PRIN/SPRIN: On Extracting Point-wise Rotation Invariant Features

Yang You, Yujing Lou, Ruoxi Shi, Qi Liu, Yu-Wing Tai,
Lizhuang Ma, Weiming Wang, Cewu Lu *Member, IEEE*

Abstract—Point cloud analysis without pose priors is very challenging in real applications, as the orientations of point clouds are often unknown. In this paper, we propose a brand new point-set learning framework PRIN, namely, **P**oint-wise **R**otation **I**nvariant **N**etwork, focusing on rotation invariant feature extraction in point clouds analysis. We construct spherical signals by Density Aware Adaptive Sampling to deal with distorted point distributions in spherical space. Spherical Voxel Convolution and Point Re-sampling are proposed to extract rotation invariant features for each point. In addition, we extend PRIN to a sparse version called SPRIN, which directly operates on sparse point clouds. Both PRIN and SPRIN can be applied to tasks ranging from object classification, part segmentation, to 3D feature matching and label alignment. Results show that, on the dataset with randomly rotated point clouds, SPRIN demonstrates better performance than state-of-the-art methods without any data augmentation. We also provide thorough theoretical proof and analysis for point-wise rotation invariance achieved by our methods. The code to reproduce our results will be made publicly available.

Index Terms—Point cloud, object analysis, rotation invariance, feature learning

✦

1 INTRODUCTION

Deep learning on point clouds has received tremendous interest in recent years. Since depth cameras capture point clouds directly, efficient and robust point processing methods like classification, segmentation and reconstruction have become key components in real-world applications. Robots, autonomous cars, 3D face recognition and many other fields rely on learning and analysis of point clouds.

Existing works like PointNet [1] and PointNet++ [2] have achieved remarkable results in point cloud learning and shape analysis. But they focus on objects with canonical orientations. In real applications, these methods fail to be applied to rotated shape analysis since the model orientation is often unknown as a priori, as shown in Figure 1. In addition, existing frameworks require massive data augmentation to handle rotations, which induces unacceptable computational cost.

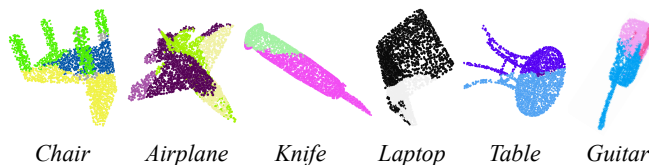*Chair   Airplane   Knife   Laptop   Table   Guitar*

Fig. 1. **PointNet++ part segmentation results on rotated shapes.** When trained on objects with canonical orientations and evaluated on rotated ones, PointNet++ is unaware of their orientations and fails to segment their parts out.

- *Yang You, Yujing Lou, Ruoxi Shi, Qi Liu, Lizhuang Ma, Weiming Wang, Cewu Lu are with Shanghai Jiao Tong University, Shanghai, 200240, China. Yu-Wing Tai is with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Hong Kong. Cewu Lu is also the member of Qing Yuan Research Institute, Shanghai Qizhi Research Institute and MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University, China.*

- *Weiming Wang and Cewu Lu are the corresponding authors. E-mail: wangweiming@sjtu.edu.cn, lucewu@sjtu.edu.cn.*

Spherical CNN [3] and a similar method [4] try to solve this problem and propose a global feature extracted from continuous meshes, while they are not suitable for point clouds since they project 3D meshes onto their enclosing spheres using a ray casting scheme. Difficulty lies in how to apply spherical convolution in continuous domain to sparse point clouds. Besides, by projecting onto a unit sphere, their method is limited to processing convex shapes, ignoring any concave structures. Therefore, we propose a point-wise rotation invariant network (PRIN) to handle these problems. Firstly, we observe the discrepancy between unit spherical space and Euclidean space, and propose Density Aware Adaptive Sampling (DAAS) to avoid biases. Secondly, we come up with Spherical Voxel Convolution (SVC) without loss of rotation invariance, which is able to capture any concave information. Furthermore, we propose Point Re-sampling module that helps to extract rotation invariant features for each point.

PRIN is a network that directly takes point clouds with random rotations as the input, and predicts both categories and point-wise segmentation labels without data augmentation. It absorbs the advantages of both Spherical CNN and PointNet-like network by keeping rotation invariant features, while maintaining a one-to-one point correspondence between the input and output. PRIN learns rotation invariant features at spherical voxel grids. Afterwards, these features could be aggregated into a global descriptor or per-point descriptor to conduct model classification or part segmentation, respectively. We rigorously prove the point-wise rotation invariance of PRIN under certain conditions.

In addition, we extend our PRIN framework and propose a sparse version of PRIN called SPRIN. SPRIN considers the input as the Dirac delta function and gives rotation invariant features when the filter is constant on the left coset of z-axis rotation.

We experimentally compare PRIN and SPRIN with various state-of-the-art approaches on the benchmark dataset: ShapeNet part dataset [5] and ModelNet40 [6]. Additionally, both PRIN and SPRIN can be applied to 3D point matching and label alignment. Both PRIN and SPRIN exhibit remarkable performance. SPRIN

also achieves the state-of-the-art performance on part segmentation.

The key contributions of this paper are as follows:

- We design two novel deep network processing pipelines PRIN/SPRIN that extracts rotation invariant point-level features.
- Three key modules: Density Aware Adaptive Sampling (DAAS), Spherical Voxel Convolution (SVC) and Point Re-sampling are proposed for PRIN.
- We propose a special spherical voxel convolution and prove that it is rotation equivariant. In addition, we extend this convolution to the sparse domain and develop a sparse version of PRIN called SPRIN. Rigorous proof of point-wise rotation invariance is given for both PRIN and SPRIN frameworks.
- We show that PRIN/SPRIN can be used for point cloud part segmentation, classification, 3D point matching and label alignment under different rotations. SPRIN achieves the state-of-the-art performance on part segmentation.

A preliminary version of this work was presented in AAAI2020 [7]. In this study, we extend it in two fundamental aspects. First, we provide thorough theoretical analysis and proof for the point-wise rotation invariance in PRIN, and some necessary filter conditions are proposed. With this revised version of spherical voxel convolution, we achieve much better results than our previous work. Second, we extend PRIN to sparse domain, where the Dirac delta function is leveraged and sparse correlation is proposed with guaranteed rotation invariance. The sparse version of PRIN, which is named SPRIN, achieves the state-of-the-art performance on ShapeNet part segmentation.

2 RELATED WORK

2.1 Rotation invariant Features

Rotation Invariance is often regarded as a preliminary to the success of template matching and object detection, in both 2D and 3D domains.

The development of rotation invariant features from geometries could be retrospected to manual designed features, including Structural Indexing [8], Signature of Histogram Orientations (SHOT) [9], CGF [10] and Snapshots [11]. They construct a local reference frame (LRF) which aligns the model into its canonical pose in order to extract rotation invariant point features. However, these methods depend on local surface variations, therefore are sensitive to noise and point densities. Besides, these descriptors rely on delicate hand-craft design, and could only capture low-level geometric features. For a more complete review on traditional feature descriptors, we refer to Guo *et al.* [12].

Recently, some papers consider generalizations of 2D CNNs that exploit larger groups of symmetries [13], [14], including the 2D/3D rotation group [15]. Spherical CNN [3] and a similar method [4] propose to extract global rotation invariant features from continuous meshes, while they are not suitable for point clouds since they project 3D meshes onto their enclosing spheres using a ray casting scheme.

In parallel to group invariant/equivalent convolutions, some researchers incorporate rotation invariant point-level convolutions. SRINet [16] proposes the point projection feature, which is invariant to the rotation of the input point cloud. It introduces an efficient key point descriptor to assign each point with different

response and help recognize the overall geometry. Poulenard *et al.* [17] employ a spherical harmonics based kernel at different layers of a point-based PCNN architecture. Kim *et al.* [18] take advantages of multi-level abstraction based on graph convolutional neural networks, which constructs a descriptor hierarchy to encode rotation invariant shape information of an input object in a bottom-up manner. Zhang *et al.* [19] use low-level rotation invariant geometric features such as distances and angles to design a convolution operator for point cloud learning. PPF-FoldNet [20] obtains unsupervised rotation invariant point-wise features via an auto encoder-decoder structure. Li *et al.* [21] present a network architecture to embed rotation invariant representations into features, encoding local relations between points and their neighbors, and the global shape structure. Using graphs as point cloud representation is another way to achieve rotation invariance like Zhang *et al.* [22] and Wang *et al.* [23].

2.2 Rotation Equivariance in Point Clouds

Rotation Equivariance is closely related to rotation invariant point features, where feature locations are equivalently transformed according to the input rotation. Tensor field networks [24] build filters from spherical harmonics and are proven to be equivariant to rotations. SE(3)-Transformers [25] combine graph networks and self-attention mechanisms to fulfill the goal of rotation-translation equivariance. However, both Tensor field networks and SE(3)-Transformers do not scale well with input points and fail to be applied to large point clouds (e.g. 2048 points). In contrast, our SPRIN can easily fit 2048 points with a large batch size, due to the introduced sparse correlation.

Cohen *et al.* [3], Esteves *et al.* [4] and Cruz [26] all decomposes SO(3) group with irreducible harmonic orthogonal basis, which is proven to strictly equivariant to SO(3) rotations. Quaternion equivariant capsule networks [27] disentangles geometry from pose with dynamic routing algorithm from capsule networks. Rotation Equivariance can be also achieved by giving a rotation invariant representation for each individual point, such as Khoury *et al.* [10] and Gojcic *et al.* [28].

2.3 Deep Learning on 3D Shapes

As the consequence of success in deep learning, various methods have been proposed for better understanding 3D models. Convolutional neural networks are applied to volumetric data since its format is similar to pixels in an image and easy to transfer to existing frameworks. 3D ShapeNet [6] and VoxNet [29] are pioneers introducing fully-connected networks to voxels. However, dealing with voxel data requires large memory and the sparsity of point sets also makes it challenging to extract particular features from big data. As a consequence, MinkowskiNet [30] and Submanifold Sparse Convolutional Networks [31] try to solve this by conducting sparse convolutions near existing points. Our SPRIN also takes inspiration from these works to improve the scalability of PRIN.

Another research branch is multi-view methods. MVCNN [32] renders 3D models into multi-view images and propagates these images into traditional convolutional neural networks. These approaches are limited to simple tasks like classification and not suitable for 3D part segmentation, key point matching or other tasks.

Dealing with point clouds directly is another popular branch, among which PointNet [1] is the pioneer in building a general
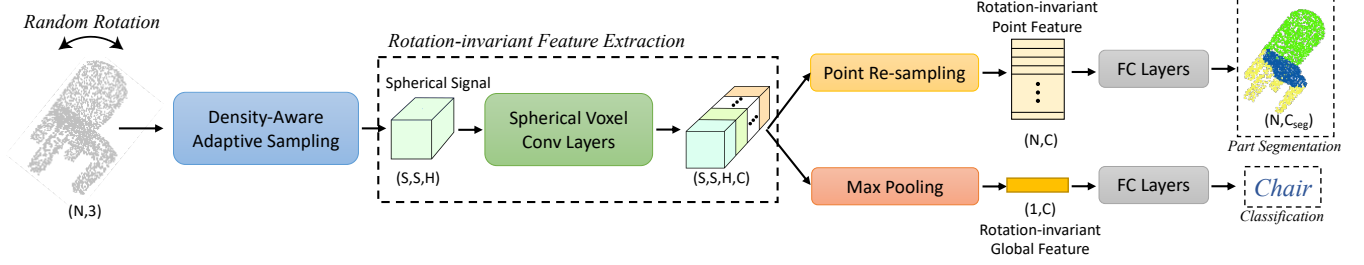
Fig. 2. **PRIN Architecture.** Our network takes sparse points as input, and then uses Density-Aware Adaptive Sampling to transform the signals into spherical voxel grids. The spherical voxel signals are then passed through several Spherical Voxel Convolution layers, ending with a feature at each spherical voxel grid. Any point feature can be extracted by point re-sampling, which is used to do point-wise part segmentation. All these voxel features can also be max-pooled to get a global feature, which is suitable for classification.

framework for learning point clouds. Since then, many networks are proposed to learn from point clouds. PointNet++ [2] extends PointNet by introducing a hierarchical structure. RSNet [33] combines a novel slice pooling layer, Recurrent Neural Network (RNN) layers, and a slice unpooling layer, to better propagate features among different parts. SplatNet [34] uses sparse bilateral convolutional layers to enable hierarchical and spatially-aware feature learning. DGCNN [35] proposes EdgeConv, which explicitly constructs a local graph and learns the embeddings for the edges in both Euclidean and semantic spaces. As a follow-up, LDGCNN [36] links the hierarchical features from different dynamic graphs based on DGCNN. SO-Net [37] models the spatial distribution of point cloud by building a Self-Organizing Map (SOM) and its receptive field can be systematically adjusted by conducting point-to-node k nearest neighbor search. SpiderCNN [38] designs the convolutional filter as a product of a simple step function that captures local geodesic information and a Taylor polynomial that ensures the expressiveness. DeepSets [39] provides a family of functions which are permutation-invariant and gives a competitive result on point clouds. Point2Sequence [40] uses a recurrent neural network (RNN) based encoder-decoder structure, where an attention mechanism is proposed to highlight the importance of different area scales. Kd-Network [41] utilizes kd-tree structures to form the computational graph, which learns from point clouds hierarchically. However, few of them are robust to random rotations, making it hard to apply them to real applications.

3　PRELIMINARIES

3.1　Unit Sphere: S^2

A point s in a two-dimensional sphere can be uniquely described by its azimuthal and polar angles: (α, β), where $\alpha \in [0, 2\pi], \beta \in [0, \pi]$. Furthermore, if we denote $n = (0, 0, 1)^T$ as the North Pole, the coordinate of s is given by $Z(\alpha)Y(\beta)n$, where $Z(\cdot)$ and $Y(\cdot)$ represent the rotation matrix around z and y axes, respectively. For more details of this space, we refer the reader to [42].

3.2　Unit Spherical Space: $S^2 \times H$

Any point x in unit ball can be uniquely parameterized by an augmented spherical coordinate: $(s(\alpha, \beta), h)$, where $s(\alpha, \beta) \in S^2$ denotes its location when projected to unit sphere, and $h \in H = [0, 1]$ represents the radial distance to the origin. It is obvious that this parametrization is bijective.

3.2.1　Rotation Transformation

Consider an arbitrary rotation transformation Q applied to a point $x \in S^2 \times H$:

$$\begin{aligned} Qx(\alpha, \beta, h) &= (Qs(\alpha, \beta), h) \\ &= (QZ(\alpha)Y(\beta)n, h). \end{aligned} \quad (1)$$

Intuitively, Q rotates the point around the origin, while keeping its radial distance towards the origin.

3.3　3D Rotation Group: $SO(3)$

The 3D rotation group, often denoted $SO(3)$, which is termed "special orthogonal group", is the group of all rotations about the origin of three-dimensional Euclidean space $\mathbb{R}^3$ under the operation of composition. Almost every element (except for singularities) in $SO(3)$ can be uniquely parameterized by ZYZ-Euler angles [43]: (α, β, γ), where $\alpha \in [0, 2\pi], \beta \in [0, \pi]$, and $\gamma \in [0, 2\pi]$. In matrix form, for any element $R \in SO(3)$, $R(\alpha, \beta, \gamma) = Z(\alpha)Y(\beta)Z(\gamma)$. This parametrization is also bijective almost everywhere.

3.3.1　Rotation Transformation

Consider an arbitrary rotation transformation Q applied to another rotation $R \in SO(3)$, which is a transformation composition:

$$QR(\alpha, \beta, \gamma) = QZ(\alpha)Y(\beta)Z(\gamma). \quad (2)$$

4　PRIN: AN EXACT POINT-WISE ROTATION INVARIANT NETWORK

In this section, we discuss the development of our point-wise rotation invariant algorithm. In Section 4.2, we propose a density aware adaptive sampling module to correct the distortion in spherical voxels. In Section 4.3, we propose a special spherical voxel convolution and prove that it is rotation equivariant (e.g., point-wise rotation invariant) theoretically. Besides, we also derive a sufficient condition on convolutional filters to ensure this rotation equivariance.

4.1　Problem Statement

Given a set of unordered points $\mathcal{X} = \{x_i\}$ with $x_i \in \mathbb{R}^{d^{in}}$ and $i = 1, \cdots, N$, where N denotes the number of input points and d^{in} denotes the dimension of input features at each point, which can be positions, colors, etc. Our goal is to produce a set of point-wise features $\mathcal{Y} = \{y_i\}$ with $y_i \in \mathbb{R}^{d^{out}}$ and $i = 1, \cdots, N$,

which are invariant to input orientations. PRIN can be modeled as a rotation invariant function $\mathcal{F} : \mathcal{X} \mapsto \mathcal{Y}$. Although the problem and the method developed are general, we focus on the case $d_{in} = 3$ using only Euclidean coordinates as the input. To implement the function $\mathcal{F}$, we design mainly three modules: (1) Density Aware Adaptive Sampling (DAAS) module $\Gamma : \mathbb{R}^{N \times 3} \to \mathbb{R}^{S^2 \times H}$ that constructs spherical signals; (2) Spherical Voxel Convolution (SVC) module $\Phi : \mathbb{R}^{S^2 \times H \times C_{in}} \to \mathbb{R}^{S^2 \times H \times C_{out}}$ that extracts rotation invariant features; (3) Point Re-sampling module $\Lambda : \mathbb{R}^{S^2 \times H \times C_{out}} \to \mathbb{R}^{N \times C_{out}}$ that re-samples points from spherical signals. We will explain these modules in the following sections and the whole pipeline is shown in Figure 2.

4.2　Density Aware Adaptive Sampling

In this step, the objective is to build spherical signals from irregular point clouds. Nonetheless, if we sample point clouds uniformly into regular spherical voxels, we will meet a problem: points around the pole appear to be more sparse than those around the equator in spherical coordinates, which brings a bias to the resulting spherical voxel signals. To address this problem, we propose Density Aware Adaptive Sampling (DAAS). DAAS leverages a non-uniform filter to adjust the density discrepancy brought by spherical coordinates, thus reducing the bias. This process can be modeled as $\Gamma : \mathbb{R}^{N \times 3} \to \mathbb{R}^{S^2 \times H}$.

4.2.1　Spherical Distortion

Specifically, we divide unit spherical space $S^2 \times H$ into spherical voxels, which are indexed by $(i, j, k) \in I \times J \times K$, where $I \times J \times K$ is the spatial resolution, also known as the bandwidth [44]. Each spherical voxel is represented by its center $(s(a_i, b_j), c_k)$ with $a_i = \frac{i}{I} \cdot 2\pi$, $b_j = \frac{j}{J} \cdot \pi$, and $c_k = \frac{k}{K}$. The division of unit spherical space is shown in the left of Figure 3.

Then we sample the input signal into a tensor of equal-sized measures of Euler angles (α, β, γ) in $SO(3)$, with the bijective mapping between $SO(3)$ and $S^2 \times H$ (Theorem 4.2). They are leveraged for discrete spherical voxel convolution discussed later in Section 4.3. This representation is shown in the right of Figure 3.

However, such an equal-angle discretization introduces a non-uniform or distorted distribution of the input signal in Euclidean space. This is illustrated in the middle of Figure 3. Since the size of the spherical voxel is smaller around the pole than on the equator, the input signal gets distorted around the pole when we stretch non-equal-sized spherical voxels to equal-sized angle bins.

To address this issue and obtain an unbiased estimate of the input signal, Density Aware Adaptive Sampling (DAAS) is proposed. Formally, we map each point $x_n \in \mathcal{X}$ from Euclidean space $\mathbb{R}^3$ to unit spherical space $S^2 \times H$ by calculating its spherical coordinates $x_n = (s(\alpha_n, \beta_n), h_n)$, and then calculate the spherical signal $f : S^2 \times H \to \mathbb{R}$ as follows:

$$f(a_i, b_j, c_k) = \frac{\sum\limits_{n=1}^{N} w_n \cdot (\xi - \|h_n - c_k\|)}{\sum\limits_{n=1}^{N} w_n}, \tag{3}$$

where w_n is a normalizing factor that is defined as

$$
\begin{aligned}
w_n = &\mathbf{1}(\|\alpha_n - a_i\| < \xi) \\
&\cdot \mathbf{1}(\|\beta_n - b_j\| < \eta\xi) \\
&\cdot \mathbf{1}(\|h_n - c_k\| < \xi),
\end{aligned}
\tag{4}
$$

where ξ is a predefined threshold filter width and η is the Density Aware Adaptive Sampling Factor. f can be viewed as an unbiased version of the empirical distribution of x, except we use $(\xi - \|h_n - c_k\|)$ instead of Dirac delta because it captures information along the H axis, which is orthogonal to S^2 and is invariant under random rotations.

Theorem 4.1 (Density Aware Adaptive Sampling Factor). $\eta = sin(\beta)$ is the Density Aware Adaptive Sampling Factor, accounting for distorted spherical voxels in Euclidean spaces.

Proof. To get the relationship between the differential spherical volumes and rectangular (e.g., Euclidean) volumes, we need to calculate the Jacobian of the transformation from spherical coordinates to Euclidean coordinates. Denote some Euclidean coordinate as (x, y, z) and its corresponding spherical coordinate as (α, β, h), we have [45],

$$
\begin{aligned}
x &= h\sin(\beta)\cos(\alpha), \\
y &= h\sin(\beta)\sin(\alpha), \\
z &= h\cos(\beta).
\end{aligned}
\tag{5}
$$

Denote the Jacobian as J:

$$J = \begin{bmatrix} \frac{\partial x}{\partial \alpha} & \frac{\partial x}{\partial \beta} & \frac{\partial x}{\partial h} \\ \frac{\partial y}{\partial \alpha} & \frac{\partial y}{\partial \beta} & \frac{\partial y}{\partial h} \\ \frac{\partial z}{\partial \alpha} & \frac{\partial z}{\partial \beta} & \frac{\partial z}{\partial h} \end{bmatrix} = h^2 \sin(\beta), \tag{6}$$

This involves h and $\sin(\beta)$. h does reflect the volume difference of spherical voxels when placed at different distances from the ball center. However, this change is isotropic and does not cause the volume distortion between the equator and the pole, which is shown in Figure 3. The other factor $\sin(\beta)$ is therefore the only reason for distorted spherical voxels. In other words, if we normalize the spherical voxel by setting $h = 1$, we get $J_{|h=1} = sin(\beta)$. □
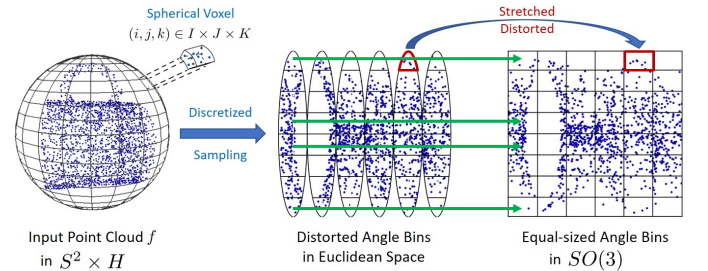

Fig. 3. Spherical distortion. An input point cloud is first sampled into discretized angle bins, though such equal-sized angle bins are distorted in Euclidean space. Therefore, we propose Density Aware Adaptive Sampling (DAAS).

4.2.2　Comparison with Other Spherical Representations

To recap, we create equal-sized angle bins/measures from distorted spherical voxels to form a compact 3D tensor in $SO(3)$. This tensor is later used by discrete spherical voxel convolution to achieve point-wise rotation invariance.

This is different from what is proposed in SPH3D-GCN [46], which also leverages spherical voxel. In SPH3D-GCN, the authors do not consider rotation invariance, and utilize a simple uniform sampling of input signals. This is okay as long as the bias

introduced by distorted spherical voxels is consistent in both training and testing. However, if an arbitrary rotation is introduced during testing but no rotation when training, the bias is inevitable, and their method will give an inferior result. This is also confirmed in our ablation studies, referred to as "uniform sampling".

Besides, one may consider using HealPix [47] that produces a subdivision of a spherical surface in which each pixel covers the same surface area as every other pixel. Though it produces equal sized surface area for each pixel, the center of each pixel is not necessarily aligned with the grid of general discrete Fast Fourier Transforms (FFTs) on the rotation group [48], which is required for a fast computation of the spherical voxel convolution discussed in Section 4.3. HealPix, though uniformly discretizes voxels in Euclidean space, produces non-uniform discretization in $SO(3)$ space, making the separation of variables impossible. As a result, FFTs are not applicable for HealPix. For more details about the requirement for FFTs on the rotation group, we refer the reader to Chapter 3 of [48].

4.3 Spherical Voxel Convolution

Given some spherical voxel signal $f : S^2 \times H \to \mathbb{R}^{C_{in}}$, we introduce Spherical Voxel Convolution (SVC) to further refine the features, while keeping it rotation invariant. This step implements the operator $\Phi : \mathbb{R}^{S^2 \times H \times C_{in}} \to \mathbb{R}^{S^2 \times H \times C_{out}}$.

Compared with Spherical CNN [4] where only spherical signals defined in S^2 get convolved. We propose the spherical voxel convolution, which takes spherical signals defined in $S^2 \times H$ as the input. To understand spherical voxel convolution, we first give some definitions and annotations. Here, we only consider functions with $C_{in} = C_{out} = 1$ to reduce clutter, while extensions to higher-dimension functions are straight-forward.

In order to define spherical voxel convolution, we need to first bridge $S^2 \times H$ and $SO(3)$ by several theorems and lemmas.

Theorem 4.2 (Bijection from $S^2 \times H$ to $SO(3)$). *There exists a bijective map (almost everywhere) $\mathcal{T} : S^2 \times H \to SO(3)$, such that $\mathcal{T}(s(\alpha, \beta), h) = R(\alpha, \beta, 2\pi h) = Z(\alpha)Y(\beta)Z(2\pi h)$.*

Proof. This map is onto because almost every element $R \in SO(3)$ can be parameterized by ZYZ-Euler angles [43]: (α, β, γ), and there is some $x = (s(\alpha, \beta), \frac{\gamma}{2\pi})$ such that $\mathcal{T}(x) = R(\alpha, \beta, \gamma)$. To prove that this map in injective, suppose $x_1 \neq x_2$, by the bijective parameterization of $S^2 \times H$, $(\alpha_1, \beta_1, h_1) \neq (\alpha_2, \beta_2, h_2)$, therefore $\mathcal{T}(x_1) \neq \mathcal{T}(x_2)$ by the bijective parameterization of $SO(3)$. $\square$

Lemma 4.3. *For any $x \in S^2 \times H$ and $Q \in SO(3)$, there exists some θ, such that:*

$$\mathcal{T}(Qx) = Q\mathcal{T}(x)Z(\theta), \tag{7}$$

where $Z(\theta)$ is the rotation around z axes by θ.

Proof. Writing left-hand side out and leverage Equation 1, we have:

$$\mathcal{T}(Qx(\alpha, \beta, h)) = \mathcal{T}(Qs(\alpha, \beta), h) \tag{8}$$
$$= \mathcal{T}(QZ(\alpha)Y(\beta)n, h), \tag{9}$$

Notice that $QZ(\alpha)Y(\beta)$ is a general rotation and can be uniquely parameterized as $Z(\alpha')Y(\beta')Z(\gamma')$. Substitute this into Equation 9:

$$\mathcal{T}(Qx) = \mathcal{T}(Z(\alpha')Y(\beta')Z(\gamma')n, h) \tag{10}$$
$$= \mathcal{T}(Z(\alpha')Y(\beta')n, h) \tag{11}$$
$$= Z(\alpha')Y(\beta')Z(2\pi h) \tag{12}$$
$$= QZ(\alpha)Y(\beta)Z(-\gamma')Z(2\pi h) \tag{13}$$
$$= QZ(\alpha)Y(\beta)Z(2\pi h)Z(-\gamma') \tag{14}$$
$$= Q\mathcal{T}(x)Z(-\gamma'). \tag{15}$$

Equation (11) follows from that rotations around z axes fix the North Pole; Equation (13) follows because $QZ(\alpha)Y(\beta) = Z(\alpha')Y(\beta')Z(\gamma')$; Equation (14) follows since rotations around z axes are commutative. $\square$

Definition 4.1 (Adjoint Function). We define the adjoint function of f as follows:

$$f_{\mathcal{T}}(x) = f(\mathcal{T}^{-1}(x)). \tag{16}$$

Remark. Notice that $f_{\mathcal{T}}$ is also the pull-back function [49] of the measurable push-forward map $\mathcal{T}$ (up to a constant):

$$\int_{SO(3)} f_{\mathcal{T}}(x)dx = \int_{SO(3)} f(\mathcal{T}^{-1}(x))dx \tag{17}$$
$$= \int_{S^2 \times H} f(y)d(\mathcal{T}(y)) \tag{18}$$
$$= 2\pi \int_{S^2 \times H} f(y)dy, \tag{19}$$

where we make use of $y := \mathcal{T}^{-1}(x)$ and Equation 19 follows by Theorem 4.2.

Now, we are ready to give a formal definition of spherical voxel convolution, by leveraging $SO(3)$ group convolution.

Definition 4.2 (Spherical Voxel Convolution). Spherical voxel convolution, evaluated at $p \in S^2 \times H$, is defined as:

$$[\psi \star f](p) = \int_\gamma [\psi_{\mathcal{T}} \star f_{\mathcal{T}}](\mathcal{T}(p)Z(\gamma))d\gamma$$
$$= \int_\gamma \int_{SO(3)} \psi_{\mathcal{T}}(R^{-1}\mathcal{T}(p))f_{\mathcal{T}}(RZ(\gamma))dRd\gamma. \tag{20}$$

$\psi, f : S^2 \times H \to \mathbb{R}$, where f is the input signal and ψ is the filter. Intuitively, our spherical voxel convolution is the corresponding adjoint $SO(3)$ convolution averaged over the coset defined by $Z(\gamma)$.

Definition 4.3 (Rotation Operator [3]). We introduce a rotation function operator L_Q:

$$[L_Q f](x) = f(Q^{-1}x), Q \in SO(3). \tag{21}$$

It follows that:

$$[L_Q f]_{\mathcal{T}}(x) = f_{\mathcal{T}}(\mathcal{T}(Q^{-1}x)) = f_{\mathcal{T}}(Q^{-1}\mathcal{T}(x)Z(\theta)). \tag{22}$$

Theorem 4.4 (**Main Result: Rotation Invariance**). *The spherical voxel convolution is point-wise rotation invariant: $[\psi \star L_Q f](Qp) = [\psi \star f](p)$, when $\psi_{\mathcal{T}}$ is constant on the right latitude: $\psi_{\mathcal{T}}(R) \equiv \psi_{\mathcal{T}}(RZ(\theta))$ for any R, θ.*

Proof. Suppose that the input point cloud is rotated by an arbitrary rotation Q, we have $p \to Qp$. As a consequence, the corresponding spherical signal is also rotated: $f \to L_Q f$, since f is sampled from the original point cloud. We are now ready to prove the rotation invariance by applying SVC to the rotated input:

$$[\psi \star L_Q f](Qp) \tag{23}$$

$$= \int_\gamma \int_{SO(3)} \psi_\mathcal{T}(R^{-1}\mathcal{T}(Qp)) f_\mathcal{T}(Q^{-1}RZ(\theta+\gamma))dRd\gamma \tag{24}$$

$$= \int_\gamma \int_{SO(3)} \psi_\mathcal{T}(R^{-1}Q\mathcal{T}(p)) f_\mathcal{T}(Q^{-1}RZ(\theta+\gamma))dRd\gamma \tag{25}$$

$$= \int_{SO(3)} \psi_\mathcal{T}(R^{-1}\mathcal{T}(p)) \int_\gamma f_\mathcal{T}(RZ(\theta+\gamma))d\gamma dR \tag{26}$$

$$= \int_{SO(3)} \psi_\mathcal{T}(R^{-1}\mathcal{T}(p)) \int_\gamma f_\mathcal{T}(RZ(\gamma))d\gamma dR \tag{27}$$

$$= \int_\gamma \int_{SO(3)} \psi_\mathcal{T}(R^{-1}\mathcal{T}(p)) f_\mathcal{T}(RZ(\gamma))dRd\gamma \tag{28}$$

$$= [\psi \star f](p). \tag{29}$$

Equation (25) follows from Lemma 4.3 and $\psi_\mathcal{T}(R) \equiv \psi_\mathcal{T}(RZ(\theta))$; Equation (26) follows from the right invariance of $SO(3)$ group integral [50]; Equation (27) is a result of corollary (2.1) in [50]. □

It is obvious that SVC extracts the same feature no matter how the input point cloud rotates, which ensures point-wise **rotation invariance**.

In practice, Spherical Voxel Convolution (SVC) can be efficiently computed by Fast Fourier Transform (FFT) [48]. Convolutions are implemented by first doing FFT to convert both the input and filters into spectral domain, then multiplying them and converting the results back to spatial domain, using Inverse Fast Fourier Transform (IFFT) [48].

4.3.1 Finding Filter ψ to Achieve Rotation Invariance

Notice that to achieve the rotation invariance stated in Theorem 4.4, we need a filter such that $\psi_\mathcal{T}(R) \equiv \psi_\mathcal{T}(RZ(\theta))$, for any R, θ. Leveraging the Euler angle representation of R, we can reform the condition as:

$$\psi_\mathcal{T}(Z(\alpha)Y(\beta)Z(\gamma)) \equiv \psi_\mathcal{T}(Z(\alpha)Y(\beta)Z(\gamma+\theta)), \forall \alpha, \beta, \gamma, \theta. \tag{30}$$

In other words,

$$\psi_\mathcal{T}(Z(\alpha)Y(\beta)Z(\gamma)) \equiv \psi_\mathcal{T}(Z(\alpha)Y(\beta)Z(\gamma')), \forall \alpha, \beta, \gamma, \gamma'. \tag{31}$$

During implementation, we regard $\psi_\mathcal{T}$ as a 3D tensor with dimensions corresponding to the three Euler angles, we have that $\psi_\mathcal{T}$ is constant on the third dimension.

To qualitatively illustrate this, we plot a sample signal f and its rotated version $L_Q f$, together with a filter ψ that satisfy the constraint 31 in Figure 4. We see that after our point-wise rotation invariant spherical voxel convolution, the output is rotation equivariant; and for each individual point, exact rotation invariance is achieved with an error up to the numeric precision.

4.4 Point Re-sampling

After Spherical Voxel Convolution (SVC), we re-sample features at the location of original points, with *Trilinear Interpolation* as our operator $\Lambda : \mathbb{R}^{S^2 \times H \times C} \to \mathbb{R}^{N \times C}$. Each point's feature is a weighted average of the nearest eight voxels, where the weights are inversely related to the distances to these spherical voxels. Formally, denote the target sampled point x as the spherical coordinate $(s(\alpha, \beta), h)$; the discrete feature map before point re-sampling as $\bar{F} \in \mathbb{R}^{I \times J \times K \times C}$, where $I \times J \times K$ are predefined spatial resolution for $S^2 \times H$; the point-wise feature map after re-sampling as $F \in \mathbb{R}^{N \times C}$, we have:

$$F(x, \cdot) = \tag{32}$$
$$\sum_{a,b,c \in \{0,1\}} \frac{w_{abc} \cdot \bar{F}(\lfloor \frac{I}{2\pi}\alpha \rfloor + a, \lfloor \frac{J}{\pi}\beta \rfloor + b, \lfloor Kh \rfloor + c, \cdot)}{\sum_{a',b',c' \in \{0,1\}} w_{a'b'c'}}, \tag{33}$$

where

$$w_{abc} = (1 - a + (2a-1)(\frac{I}{2\pi}\alpha - \lfloor \frac{I}{2\pi}\alpha \rfloor)) \tag{34}$$

$$\cdot (1 - b + (2b-1)(\frac{J}{\pi}\beta - \lfloor \frac{J}{\pi}\beta \rfloor)) \tag{35}$$

$$\cdot (1 - c + (2c-1)(Kh - \lfloor Kh \rfloor)), \tag{36}$$

and $\lfloor \cdot \rfloor$ is the integer floor function.

Finally, F are passed through several fully connected layers to get refined point-wise features.

4.5 Architecture

To summarize, our rotation invariant function $\mathcal{F}$ is a concatenation of the three modules/operators mentioned above: $\mathcal{Y} = \Lambda(\Phi(\Gamma(\mathcal{X})))$. We first transform points from Euclidean space to unit spherical space by operator Γ, and then conduct a rotation invariant feature transform by operator Φ, and finally re-sample the point features in Euclidean space by operator Λ.

After extracting point-wise rotation invariant features, we are able to do part segmentation by concatenating some fully connected layers. Our network could also realize object classification by placing a different head. In this case, we maxpool all the features in spherical voxels and pass this global feature through several fully connected layers, as shown in Figure 2.

5 SPRIN: SPARSE POINT-WISE ROTATION INVARIANT NETWORK

Though PRIN achieves point-wise rotation invariance by conducting DAAS and SVC, it is limited by the resolution of spherical voxel grids in $S^2 \times H$. In practice, the maximum resolution allowed by GPU memory is only 64^3, while a large quantity of spherical voxels are empty containing no points. This is extremely inefficient and motivates us to propose a sparse version of PRIN, by directly taking the point cloud without sampling into dense grids.

In this section, we provide a sparse version of PRIN, which directly operates on the original sparse point cloud, making it more efficient and achieve the state-of-the-art performance. We borrow the idea of spherical voxel convolution and propose sparse correlation on the input points. Besides, since our sparse correlation directly operates on the input points, there is no need to conduct DAAS or point re-sampling to convert input or output signals across different domains. Details are given below.
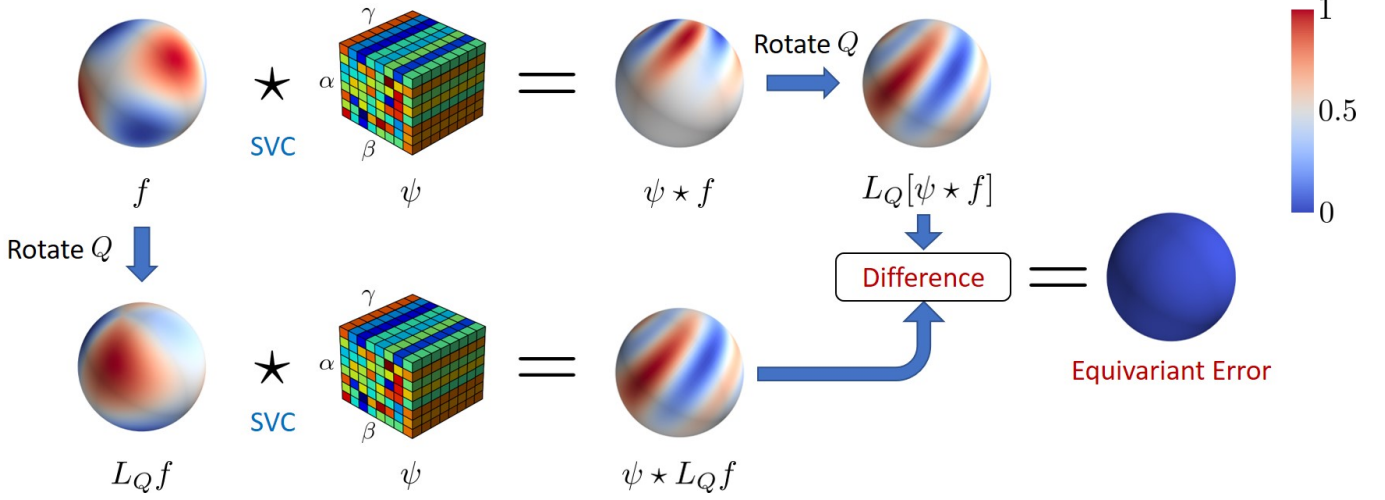
Fig. 4. Our spherical voxel convolution that achieves point-wise rotation invariance, a.k.a. rotation equivariant. Applying a rotation to the input signal and do Spherical Voxel Convolution is equivalent to applying the same rotation to the original convolution results. The absolute equivariant error is almost zero, with maximum 4.77×10^{-7} due to the floating point precision. Also notice that ψ is constant on the third dimension γ in order to fulfill the constraint 31. We slightly abuse the notation ψ for $\psi_\mathcal{T}$ for a better illustration.

								Arbitrary Rotation											No Rotation	
	avg. inst.	avg. cls.	air plane	bag	cap	car	chair	ear phone	guitar	knife	lamp	laptop	motor bike	mug	pistol	rocket	skate board	table	avg. inst.	avg. cls.
PointNet [1]	31.30	29.38	19.90	46.25	43.27	20.81	27.04	15.63	34.72	34.64	42.10	36.40	19.25	49.88	33.30	22.07	25.71	29.74	83.15	78.95
PointNet++ [2]	36.66	35.00	21.90	51.70	40.06	23.13	43.03	9.65	38.51	40.91	45.56	41.75	18.18	53.42	42.19	28.51	38.92	36.57	84.63	81.52
RS-Net [33]	50.38	32.99	38.29	15.45	53.78	33.49	60.83	31.27	9.50	43.48	57.37	9.86	20.37	25.74	20.63	11.51	30.14	66.11	84.92	81.41
PCNN [51]	28.80	31.72	23.46	46.55	35.25	22.62	24.27	16.67	32.89	39.80	52.18	38.60	18.54	48.90	27.83	27.46	27.60	24.88	85.13	81.80
SPLATNet [34]	32.21	38.25	34.58	68.10	46.96	19.36	16.25	24.72	88.39	52.99	49.21	31.83	17.06	48.56	21.20	34.98	28.99	28.86	84.97	82.34
DGCNN [35]	43.79	30.87	24.84	51.29	36.69	20.33	30.07	27.86	38.00	45.50	42.29	34.84	20.51	48.74	26.25	26.88	26.95	28.85	85.15	82.33
SO-Net [37]	26.21	14.37	21.08	8.46	1.87	11.78	27.81	11.99	8.34	15.01	43.98	1.81	7.05	8.78	4.41	6.38	16.10	34.98	84.83	81.16
SpiderCNN [38]	31.81	35.46	22.28	53.07	54.2	22.57	28.86	23.17	35.85	42.72	44.09	55.44	19.23	48.93	28.65	25.61	31.36	31.32	**85.33**	**82.40**
SHOT+PointNet [9]	32.88	31.46	37.42	47.30	49.53	27.71	28.09	16.34	9.79	27.66	37.33	25.22	16.31	50.91	25.07	21.29	43.10	40.27	32.75	31.25
CGF+PointNet [10]	50.13	46.26	50.97	70.34	60.44	25.51	59.08	33.29	50.92	71.64	40.77	31.91	23.93	63.17	27.73	30.99	47.25	52.06	50.13	46.31
SRINet [16]	76.95	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	76.95	-
RIConv [19]	79.31	74.60	78.64	78.70	73.19	68.03	86.82	71.87	89.36	82.95	74.70	76.42	56.58	88.44	72.16	51.63	66.65	77.47	79.55	74.43
Kim et al. [18]	79.56	74.41	77.53	73.43	76.95	66.13	87.22	**75.44**	87.42	80.71	78.44	71.21	51.09	90.76	73.69	53.86	68.10	78.62	79.92	74.69
Li et al. [52]	82.17	78.78	81.49	80.07	**85.55**	74.83	**88.62**	71.34	90.38	82.82	**80.34**	81.64	68.87	92.23	74.51	54.08	74.59	79.11	82.47	79.40
Ours(PRIN)	71.20	66.75	69.29	55.90	71.49	56.31	78.44	65.92	86.01	73.58	66.97	59.29	47.56	81.47	71.99	49.02	64.70	70.12	72.04	68.39
Ours(SPRIN)	**82.67**	**79.50**	**82.07**	**82.01**	76.48	**75.53**	88.17	71.45	**90.51**	**83.95**	79.22	**83.83**	**72.59**	**93.24**	**78.99**	**58.85**	**74.77**	**80.31**	82.59	79.31

TABLE 1

Shape part segmentation results on ShapeNet part dataset. Both average instance and average class IoUs (%) are reported. All models are trained on the non-rotated training set, then evaluated on the non-rotated and rotated test set, indicated by the labels.

5.1 Sparse Rotation Invariance

Definition 5.1 (Point Cloud). A point cloud f can be viewed as a proper (normalized) empirical discrete probabilistic distribution of input points:

$$f(x) = \frac{1}{N} \sum_i^N \delta(x - x_i), \qquad (37)$$

where δ is the Dirac delta function, x_i are input points. This is different from that in Section 4.2, where we reconstruct the dense signal through an adaptive sampling method.

Definition 5.2 (Sparse Correlation). In SPRIN, both input and output are the sparse points of the original point cloud. We directly take $S^2 \times H$ correlation between the filter and the point cloud,

evaluated at some point x_j:

$$[\psi * f](x_j) = \int_{S^2 \times H} \psi(\mathcal{T}(x_j)^{-1}x) f(x) dx \qquad (38)$$

$$= \frac{1}{N} \sum_i^N \int_{S^2 \times H} \psi(\mathcal{T}(x_j)^{-1}x) \delta(x - x_i) dx \quad (39)$$

$$= \frac{1}{N} \sum_i^N \psi(\mathcal{T}(x_j)^{-1}x_i) \qquad (40)$$

Remark. The correlation output $[\psi * f](x_j)$ can be also viewed as the empirical filter expectation $\mathbb{E}_{x \sim f(x)}[\psi(\mathcal{T}(x_j)^{-1}x)]$.

Theorem 5.1 (Sparse Point-wise Rotation Invariance). *The sparse correlation is point-wise rotation invariant when ψ is constant on the left coset of $Z(\theta)$: $\psi(x) \equiv \psi(Z(\theta)x)$, for any x, θ.*

Proof.

$$[\psi * L_Q f](Qx_j) \tag{41}$$

$$= \int_{S^2 \times H} \psi(\mathcal{T}(Qx_j)^{-1}x) f(Q^{-1}x) dx \tag{42}$$

$$= \int_{S^2 \times H} \psi(Z(-\theta)\mathcal{T}(x_j)^{-1}Q^{-1}x) f(Q^{-1}x) dx \tag{43}$$

$$= \int_{S^2 \times H} \psi(\mathcal{T}(x_j)^{-1}x) f(x) dx \tag{44}$$

$$= [\psi * f](x_j). \tag{45}$$

$\square$

In practice, we sum ψ over k nearest neighbors of x_j to reduce memory footprint. ψ is implemented as fully connected layers, which is constant on the latitude. Since $[\psi * f](x_j)$ can be interpreted as the expectation $\mathbb{E}_{x \sim f(x)}[\psi(\mathcal{T}(x_j)^{-1}x)]$, we randomly select a subset of k nearest neighbors for filtering without bias. This trick is also mentioned in [18] as dilated kNN module.

In order to include more context, besides $\mathcal{T}(x_j)^{-1}x_i$, we augment ψ with six rotation invariant features, which are sides and inner-angles of the triangle formed by x_i, x_j and $\frac{1}{N}\sum_i^N x_i$. Though some other rotation invariant features are also available, such as moments $\frac{1}{N}\sum_i^N x_i^2$, $\frac{1}{N}\sum_i^N x_i^3$, we empirically find that the triangle formed by x_i, x_j and $\frac{1}{N}\sum_i^N x_i$ gives a better result by the ablation study. We suspect this is because the triangle sides and angles provide more structural information than simple moments.

5.2 Architecture

SPRIN architecture is shown in Figure 5. It progressively conducts sparse rotation invariant spherical correlation 5.2 between every point and its k nearest neighbors. The dilated kNN module is leveraged, which is similar to that of Kim *et al.* [18]. Given a center point x_j and a dilation rate d, the output of the dilated kNN search is an index set of k/d points, which are randomly chosen from the nearest k neighbors. To hierarchical propagate low-level information to a larger high-level region, we leverage a similar operation *set abstraction* with PointNet++ [53]. Our *set abstraction* layer only contains the farthest point sampling module, which defines the evaluating point of following sparse spherical correlation layers.

For part segmentation, we incorporate two feature propagation layers are concatenated to up-sample the point cloud. This operation is similar to that in PointNet++ [53] while our model does not induce any weight interpolation. It directly conducts sparse spherical correlation at those up-sampled points.

6 EXPERIMENTS

In this section, we arrange comprehensive experiments to evaluate PRIN for point cloud analysis on different tasks. First, we demonstrate that our model can be applied to 3D shape part segmentation and classification with random rotations. Then, we provide some applications on 3D point matching and shape alignment. At last, we conduct an ablation study to validate the design of each part.

6.1 Implementation Details.

PRIN and SPRIN are all implemented in Python using PyTorch on one NVIDIA GTX 1080 Ti. Adam [54] optimization algorithm with learning rate 1e-2 is employed for PRIN, with a batch size of 8. Adam optimization algorithm with learning rate 1e-3 is employed for SPRIN, with a batch size of 16. The learning rate for PRIN decays with a rate of 0.5 every 5 epochs.

For PRIN, we set $\xi = 1/32$. One spherical voxel convolution layer is utilized with 40 channels. It is followed by two $SO(3)$ group convolution layers with channel 40 and 50. All layers share the same bandwidth, which is 32. Two fully-connected layers with channels 50, 50 are concatenated for part segmentation.

For SPRIN, the first two sparse spherical correlation layers are evaluated on all input points, with 2-strided 64-NN neighborhoods. Next three sparse spherical correlation layers are evaluated on 128 sampled points from the sampling & grouping module, with 3-strided 72-NN, 1-strided 32-NN and 1-strided 32-NN neighborhoods, respectively. Then three sparse spherical correlation layers are evaluated on 32 sampled points from the sampling & grouping module, with 1-strided 32-NN neighborhoods. For classification, the max and avg pooling layer concatenates max-pooling and avg-pooling results, followed by three fully-connected layers, with 256, 64, 40 channels respectively. For part segmentation, two sparse spherical correlation layers with 1-strided 16-NN and 1-strided 32-NN are proposed at the first up-sampling level. Three sparse spherical correlation layers with 1-strided 32-NN, 2-strided 48-NN and 3-strided 96-NN are proposed at the second up-sampling level. In the end, three fully connected layers with channels 128, 256, 50 are concatenated.

6.2 Part Segmentation on Rotated Shapes

6.2.1 Dataset

ShapeNet part dataset [5] contains 16,881 shapes from 16 categories in which each shape is annotated with expert verified part labels from 50 different labels in total. Most shapes are composed of two to five parts. We follow the data split in PointNet [1]. 2048 points are sampled per shape for both PRIN and SPRIN.

6.2.2 Evaluation Results

Part segmentation is a challenging task for rotated shape analysis. We compare our method with various baseline methods and train each model on the non-rotated training set and evaluate them on the non-rotated and rotated test set. All the results of baselines are either taken from their original papers or, if not available, reproduced with their released code. RIConv [19], Kim *et al.* [18] and Li *et al.* [52] reported their results on arbitrary rotated test shapes with z-axis data augmentation at training time. In order to make a fair comparison, we re-trained and re-evaluated their models with no rotation augmentation at training time.

Qualitative results are shown in Figure 6. Both PRIN and SPRIN can accomplish part segmentation task on randomly rotated shapes without seeing these rotations. Even though state-of-the-art vanilla deep learning networks like PointNet++ [2], DGCNN [35] and SpiderCNN [38] can achieve fairly good results, they fail on the rotated point clouds. Besides, traditional local features based on local reference frames (LRF) like CGF [10] and SHOT [9] concatenated are also compared. For these descriptors, we train a PointNet-like network with CGF/SHOT features as the input. Although these descriptors are rotation invariant, their performance is inferior to ours. Point-level rotation invariant
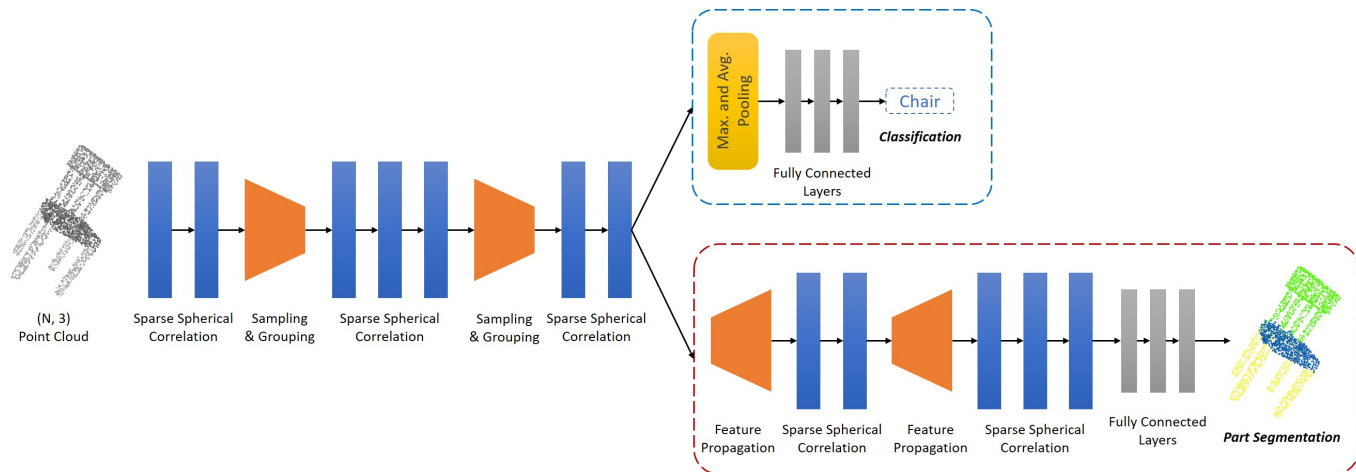
Fig. 5. **SPRIN Architecture.** SPRIN directly operates on sparse point clouds with several sparse spherical correlation layers. Farthest point sampling and kNN grouping are leveraged to aggregate information from low level to high level. Max and average pooling layers follows to extract global features, and finally a classification score is given by fully-connected layers.

convolutions such as RIConv [19] and Kim *et al.* [18] are trained with z-axes rotation augmentation but still inferior to our SPRIN network.

Table 1 shows quantitative results of PRIN/SPRIN and baseline methods. We compare both mean instance IoU and mean class IoU for all the methods. Our SPRIN network even approaches the performance of modern deep learning methods when tested with no rotation. Figure 6 gives some visualization of results between state-of-the-art deep learning methods and PRIN/SPRIN over the ShapeNet part dataset. Biased by the canonical orientation of point clouds in the training set, networks like PointNet just learn a simple partition of Euclidean space, regardless of how the object is oriented. RIConv gives inaccurate segmentation results on part boundaries, while our methods are able to give accurate results under arbitrary orientations.

It should be also noticed that both PRIN and SPRIN have little performance drop when evaluated on the rotated test set, compared with that on the non-rotated set. This is also consistent with our theoretical analysis.

6.3 Classification on Rotated Shapes

In this section, we evaluate our method on ModelNet40 and ScanObjectNN, which are synthetic and real-world datasets respectively.

6.3.1 Dataset

ModelNet40 [6] is a 3D shape classification dataset which contains 12,308 shapes from 40 categories. Here, we use its corresponding point clouds dataset provided by PointNet [1]. ScanObjectNN [55] is a real-world point cloud object dataset, where each object is retrieved from real scans with potential clutter backgrounds. 2048 points are sampled per shape for both PRIN and SPRIN.

6.3.2 Evaluation Results

Though classification does not require point-wise rotation invariant features but a global feature, our network still benefits from DAAS and SVC.

We compare PRIN/SPRIN with several state-of-the-art deep learning methods that take point Euclidean coordinates and local

Method	Input	NR	AR
PointNet [1]		88.45	12.47
PointNet++ [2]		89.82	21.35
PCNN [51]		92.30	17.38
Point2Sequence [40]		92.60	10.53
Kd-Network [41]		86.20	8.49
Spherical CNN [3]	Point XYZ	81.73	55.62
DeepSets [39]		88.73	9.72
LDGCNN [36]		92.91	17.82
SO-Net [37]		**93.44**	9.64
Thomas *et al.* [24]		31.24	32.29
QE-Net [27]		74.43	74.07
SPHNet [17]		87.70	86.60
SRINet [16]		87.01	87.01
RIConv [19]		87.56	87.24
Kim *et al.* [18]		88.49	88.40
Li *et al.* [52]		89.40	**89.32**
Ours(PRIN)	Point XYZ	79.76	72.43
Ours(SPRIN)		86.01	86.13
SHOT+PointNet [9]	Local Features	48.79	48.79
CGF+PointNet [10]		57.70	57.89

TABLE 2
Classification results on ModelNet40 dataset. Performance is evaluated in average instance accuracy. NR means to train with no rotations and test with no rotations. AR means to train with no rotations and test with arbitrary rotations.

rotation invariant features as the input. All the results of baselines are either taken from their original papers or, if not available, reproduced with their released code. RIConv [19], Kim *et al.* [18] and Li *et al.* [52] reported their results on arbitrary rotated test shapes with z-axis data augmentation at training time. In order to make a fair comparison, we re-trained and re-evaluated their models with no rotation augmentation at training time.

Local rotation invariant features like CGF and SHOT are inferior to our method. The results are shown in Table 2 and Table 3. Almost all other deep learning methods fail to generalize to unseen orientations, except for a few recent rotation invariant models. Our model achieves competitive results with current state-of-the-arts, on both ModelNet40 and ScanObjectNN.
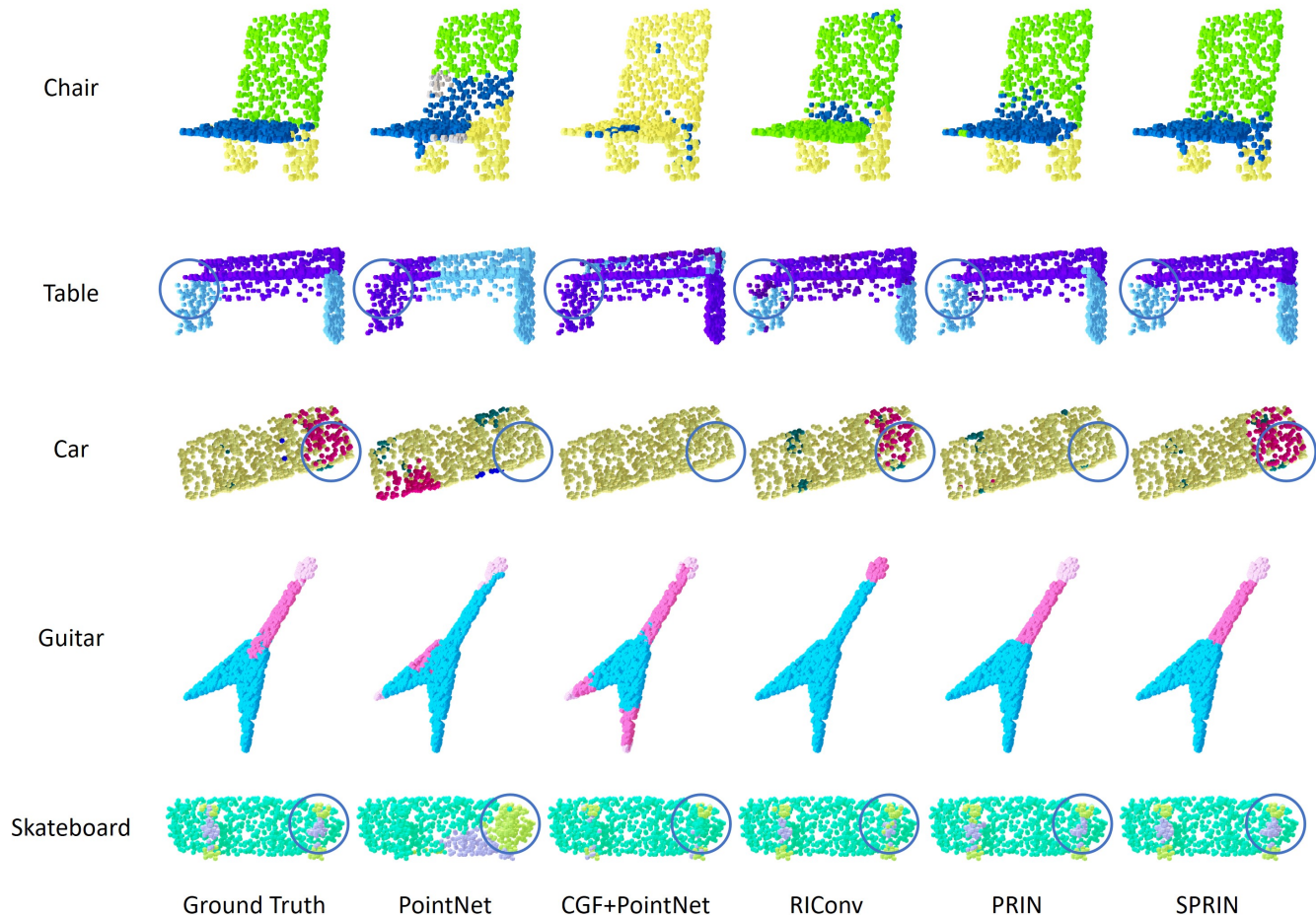
Fig. 6. **Visualization of results.** We compare the results of various methods on rotated point clouds, which are trained on the non-rotated dataset. Both PRIN and SPRIN generalize well to unseen orientations.

Method	Input	NR	AR
PointNet [1]		73.32	21.35
PointNet++ [2]		82.31	18.61
SO-Net [37]		86.18	13.26
SPHNet [17]		67.43	66.52
SRINet [16]		69.53	69.18
RIConv [19]	Point XYZ	69.32	68.44
Kim *et al.* [18]		68.76	68.65
Li *et al.* [52]		**73.42**	**73.44**
Ours(PRIN)	Point XYZ	58.43	52.14
Ours(SPRIN)		70.12	69.83
SHOT+PointNet [9]	Local Features	7.43	7.43
CGF+PointNet [10]		2.87	2.87

TABLE 3
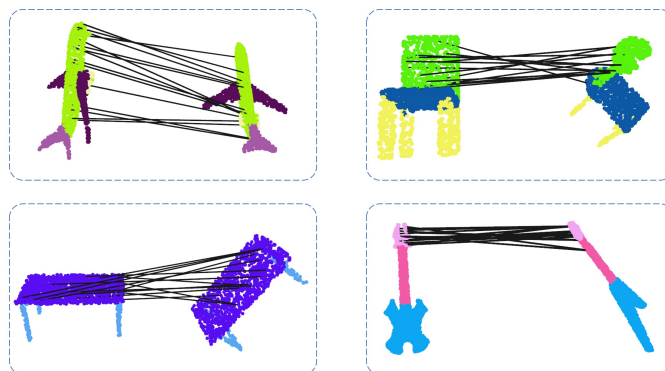**Classification results on ScanObjectNN dataset.** Performance is evaluated in average instance accuracy.

Fig. 7. **3D point matching results for PRIN.** Point matching results between the query object features (left) and the retrieved ones (right) under different orientations.

6.4 Application

3D Rotation Invariant Point Descriptors. For 3D objects, SHOT, CGF and other local features exploits local reference frames (LRF) as local descriptors. Similarly, PRIN/SPRIN is able to produce high quality rotation invariant 3D point descriptors, which is pretty useful as pairwise matching become possible regardless of rotations. In this application, we retrieve the closest descriptor from similar objects under arbitrary orientations. After that, point-

level matching is conducted based on these rotation invariant point descriptors. A demonstration is shown in Figure 7. Through this matching process, we can find out where each point of the query object locates on another object. Moreover, such 3D point descriptors could have the potentiality to do scene searching and parsing as the degree of freedom reduces from six to three, leaving only translations.

To test the matching accuracy on ShapeNet part dataset, we compute and store point-wise features of 80% test objects (non-rotated) in ShapeNet as a database. Then we compute point-wise features of the other 20% test objects that are randomly rotated as queries. Feature retrievals are done by finding queries' nearest neighbors in the database. We evaluate against three different baselines to see if points corresponding to the same part are matched. The results are summarized in Table 4. Both PRIN and SPRIN outperform baseline methods by a large margin.

Method	Matching Accuracy
PointNet [1]	38.91
SHOT [9]	17.11
CGF [10]	52.51
Ours(PRIN)	**86.12**
Ours(SPRIN)	**78.92**

TABLE 4
3D descriptor matching results for various methods. Accuracy is the number of matched points corresponding to the same part divided by the number of all matched points.
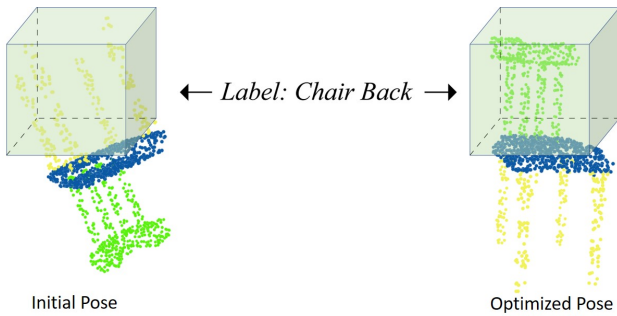

Fig. 8. **Chair alignment with its back on the top. Left:** A misalignment induces large KL divergence. **Right:** Required labels fulfilled with small KL divergence.

Shape Alignment with Label Priors. We now introduce a task that, given some label prior in the Euclidean space, the goal is to align the point cloud to satisfy this requirement. For example, one may align a chair so that its back is on the top. So we add virtual points describing the label requirement. Once the KL divergence between the predicted scores and the ground-truth labels of these virtual points is minimized, the chair is aligned with its back on the top. This is demonstrated in Figure 8.

6.5 Ablation Study

In this section, we evaluate numerous variations of our method to verify the rationality of network design. Experiment results are summarized in Table 5, 7 and Figure 10.

Bandwidth	DAAS	PRIN Acc./mIoU
32	**Yes**	**85.99/71.20**
16	Yes	84.84/68.09
8	Yes	82.58/65.43
4	Yes	72.20/52.98
32	No	83.13/67.47

TABLE 5
Ablation study. Test accuracy/mIoU of PRIN on the rotated ShapeNet part dataset. Models with various bandwidths and sampling strategies are tested.

6.5.1 Network Bandwidth

One decisive factor of our network is the bandwidth. Bandwidth is used to describe the sphere precision, which is also the resolution on $S^2 \times H$. Mostly, large bandwidth offers more details of spherical voxels, such that our network can extract more specific point features of point clouds. While large bandwidth assures more specific representation of part knowledge, more memory cost is accompanied. Increasing bandwidth from 4 to 32 leads to a relative improvement of 16.04% and 25.59% on accuracy and mIoU, which is shown in Table 5.

6.5.2 DAAS v.s. Uniform Sampling

Recall that in Equation 3, we construct our signals on each spherical voxel with a density aware sampling factor. We now compare it with a baseline where uniform sampling is applied and results are given in the last row of Table 5. We see that using the $sin(\beta)$ corrected sampling filter gives superior performance, which confirms our theory.

We also compare extracted features of DAAS and uniform sampling qualitatively in Figure 9. After random rotations of the input canonical point cloud, our DAAS sampled signal obtains much better rotation equivariance than the uniform sampled signal.
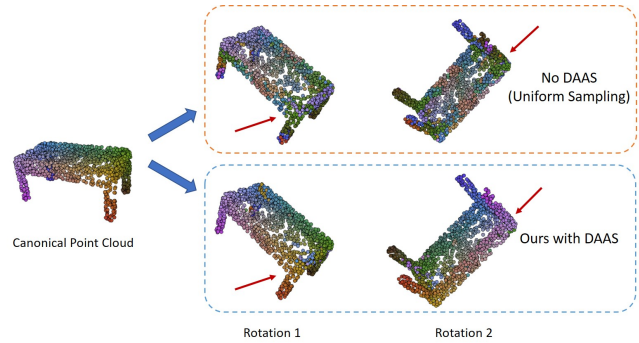

Fig. 9. **Feature visualization of rotated point clouds.** The input point cloud is rotated twice and sampled with DAAS or uniform sampling for further computation. Colors indicate correspondence in the feature space. Our DAAS sampled signal achieves much better correspondence than uniform sampling.

Correlation Features	Triangle Features	SPRIN Acc./mIoU
✓		90.75/79.70
	✓	91.59/81.38
✓	✓	**92.08/82.67**

TABLE 6
Ablation study on the selection of rotation invariant features. Test accuracy/mIoU of SPRIN on rotated ShapeNet part dataset.

# of points	PRIN Acc./mIoU	SPRIN Acc./mIoU
2048	**85.99/71.20**	**92.08/82.67**
1024	84.19/67.43	90.92/79.79
512	71.19/51.50	85.59/72.04
256	55.23/36.80	64.17/51.38

TABLE 7
Qualitative results for segmentation robustness. Test accuracy/mIoU of PRIN/SPRIN on rotated ShapeNet part dataset. Various number of points are sub-sampled.

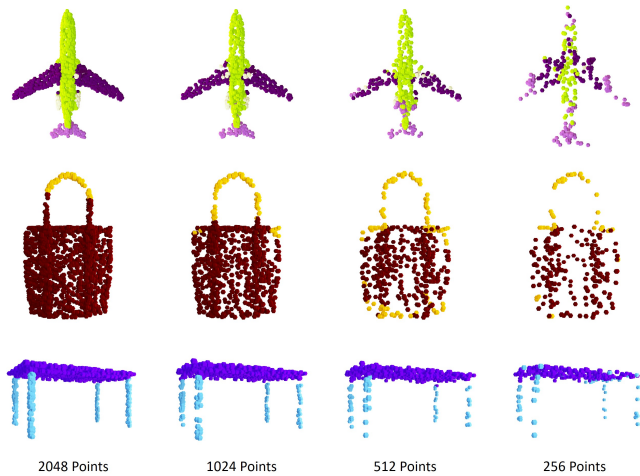2048 Points　　　1024 Points　　　512 Points　　　256 Points

Fig. 10. **Segmentation robustness visualization for PRIN. From left to right**: we sample a subset of 2048, 1024, 512, 256 points from test point clouds respectively. We observe that our method is robust to missing points and gives consistent results.

6.5.3 Selection of Rotation Invariant Features

In SPRIN, we use the correlation feature $\mathcal{T}(x_j)^{-1}x_i$ and six invariant features of the triangle formed by x_i, x_j and $\frac{1}{N}\sum_i^N x_i$. We explore whether these features contribute to the final segmentation. Results are given in Table 6.

6.5.4 Segmentation Robustness

PRIN and SPRIN also demonstrate some robustness to corrupted and missing points. Although the density of point clouds declines, our network still segments correctly for each point. Qualitative results are given in Table 7. Both PRIN and SPRIN maintain a good accuracy up to 4x down-sampling (512 points). In Figure 10, we can see that PRIN predicts consistent labels regardless of the point density.

7 CONCLUSION

We present PRIN, a network that takes any input point cloud and leverages Density Aware Adaptive Sampling (DAAS) to construct signals on spherical voxels. Then Spherical Voxel Convolution (SVC) and Point Re-sampling follow to extract point-wise rotation invariant features. We place two different output heads to do both 3D point clouds classification and part segmentation. In addition, we extend PRIN to a sparse version called SPRIN, which directly operates on sparse point clouds. Our experiments show that both PRIN and SPRIN are robust to arbitrary orientations. Our network can be applied to 3D point feature matching and shape alignment with label priors. We show that our model can naturally handle arbitrary input orientations and provide detailed theoretical analysis to help understand our network.

ACKNOWLEDGMENTS

REFERENCES

[1] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017, pp. 77–85.

[2] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in neural information processing systems*, 2017, pp. 5099–5108.

[3] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling, "Spherical cnns," *International Conference on Learning Representations (ICLR)*, 2018.

[4] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning so (3) equivariant representations with spherical cnns," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 52–68.

[5] L. Yi, V. G. Kim, D. Ceylan, I.-C. Shen, M. Yan, H. Su, C. Lu, Q. Huang, A. Sheffer, and L. Guibas, "A scalable active framework for region annotation in 3d shape collections," *SIGGRAPH Asia*, 2016.

[6] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, "3d shapenets: A deep representation for volumetric shapes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1912–1920.

[7] Y. You, Y. Lou, Q. Liu, Y.-W. Tai, L. Ma, C. Lu, and W. Wang, "Pointwise rotation-invariant network with adaptive sampling and 3d spherical voxel convolution," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 07, 2020, pp. 12 717–12 724.

[8] F. Stein and G. Medioni, "Structural indexing: Efficient 3-d object recognition," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 2, pp. 125–145, 1992.

[9] F. Tombari, S. Salti, and L. Di Stefano, "Unique signatures of histograms for local surface description," in *Computer Vision – ECCV 2010*, K. Daniilidis, P. Maragos, and N. Paragios, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 356–369.

[10] M. Khoury, Q.-Y. Zhou, and V. Koltun, "Learning compact geometric features," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 153–161.

[11] S. Malassiotis and M. G. Strintzis, "Snapshots: A novel local surface descriptor and matching algorithm for robust 3d surface alignment," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 29, no. 7, pp. 1285–1290, 2007.

[12] Y. Guo, M. Bennamoun, F. Sohel, M. Lu, and J. Wan, "3d object recognition in cluttered scenes with local surface features: a survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 11, pp. 2270–2287, 2014.

[13] R. Gens and P. M. Domingos, "Deep symmetry networks," in *Advances in neural information processing systems*, 2014, pp. 2537–2545.

[14] T. S. Cohen and M. Welling, "Steerable cnns," *arXiv preprint arXiv:1612.08498*, 2016.

[15] S. Dieleman, K. W. Willett, and J. Dambre, "Rotation-invariant convolutional neural networks for galaxy morphology prediction," *Monthly notices of the royal astronomical society*, vol. 450, no. 2, pp. 1441–1459, 2015.

[16] X. Sun, Z. Lian, and J. Xiao, "Srinet: Learning strictly rotation-invariant representations for point cloud classification and segmentation," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 980–988.

[17] A. Poulenard, M.-J. Rakotosaona, Y. Ponty, and M. Ovsjanikov, "Effective rotation-invariant point cnn with spherical harmonics kernels," in *2019 International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 47–56.

[18] S. Kim, J. Park, and B. Han, "Rotation-invariant local-to-global representation learning for 3d point cloud," *arXiv preprint arXiv:2010.03318*, 2020.

[19] Z. Zhang, B.-S. Hua, D. W. Rosen, and S.-K. Yeung, "Rotation invariant convolutions for 3d point clouds deep learning," in *2019 International Conference on 3D Vision (3DV)*. IEEE, 2019, pp. 204–213.

[20] H. Deng, T. Birdal, and S. Ilic, "Ppf-foldnet: Unsupervised learning of rotation invariant 3d local descriptors," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 602–618.

[21] X. Li, R. Li, G. Chen, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "A rotation-invariant framework for deep point cloud analysis," *arXiv preprint arXiv:2003.07238*, 2020.

[22] Y. Zhang and M. Rabbat, "A graph-cnn for 3d point cloud classification," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 6279–6283.

[23] W. Wang, Y. You, W. Liu, and C. Lu, "Point cloud classification with deep normalized reeb graph convolution," *Image and Vision Computing*, vol. 106, p. 104092, 2021.

[24] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, "Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds," *arXiv preprint arXiv:1802.08219*, 2018.

[25] F. B. Fuchs, D. E. Worrall, V. Fischer, and M. Welling, "Se (3)-transformers: 3d roto-translation equivariant attention networks," *arXiv preprint arXiv:2006.10503*, 2020.

[26] J. Cruz-Mota, I. Bogdanova, B. Paquier, M. Bierlaire, and J.-P. Thiran, "Scale invariant feature transform on the sphere: Theory and applications," *International journal of computer vision*, vol. 98, no. 2, pp. 217–241, 2012.

[27] Y. Zhao, T. Birdal, J. E. Lenssen, E. Menegatti, L. Guibas, and F. Tombari, "Quaternion equivariant capsule networks for 3d point clouds," in *European Conference on Computer Vision*. Springer, 2020, pp. 1–19.

[28] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 5545–5554.

[29] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept 2015, pp. 922–928.

[30] C. Choy, J. Gwak, and S. Savarese, "4d spatio-temporal convnets: Minkowski convolutional neural networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3075–3084.

[31] B. Graham and L. van der Maaten, "Submanifold sparse convolutional networks," *arXiv preprint arXiv:1706.01307*, 2017.

[32] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3d shape recognition," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 945–953.

[33] Q. Huang, W. Wang, and U. Neumann, "Recurrent slice networks for 3d segmentation of point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2626–2635.

[34] H. Su, V. Jampani, D. Sun, S. Maji, E. Kalogerakis, M.-H. Yang, and J. Kautz, "SPLATNet: Sparse lattice networks for point cloud processing," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 2530–2539.

[35] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (TOG)*, 2019.

[36] K. Zhang, M. Hao, J. Wang, C. W. de Silva, and C. Fu, "Linked dynamic graph cnn: Learning on point cloud via linking hierarchical features," *arXiv preprint arXiv:1904.10014*, 2019.

[37] J. Li, B. M. Chen, and G. Hee Lee, "So-net: Self-organizing network for point cloud analysis," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 9397–9406.

[38] Y. Xu, T. Fan, M. Xu, L. Zeng, and Y. Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 87–102.

[39] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Advances in neural information processing systems*, 2017, pp. 3391–3401.

[40] X. Liu, Z. Han, Y.-S. Liu, and M. Zwicker, "Point2sequence: Learning the shape representation of 3d point clouds with an attention-based sequence to sequence network," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8778–8785.

[41] R. Klokov and V. Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Oct 2017, pp. 863–872.

[42] P. Moon and D. E. Spencer, *Field theory handbook: including coordinate systems, differential equations and their solutions*. Springer, 2012.

[43] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: modelling, planning and control*. Springer Science & Business Media, 2010.

[44] P. J. Kostelec and D. N. Rockmore, "Soft: So (3) fourier transforms," *Department of Mathematics, Dartmouth College, Hanover, NH*, vol. 3755, 2007.

[45] G. B. Thomas Jr, M. D. Weir, J. Hass, C. Heil, and T. Edition, "Early transcendentals," pp. 927–928, 2014.

[46] H. Lei, N. Akhtar, and A. Mian, "Spherical kernel for efficient graph convolution on 3d point clouds," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[47] K. M. Gorski, E. Hivon, A. J. Banday, B. D. Wandelt, F. K. Hansen, M. Reinecke, and M. Bartelmann, "Healpix: A framework for high-resolution discretization and fast analysis of data distributed on the sphere," *The Astrophysical Journal*, vol. 622, no. 2, p. 759, 2005.

[48] P. J. Kostelec and D. N. Rockmore, "Ffts on the rotation group," *Journal of Fourier analysis and applications*, vol. 14, no. 2, pp. 145–179, 2008.

[49] J. Solomon, "Computational optimal transport," 2017.

[50] "Harmonic analysis on so(3)," http://www2.math.ou.edu/~cremling/teaching/lecturenotes/ln-so3.pdf, accessed: 2021-02-09.

[51] M. Atzmon, H. Maron, and Y. Lipman, "Point convolutional neural networks by extension operators," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 71:1–71:12, Jul. 2018. [Online]. Available: http://doi.acm.org/10.1145/3197517.3201301

[52] X. Li, R. Li, G. Chen, C.-W. Fu, D. Cohen-Or, and P.-A. Heng, "A rotation-invariant framework for deep point cloud analysis," *IEEE Transactions on Visualization and Computer Graphics*, 2021.

[53] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in Neural Information Processing Systems 30*, 2017, pp. 5099–5108. [Online]. Available: http://papers.nips.cc/paper/7095-pointnet-deep-hierarchical-feature-learning-on-point-sets-in-a-metric-space.pdf

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[55] M. A. Uy, Q.-H. Pham, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1588–1597.

Yang You received the BS and MS degrees from Shanghai Jiao Tong University and University of Virginia in 2016 and 2017 respectively. He is now a third-year student pursuing his doctoral degree at Mechanical Engineering department in Shanghai Jiao Tong University. His main interests lie in 3D Computer Vision, Computer Graphics and Robotics.

Yujing Lou received the B.S. degree in computer science and technology from Harbin Institute of Technology, China, in 2018 and the M.S. degree in computer science and technology from Shanghai Jiao Tong University, China, in 2020. He is currently a Ph.D. student with MVIG lab, Shanghai Jiao Tong University. His research interests include 3D scene/object perception and robot vision.

Ruoxi Shi is currently an undergraduate student in School of Electronic Information and Electrical Engineering at Shanghai Jiao Tong University, majoring in Artificial Intelligence. He has been working with Prof. Cewu Lu and Dr. Yang You at the Machine Vision and Intelligence Group since 2020. He is currently interested in 3D computer vision and deep learning.

Qi Liu received his B.S. degrees from Dalian University of Technology in 2016. He is now a PhD candidate in Shanghai Jiao Tong University. His research interests include 3D computer vision and computer graphics.

Cewu Lu is an Associate Professor at Shanghai Jiao Tong University (SJTU). Before he joined SJTU, he was a research fellow at Stanford University, working under Prof. Fei-Fei Li and Prof. Leonidas J. Guibas. He was a Research Assistant Professor at Hong Kong University of Science and Technology with Prof. Chi Keung Tang. He got his PhD degree from the Chinese Univeristy of Hong Kong, supervised by Prof. Jiaya Jia. He is one of the core technique members in Stanford-Toyota autonomous car project. He serves as an associate editor for Journal CVPR and reviewer for Journal TPAMI and IJCV. His research interests fall mainly in computer vision, deep learning, deep reinforcement learning and robotics vision.

Yu-Wing Tai received his B.Eng. (First Class Honors) and M Phil degrees from the Department of Computer Science and Engineering, HKUST in 2003 and 2005 and PhD degree from the National University of Singapore in 2009. He is currently an Adjunct Professor at the Department of Computer Science and Engineering, HKUST. He is a research director of YouTu research lab of Social Network Group of Tencent. He was a principle research scientist of SenseTime Group Limited from September 2015 to December 2016. He was an associate professor at the Korea Advanced Institute of Science and Technology (KAIST) from July 2009 to August 2015. From Sept 2007 to June 2008, he worked as a full-time student internship in the Microsoft Research Asia (MSRA). He was awarded the Microsoft Research Asia Fellowship in 2007, and the KAIST 40th Anniversary Academic Award for Excellent Professor in 2011 respectively. His research interests include deep learning, computer vision and image/video processing.

Lizhuang Ma received his B.S. and Ph.D. degrees from the Zhejiang University, China in 1985 and 1991, respectively. He is now a Distinguished Professor, at the Department of Computer Science and Engineering, Shanghai Jiao Tong University, China and the School of Computer Science and Technology, East China Normal University, China. He was a Visiting Professor at the Frounhofer IGD, Darmstadt, Germany in 1998, and a Visiting Professor at the Center for Advanced Media Technology, Nanyang Technological University, Singapore from 1999 to 2000. His research interests include computer vision, computer aided geometric design, computer graphics, scientific data visualization, computer animation, digital media technology, and theory and applications for computer graphics, CAD/CAM.

Weiming Wang is a Professor with the School of Mechanical Engineering, Shanghai Jiao Tong University, Shanghai, China. His research interests include machine vision, flexible robots, and human-robot interaction.