RISurConv: Rotation Invariant Surface Attention-Augmented Convolutions for 3D Point Cloud Classification and Segmentation

Zhiyuan Zhang[1], Licheng Yang[2⋆], and Zhiyu Xiang[2]

[1] School of Computing and Information Systems, Singapore Management University
https://github.com/cszyzhang/RISurConv
[2] College of Information Science and Electronic Engineering, Zhejiang University

Abstract. Despite the progress on 3D point cloud deep learning, most prior works focus on learning features that are invariant to translation and point permutation, and very limited efforts have been devoted for rotation invariant property. Several recent studies achieve rotation invariance at the cost of lower accuracies. In this work, we close this gap by proposing a novel yet effective rotation invariant architecture for 3D point cloud classification and segmentation. Instead of traditional point-wise operations, we construct local triangle surfaces to capture more detailed surface structure, based on which we can extract highly expressive rotation invariant surface properties which are then integrated into an attention-augmented convolution operator named RISurConv to generate refined attention features via self-attention layers. Based on RISurConv we build an effective neural network for 3D point cloud analysis that is invariant to arbitrary rotations while maintaining high accuracy. We verify the performance on various benchmarks with supreme results obtained surpassing the previous state-of-the-art by a large margin. We achieve an overall accuracy of **96.0%** (+4.7%) on ModelNet40, **93.1%** (+12.8%) on ScanObjectNN, and class accuracies of **91.5%** (+3.6%), **82.7%** (+5.1%), and **78.5%** (+9.2%) on the three categories of the FG3D dataset for the fine-grained classification task. Additionally, we achieve **81.5%** (+1.0%) mIoU on ShapeNet for the segmentation task.

Keywords: Point cloud · Rotation invariant · Attention

1 Introduction

Point cloud has become the most promising 3D data representation for a wide range of immersive applications from robot navigation to autonomous driving. The increasing availability of 3D sensors and the emergence of large and mid-scale point cloud datasets [1,2,12,19,31,35] have spurred significant research in this area, leading to the development of numerous deep learning models for point cloud classification and segmentation, which are the fundamentals for various computer vision tasks.

⋆ co-first author.

However, analyzing 3D point clouds remains challenging, mainly due to the irregular nature of point clouds and their inherent invariances such as translation, point permutation, and rotation. While significant progress has been made in learning translation and permutation-invariant features [17, 23, 25, 26, 44], achieving rotation invariance in point cloud convolution has been a relatively unexplored area.

Rotation invariance is essential for 3D object classification and segmentation as objects can be viewed from different viewpoints, leading to variations in their orientation. Prior approaches [23,25,33] usually rely on rotation augmentation in training stage to relief the rotation affections for testing. However, such scheme is less effective as 3D data has more degrees of freedom. A rotation invariant model is, therefore, critical to accurately classify and segment objects in 3D point clouds.

Various attempts have been made to address the problem of rotation invariance in 3D point clouds, as evidenced by several works [3, 4, 22, 27, 41]. These efforts have focused on designing rotation invariant properties to achieve consistent accuracies under arbitrary rotations without resorting to rotation augmentation. However, these methods have demonstrated inferior performance when compared to translation invariant approaches, primarily due to the loss of global information during the generation of rotation invariant properties. Recent research efforts [15,30,40] have attempted to overcome this limitation by employing local reference frame (LRF) or local reference axis (LRA) [42] to transform the data into a canonical coordinate system and encode global information. Despite the improvements, their accuracies are still lower than those of state-of-the-art non-rotation invariant methods because of the unstable LRF/LRA and the less descriptive rotation invariant properties that the useful surface information is not well preserved.

To address the above-mentioned issues, we propose a novel yet effective rotation invariant architecture. Specifically, we construct local triangle surfaces for each reference point of the input data to better capture the local surface structure. On the local surfaces, we design highly expressive rotation invariant surface properties which are then integrated into an attention-augmented convolution operator named RISurConv to extract refined rotation invariant features. Finally, we build up network based on RISurConv for rotation invariant object classification and segmentation. In summary, our main contributions include:

- **Rotation Invariant Surface Properties**. We construct local triangle surfaces from the reference point and its neighbors, based on which we are able to design highly expressive rotation invariant surface properties;
- **RISurConv**. We integrate the Rotation Invariant Surface Properties into an attention-augmented architecture named RISurConv comprising two self-attention layers to learn and generate refined rotation invariant features;
- Extensive experiments on a variety of classification and segmentation tasks. Our approach shows supreme performance, **surpassing the state-of-the-art by a large margin** under challenging rotation scenarios including an analysis of rotation-invariant features and an ablation study of our neural

network to provide insights into the key factors contributing to the exceptional performance.

2 Related Works

In this section, we review the representative works in 3D point cloud deep learning and rotation invariant learning.

3D Point Cloud Deep Learning. 3D point clouds is a more compact and intuitive representation for feature learning. PointNet [23] pioneered a point cloud convolution with global features by max-pooling per-point features from MLPs, and follow-up works [17, 25, 38] have focused on exploring convolution kernels that exploit geometric features [29], adding edges on top of points [33], parameterizing convolution using polynomials [38], and leveraging shape context [37]. Some methods are designed to combine with recurrent neural networks [13] and sequence models [18].

In recent years, attention mechanism has become increasingly popular in various natural language processing (NLP) [32] and computer vision [6] tasks. With their ability to handle sequential and spatial information, attention mechanism has also shown promise in 3D point cloud processing tasks. Pioneer works that apply attention mechanism in point clouds include Point Transformer [43] and PCT [9], which use a self-attention mechanism to learn local and global features of point clouds. Other works have explored different types of transformers. Inspired by BERT [5], Point-BERT [39] was proposed to pre-train pure Transformer-based models with a Mask Point Modeling task for point cloud classification. Dual Transformer Network [11] aggregated the point-wise and channel-wise multi-head self-attention models simultaneously such that long-range context dependencies can be captured by investigating the point-wise and channel-wise relationships. Overall, these developments show promise in enhancing the efficiency and accuracy of point cloud analysis for a variety of applications. Readers can refer to the survey [10] for a comprehensive overview of 3D point cloud deep learning. However, most of these methods lacks the property of rotation invariance which is critical for object classification and segmentation as the object can be viewed at any angle in reality. To address this issue, a common approach is to increase the training data by augmenting it with arbitrary rotations [17, 25, 33]. However, such approach has difficulty in generalizing predictions to unseen rotations, resulting in deteriorated performance. Therefore, it would be desirable to design a specific convolution possessing rotation-invariant features.

Rotation Invariant Learning. Various methods have been proposed to address the issue of rotation invariance in feature learning of point clouds. For instance, Rao et al. [27] used a spherical domain to define a rotation-invariant convolution for point clouds. However, the discretized sphere is sensitive to global rotations, which can lead to a notable drop in performance for objects with arbitrary rotations. To overcome this limitation, Poulenard et al. [22] integrated spherical harmonics to their convolution. Similarly, Chen et al. [3] proposed a

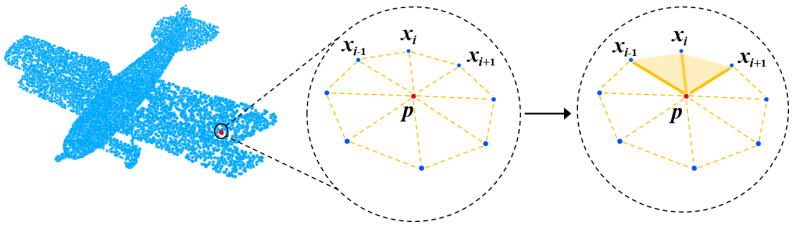

Fig. 1: Rotation Invariant Surface Property (RISP) construction: Given a point p as the reference point, K ($K = 8$ in this example) nearest points are selected (middle). For each neighbor x_i, two adjacent neighbors x_{i-1} and x_{i+1} are used to form two triangular local surfaces (right), based on which rotation invariant properties are constructed.

hierarchical clustering scheme to encode the relative angles between two-point vectors and used vector norm to maintain rotation invariance. Zhang et al. [41] presented a simple convolution named RIConv that operates on handcrafted features built from Euclidean distances and angles that are rotation invariant by nature. However, this approach only considers local features resulting in accuracy degradation. GCAConv [40] addressed this limitation by building a global context-aware convolution based on anchors and Local Reference Frame (LRF) to achieve rotation invariance. RI-GCN [15] learned rotation-invariant local descriptors and applied graph convolutional neural networks to aggregate local features based on LRF. Thomas [30] also relied on LRF and used a multiple alignment scheme to achieve better results. RIF [16] presented a framework to construct both local and global features based on distances, angles, and reference points. To remove the uncertainty of LRF in x and y directions, RIConv++ [42] extracts informative features based on local reference axis (LRA). However, the LRF/LRA and global reference used in these methods may not be stable enough limiting their overall performance. We can clearly see the accuracy discrepancy of the object classification task on ModelNet40 dataset [36]. State-of-the-art translation-invariant convolutions such as PointNet++ [25] or Point Transformer v2 [34] achieve 89%-94% of accuracy while rotation-invariant convolutions only report up to 87%-91% of accuracy [14, 42].

Our target in this paper is not only closing this performance gap but also surpassing the state-of-the-art methods while maintaining rotation invariance.

3 Rotation Invariant Surface Property

In this section, we detail the Rotation Invariant Surface Property construction which is the first step of our method. Our goal for this step is to design highly expressive rotation invariant properties from underlying local surfaces. Different from previous works that rely on pointwise operations, we construct local surfaces around the reference point to better capture the local surface structure, based on which we then extract more expressive rotation invariant properties.

The construction of rotation invariant surface property is shown in Fig. 1. Given a point p as the reference point (red), we get K nearest neighbors to form a local point set. In this case, K is 8. For each neighbor x_i, two adjacent neighbors, x_{i-1} and x_{i+1}, are identified based on the Euclidean distances, forming two triangular local surfaces, as shown in the figure. Then, we construct the rotation invariant surface properties (RISP) as follows:

$$\begin{aligned} \mathrm{RISP}(x_i) = [&L_0, \phi_1, \phi_2, \phi_3, \phi_4, \phi_5, \\ &\alpha_1, \alpha_2, \beta_1, \beta_2, \theta_1, \theta_2, \gamma_1, \gamma_2] \end{aligned} \tag{1}$$

where L_0 measures the distance from reference p to neighbor x_i, and ϕ_1 to ϕ_5 measure the two triangles as well as the relationship between the two triangle surfaces with regard to the edge $\overrightarrow{px_i}$ in the Euclidean space:

$$\begin{aligned} &\phi_1 = \angle\left(\overrightarrow{x_{i-1}p}, \overrightarrow{x_ip}\right), \phi_2 = \angle\left(\overrightarrow{x_{i+1}p}, \overrightarrow{x_ip}\right), \\ &\phi_3 = \angle\left(\overrightarrow{x_{i-1}x_i}, \overrightarrow{x_{i-1}p}\right), \phi_4 = \angle\left(\overrightarrow{x_{i+1}p}, \overrightarrow{x_{i+1}x_i}\right), \\ &\phi_5 = \angle\left(\overrightarrow{x_{i+1}p} \times \overrightarrow{x_ip}, \overrightarrow{x_{i-1}p} \times \overrightarrow{x_ip}\right), \end{aligned} \tag{2}$$

while other properties describe the two surfaces in the tangent space, e.g., normal vectors can define the directions in which the surface is bending away from the tangent space:

$$\begin{aligned} &\alpha_1 = \angle\left(\overrightarrow{n_p}, \overrightarrow{x_ip}\right), \alpha_2 = \angle\left(\overrightarrow{n_p}, \overrightarrow{x_{i-1}p}\right), \\ &\beta_1 = \angle\left(\overrightarrow{n_i}, \overrightarrow{x_ip}\right), \beta_2 = \angle\left(\overrightarrow{n_i}, \overrightarrow{x_{i-1}x_i}\right), \\ &\theta_1 = \angle\left(\overrightarrow{n_{i-1}}, \overrightarrow{x_{i-1}p}\right), \theta_2 = \angle\left(\overrightarrow{n_{i-1}}, \overrightarrow{x_{i-1}x_i}\right), \\ &\gamma_1 = \angle\left(\overrightarrow{n_{i+1}}, \overrightarrow{x_{i+1}x_i}\right), \gamma_2 = \angle\left(\overrightarrow{n_{i+1}}, \overrightarrow{x_{i+1}p}\right). \end{aligned} \tag{3}$$

RISP is able to fully describe the dual triangles as well as their relationship along different directions. Readers can refer to the supplementary for the mathematical **proof of the completeness** of RISP defined in Eq. (1).

Number of local triangle surfaces. Throughout the paper, we define the number of local triangle surfaces as 2. Such setting is due to the fact that we wish to capture local surface as much as possible to extract more useful information to reflect the real shape of the object. So we construct 2 triangles rather than single triangle. However, this does not mean the more the better since more triangles can result in distorted surface and affect the computing efficiency. Please refer to the ablation study section for more details.

4 The RISurConv Operator

Based on the Rotation Invariant Surface Properties (RISP), we are able to build up RISurConv. The main steps are detailed in Fig. 2. To start, we utilize a farthest point sampling strategy to generate uniformly distributed representative

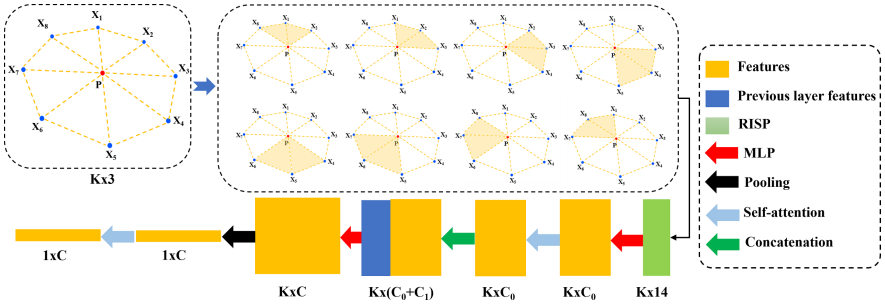

Fig. 2: RISurConv operator. For a local point set with p as the reference (red), K nearest neighbors are labelled as blue. Then, we compute the Rotation Invariant Surface Properties at each neighbor by constructing local dual triangle surfaces (Sec. 3), which is embedded to a high-dimensional space by a shared multi-layer perceptron (MLP) followed by a self-attention layer to produce refined features. Concatenated with previous layer features (if any), the features of these local points are further passed to MLPs, which are then summarized by maxpooling. To further refine the features, another self-attention layer follows.

Algorithm 1 RISurConv operator.

Input: Reference point p, point set Ω, point features $\mathbf{f}_{prev}$ from previous layer (if any)
Output: Convoluted features $\mathbf{f}$

$\mathbf{f} \leftarrow \{\text{RISP}(x_i) : \forall x_i \in \Omega\}$ * Construct Rotation Invariant Surface Properties (Sec. 3)
$\mathbf{f} \leftarrow \text{MLP}(\mathbf{f})$; * Embed each feature to a high-dimensional feature space
$\mathbf{f} \leftarrow \text{SA}(\mathbf{f})$; * Refine features via self-attention layer
$\mathbf{f}_{in} \leftarrow [\mathbf{f}_{prev}, \mathbf{f}]$ * Concatenate the features from the local and the previous layer (if any)
$\mathbf{f}_{out} \leftarrow \text{MLP}(\mathbf{f}_{in})$ * Feature embedding
$\mathbf{f}_{out} \leftarrow \text{maxpool}(\mathbf{f}_{out})$ * Maxpool features
return $\text{SA}(\mathbf{f}_{out})$ * Self-attention and return

points. From this initial set, we perform K-nearest neighbor searches to obtain local point sets. Let us denote a local point set by $\Omega = \{x_i\}$, where x_i represents the 3D coordinates of point i. We define the attention-augmented convolution operation as follows:

$$\mathbf{f}(\Omega) = SA(\sigma(\mathcal{A}(\{\mathcal{T}(\mathbf{f}_{x_i}) : \forall i\}))). \tag{4}$$

This formula indicates that features of each point in the point set are first transformed by $\mathcal{T}$ before being aggregated by the aggregation function $\mathcal{A}$ and passed to an activation function σ. SA is a channel-wise self-attention layer used to output refined features. We set the input features to our expressive rotation-invariant features $\mathbf{f}_{x_i} = \text{RISP}(x_i)$. We define the transformation function as

$$\mathcal{T}(\mathbf{f}_{x_i}) = \mathbf{w}_i \cdot \mathbf{f}_{x_i} = \mathbf{f}'_{x_i} \tag{5}$$

where $\cdot$ indicates the element-wise product, and $\mathbf{w}_i$ is the weight parameter to be learned by the network. Our transformation function is similar to PointNet++. A popular choice of the aggregation function $\mathcal{A}$ is maxpooling, which supports permutation invariance of the input point features.

To proceed with feature learning and be invariant to point permutation, we construct surfaces for each neighbor from which we construct RISPs. In Fig. 2, there are eight neighbor, so the resulted RISPs are in size of 8×14. Here, 14 is the length of the RISP as defined in Eq. (1) and 8 is the neighborhood size. Since RISPs are used as the input which are already rotation invariant, RISurConv is rotation invariant by nature. We then embed the RISPs into feature space using two layers of MLPs followed by a self-attention (SA) layer for feature refinement. We concatenate the refined features with previous layer features (if any), and embed the concatenated features into higher dimensional space via MLPs again. After the maxpooling, the aggregated features are passed into another SA layer again for further refinement. The detailed steps are shown in the Algorithm 1. Here, the two SA layers used in RISurConv are the standard SA module of the transformer [32].

Self Attention Layers. In RISurConv there are two Self-Attention (SA) layers by which the extracted features are enhanced. Below, we provide a more detailed illustration of the structure of the SA layers. The SA structure is the standard SA module of the famous transformer [32] (Attention is all you need) which runs twice in RISurConv. The first SA is for the K neighbors, and the second SA works for the N representative points. The input tensor shape is $[B, N, K, C_0]$ for the first SA which goes into linear layers and outputs Q,K, and V respectively in shape of $[B, N, K, C_0]$. According to $attn(Q, K, V) = Softmax(QK^\top / sqrt(d_k))V$, we have the 1st attention score in $[B, N, K, K]$. By multiplying with V, we get the refined feature in $[B, N, K, C_0]$. The same goes with the second SA where the input shape is $[B, N, C]$ and the attention score shape is $[B, N, N]$.

5 RISurConv Networks

We employed the RISurConv to develop neural networks for object classification and segmentation, as illustrated in Fig. 3. Our classification network follows a standard architecture similar to the single-scale grouping version of PointNet++, consisting of five consecutive layers of RISurConv followed by a transformer encoder [32] with 8 heads to enhance the extracted features. Finally, we connect the network to fully connected layers to produce the probability map. One of the major advantages of RISurConv is its ability to handle arbitrary rotation and point orders, which enables us to place each RISurConv layer consecutively without the need for complex preprocessing steps. Additionally, we apply batch normalization and ReLU activation to each convolution layer by default. The segmentation network utilizes an encoder-decoder structure similar to the U-Net [28]. We adopt the same definition of deconvolution as RISurConv. The key distinction lies in that the convolution produces a point subset with a greater

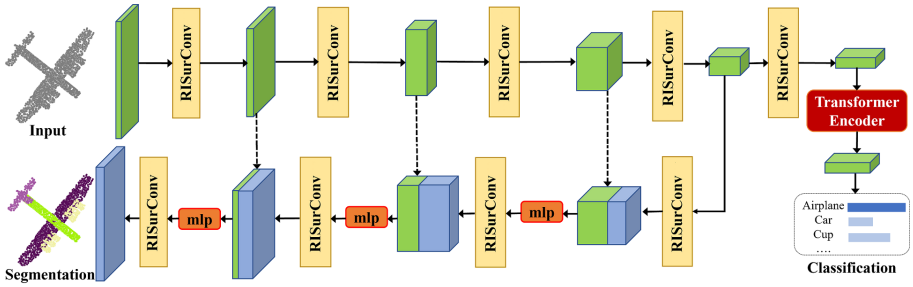

Fig. 3: Our neural network architecture comprises five RISurConv layers to extract rotation invariant features followed by a Transformer Encoder to enhance the learnt features before fully connected layers for object classification. We add a decoder with skip connections for segmentation task.

number of feature channels and less points while the deconvolution outputs to a point set with more points compared to the input with fewer feature channels. Please refer to the supplementary for more details regarding the network architecture.

6 Experiments

We report our evaluation results in this section. Our network is implemented in PyTorch using an Adam optimizer with initial learning rate of 0.001 for both classification and segmentation. Our training is executed on a computer with an Intel(R) Core(TM) i7-10700K CPU equipped with a NVIDIA GTX 3090 GPU.

We evaluated the performance of our method extensively on a series of classification and segmentation benchmarks. To train our model, we utilized 450 epochs for classification and 250 epochs for segmentation for most datasets except the Car and Chair categories of the fine-grained FG3D dataset due to small size and subtle variation where we applied 900 epochs. Our method can achieve convergence within 11 hours for classification and 24 hours for segmentation. To assess the robustness of our method, we followed the experimental design proposed by [8] and conducted experiments in three cases. The first case involved training and testing with data augmented with rotation about the gravity axis (z/z), which is a common approach for evaluating translation-invariant point cloud learning methods. The second case involved training and testing with data augmented with arbitrary SO3 rotations (SO3/SO3), and the third case involved training with data by z-rotations and testing with data by SO3 rotations (z/SO3), which are used to evaluate rotation invariance. A rotation-invariant method should produce consistent results across all three cases. Our method demonstrated superior performance across all three cases, highlighting its robustness and effectiveness.

6.1 Human-made Object Classification

ModelNet40 [36] that comprises 9843 training models and 2468 test models, divided into 40 categories is one the of the most popular datasets for point cloud classification evaluation. The input data size is 1024 with each point equipped with (x; y; z; nx; ny; nz) which are 3D coordinates and 3D normals in the Euclidean space. Note that **normals are optional** to our method. When normals are not available (w/o n), we use weighted eigen vector corresponding to the smallest eigen value as the normal. This strategy has been used in SHOT and RIConv++ [42] which is highly robust to noise and efficient to compute.

The experimental results are shown in Tab. 1. We use two criteria for evaluation: overall accuracy and accuracy standard deviation (Std.). True rotation invariant methods are expected to be unaffected by the rotations present in the training and testing data, resulting in low accuracy deviation.

From Tab. 1, we see that our method surpasses the state-of-the-art performance in all cases while maintaining zero accuracy deviation. It is worth noting that our method surpasses both rotation invariant and non-rotation invariant methods. To the best of our knowledge, this is the first work that achieves such high accuracy. It outperforms the state-of-the-art rotation invariant method RIConv++ by **4.7%**, and outperforms the state-of-the-art non-rotation invariant method Point Transformer v2 by **1.8%** under z/z case.

6.2 Real World Object Classification

We also evaluate the classification performance on a real-world 3D point cloud dataset ScanObjectNN [31]. It is composed of 2902 point clouds categorized into 15 categories sampled from real world indoor scenes. For our evaluation, we use the processed files and choose the hardest variant PB_T50_RS with 50% bounding box translation, rotation around the gravity axis, and random scaling that result in rotated and partial data. The results are shown in Tab. 2. We see that RISurConv surpasses all the compared methods by a large margin. Particularly, our method **significantly outperforms** the state-of-the-art rotation invariant approach RIConv++ [42] by **12.8%**. This verifies that our method is effective for both synthetic and real world data. Note that we only test the 'w/o normal' case as the normal vectors are not provided in the processed files of this dataset.

Table 2: Comparisons of real world 3D point cloud classification on hardest variant of ScanObjectNN dataset (**Overall Accuracy**).

	PB_T50_RS		
Method	z/z	SO3/SO3	z/SO3
PointNet [23]	68.2	42.2	17.1
PointNet++ [25]	77.9	60.1	15.8
PointCNN [17]	78.5	51.8	14.9
DGCNN [33]	78.1	63.4	16.1
RIConv [41]	68.1	68.3	68.3
GCAConv [40]	69.8	70.0	69.8
RIConv++ [42]	80.3	80.3	80.3
Ours (w/o normal)	**93.1**	**93.1**	**93.1**

Table 1: Comparisons of the classification accuracy (%) on the ModelNet40 dataset. On average, our method has the best accuracy and lowest accuracy deviation in all cases (**Overall Accuracy**).

	Method	Format	Input Size	Params.	z/z↑	SO3/SO3↑	z/SO3↑	Std.↓
Traditional	VoxNet [21]	voxel	30^3	0.90M	83.0	87.3	-	3.0
	SubVolSup [24]	voxel	30^3	17.00M	88.5	82.7	36.6	28.4
	PointNet [23]	xyz	1024×3	3.50M	87.0	80.3	21.6	41.0
	PointCNN [17]	xyz	1024×3	0.60M	91.3	84.5	41.2	27.2
	PointNet++ [25]	xyz + nor	1024×6	1.40M	89.3	85.0	28.6	33.8
	DGCNN [33]	xyz	1024×3	1.84M	92.2	81.1	20.6	38.5
	RS-CNN [20]	xyz	1024×3	1.41M	90.3	82.6	48.7	22.1
	Pt Transformer [43]	xyz	1024×3	-	93.7	85.9	50.1	19.1
	Pt Transformer v2 [34]	xyz	1024×3	-	94.2	88.3	51.8	23.0
Rotation-invariant	Spherical CNN [7]	voxel	2×64^2	0.50M	88.9	86.9	78.6	5.5
	RIConv [41]	xyz	1024×3	0.70M	86.5	86.4	86.4	0.1
	SPHNet [22]	xyz	1024×3	2.90M	87.0	87.6	86.6	0.5
	SFCNN [27]	xyz	1024×3	-	91.4	90.1	84.8	3.5
	ClusterNet [3]	xyz	1024×3	1.40M	87.1	87.1	87.1	**0.0**
	GCAConv [40]	xyz	1024×3	0.41M	89.0	89.2	89.1	**0.0**
	RIF [16]	xyz	1024×3	-	89.4	89.3	89.4	**0.0**
	RI-GCN [15]	xyz + nor	1024×6	4.38M	91.0	91.0	91.0	**0.0**
	RIConv++ [42]	xyz	1024×3	0.40M	91.2	91.2	91.2	**0.0**
	RIConv++ [42]	xyz + nor	1024×6	0.40M	91.3	91.3	91.3	**0.0**
	Ours (w/o normal)	xyz	1024×3	14.0M	<u>95.6</u>	<u>95.6</u>	<u>95.6</u>	**0.0**
	Ours (w/ normal)	xyz + nor	1024×6	14.0M	**96.0**	**96.0**	**96.0**	**0.0**

6.3 Fine-Grained Object Classification

Fine-grained object classification is more challenging because the differences between subcategories are subtle and require a high level of precision in distinguishing them. We conduct experiments on the three categories from FD3D dataset [19]: Airplane, Chair and Car. To the best of our knowledge, this is the first work to test the performance of rotation invariant methods on fine-grained 3D point cloud dataset. The class accuracies are shown in Tab. 3. Again, our method outperforms the state-of-the-art approaches by large margins for all rotation cases and all the categories. Specifically, our method improves non-rotation invariant method FG3D-Net by **2.1%**, **2.7%**, and **4.5%** for Airplane, Chair, and Car categories respectively under the z/z rotation case, and the accuracy consistency is well preserved. Compared to the rotation invariant methods, our method outperforms the latest RIConv++ by **3.6%**, **5.1%**, and **9.2%** on the three categories. Such supreme performance shows that features extracted by RISurConv are not only rotation invariant but also highly expressive.

6.4 Part Segmentation on ShapeNet

Table 3: Comparisons of fine-grained 3D point cloud classification on FG3D dataset (**Class Accuracy**).

Method	Modality	Airplane			Chair			Car		
		z/z	SO3/SO3	z/SO3	z/z	SO3/SO3	z/SO3	z/z	SO3/SO3	z/SO3
VoxNet	Voxel	79.2	57.5	28.7	73.3	54.7	16.7	68.2	42.2	17.1
MVCNN	View	82.6	66.9	21.9	76.3	63.7	14.6	71.9	51.8	14.9
View-GCN	View	87.4	56.6	22.7	79.7	50.2	19.7	73.7	44.4	18.1
RotationNet	View	89.1	50.8	29.8	78.5	36.2	18.4	72.5	45.3	28.3
Part4Feature	View	82.6	49.1	32.2	77.1	52.2	30.2	73.4	35.0	23.8
FG3D-Net	View	89.4	61.1	26.9	80.0	43.2	18.4	74.0	56.4	20.8
PointNet	Point	82.7	55.5	29.7	72.1	33.9	18.2	68.1	32.1	16.4
PointNet++	Point	87.3	57.1	25.6	78.1	40.4	16.0	70.3	50.4	20.8
RS-CNN	Point	82.8	45.9	32.2	75.1	50.7	18.6	71.2	31.8	13.9
DGCNN	Point	88.4	60.6	22.7	71.7	52.3	19.7	65.3	53.4	18.1
RIConv	Point	85.8	85.8	85.8	76.3	76.3	76.3	67.3	67.3	67.3
RIConv++	Point	87.9	87.9	87.9	77.6	77.6	77.6	69.3	69.3	69.3
Ours	Point	**91.5**	**91.5**	**91.5**	**82.7**	**82.7**	**82.7**	**78.5**	**78.5**	**78.5**

To evaluate the segmentation performance, we employed the ShapeNet dataset comprising 16880 CAD models distributed across 16 distinct categories. These models are annotated with 2 to 6 parts each, culminating in a dataset of 50 object parts. We follow the standard train/test split with 14006 models for training and 2874 models for testing, respectively.

Table 4: Comparisons of object part segmentation performed on ShapeNet dataset. The mean per-class IoU (mIoU, %) is used to measure the accuracy under two challenging rotation modes: SO3/SO3 and z/SO3.

Method	Input	SO3/SO3	z/SO3
PointNet [23]	xyz	74.4	37.8
PointNet++ [25]	xyz+nor	76.7	48.2
PointCNN [17]	xyz	71.4	34.7
DGCNN [33]	xyz	73.3	37.4
RS-CNN [20]	xyz	72.5	36.5
SpiderCNN	xyz+nor	72.3	42.9
RIConv [41]	xyz	75.5	75.3
GCAConv [40]	xyz	77.3	77.2
RI-GCN [15]	xyz	77.0	77.0
RIF [16]	xyz	79.4	79.2
RIConv++ [42]	xyz	80.3	80.3
RIConv++ [42]	xyz+nor	80.5	80.5
Ours (w/o normal)	xyz	<u>81.3</u>	<u>81.3</u>
Ours (w/ normal)	xyz+nor	**81.5**	**81.5**

The evaluation results are shown in Tab. 4. Our method outperforms both traditional translation-invariant and latest rotation invariant methods significantly in the SO3/SO3 and z/SO3 scenario. Our method outperforms the RIConv++ by **1.0%** mIoU. This result aligns well with the performance reported in the object classification task. We also show the qualitative results by error maps in Fig. 4. The wrong segmentation points are plotted as red. It clearly shows that our predictions are the closest to the ground truth.

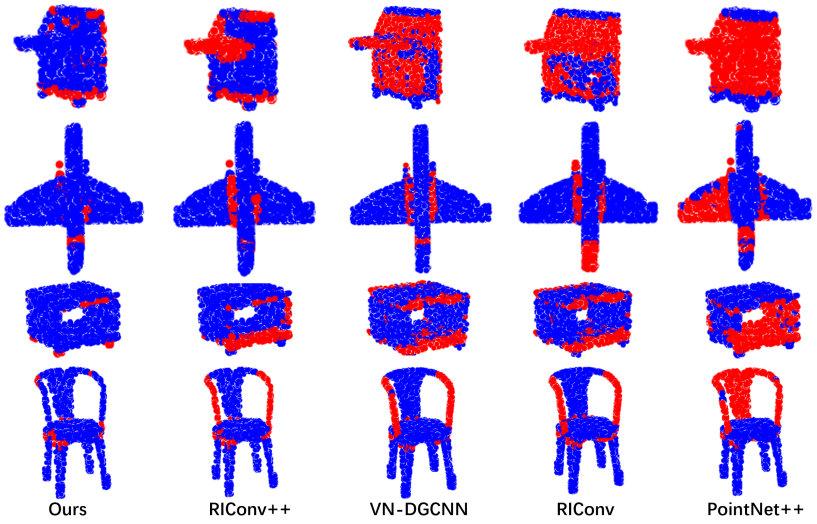

Fig. 4: Qualitative comparisons (Red indicates wrong).

6.5 Ablation Studies

We ablate some vital designs of our method on ModelNet40 [36] for an insightful exploration.

RISP Design. We first test the performance by turning on/off and adding different rotation invariant components used in the RISP construction (Algorithm 1). The results are shown in Tab. 5. Model A is our baseline setting with all rotation invariant features activated. Model B has only angle features but still achieves relatively high accuracy. Model C turns off both L_0 and $\phi_{1...5}$ with accuracy decreases to 90.9%. This means that the distance L_0 is not as important as angles. In Model D, we turn off the angle features on the tangent direction with only L_0 and $\phi_{1...5}$ are kept which is more like RIConv [41]. Compared to Model A, it shows that our proposed features is more effective than those by RIConv, and the improvement is explained by the additional consideration of the relations among the neighbor points. Model E adds more rotation invariant

Table 5: Ablation study on the RISP design.

Model	L_0	$\phi_{1...5}$	$\alpha_{1,2}$	$\beta_{1,2}$	$\theta_{1,2}$	$\gamma_{1,2}$	λ	μ	Acc.
A	✓	✓	✓						96.0
B		✓	✓						95.5
C			✓						90.9
D	✓	✓							88.2
E	✓	✓	✓				✓		95.7

Table 6: Ablation study on the Self-Attention module.

Model	SA1	SA2	Transformer Encoder	Acc.
A	✓	✓	✓	96.0
B		✓	✓	95.6
C	✓		✓	95.2
D	✓	✓		94.3
E				92.8

features. λ and μ represent the angles $\angle(\boldsymbol{n_p}, \boldsymbol{n_{i+1}})$ and $\angle(\boldsymbol{n_p}, \boldsymbol{n_{i-1}})$ respectively. The results show that the accuracy does not increase. This verifies the completeness of the proposed RISP.

Self-Attention Effects. We employ multiple self-attention (SA) modules to help produce refined rotation invariant features. So it is necessary to analyse the effects of different SA modules. In RISurConv, there are two SA modules and we name them as SA1 and SA2 respectively. Before fully connected layers, there is a Transformer Encoder module. We test these three modules by turning off one of them each time. From Tab. 6, we see that when removing the first SA module in RISurConv, the performance drops a bit, while removing the second SA module also decreases the accuracy. This indicates that both SA1 and SA2 are important for the success of our method. In addition, we also test the performance when Transformer Encoder is removed. The overall accuracy decreases to 94.3% but it is still higher than most existing methods. In model E, we test the performance when all SA modules are removed. The accuracy decreases to 92.8% which is still better than the state-of-the-art rotation invariant method [42] thanks to the well designed RISPs which capture sufficient surface structures. We also visualize the global feature vector before classification as shown in Fig. 5. We normalize the feature values to $[0, 1]$ and plot the histograms. We can see that by turning on the SA modules, the feature values become more evenly distributed and have less zeros because SA mechanism can reweight the features making the features more effective.

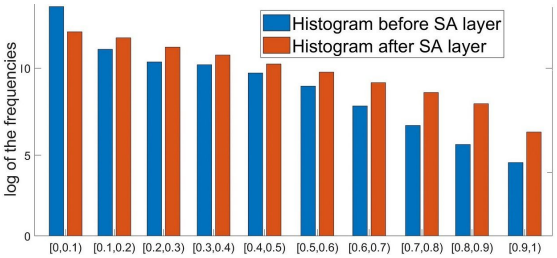

Fig. 5: Histogram comparison for normalized feature values without and with self-attention layers.

Network Efficiency. In the experiments, we acknowledge that our method employs a higher number of parameters. Thus, it is imperative to conduct a comprehensive analysis of network efficiency during both the training and testing stages. We list the trainable parameters, FLOPs and running time comparisons in Tab. 7. We compare the performance with four recent works: PointNet++ [25], RIConv [41], RI-GCN [15], RIConv++ [42].

Table 7: Trainable parameters, FLOPs and timing comparisons on ModelNet40. All are tested on the same platform. Time is measured per batch (batch size=16).

Methods	Params	FLOPs	Time (Train/Infer)
PointNet++ [25]	1.41M	0.86G	0.145s / 0.129s
RIConv [41]	0.70M	0.92G	0.121s / 0.092s
RI-GCN [15]	4.19M	1.24G	0.148s / 0.112s
RIConv++ [42]	0.42M	0.72G	0.074s / 0.029s
Ours	13.96M	1.12G	0.114s / 0.049s
Ours w/o TR	1.35M	1.11G	0.105s / 0.047s

From the results, we can see that the number of parameters mainly comes from the transformer architectures used in RISurConv. When we turn off all

the transformer component, our method only takes up 1.35M parameters. More parameters do not mean longer inference time. It depends on lots of things including depth, parameter, number of operations etc. Our approach runs faster than most existing methods. It takes more time for training because more epochs are needed for self attention layers to converge. This is a common phenomenon of transformer based methods. The accuracy is also not due to the complexity model. As is shown in the ablation studies, by removing transformer encoder, it still achieves state-of-the-art performance due to the effectiveness of RISurConv.

Number of Local Surfaces. Local triangle surfaces used in Sec. 3 are the base for rotation invariant surface property (RISP) construction. Thus, it is important to analyse the effects of different number of surfaces. We conduct performance tests by setting the number of local surfaces to 1, 2, 3, and 4, resulting in overall accuracies of 94.7, 96.0, 91.9, and 89.8 respectively. It worth noting that for more surfaces, we can define more RISPs. From the results, we see the accuracy drops a bit when single surface is constructed. This is because smaller area cannot capture sufficient surface structures and less effective features can lower the performance. When we increase the number of surfaces to 3 and 4, the performance decreases. This shows that more surfaces can result in inferior performance because we construct triangle surfaces just to approximate the true surface. More surfaces may mess up the surface structure and affect the performance. Take Fig. 6 for example, suppose we are constructing the local surfaces ($\triangle px_ix_{i+1}$ $\triangle px_ix_{i-1}$) with regard to the neighbor point x_i, adding one more triangle $\triangle px_ix_{i+2}$ will result in distorted/non-manifold surfaces. Thus, we set number of surfaces as 2 throughout the paper.

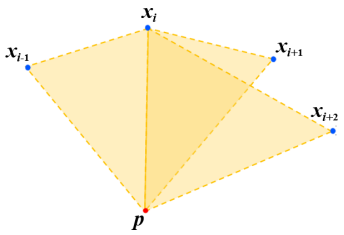

Fig. 6: Distorted surface caused by 3 triangles.

7 Discussion

Limitation. Though effective, RISurConv may require longer training time (e.g. more epochs) to converge due to large number of training parameters (Tab. 1). This is a common issue of transformer. But this does not affect the inference speed as shown in the ablation study (Tab. 7).

Conclusion. We have presented a new framework for achieving rotation invariance on 3D point cloud. We construct local triangle surfaces to better represent the local surface structures based on which we design highly expressive rotation invariant surface properties. We integrate the properties into an architecture named RISurConv to extract refined rotation invariant features. Based on RISurConv, we finally build up effective rotation invariant neural networks for 3D point cloud classification and segmentation with supreme performance achieved not only closing the performance gap with non-rotation invariant approaches but also surpassing the state-of-the-art methods by a large margin.

Acknowledgements

This research is supported by the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 1 grant (MSS23C010), Ningbo 2025 Science & Technology Innovation Major Project (No. 2022Z072, No.2023Z044), and Key Research & Development Plan of Zhejiang Province (2024C01017).

References

1. Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3d semantic parsing of large-scale indoor spaces. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1534–1543 (2016)
2. Chang, A.X., Funkhouser, T.A., Guibas, L.J., Hanrahan, P., Huang, Q.X., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: Shapenet: An information-rich 3d model repository. arXiv preprint arXiv:1512.03012 (2015)
3. Chen, C., Li, G., Xu, R., Chen, T., Wang, M., Lin, L.: Clusternet: Deep hierarchical cluster network with rigorously rotation-invariant representation for point cloud analysis. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4994–5002 (2019)
4. Deng, H., Birdal, T., Ilic, S.: Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors. In: Eur. Conf. Comput. Vis. pp. 602–618 (2018)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (2018)
6. Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al.: An image is worth 16x16 words: Transformers for image recognition at scale. In: Int. Conf. Learn. Represent. (2020)
7. Esteves, C., Allen-Blanchette, C., Makadia, A., Daniilidis, K.: Learning so (3) equivariant representations with spherical cnns. In: Eur. Conf. Comput. Vis. pp. 52–68 (2018)
8. Esteves, C., Xu, Y., Allen-Blanchette, C., Daniilidis, K.: Equivariant multi-view networks. In: Int. Conf. Comput. Vis. pp. 1568–1577 (2019)
9. Guo, M.H., Cai, J.X., Liu, Z.N., Mu, T.J., Martin, R.R., Hu, S.M.: Pct: Point cloud transformer. Computational Visual Media **7**, 187–199 (2021)
10. Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., Bennamoun, M.: Deep learning for 3d point clouds: A survey. IEEE Trans. Pattern Anal. Mach. Intell. **43**(12), 4338–4364 (2020)
11. Han, X.F., Jin, Y.F., Cheng, H.X., Xiao, G.Q.: Dual transformer for point cloud analysis. IEEE Trans. Multimedia (2022)
12. Hua, B.S., Pham, Q.H., Nguyen, D.T., Tran, M.K., Yu, L.F., Yeung, S.K.: Scenenn: A scene meshes dataset with annotations. In: International Conference on 3D Vision. pp. 92–101 (2016)
13. Huang, Q., Wang, W., Neumann, U.: Recurrent slice networks for 3d segmentation of point clouds. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 2626–2635 (2018)
14. Kim, J., Jung, W., Kim, H., Lee, J.: Cycnn: a rotation invariant cnn using polar mapping and cylindrical convolution layers. arXiv preprint arXiv:2007.10588 (2020)
15. Kim, S., Park, J., Han, B.: Rotation-invariant local-to-global representation learning for 3d point cloud. Adv. Neural Inform. Process. Syst. **33**, 8174–8185 (2020)

16. Li, X., Li, R., Chen, G., Fu, C.W., Cohen-Or, D., Heng, P.A.: A rotation-invariant framework for deep point cloud analysis. IEEE Trans. Vis. Comput. Graph. (2021)
17. Li, Y., Bu, R., Sun, M., Wu, W., Di, X., Chen, B.: Pointcnn: Convolution on x-transformed points. In: Adv. Neural Inform. Process. Syst. pp. 820–830 (2018)
18. Liu, X., Han, Z., Liu, Y.S., Zwicker, M.: Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network. In: AAAI. vol. 33, pp. 8778–8785 (2019)
19. Liu, X., Han, Z., Liu, Y.S., Zwicker, M.: Fine-grained 3d shape classification with hierarchical part-view attention. IEEE Trans. Image Process. **30**, 1744–1758 (2021)
20. Liu, Y., Fan, B., Xiang, S., Pan, C.: Relation-shape convolutional neural network for point cloud analysis. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 8895–8904 (2019)
21. Maturana, D., Scherer, S.: Voxnet: A 3d convolutional neural network for real-time object recognition. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 922–928 (2015)
22. Poulenard, A., Rakotosaona, M.J., Ponty, Y., Ovsjanikov, M.: Effective rotation-invariant point cnn with spherical harmonics kernels. International Conference on 3D Vision (2019)
23. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 652–660 (2017)
24. Qi, C.R., Su, H., Nießner, M., Dai, A., Yan, M., Guibas, L.J.: Volumetric and multi-view cnns for object classification on 3d data. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 5648–5656 (2016)
25. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: Adv. Neural Inform. Process. Syst. pp. 5105–5114 (2017)
26. Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B.: Pointnext: Revisiting pointnet++ with improved training and scaling strategies. Adv. Neural Inform. Process. Syst. **35**, 23192–23204 (2022)
27. Rao, Y., Lu, J., Zhou, J.: Spherical fractal convolutional neural networks for point cloud recognition. In: IEEE Conf. Comput. Vis. Pattern Recog. (2019)
28. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241 (2015)
29. Shen, Y., Feng, C., Yang, Y., Tian, D.: Mining point cloud local structures by kernel correlation and graph pooling. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4548–4557 (2018)
30. Thomas, H.: Rotation-invariant point convolution with multiple equivariant alignments. In: International Conference on 3D Vision (3DV). pp. 504–513 (2020)
31. Uy, M.A., Pham, Q.H., Hua, B.S., Nguyen, D.T., Yeung, S.K.: Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In: Int. Conf. Comput. Vis. (2019)
32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Adv. Neural Inform. Process. Syst. **30** (2017)
33. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Trans. Graph. (2019)
34. Wu, X., Lao, Y., Jiang, L., Liu, X., Zhao, H.: Point transformer v2: Grouped vector attention and partition-based pooling. Adv. Neural Inform. Process. Syst. **35**, 33330–33342 (2022)

35. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1912–1920 (2015)
36. Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 1912–1920 (2015)
37. Xie, S., Liu, S., Chen, Z., Tu, Z.: Attentional shapecontextnet for point cloud recognition. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 4606–4615 (2018)
38. Xu, Y., Fan, T., Xu, M., Zeng, L., Qiao, Y.: Spidercnn: Deep learning on point sets with parameterized convolutional filters. In: Eur. Conf. Comput. Vis. pp. 87–102 (2018)
39. Yu, X., Tang, L., Rao, Y., Huang, T., Zhou, J., Lu, J.: Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In: IEEE Conf. Comput. Vis. Pattern Recog. pp. 19313–19322 (2022)
40. Zhang, Z., Hua, B.S., Chen, W., Tian, Y., Yeung, S.K.: Global context aware convolutions for 3d point cloud understanding. In: International Conference on 3D Vision (2020)
41. Zhang, Z., Hua, B.S., Rosen, D.W., Yeung, S.K.: Rotation invariant convolutions for 3d point clouds deep learning. In: International Conference on 3D Vision. pp. 204–213 (2019)
42. Zhang, Z., Hua, B.S., Yeung, S.K.: Riconv++: Effective rotation invariant convolutions for 3d point clouds deep learning. Int. J. Comput. Vis. **130**(5), 1228–1243 (2022)
43. Zhao, H., Jiang, L., Jia, J., Torr, P.H., Koltun, V.: Point transformer. In: Int. Conf. Comput. Vis. pp. 16259–16268 (2021)
44. Zhao, Y., Birdal, T., Lenssen, J.E., Menegatti, E., Guibas, L., Tombari, F.: Quaternion equivariant capsule networks for 3d point clouds. In: Eur. Conf. Comput. Vis. pp. 1–19. Springer (2020)