6

[6] Zan Gojcic, Caifa Zhou, Jan D Wegner, and Andreas Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *CVPR*, pages 5545–5554, 2019. 6

[7] Shengyu Huang, Zan Gojcic, Mikhail Usvyatsov, Andreas Wieser, and Konrad Schindler. Predator: Registration of 3d point clouds with low overlap. In *CVPR*, pages 4267–4276, 2021. 2, 3, 4, 5, 6

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2015. 2

[9] Junha Lee, Seungwook Kim, Minsu Cho, and Jaesik Park. Deep hough voting for robust global registration. In *ICCV*, pages 15994–16003, 2021. 4, 5

[10] Fan Lu, Guang Chen, Yinlong Liu, Lijun Zhang, Sanqing Qu, Shu Liu, and Rongqi Gu. Hregnet: A hierarchical network for large-scale outdoor lidar point cloud registration. In *ICCV*, pages 16014–16023, 2021. 4

[11] G Dias Pais, Srikumar Ramalingam, Venu Madhav Govindu, Jacinto C Nascimento, Rama Chellappa, and Pedro Miraldo. 3dregnet: A deep neural network for 3d point registration. In *CVPR*, pages 7193–7203, 2020. 4, 5

[12] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *NeurIPS*, 32:8026–8037, 2019. 2

[13] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, pages 4938–4947, 2020. 1

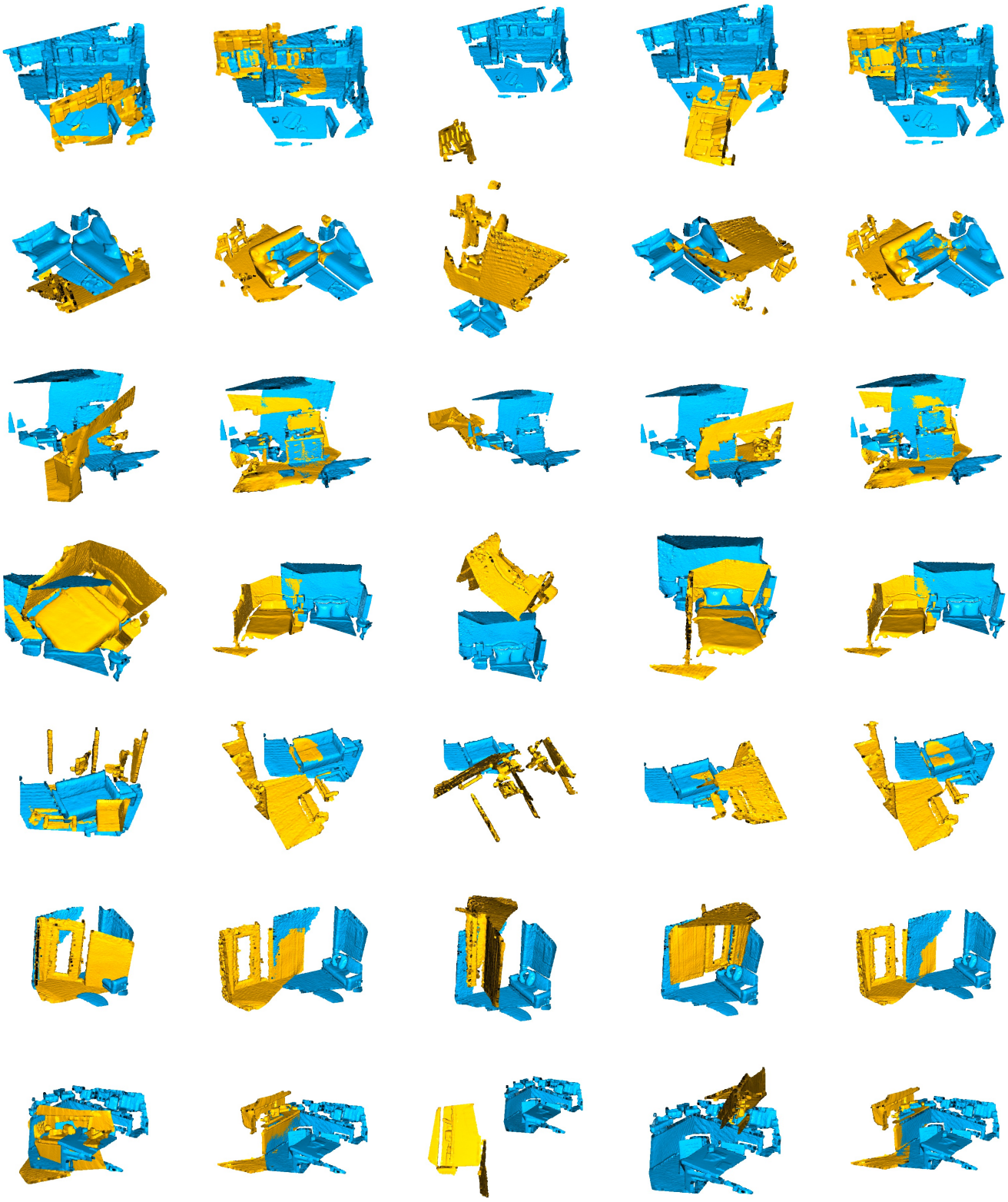
[14] Richard Sinkhorn and Paul Knopp. Concerning nonnegative matrices and doubly stochastic matrices. *Pacific Journal of Mathematics*, 21(2):343–348, 1967. 1

[15] Hugues Thomas, Charles R Qi, Jean-Emmanuel Deschaud, Beatriz Marcotegui, François Goulette, and Leonidas J Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *ICCV*, pages 6411–6420, 2019. 2

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. 1

[17] Yuxin Wu and Kaiming He. Group normalization. In *ECCV*, pages 3–19, 2018. 2

[18] Hao Yu, Fu Li, Mahdi Saleh, Benjamin Busam, and Slobodan Ilic. Cofinet: Reliable coarse-to-fine correspondences for robust point cloud registration. *arXiv preprint arXiv:2110.14076*, 2021. 3, 4, 5, 6



(a) Input (b) Ground truth (c) Predator (d) CoFiNet (e) GeoTransformer

Figure 3. Registration results on 3DMatch and 3DLoMatch.

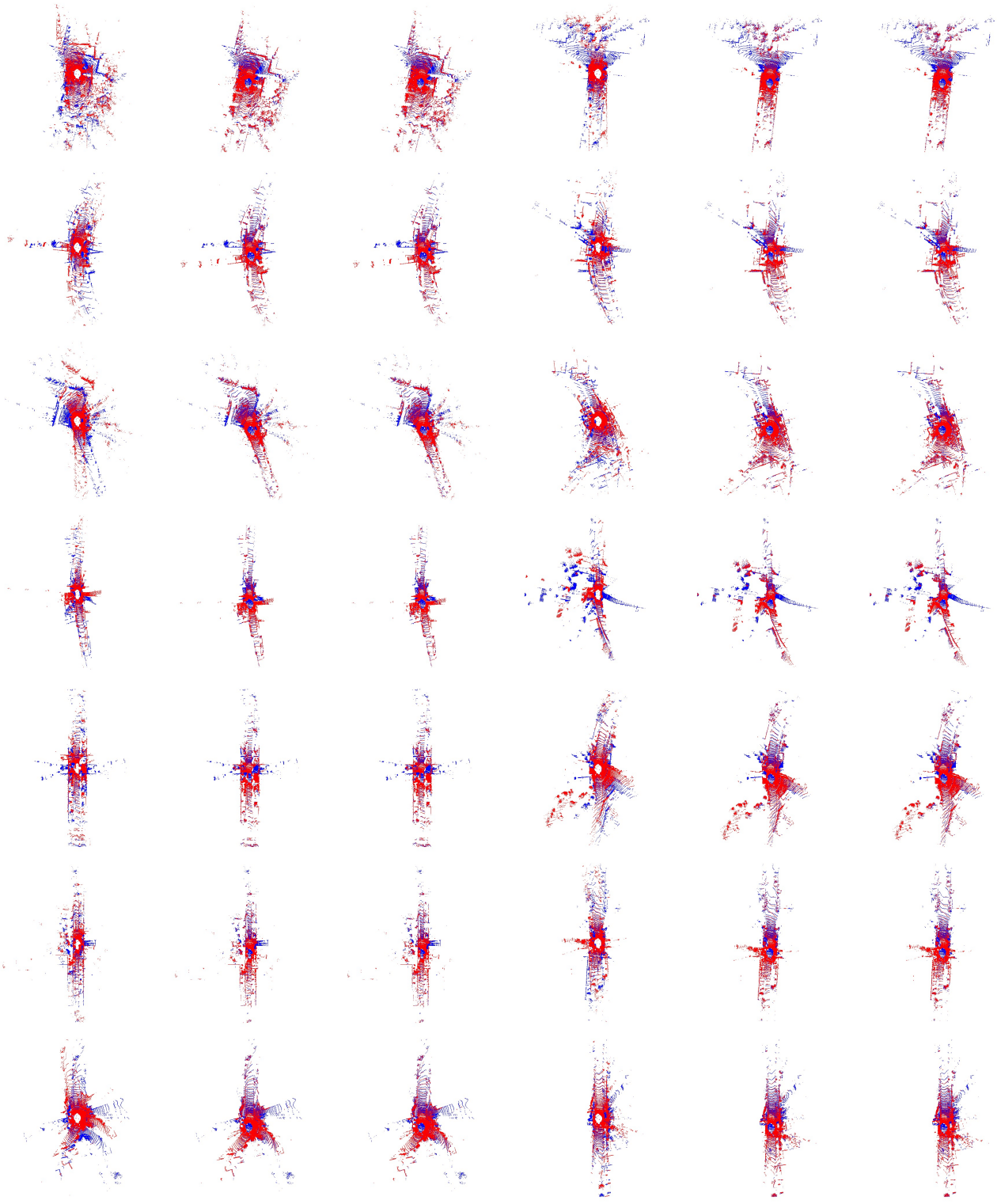
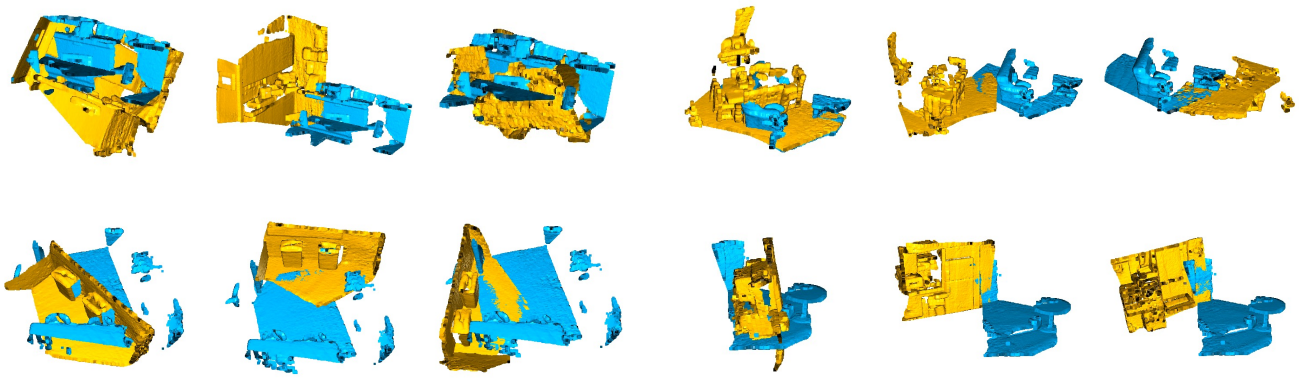


Figure 4. Registration results on KITTI odometry.



(a) Input (b) Ground truth (c) Ours (a) Input (b) Ground truth (c) Ours

Figure 5. Failed cases on 3DLoMatch.