REGTR: End-to-end Point Cloud Correspondences with Transformers - Supplementary

In this supplementary, we first provide additional details on the datasets and their preprocessing (Sec. A). We then describe the procedure for recovering the rigid transformation from the correspondences (Sec. B), our sinusoidal position encodings (Sec. C) and additional information on the network architecture (Sec. D). Finally, we show detailed results for ScanNet (Sec. E) and additional qualitative results (Sec. F).

A. Dataset Details

3DMatch. The 3DMatch [23] dataset comprises RGB-D frames obtained from several sources (Tab. 1). The data is captured from diverse scenes (*e.g.* bedrooms, kitchens, offices) and different sensors (*e.g.* Microsoft Kinect, Intel Realsense), and each point cloud is generated by fusing 50 consecutive depth frames using TSDF volumetric fusion [5]. We use the voxel-grid downsampled data from Predator [9], and the same point cloud pairs for training and evaluation. The dataset contains 46 train, 8 validation, and 8 test scenes. The training and validation scenes contains a total of 20,586[1] and 1,331 point cloud pairs respectively, and the test scenes contain 1,279 (3DMatch) and 1,726 (3DLoMatch) pairs. We apply training data augmentation by applying a small rigid perturbation with magnitudes sampled from a Gaussian distribution with $\sigma_r = 0.1\pi$ and $\sigma_t = 0.1$ for the rotation and translation, respectively. We then apply a Gaussian noise ($\sigma = 0.05$) on the individual point locations, and shuffling of point order.

ModelNet40. The ModelNet40 [18] dataset provides 3D CAD models from 40 object categories for academic use. We follow previous works [17,22] in using the preprocessed data from [12]. These data are generated by sampling 2,048 points from the mesh faces and then scaling them to fit into a unit sphere. The partial scans are generated from the procedure in [22]: A half-space with random direction is sampled, and shifted such that a proportion p of points lie within the half space. Subsequently, random rotation of up to $45°$, translation up to 0.5 units, Gaussian noise ($\sigma = 0.05$) on the individual point locations, and shuffling of point order are applied to the point clouds. The point clouds are finally resampled to 717 points. Following [9], p is set to 0.7 and 0.5 for ModelNet and ModelLoNet benchmarks, respectively. We use the first 20 categories for training and validation, and the other 20 categories for testing.

[1] one point cloud (7-scenes-fire/19) has a wrong groundtruth pose and we exclude training pairs containing this point cloud.

Datasets	License
SUN3D [8, 19]	CC BY-NC-SA 4.0
7-Scenes [13]	Non-commercial use only
RGB-D Scenes v2 [11]	(License not stated)
BundleFusion [6]	CC BY-NC-SA 4.0
Analysis-by-Synthesis [15]	CC BY-NC-SA 4.0

Table 1. Raw data used in the 3DMatch [23] dataset and their licenses.

B. Estimation of Rigid Transformation

In this section, we describe the closed form solution for the rigid transformation $\{\mathbf{R}, \mathbf{t}\}$, given correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{y}_i\}$ with their weights $\{o_i\}$, as used in Sec. 4.4 in the main paper:

$$\hat{\mathbf{R}}, \hat{\mathbf{t}} = \arg\min_{\mathbf{R},\mathbf{t}} \sum_{i}^{N} o_i \|\mathbf{R}\mathbf{x}_i + \mathbf{t} - \mathbf{y}_i\|^2. \tag{1}$$

Step 1. Compute the weighted centroids of the 2 point sets:

$$\bar{\mathbf{x}} = \frac{\sum_{i=1}^{N} o_i \mathbf{x}_i}{\sum_{i=1}^{N} o_i}, \qquad \bar{\mathbf{y}} = \frac{\sum_{i=1}^{N} o_i \mathbf{y}_i}{\sum_{i=1}^{N} o_i}. \tag{2}$$

Step 2. Center the point clouds by subtracting away the centroid:

$$\tilde{\mathbf{x}}_i = \mathbf{x}_i - \bar{\mathbf{x}}, \quad \tilde{\mathbf{y}}_i = \mathbf{y}_i - \bar{\mathbf{y}}, \quad \forall i = 1, \dots, N. \tag{3}$$

Step 3. Recover the rotation $\mathbf{R}$. For this, we can use the Kabsch algorithm [10]. First construct the following 3×3 weighted covariance matrix:

$$\mathbf{H} = \sum_{i=1}^{N} o_i \tilde{\mathbf{x}}_i \tilde{\mathbf{y}}_i^\top. \tag{4}$$

Considering the singular value decomposition $\mathbf{H} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$, the desired rotation $\hat{\mathbf{R}}$ is given by:

$$\hat{\mathbf{R}} = \mathbf{V} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det\left(\mathbf{V}\mathbf{U}^\top\right) \end{bmatrix} \mathbf{U}^\top, \tag{5}$$

where $\det(\cdot)$ denotes the matrix determinant.

Step 4. Lastly, the translation can be computed as:

$$\hat{\mathbf{t}} = \bar{\mathbf{y}} - \hat{\mathbf{R}}\bar{\mathbf{x}}. \tag{6}$$

C. Position Encodings

We encode the point coordinates by generalizing the sinusoidal positional encodings in [16] to 3D continuous coordinates. The position encodings have the same dimension $d = 256$ as the feature embeddings used in the attention layers.

For a point $\mathbf{x} = (x, y, z)$, we separately transform each coordinate to their embeddings $\mathbf{p}_x^\mathbf{x}, \mathbf{p}_y^\mathbf{x}, \mathbf{p}_z^\mathbf{x} \in \mathbb{R}^{2\lfloor d/6 \rfloor}$. The x-coordinate is transformed as:

$$\mathbf{p}_x^\mathbf{x}[2i] = \sin\left(\frac{x}{10000^{2i/\lfloor d/3 \rfloor}}\right) \tag{7a}$$

$$\mathbf{p}_x^\mathbf{x}[2i+1] = \cos\left(\frac{x}{10000^{2i/\lfloor d/3 \rfloor}}\right). \tag{7b}$$

The y and z coordinates are transformed in a similar manner. We then concatenate the embeddings for all three dimensions. Since the embedding dimension $d = 256$ is not divisible by 6, we pad the remaining 4 elements with zeros to obtain the final embedding.

Importance of position encodings. Table 2 compares different choices of position encodings. The position encodings can only be removed when decoding the correspondences as a weighted sum (Eq. 4 in main paper). Without position encodings, the network suffered a significant drop in performance both in terms of registration recall (RR) and accuracy (RTE and RRE), further supporting our hypothesis that our attention mechanism utilizes rigidity constraints to correct bad matches. We also compare with learned embeddings (using a 5-layer MLP with 32-64-128-256-256 channels), which has a slightly lower performance in most metrics. We therefore chose to use sinusoidal encodings, which also reduces the number of learnable weights.

D. Network Architecture

KPConv backbone. We show the detailed network architecture of our KPConv [14] backbone in Fig. 2. We use the same KPConv backbone as Predator, but we modify it to apply instance normalization on each point cloud individually instead of over all point clouds to allow for correct behavior over batch sizes larger than one. We do not make any other changes to the backbone for the 3DMatch dataset. However, to maintain a reasonable resolution for the downsampled keypoints for ModelNet, we use a shallower backbone consisting of only a single downsampling, and the voxel size used in the first level is set to 0.03 instead of 0.06.

		3DMatch			*3DLoMatch*		
Dec.	Pos.	RR(%)	RRE(°)	RTE(m)	RR(%)	RRE(°)	RTE(m)
Wt.	None	87.9	1.958	0.062	55.0	3.389	0.093
Wt.	Learned	89.0	1.519	0.047	61.3	2.812	0.083
Wt.	Sine	90.9	**1.468**	**0.046**	63.1	**2.540**	**0.076**
Reg.	Learned	91.6	1.576	0.047	**64.8**	2.999	0.080
Reg.	Sine	**92.0**	1.567	0.049	**64.8**	2.827	0.077

Table 2. Effects of different position encodings. "Dec." denotes correspondence decoding scheme, which can be either weighted coordinates using Eq. 4 in main paper (Wt.) or regression using Eq. 3 in main paper (Reg.). "Pos." denotes position encoding type.

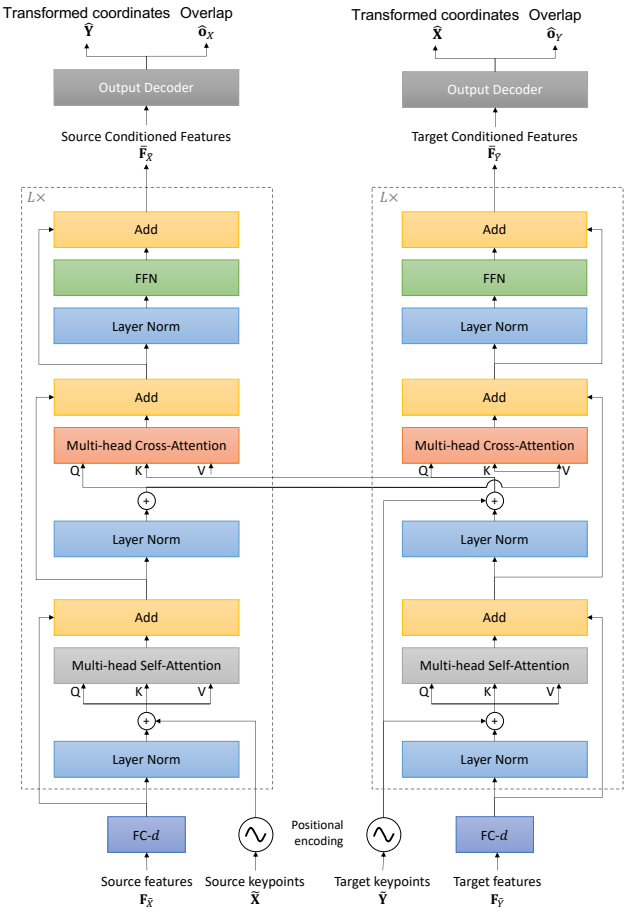

Figure 1. REGTR's transformer cross-encoder layers.

Transformer. Figure 1 provides the detailed description of our transformer cross-encoder. Geometric features from the KPConv backbone are first projected to $d = 256$ dimensions, and then passed through $L = 6$ transformer cross-

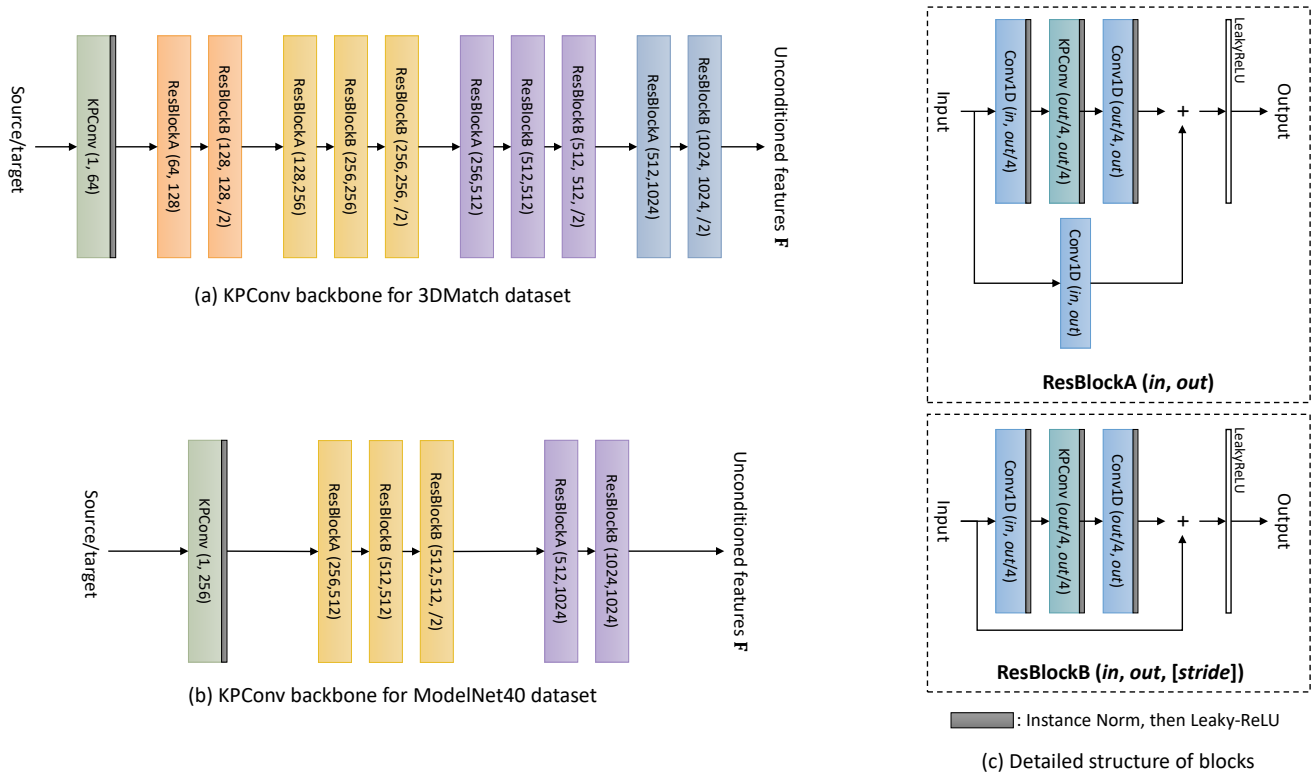

Figure 2. KPConv backbone used for (a) 3DMatch and (b) ModelNet. (c) shows the detailed structure of the residual blocks.

encoder layers to obtain the conditioned features $\bar{\mathbf{F}}_{\tilde{X}}, \bar{\mathbf{F}}_{\tilde{Y}}$, which can be used to predict the output correspondences and overlap scores via our output decoder. We use the pre-LN [20] configuration for the self-attention, cross-attention, and position-wise feed-forward networks (FFN). Positional encodings (Sec. C) are added to the queries, keys and values before every self- and cross-attention layer.

E. Detailed Registration Results for 3DMatch

We report the breakdown of the Registration Recall, Relative Rotation Error, and Relative Translation Error for each individual scene in Tab. 3. REGTR obtains the highest registration recall for three (3DMatch) and four (3DLoMatch) of the scenes, and the lowest rotation/translation errors for majority of the scenes in both settings, despite using downsampled features.

F. Additional Qualitative Results

We show additional qualitative results for both 3DMatch and ModelNet datasets in Fig. 3. The last two rows show example failure cases. During failures, usually both overlap and correspondences are predicted wrongly.

References

[1] 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. 5

[2] 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. 5

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

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

[5] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996. 1

[6] Angela Dai, Matthias Nießner, Michael Zollöfer, Shahram Izadi, and Christian Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface re-integration. *ACM TOG*, 2017. 1

[7] 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. 5

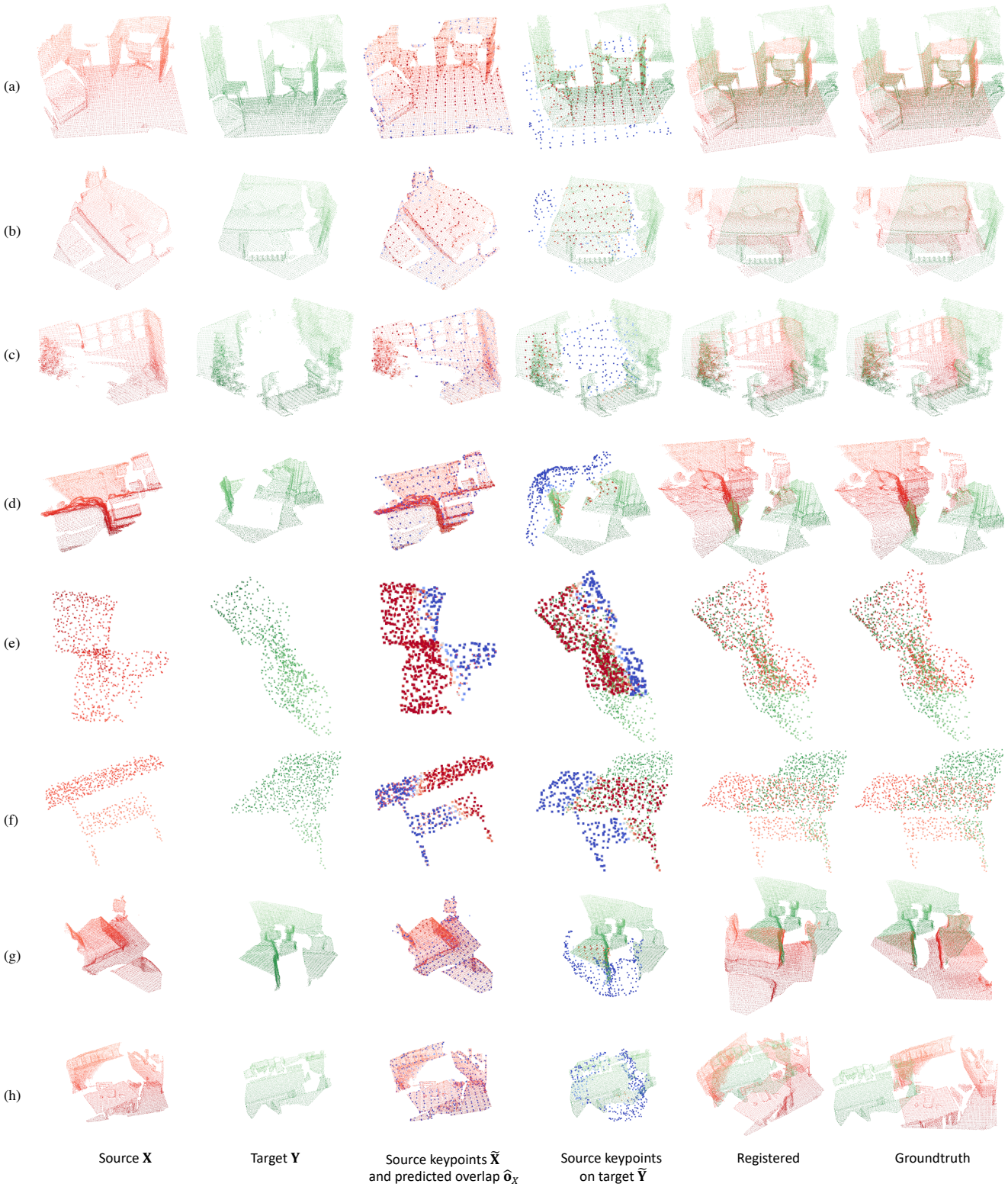

Figure 3. Additional qualitative results on (a,b) 3DMatch, (c,d) 3DLoMatch, (e) ModelNet, and (f) ModelLoNet benchmarks. Keypoints are colored by their predicted overlap scores where red indicates high overlap. The last two rows (g,h) show example failure cases on the 3DMatch dataset. Best viewed in color.

	3DMatch (≥ 30% overlap)									3DLoMatch (10-30% overlap)								
	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Avg.	Kitchen	Home 1	Home 2	Hotel 1	Hotel 2	Hotel 3	Study	MIT Lab	Avg.
	# pairs																	
	449	106	159	182	78	26	234	45	160	524	283	222	210	138	42	237	70	191
	Registration Recall (%) ↑																	
3DSN [7]	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 [4]	**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 [1]	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-5k [9]	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
Predator-1k [9]	97.1	**98.1**	74.8	97.8	**96.2**	88.5	87.2	84.4	90.5	70.6	**62.8**	63.1	**80.9**	**64.2**	61.0	50.0	47.8	62.5
Predator-NR [9]	60.8	74.5	52.2	80.8	65.4	57.7	54.3	55.6	62.7	25.6	19.9	37.8	32.5	22.6	24.4	11.9	17.4	24.0
OMNet [21]	39.0	39.6	27.7	30.8	38.5	46.2	21.4	44.4	35.9	9.0	6.7	9.0	3.3	8.8	14.6	2.5	13.0	8.4
DGR [3]	97.8	94.3	62.3	95.1	88.5	84.6	84.2	75.6	85.3	60.2	45.0	52.7	54.5	48.9	41.5	38.6	47.8	48.7
PCAM [2]	96.9	93.4	**78.6**	96.7	83.3	84.6	81.6	68.9	85.5	71.1	56.7	60.8	70.8	59.9	41.5	36.4	42.0	54.9
Ours	97.8	90.6	75.5	97.8	94.9	**100**	**88.5**	**91.1**	**92.0**	66.2	58.5	**64.9**	72.7	61.3	**70.7**	**53.0**	**71.0**	**64.8**
	Relative Rotation Error (°) ↓																	
3DSN [7]	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 [4]	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 [1]	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-5k [9]	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
Predator-1k [9]	1.902	1.739	2.306	1.897	1.817	2.289	2.278	2.271	2.062	3.049	2.679	3.247	2.857	2.782	3.340	3.778	3.538	3.159
Predator-NR [9]	3.052	2.223	2.805	2.996	1.900	2.117	3.711	1.854	2.582	6.445	4.508	5.272	5.224	4.889	6.975	8.457	5.320	5.886
OMNet [21]	4.142	3.166	3.664	5.450	3.952	5.518	4.142	3.296	4.166	6.924	8.747	8.199	8.590	10.901	7.613	4.297	3.123	7.299
DGR [3]	2.181	1.809	2.474	1.842	1.966	2.313	2.653	1.588	2.103	4.049	3.967	4.433	3.666	4.119	3.742	4.188	3.469	3.954
PCAM [2]	1.965	1.644	2.145	1.874	1.434	**1.631**	2.250	1.521	1.808	3.501	3.518	3.571	3.649	3.197	3.278	4.148	3.368	3.529
Ours	**1.729**	**1.347**	**1.797**	**1.639**	**1.289**	1.810	**1.570**	**1.357**	**1.567**	3.366	**2.446**	3.244	2.732	**2.439**	2.919	**3.044**	**2.428**	**2.827**
	Relative Translation Error (m) ↓																	
3DSN [7]	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 [4]	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 [1]	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-5k [9]	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
Predator-1k [9]	0.052	0.062	0.071	0.062	0.058	0.055	0.088	0.094	0.068	**0.077**	0.084	**0.074**	**0.090**	0.093	0.096	0.126	0.128	0.096
Predator-NR [9]	0.089	0.065	0.072	0.084	0.061	**0.028**	0.124	0.074	0.075	0.137	0.106	0.118	0.158	0.112	0.152	0.206	0.193	0.148
OMNet [21]	0.103	0.098	0.097	0.144	0.099	0.079	0.124	0.095	0.105	0.137	0.192	0.146	0.228	0.198	0.098	0.119	0.094	0.151
DGR [3]	0.057	0.064	0.070	0.068	0.052	0.062	0.094	0.072	0.067	0.104	0.111	0.119	0.111	0.092	0.085	0.147	0.134	0.113
PCAM [2]	0.050	0.051	0.060	0.066	0.051	0.058	0.081	**0.052**	0.059	0.088	0.108	0.090	0.112	0.098	0.068	0.117	0.110	0.099
Ours	**0.040**	**0.041**	**0.057**	**0.057**	**0.042**	0.039	**0.054**	0.058	**0.049**	0.079	**0.064**	0.078	0.094	**0.074**	**0.060**	**0.093**	**0.077**	**0.077**

Table 3. Detailed results on the 3DMatch and 3DLoMatch datasets. Results of 3DSN, FCGF, D3Feat and Predator-5k are taken from [9].

[8] Maciej Halber and Thomas Funkhouser. Fine-to-coarse global registration of rgb-d scans. In *CVPR*, pages 1755–1764, 2017. 1

[9] 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. 1, 5

[10] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976. 1

[11] Kevin Lai, Liefeng Bo, and Dieter Fox. Unsupervised feature learning for 3d scene labeling. In *ICRA*, pages 3050–3057, 2014. 1

[12] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017. 1

[13] 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 *CVPR*, pages 2930–2937, 2013. 1

[14] 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

[15] Julien Valentin, Angela Dai, Matthias Niessner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. In *3DV*, pages 323–332, 2016. 1

[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. 2

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

[18] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3D ShapeNets: A deep representation for volumetric shapes. In *CVPR*, pages 1912–1920, 2015. 1

[19] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. SUN3D: A database of big spaces reconstructed using sfm and object labels. In *ICCV*, pages 1625–1632, 2013. 1

[20] Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. On layer normalization in the transformer architecture. In *ICML*, pages 10524–10533, 2020. 3

[21] Hao Xu, Shuaicheng Liu, Guangfu Wang, Guanghui Liu, and Bing Zeng. OMNet: Learning overlapping mask for partial-to-partial point cloud registration. In *ICCV*, pages 3132–3141, 2021. 5

[22] Zi Jian Yew and Gim Hee Lee. RPM-Net: Robust point matching using learned features. In *CVPR*, pages 11824–11833, 2020. 1

[23] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3DMatch: Learning local geometric descriptors from RGB-D reconstructions. In *CVPR*, pages 1802–1811, 2017. 1