Rethinking Rotation Invariance with Point Cloud Registration **Supplementary Material**

Jianhui Yu, Chaoyi Zhang, Weidong Cai

School of Computer Science, University of Sydney, Australia
{jianhui.yu, chaoyi.zhang, tom.cai}@sydney.edu.au

1 Mathematical Proofs

Proof of Rotation Equivariance of Orthonormal Basis $\mathbf{M}_i^\ell$ of LRFs. We will prove the rotation equivariance of the orthonormal basis $\mathbf{M}_i^\ell$ designed for LRFs as mentioned in Section 3.1 of the main work. Given a random rotation matrix $\mathbf{R} \in \mathbb{R}^{3\times3}$, it is easy to derive that $\vec{x_i}^\ell$ is equivariant to rotations given the rotated version $\vec{x}_{i,rot}^\ell$:

$$\begin{aligned}\vec{x}_{i,rot}^\ell &= \frac{\mathbf{R}p_i - \mathbf{R}p_m}{\|\mathbf{R}p_i - \mathbf{R}p_m\|_2} = \frac{\mathbf{R}(p_i - p_m)}{\sqrt{\left(\mathbf{R}(p_i-p_m)\right)^\top \mathbf{R}(p_i - p_m)}} \\ &= \frac{\mathbf{R}\overrightarrow{p_m p_i}}{\sqrt{\overrightarrow{p_m p_i}^\top \mathbf{R}^\top \mathbf{R} \overrightarrow{p_m p_i}}} = \mathbf{R}\frac{\overrightarrow{p_m p_i}}{\|\overrightarrow{p_m p_i}\|_2} = \mathbf{R}\vec{x_i}^\ell,\end{aligned} \tag{1}$$

where the subscript rot represents the axis after rotations. Moreover, $\mathbf{\Sigma}_i^\ell$ from Eq. (1) of the main work after rotations can be represented as follows:

$$\begin{aligned}\mathbf{\Sigma}_{i,rot}^\ell &= \sum_{j=1}^{|\mathcal{N}(p_i)|} \alpha_j \mathbf{R}\overrightarrow{p_i p_j}\overrightarrow{p_i p_j}^\top \mathbf{R}^\top \\ &= \mathbf{R}\left(\sum_{j=1}^{|\mathcal{N}(p_i)|} \alpha_j \overrightarrow{p_i p_j}\overrightarrow{p_i p_j}^\top\right)\mathbf{R}^\top = \mathbf{R}\mathbf{\Sigma}_i^\ell \mathbf{R}^\top.\end{aligned} \tag{2}$$

As mentioned in the main work, eigenvalue decomposition can be directly applied to $\mathbf{\Sigma}_i^\ell$, resulting in the following expressions:

$$\mathbf{R}\mathbf{\Sigma}_i^\ell\mathbf{R}^\top = \mathbf{R}\mathbf{U}_i^\ell \mathbf{\Lambda}_i^\ell {\mathbf{U}_i^\ell}^\top \mathbf{R}^\top = \left(\mathbf{R}\mathbf{U}_i^\ell\right)\mathbf{\Lambda}_i^\ell\left(\mathbf{R}\mathbf{U}_i^\ell\right)^\top. \tag{3}$$

Since $\vec{z_i}^\ell$ is defined to have the same direction as the eigenvector with the smallest eigenvalue, hence after rotation, $\vec{z}_{i,rot}^\ell = \mathbf{R}\vec{z_i}^\ell$. Thus, the *rotated* y-axis is:

$$\begin{aligned}\vec{y}_{i,rot}^\ell &= \vec{z}_{i,rot}^\ell \times \vec{x}_{i,rot}^\ell = \mathbf{R}\vec{z_i}^\ell \times \mathbf{R}\vec{x_i}^\ell \\ &= \det(\mathbf{R})\left(\mathbf{R}^{-1}\right)^\top\left(\vec{z_i}^\ell \times \vec{x_i}^\ell\right) \\ &= \mathbf{R}\left(\vec{z_i}^\ell \times \vec{x_i}^\ell\right) = \mathbf{R}\vec{y_i}^\ell.\end{aligned} \tag{4}$$

Since all basis vectors are rotation-equivariant, hence the local orthonormal basis $\mathbf{M}_i^\ell = [\vec{x_i}^\ell, \vec{y_i}^\ell, \vec{z_i}^\ell]$ is rotation-equivariant.

Proof of Rotation Invariance of Local Shape Descriptors p_{ij}^ℓ. Here, we show the rotation invariance of local shape descriptors p_{ij}^ℓ introduced in Section 3.1 of the main work. Based on the proof shown above, it is easy to show the rotation invariance of p_{ij}^ℓ after rotation $\mathbf{R}$:

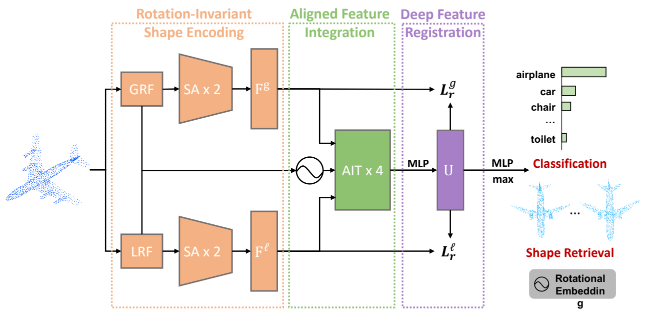
$$\begin{aligned}p_{ij,rot}^\ell &= \overrightarrow{p_i p_j}_{,rot}^\top \mathbf{M}_{i,rot}^\ell = \left(\mathbf{R}\overrightarrow{p_i p_j}\right)^\top\left(\mathbf{R}\mathbf{M}_i^\ell\right) \\ &= \overrightarrow{p_i p_j}^\top \mathbf{M}_i^\ell = p_{ij}^\ell.\end{aligned} \tag{5}$$

2 Model Details

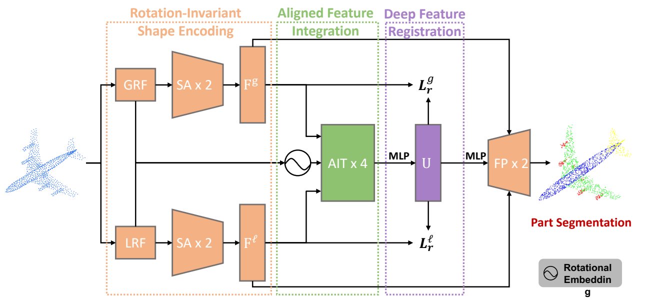
Architectures. The model overview for the 3D classification, segmentation, and retrieval tasks is shown in Figure 1, where details of each module design are explained in Section 3 of the main work. The detailed designs of intra-frame aligned self-attention and inter-frame aligned cross-attention are illustrated in Figure 2. Please refer to Section 3.2 of the main work for more details.

Cease of Encoding of $\mathbf{F}^g$ in AIT. We propose in our main work that we do not apply additional learnable modules on $\mathbf{F}^g$ in AIT to alleviate the model overfitting. This implementation creates a shot path from low-level feature space to high-level feature space, which shares a similar spirit with the skip connection in ResNet (He et al. 2016): allowing lower-level information to flow effectively across layers to help the learning of higher-level features and hence reducing the model overfitting. Meanwhile, we encode local features along with $\mathbf{F}^g$ in AIT blocks as abstracting information from local structures can increase the model's ability on fine-grained pattern recognition and generalizability to complex scenes (Qi et al. 2017b). Hence, we find that not having additional learnable modules to further process $\mathbf{F}^g$ in the AIT module is beneficial to the our model performance.

Registration Loss. In this this part, we explain the benefits of having registration loss to preserve the rotation invariance and give more details about Eq. (10) of the main work. Suppose $\mathbf{U}^\ell$ and $\mathbf{U}^g$ are local and global part information of the final integrated feature $\mathbf{U}$. By maximizing the mutual information between $(\mathbf{U}, \mathbf{F}^\ell)$ and $(\mathbf{U}, \mathbf{F}^g)$, $(\mathbf{U}^\ell, \mathbf{F}^\ell)$ and $(\mathbf{U}^g, \mathbf{F}^g)$ are implicitly maximized. In this case, the shared geometric information between the local $\mathbf{F}^\ell$/global $\mathbf{F}^g$ and the integrated domain $\mathbf{U}$ are refined, increasing the representation power of $\mathbf{U}$. Besides, the maximized similarities of



(a) classification/retrieval



(b) segmentation

Figure 1: Model overview for (a) classification/retrieval and (b) segmentation. GRF: global reference frame; LRF: local reference frame; SA: set abstraction; AIT: Aligned Integration Transformer; and FP: forward passing.

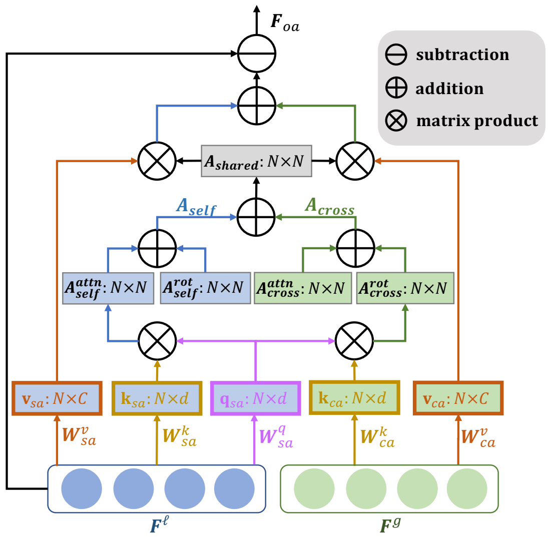

F_{oa}
subtraction
addition
matrix product
$A_{shared}: N\times N$
A_{self}
A_{cross}
$A^{attn}_{self}: N\times N$
$A^{rot}_{self}: N\times N$
$A^{attn}_{cross}: N\times N$
$A^{rot}_{cross}: N\times N$
$\mathbf{v}_{sa}: N\times C$
$\mathbf{k}_{sa}: N\times d$
$\mathbf{q}_{sa}: N\times d$
$\mathbf{k}_{ca}: N\times d$
$\mathbf{v}_{ca}: N\times C$
W^v_{sa}
W^k_{sa}
W^q_{sa}
W^k_{ca}
W^v_{ca}
F^ℓ
F^g

Figure 2: Illustrations of attention-based feature integration, where blue and green boxes indicate self- and cross-attention. Brown, gold and purple colored components correspond to $\mathbf{v}$, $\mathbf{k}$ and $\mathbf{q}$ implementations.

$(\mathbf{U}^\ell, \mathbf{F}^\ell)$ and $(\mathbf{U}^g, \mathbf{F}^g)$ also tend to learn rotation invariance in an unsupervised manner. Specifically, although $\mathbf{F}^\ell/\mathbf{U}^\ell$ encodes local patches with different poses (since LRFs are unaligned), and $\mathbf{F}^g/\mathbf{U}^g$ encodes the whole 3D object in a canonical pose, their feature similarity should be enforced to be similar as they represent the same 3D object, no matter what their poses are. Moreover, the mutual information between a local scale of a 3D object and a global scale of the same object is maximized, which embeds $\mathbf{U}$ with more accurate geometric information to distinguish it from objects in different classes.

We then explain the symbols in Eq. (10) of the main work. M stands for the set of all $B \times N$ pairs of positive samples across mini-batches, i.e., $M = \{(0, 0), ..., (B\times N, B\times N)\}$, where B is the number of batch size and N is the number of points. The point feature $\mathbf{U}_k$ are the set of negative keys where $(\cdot, k) \in M$ and $k \neq j$. Note that since the registration loss function is applied to both $\mathbf{F}^\ell$ and $\mathbf{F}^g$, the point features of the same point encoded from the local and global scales are pushed close to each other, where the mutual information is increased such that shared geometric information can be discovered across the local and global scales.

Training Details. For all three tasks, we set the batch size to 32 for training and 16 for testing. We use farthest point sampling to re-sample the points from the initial 10k points to 1024 points for classification and retrieval and 2048 points for segmentation. Random point translation within $[-0.2, 0.2]$ and rescaling within $[0.67, 1.5]$ were adopted for augmentation. We trained the model for 250 epochs with t_α = 15 and t = 0.017. SGD is adopted as the optimizer, where the learning rate was set to 1e-2 with momentum of 0.9 and weight decay of 1e-4. Cosine annealing was applied to reschedule the learning rate for each epoch. For classification and retrieval, we used one RTX2080Ti GPU with PyTorch for model implementation, and we used two GPUs for the segmentation task. The normal vector information is ignored for all experiments.

3 More Analysis Experiments

Influence of Randomness. We report the variance and mean values of each model in Table 5 of the main work to derive a more accurate and reliable estimate of our model performance. We hence report the variance and mean values of performance of each model in Table 5 on ModelNet40 with 5 training rounds. As shown in the Table 1, we can see that even for our model weights with the lowest performance 90.6% (among our five repeated runs), it still surpasses the highest performance among models from A to H.

Model	A	B	C
Acc. (%)	89.8±0.2	90.4±0.2	90.1±0.1
Model	D	E	F
Acc. (%)	89.8±0.4	90.1± 0.3	89.6±0.4
Model	G	H	Best
Acc. (%)	90.0±0.2	90.3±0.3	90.8±0.2

Table 1: Variance and Mean values of different model performances on ModelNet40 with z/SO(3).

Point Re-sampling and Down-sampling. We examine different point re-sampling strategies from the initial 10k input points down to 1024 input points. Experimental results of applying different sampling techniques on ModelNet40 are shown in Table 2, where we use random sampling (RS), farthest point sampling (FPS), uniform sampling (US), and inverse density importance sampling (IDIS) from (Groh, Wieschollek, and Lensch 2018) to examine the impact of different sampling methods on rotation invariance. Note that point sampling affects both LRF and GRF constructions in our design, so we can only give analysis when considering both reference frames. We can see that random sampling gives the lowest model performance with 89.7%, with 1.3% absolute performance drop compared to our method using FPS. Inverse density importance sampling can achieve a comparable result as our method, while it is not strictly invariant to rotations. We argue that due to the information compensation between features encoded from LRFs and GRF, different sampling strategies will not affect our model performance quite much.

Sampling Method	RS	FPS (ours)	US	IDIS
z/z	89.7	91.0	90.2	90.6
z/SO(3)	89.7	91.0	90.2	90.6
SO(3)/SO(3)	89.7	91.0	90.2	90.6

Table 2: Classification results (%) on ModelNet40 with different re-sampling techniques.

Visualization of U. To better present the discriminability of the learned features, we summarize the shape feature representation **U** by maxpooling and visualize it via t-SNE (Van der Maaten and Hinton 2008). Experiments are conducted on object classification under z/z and z/SO(3). Only the first 16 classes are selected for a clear representation purpose as shown in Figure 3. Although it is difficult to correctly separate all categories, we can see that some shape classes can be perfectly predicted, and the overall representation ability of **U** under different testing protocols is satisfactory and consistent.

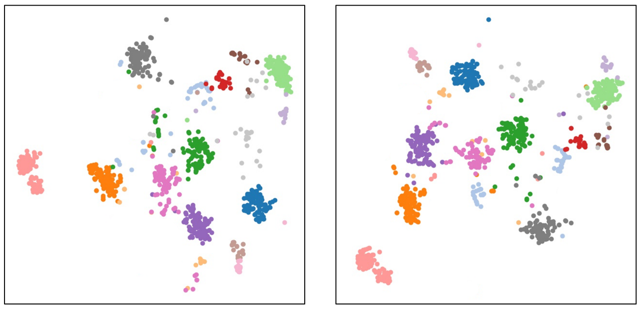


Figure 3: t-SNE of the aggregated **U** with z/SO(3) (**Left**) and SO(3)/SO(3) (**Right**). Clusters indicate good predictions in object classification.

Local Rotation Invariance. In this part, we examine the model performance when using different methods to construct the local rotation-invariant feature p_{ij}^{ℓ}. Besides the proposed method that builds p_{ij}^{ℓ} on LRFs, we also examine building p_{ij}^{ℓ} with point-pair features (PPFs) following Deng, Birdal, and Ilic (2018). As mentioned in (Deng, Birdal, and Ilic 2018), the normal information is assumed to be known and is utilized to construct PPFs. However, in our case, we only consider xyz positions of the point cloud as the known information, and we treat the point normals as unknown since in most cases normal information is hard to obtain. Hence, in Table 3, we report both cases when using the reference vector $\vec{z}$ as the point normals and the actual normal vectors for both PPF-based and LRF-based rotation-invariant features at low levels. We find that the model performance of using PPFs is lower than our LRF-based method. The reason is that point positions are important for shape learning, since they provide information of exact shape of the 3D objects. However, point-pair features give information about the topology of a 3D shape, and different 3D shapes can have the same topology, which introduces difficulty for exact 3D shape learning.

Method	PPF (w. normal)	PPF (w.o. normal)	Ours (w. normal)	Ours (w.o. normal)
Acc. (%)	89.3	88.1	91.9	91.0

Table 3: Classification results (%) on ModelNet40 with z/SO(3).

Model Complexity. Inference model sizes of different methods along with the corresponding construction time for LRFs and inference speed are reported in Table 4. The construction time measured in seconds (s) shows time cost for different models generating their low-level rotation-invariant shape codes, where we record the total time for local and global representation constructions of RI-Framework and our work. VN-DGCNN does not compute the rotation-invariant shape codes, so no result can be reported. The inference speed with the unit of number of instances evaluated within one second (ins./s) is measured for each method with a batch size of 1. When computing the inference speed, the amount of time for low-level rotation-invariant feature construction of methods (Li et al. 2021b; Kim, Park, and Han 2020; Zhang et al. 2019; Li et al. 2021a) is also considered. Table 4 shows that our method only needs a relatively short construction time for both LRFs and GRF. Meanwhile, the trade-off between the accuracy and infer-

Method	Params (M)	Times (s)	Speed (ins./s)	Acc (%)
RIConv	0.68	0.041	396.4	86.4
RI-GCN	4.19	0.057	139.1	89.5
RI-Framework	2.36	0.134	43.1	89.4
VN-DGCNN	2.77	-	77.3	89.5
Li et al. (2021a)	2.76	0.047	35.8	90.2
Ours	3.11	0.043	205.3	91.0

Table 4: Model complexity construction time for LRFs, and inference speed on ModelNet40 with z/SO(3), where Li et al. (2021a) is considered without test time augmentation.

ence speed is hard to balance. We will investigate the model design for a much high accuracy and faster speeds in the future work.

Sign Ambiguity. As mentioned in the main work, we propose simple techniques to address the sign ambiguity issue introduced by eigenvalue decomposition when computing the LRFs and GRF. We thus examine the model performance with no sign disambiguation techniques applied, of which the results are reported in Table 5. It can be seen that sign ambiguity negatively affects the model performance, where performances drop by 0.7% and 0.9% when uncertainty of vector directions is introduced to the model training. With our proposed solutions, the model behavior can be stabilized hence the classification accuracy increases.

Method	no@$\mathbf{M}^\ell$	no@$\mathbf{M}^g$	no@$\mathbf{M}^\ell$ and $\mathbf{M}^g$
Acc. (%)	90.3	90.1	89.8

Table 5: Classification results (%) on ModelNet40 with z/SO(3), where *no*@ denotes no sign disambiguation technique applied.

Rotational Effect of $\mathbf{M}^g$. As mentioned in (Li et al. 2021a), different ways to ensure $\mathbf{M}^g$ is a valid rotation matrix would result in different model performances. In this part, we examine four different methods to ensure $\mathbf{M}^g$ is a valid rotation as follows: (a) we randomly permute two basis vectors regardless of the S value; (b) we randomly negate the value of a basis vector regardless of the S value; (c) we permute two basis vectors of S values being the smallest two; (d) we simply reverse the direction of the basis vector whose S value is the smallest, which is the proposed method in our implementation. We can see from Table 6 that our simple design achieves the highest value, while all the others decrease the model performance, which shows the effectiveness of our proposed method.

3D Semantic Segmentation. To check our model's effectiveness on real-world large scenes, additional experiments are conducted on S3DIS dataset (Armeni et al. 2016), which includes six indoor areas of three different buildings. Each point is labeled by one of the 13 categories (e.g., ceiling, chair or clutter). Following the same pre-processing steps as (Qi et al. 2017b; Wang et al. 2019), each room is divided into 1m$\times$1m blocks and for each block 4096 points are sampled during training process. We use area-5 for testing and all the other areas for training. The quantitative results are shown in Table 7 following (Zhao et al. 2022), where it shows that under random rotations, our model outperforms LGR-Net by 7.8%, showing a more effective way to process large indoor scenes. For a more intuitive understanding of our model performance, qualitative results are shown in Figure 4 for reference.

3D Part Segmentation. For visualization purposes (see Figure 4 in the main work) as well as a detailed analysis of model behavior for each category, we report the per-class mIoU accuracies under z/SO(3) and SO(3)/SO(3) in

Method	a	b	c	d (ours)
Acc. (%)	90.1	90.1	90.5	91.0

Table 6: Classification results (%) on ModelNet40 with z/SO(3).

Method	z/z (%)	z/SO(3) (%)	SO(3)/SO(3)(%)
PointNet (Qi et al. 2017a)	41.1	4.1	29.3
DGCNN (Wang et al. 2019)	48.4	3.6	34.3
RIConv (Zhang et al. 2019)	22.0	22.0	22.0
LRG-Net (Zhao et al. 2022)	43.4	43.4	43.4
Ours	**51.2**	**51.2**	**51.2**

Table 7: Semantic segmentation results (mIoU) on S3DIS area-5.

Tables 8 and 9, where bold numbers indicate the best result for each category. As per-class mIoU scores are not reported in VN-DGCNN (Deng et al. 2021), we follow their official implementation[1] and report the per-class mIoU results in both tables. However, the implemented results obtained are much lower than their reported results, and we will later consult with the authors for their trained model weights and will report accurate values in the future. Beside VN-DGCNN, our model outperforms recent SoTA methods (Zhao et al. 2022; Luo et al. 2022) in segmentation of several classes such as airplane, chair, table and etc under different testing conditions. Furthermore, it is also obvious that our model performance is consistent across all categories when tested under different rotations. In addition, we have shown more qualitative examples in Figure 5, which includes all 16 classes and for each class, we present two pairs of ground truth and predicted samples. We can see that although errors occur when the boundary between the different parts have marginal difference, our model achieves great performance for most classes.

References

Armeni, I.; Sener, O.; Zamir, A. R.; Jiang, H.; Brilakis, I.; Fischer, M.; and Savarese, S. 2016. 3d semantic parsing of large-scale indoor spaces. In *CVPR*.

Deng, C.; Litany, O.; Duan, Y.; Poulenard, A.; Tagliasacchi, A.; and Guibas, L. J. 2021. Vector neurons: A general framework for SO (3)-equivariant networks. In *ICCV*.

Deng, H.; Birdal, T.; and Ilic, S. 2018. Ppfnet: Global context aware local features for robust 3d point matching. In *CVPR*.

Groh, F.; Wieschollek, P.; and Lensch, H. 2018. Flex-convolution. In *ACCV*.

Guo, M.-H.; Cai, J.-X.; Liu, Z.-N.; Mu, T.-J.; Martin, R. R.; and Hu, S.-M. 2021. PCT: Point cloud transformer. In *CVM*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *CVPR*.

Kim, S.; Park, J.; and Han, B. 2020. Rotation-Invariant Local-to-Global Representation Learning for 3D Point Cloud. In *NeurIPS*.

[1] https://github.com/FlyingGiraffe/vnn-pc

Method	mIoU	air plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor bike	mug	pistol	rocket	skate board	table
PointNet (Qi et al. 2017a)	37.8	40.4	48.1	46.3	24.5	45.1	39.4	29.2	42.6	52.7	36.7	21.2	55.0	29.7	26.6	32.1	35.8
PointNet++ (Qi et al. 2017b)	48.3	51.3	66.0	50.8	25.2	66.7	27.7	29.7	65.6	59.7	70.1	17.2	67.3	49.9	23.4	43.8	57.6
PCT (Guo et al. 2021)	38.5	32.2	44.8	36.3	26.1	36.2	40.2	48.1	42.1	54.0	40.9	18.7	50.5	25.6	27.7	44.7	47.6
RIConv (Zhang et al. 2019)	75.3	80.6	80.0	70.8	68.8	86.8	70.3	87.3	84.7	77.8	80.6	57.4	91.2	71.5	52.3	66.5	78.4
GCAConv (Zhang et al. 2020)	77.2	80.9	82.6	81.0	70.2	88.4	70.6	87.1	**87.2**	81.8	78.9	58.7	91.0	77.9	52.3	66.8	80.3
RI-Framework (Li et al. 2021b)	79.2	81.4	82.3	**86.3**	75.3	88.5	72.8	90.3	82.1	81.3	81.9	**67.5**	92.6	75.5	54.8	75.1	78.9
LGR-Net (Zhao et al. 2022)	80.0	81.5	80.5	81.4	75.5	87.4	72.6	88.7	83.4	83.1	**86.8**	66.2	92.9	76.8	62.9	**80.0**	80.0
VN-DGCNN⋆ (Deng et al. 2021)	74.4	80.8	80.0	77.3	70.9	89.2	60.4	88.8	82.9	81.1	83.8	28.7	92.8	73.9	51.3	68.0	80.5
OrientedMP (Luo et al. 2022)	80.1	81.7	79.0	85.0	**78.1**	89.7	**76.5**	**91.6**	85.9	81.6	82.1	67.6	**95.0**	79.6	**64.4**	76.9	80.7
Ours	**80.3**	**84.5**	**82.7**	83.9	76.6	**90.2**	76.1	**91.6**	86.6	**83.5**	84.6	50.1	94.4	**81.9**	60.3	75.3	**81.8**

Table 8: Segmentation results of class-wise and averaged mIoU on ShapeNetPart under z/SO(3), where ⋆ indicates that our reproducing results of VN-DGCNN following the official code[1].

Method	mIoU	air plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor bike	mug	pistol	rocket	skate board	table
PointNet (Qi et al. 2017a)	74.4	81.6	68.7	74.0	70.3	87.6	68.5	88.9	80.0	74.9	83.6	56.5	77.6	75.2	53.9	69.4	79.9
PointNet++ (Qi et al. 2017b)	76.7	79.5	71.6	87.7	70.7	88.8	64.9	88.8	78.1	79.2	**94.9**	54.3	92.0	76.4	50.3	68.4	81.0
PCT (Wang et al. 2019)	75.2	80.1	69.0	82.5	66.8	88.4	69.4	90.4	85.3	81.8	79.6	39.9	89.2	76.5	51.8	72.6	80.0
RIConv (Zhang et al. 2019)	75.5	80.6	80.2	70.7	68.8	86.8	70.4	87.2	84.3	78.0	80.1	57.3	91.2	71.3	52.1	66.6	78.5
GCAConv (Zhang et al. 2020)	77.3	81.2	82.6	81.6	70.2	88.6	70.6	86.2	86.6	81.6	79.6	58.9	90.8	76.8	53.2	67.2	81.6
RI-Framework (Li et al. 2021b)	79.4	81.4	**84.5**	**85.1**	75.0	88.2	72.4	90.7	84.4	80.3	84.0	**68.8**	92.6	76.1	52.1	74.1	80.0
LGR-Net (Zhao et al. 2022)	80.1	81.7	78.1	82.5	75.1	87.6	74.5	89.4	86.1	83.0	86.4	65.3	92.6	75.2	**64.1**	**79.8**	80.5
VN-DGCNN⋆ (Deng et al. 2021)	74.7	80.0	79.4	79.1	71.5	89.2	66.1	89.0	83.5	80.6	82.0	29.3	91.4	73.4	51.5	67.8	81.0
OrientedMP (Luo et al. 2022)	**80.9**	81.8	78.8	85.4	**78.0**	89.6	**76.7**	**91.6**	85.7	81.7	82.1	67.6	**95.0**	**79.1**	63.5	76.5	81.0
Ours	80.4	**84.3**	82.2	84.6	77.9	**89.9**	76.6	91.3	**86.7**	**84.1**	84.3	50.1	93.4	79.0	63.7	75.3	**82.3**

Table 9: Segmentation results of class-wise and averaged mIoU on ShapeNetPart under SO(3)/SO(3).

Li, F.; Fujiwara, K.; Okura, F.; and Matsushita, Y. 2021a. A Closer Look at Rotation-Invariant Deep Point Cloud Analysis. In *ICCV*.

Li, X.; Li, R.; Chen, G.; Fu, C.-W.; Cohen-Or, D.; and Heng, P.-A. 2021b. A rotation-invariant framework for deep point cloud analysis. In *TVCG*.

Luo, S.; Li, J.; Guan, J.; Su, Y.; Cheng, C.; Peng, J.; and Ma, J. 2022. Equivariant Point Cloud Analysis via Learning Orientations for Message Passing. In *CVPR*.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2017a. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*.

Qi, C. R.; Yi, L.; Su, H.; and Guibas, L. J. 2017b. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*.

Van der Maaten, L.; and Hinton, G. 2008. Visualizing data using t-SNE. In *JMLR*.

Wang, Y.; Sun, Y.; Liu, Z.; Sarma, S. E.; Bronstein, M. M.; and Solomon, J. M. 2019. Dynamic graph CNN for learning on point clouds. In *ACM ToG*.

Zhang, Z.; Hua, B.-S.; Chen, W.; Tian, Y.; and Yeung, S.-K. 2020. Global context aware convolutions for 3D point cloud understanding. In *3DV*.

Zhang, Z.; Hua, B.-S.; Rosen, D. W.; and Yeung, S.-K. 2019. Rotation invariant convolutions for 3D point clouds deep learning. In *3DV*.

Zhao, C.; Yang, J.; Xiong, X.; Zhu, A.; Cao, Z.; and Li, X. 2022. Rotation invariant point cloud analysis: Where local geometry meets global topology. In *Pattern Recognition*.

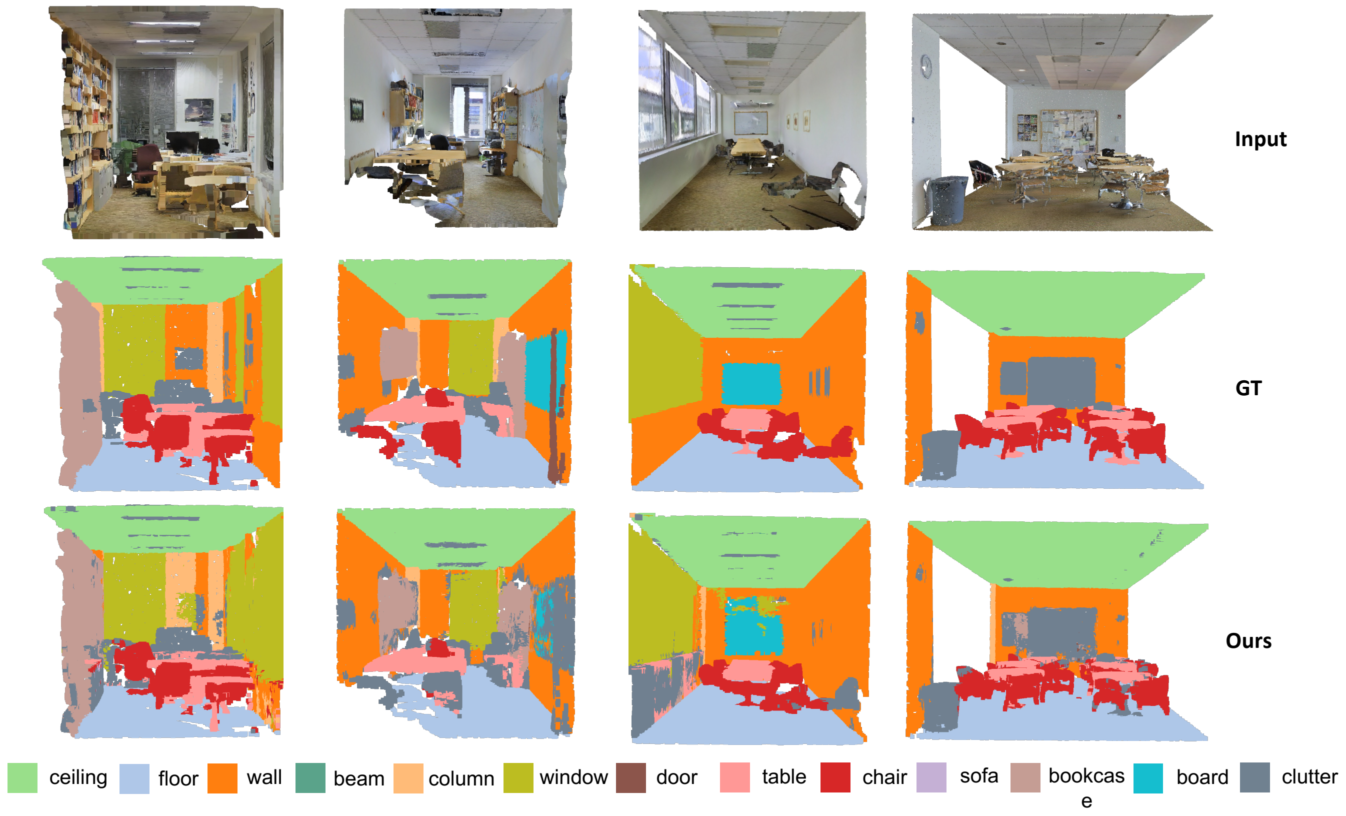


Figure 4: Visualization of semantic segmentation results on S3DIS area-5. The first row is the original inputs, the second row is the ground truth (GT) samples and the last row is our predicted results.

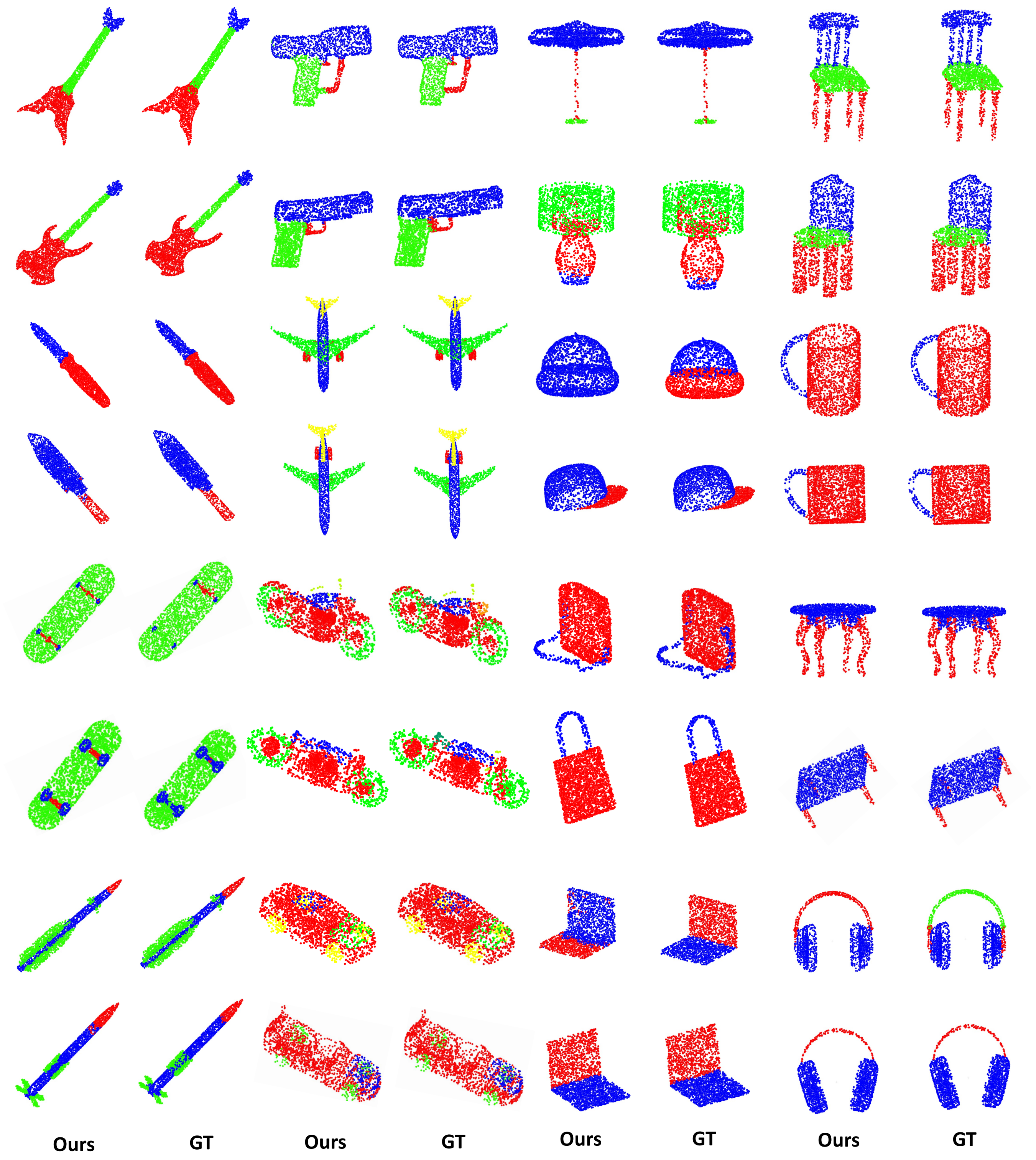


Figure 5: Segmentation comparisons between the ground truth (GT) and our model on ShapeNetPart dataset under z/SO(3).