

Learned and Hand-crafted Feature Fusion in Unit Ball for 3D Object Classification

Sameera Ramasinghe¹ ^a, Salman Khan^{2,1} and Nick Barnes¹

¹Australian National University, Australia

²Inception Institute of Artificial Intelligence, U.A.E.

Keywords: Convolution Neural Networks, Volumetric Convolution, Zernike Polynomials, Deep Learning.

Abstract: Convolution is an effective technique that can be used to obtain abstract feature representations using hierarchical layers in deep networks. However, performing convolution in non-Euclidean topological spaces such as the unit ball (\mathbb{B}^3) is still an under-explored problem. In this paper, we propose a light-weight experimental architecture for 3D object classification, that operates in \mathbb{B}^3 . The proposed network utilizes both hand-crafted and learned features, and uses capsules in the penultimate layer to disentangle 3D shape features through pose and view equivariance. It simultaneously maintains an intrinsic co-ordinate frame, where mutual relationships between object parts are preserved. Furthermore, we show that the optimal view angles for extracting patterns from 3D objects depend on its shape and achieve compelling results with a relatively shallow network, compared to the state-of-the-art.


1 INTRODUCTION

Convolution is an extremely effective technique that can be used to extract useful features from spatially correlated data structures, with minimal supervision ((Krizhevsky et al., 2012; He et al., 2016)). Interestingly, extending convolution to topological data structures other than Euclidean spaces, such as spheres, is beneficial to many research areas such as robotics, geoscience and medical imaging, as real-world 3D data naturally lie on a non-Euclidean manifold. The aforementioned extension, however, is not straightforward due to the non-uniform grid structures of non-Euclidean spaces, and is an open research problem.

Recently, there have been some key efforts to extend traditional convolution to spherical space. A preliminary study of this was presented by (Boomsma and Frellsen, 2017), where they apply a cube-sphere transformation on 3D data, and add padding prior to perform 2D convolution on the transformed data. The work by (Cohen et al., 2018) recently received much attention as they used spherical harmonics to efficiently perform convolution on the surface of the sphere (\mathbb{S}^2), while achieving 3D rotational equivariance. A key limitation of their work, however, is that the proposed convolution operation is limited to polar shapes, as the

objects are represented in (\mathbb{S}^2). In a slightly different approach, (Weiler et al., 2018) proposed a method to achieve SE(3) equivariance by modeling 3D data as dense vector fields in 3D Euclidean space.

On the contrary, (Ramasinghe et al., 2019) presented a novel convolution operation that can perform convolution in \mathbb{B}^3 . In this work, we adapt their derived formulas and present a novel experimental architecture which can be used to classify 3D objects. Our architecture can handle non-polar shapes, and capture both 2D texture and 3D shape features simultaneously. We use a capsule network after the convolution layer as it allows us to directly compare feature discriminability of spherical convolution and volumetric convolution without any bias. In other words, the optimum deep architecture for spherical convolution may not be the same for volumetric convolution. Capsules, however, do not deteriorate extracted features and the final accuracy only depends on the richness of input shape features. Therefore, a fair comparison between spherical and volumetric convolutions can be done by simply replacing the convolution layer. Additionally, the proposed architecture exploits both hand-crafted and learned features and demonstrates that fusing these feature types results in an improved performance. We show that the optimum view-angles for extracting features from a 3D object depends on its shape. We also demonstrate that different learning models such as convolution and cap-

^a  <https://orcid.org/0000-0002-3200-9291>

sule networks can work in unison towards a common goal. Furthermore, our network achieves competitive results with a significantly shallow design, compared to the state-of-the-art.

It is important to note that the proposed experimental architecture is only a one possible example out of many possible designs, and is focused on three factors: 1) Capture useful features with a relatively shallow network compared to state-of-the-art. 2) Show richness of computed features through clear improvements over spherical convolution. 3) Demonstrate the usefulness of the volumetric convolution and axial symmetry feature layers as fully differentiable and easily pluggable layers, which can be used as building blocks for end-to-end deep architectures.

The rest of the paper is structured as follows. In Sec. 2 we introduce the overall problem and our proposed solution. Sec. 3 presents an overview of the necessary theoretical background. Sec. 4.2 presents our experimental architecture, and in Sec. 5 we show the effectiveness of the derived operators through extensive experiments. Finally, we conclude the paper in Sec. 6.

2 PROBLEM DEFINITION

Convolution is an effective method to capture useful features from uniformly spaced grids in \mathbb{R}^n , within each dimension of n , such as gray scale images (\mathbb{R}^2), RGB images (\mathbb{R}^3), spatio-temporal data (\mathbb{R}^3) and stacked planar feature maps (\mathbb{R}^n). In such cases, uniformity of the grid within each dimension ensures the translation equivariance of the convolution. However, for topological spaces such as \mathbb{S}^2 and \mathbb{B}^3 , it is not possible to construct such a grid due to non-linearity. A naive approach to perform convolution in \mathbb{B}^3 would be to create a uniformly spaced three dimensional grid in (r, θ, ϕ) coordinates (with necessary padding) and perform 3D convolution. However, the spaces between adjacent points in each axis are dependant on their absolute position and hence, modeling such a space as a uniformly spaced grid is not accurate.

To overcome these limitations, we propose a novel experimental architecture which can effectively operate on functions in \mathbb{B}^3 . It is important to note that ideally, the convolution in \mathbb{B}^3 should be a signal on both 3D rotation group and 3D translation. However, since Zernike polynomials do not have the necessary properties to automatically achieve translation equivariance, we stick to 3D rotation group in this work. In Sec. 3, we present an overview of the theoretical background that is relevant to the context of this paper.

3 THEORETICAL BACKGROUND

3.1 3D Zernike Polynomials

3D Zernike polynomials are a complete and orthogonal set of basis functions in \mathbb{B}^3 , that exhibits a ‘*form invariance*’ property under 3D rotation ((Canterakis, 1999)). A $(n, l, m)^{th}$ order 3D Zernike basis function is defined as,

$$Z_{n,l,m} = R_{n,l}(r)Y_{l,m}(\theta, \phi) \quad (1)$$

where $R_{n,l}$ is the Zernike radial polynomial, $Y_{l,m}(\theta, \phi)$ is the spherical harmonics function, $n \in \mathbb{Z}^+$, $l \in [0, n]$, $m \in [-l, l]$ and $n - l$ is even. Since 3D Zernike polynomials are orthogonal and complete in \mathbb{B}^3 , an arbitrary function $f(r, \theta, \phi)$ in \mathbb{B}^3 can be approximated using Zernike polynomials as follows.

$$f(\theta, \phi, r) = \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) Z_{n,l,m}(\theta, \phi, r) \quad (2)$$

where $\Omega_{n,l,m}(f)$ could be obtained using,

$$\Omega_{n,l,m}(f) = \int_0^1 \int_0^{2\pi} \int_0^\pi f(\theta, \phi, r) Z_{n,l,m}^\dagger r^2 \sin\phi dr d\phi d\theta \quad (3)$$

where † denotes the complex conjugate. In Sec. 3.2, we will derive the proposed volumetric convolution.

3.2 Volumetric Convolution

Consider a 3D rotation operation, which moves a point p on the surface of the unit sphere to another point p' . If we decompose the rotation using Euler angles as $R(\theta, \phi) = R(\theta)_y R(\phi)_z R(\theta)_y$, the first rotation $R(\theta)_y$ can differ while mapping p to p' (if y is the north pole). In another words, there is no unique rotation operation which can map p to p' . However, enforcing the kernel to be symmetric around the north pole (y -axis) makes the rotated kernel function depend only on p and p' , since then the initial rotation around y does not affect the kernel function. Following this observation, we are able to define a 3D rotation kernel which only depends on azimuth and polar angles, as shown in upcoming derivations.

Let the kernel be symmetric around y and $f(\theta, \phi, r)$, $g(\theta, \phi, r)$ be the functions of object and kernel respectively. Then we define volumetric convolution as,

$$f * g(\alpha, \beta) := \langle f(\theta, \phi, r), \tau_{(\alpha, \beta)}(g(\theta, \phi, r)) \rangle \quad (4)$$

$$= \int_0^1 \int_0^{2\pi} \int_0^\pi f(\theta, \phi, r) \tau_{(\alpha, \beta)}(g(\theta, \phi, r)) \sin\phi d\phi d\theta dr, \quad (5)$$

where $\tau_{(\alpha,\beta)}$ is an arbitrary rotation, that aligns the north pole with the axis towards (α, β) direction (α and β are azimuth and polar angles respectively). Eq. 4 is able to capture more complex patterns compared to spherical convolution due to two reasons: 1) the inner product integrates along the radius and 2) the projection onto spherical harmonics forces the function into a polar function, that can result in information loss.

(Ramasinghe et al., 2019) present the following theorem to present volumetric convolution. A short version of the proof is then provided. Please see Appendix 6 for the complete derivation.

Theorem 1: Suppose $f, g : X \rightarrow \mathbb{R}^3$ are square integrable complex functions defined in \mathbb{B}^3 so that $\langle f, f \rangle < \infty$ and $\langle g, g \rangle < \infty$. Further, suppose g is symmetric around north pole and $\tau(\alpha, \beta) = R_y(\alpha)R_z(\beta)$ where $R \in SO(3)$. Then,

$$\begin{aligned} & \int_0^1 \int_0^{2\pi} \int_0^\pi f(\theta, \phi, r) \tau_{(\alpha,\beta)}(g(\theta, \phi, r)) \sin \phi d\phi d\theta dr \\ & \equiv \frac{4\pi}{3} \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n,l,0}(g) Y_{l,m}(\alpha, \beta), \end{aligned} \quad (6)$$

where $\Omega_{n,l,m}(f)$, $\Omega_{n,l,0}(g)$ and $Y_{l,m}(\theta, \phi)$ are $(n, l, m)^{th}$ 3D Zernike moment of f , $(n, l, 0)^{th}$ 3D Zernike moment of g , and spherical harmonics function respectively.

Proof: Completeness property of 3D Zernike Polynomials ensures that it can approximate an arbitrary function in \mathbb{B}^3 , as shown in Eq. 2. Leveraging this property, Eq. 4 can be rewritten as,

$$\begin{aligned} f * g(\alpha, \beta) &= \left\langle \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) Z_{n,l,m}, \right. \\ & \left. \tau_{(\alpha,\beta)} \left(\sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m'=-l'}^l \Omega_{n',l',m'}(g) Z_{n',l',m'} \right) \right\rangle. \end{aligned} \quad (7)$$

However, since $g(\theta, \phi, r)$ is symmetric around y , the rotation around y should not change the function. This ensures,

$$g(r, \theta, \phi) = g(r, \theta - \alpha, \phi) \quad (8)$$

and hence,

$$\begin{aligned} & \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m'=-l'}^l \Omega_{n',l',m'}(g) R_{n',l'}(r) Y_{l',m'}(\theta, \phi) \\ &= \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m'=-l'}^l \Omega_{n',l',m'}(g) R_{n',l'}(r) Y_{l',m'}(\theta, \phi) e^{-im'\alpha}. \end{aligned} \quad (9)$$

This is true, if and only if $m' = 0$. Therefore, a symmetric function around y , defined inside the unit sphere

can be rewritten as,

$$\sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \Omega_{n',l',0}(g) Z_{n',l',0} \quad (10)$$

which simplifies Eq. 7 to,

$$\begin{aligned} f * g(\alpha, \beta) &= \left\langle \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) Z_{n,l,m}, \right. \\ & \left. \tau_{(\alpha,\beta)} \left(\sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \Omega_{n',l',0}(g) Z_{n',l',0} \right) \right\rangle \end{aligned} \quad (11)$$

Using the properties of inner product, Eq. 11 can be rearranged as,

$$\begin{aligned} f * g(\alpha, \beta) &= \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n',l',0}(g) \\ & \langle Z_{n,l,m}, \tau_{(\alpha,\beta)}(Z_{n',l',0}) \rangle. \end{aligned} \quad (12)$$

Using the rotational properties of Zernike polynomials, we obtain (see Appendix 6 for our full derivation),

$$f * g(\theta, \phi) = \frac{4\pi}{3} \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n,l,0}(g) Y_{l,m}(\theta, \phi) \quad (13)$$

Since we can calculate $\Omega_{n,l,m}(f)$ and $\Omega_{n,l,0}(g)$ easily using an iterative method (Ramasinghe et al., 2019)), $f * g(\theta, \phi)$ can be found using a simple matrix multiplication. It is interesting to note that, since the convolution kernel does not translate, the convolution produces a polar shape, which can be further convolved—if needed—using the relationship

$$f * g(\theta, \phi) = \sqrt{\frac{4\pi}{2l+1}} \sum_l \hat{f}(l, m) \hat{g}(l, m) Y_{l,m}(\theta, \phi)$$

where, $\hat{f}(l, m)$ and $\hat{g}(l, m)$ are the $(l, m)^{th}$ frequency components of f and g in spherical harmonics space.

3.3 Equivariance to 3D Rotation Group

The equivariance of the volumetric convolution to 3D rotation group can be shown using the following theorem.

Theorem 1: Suppose $f, g : X \rightarrow \mathbb{R}^3$ are square integrable complex functions defined in \mathbb{B}^3 so that $\langle f, f \rangle < \infty$ and $\langle g, g \rangle < \infty$. Also, let $\eta_{\alpha,\beta,\gamma}$ be a 3D rotation operator that can be decomposed into three Euler rotations $R_y(\alpha)R_z(\beta)R_y(\gamma)$ and $\tau_{\alpha,\beta}$ another rotation operator that can be decomposed into $R_y(\alpha)R_z(\beta)$. Suppose $\eta_{\alpha,\beta,\gamma}(g) = \tau_{\alpha,\beta}(g)$. Then, $\eta_{(\alpha,\beta,\gamma)}(f) * g(\theta, \phi) = \tau_{(\alpha,\beta)}(f * g)(\theta, \phi)$, where $*$ is the volumetric convolution operator.

The proof to above theorem can be found in Appendix 6.

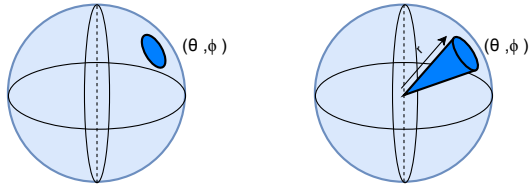


Figure 1: Difference between spherical convolution (*left*) and volumetric convolution (*right*). In volumetric convolution, inner product between kernel and the shape is taken in \mathbb{B}^3 and in spherical convolution, it is taken in \mathbb{S}^2 . Modeling and convolving in \mathbb{B}^3 allows encoding non-polar 3D shapes with texture.

3.4 Axial Symmetry of Functions in \mathbb{B}^3

Axial symmetry of a function in \mathbb{B}^3 around an arbitrary axis can be found using the following formula.

Proposition: Suppose $g : X \rightarrow \mathbb{R}^3$ is a square integrable complex function defined in \mathbb{B}^3 such that $\langle g, g \rangle < \infty$. Then, the power of projection of g in to $S = \{Z_i\}$ where S is the set of Zernike basis functions that are symmetric around an axis towards (α, β) direction is given by,

$$\|sym_{(\alpha, \beta)}\| = \sum_n \sum_{l=0}^n \left\| \sum_{m=-l}^l \Omega_{n,l,m} Y_{m,l}(\alpha, \beta) \right\|^2 \quad (14)$$

where α and β are azimuth and polar angles respectively.

The proof the above proposition is given in Appendix 6.

4 A CASE STUDY: 3D OBJECT RECOGNITION

4.1 3D Objects as Functions in \mathbb{B}^3

A polar 3D object can be expressed as a single valued function on the \mathbb{S}^2 . Performing convolution on \mathbb{S}^2 can be considered as moving the kernel on \mathbb{S}^2 and then calculating inner product with the shape function. However, representing non-polar shapes as polar objects can lead to critical information loss. Furthermore, since the inner product happens on \mathbb{S}^2 , it is not possible to capture patterns across radius.

The aforementioned limitations can be avoided by performing convolution inside the unit ball (\mathbb{B}^3). Modeling the shape functions inside \mathbb{B}^3 allows us to represent non-polar shapes without loss of information and it allows encoding 2D texture information, as each point inside \mathbb{B}^3 can be allocated a scalar value. Figure 1 illustrates the difference between volumetric and spherical convolutions. However, we experiment

only on uniform textured 3D objects in this work, and therefore, apply an artificial surface function to the objects as follows:

$$f(\theta, \phi, r) = \begin{cases} r, & \text{if surface exists at } (\theta, \phi, r) \\ 0, & \text{otherwise} \end{cases} \quad (15)$$

4.2 An Experimental Architecture

We implement an experimental architecture to demonstrate the usefulness of the proposed operations. While these operations can be used as building-tools to construct any deep network, we focus on three key factors while developing the presented experimental architecture: 1) **Shallowness:** Volumetric convolution should be able to capture useful features compared to other methodologies with less number of layers. 2) **Modularity:** The architecture should have a modular nature so that a fair comparison can be made between volumetric and spherical convolution. We use a capsule network after the convolution layer for this purpose. 3) **Flexibility:** It should clearly exhibit the usefulness of axial symmetry features as a hand-crafted and fully differentiable layer. The motivation is to demonstrate one possible use case of axial symmetry measurements in 3D shape analysis.

The proposed architecture consists of four components. **First**, we obtain three view angles, and later generate features for each view angle separately. We optimize the view angles to capture complimentary shape details such that the total information content is maximized. For each viewing angle 'k', we obtain two point sets P_k^+ and P_k^- consisting of tuples denoted as:

$$\begin{aligned} P_k^+ &= \{(x_i, y_i, z_i) : y_i > 0\}, \text{ and} \\ P_k^- &= \{(x_i, y_i, z_i) : y_i < 0\}, \end{aligned} \quad (16)$$

such that y denotes the horizontal axis. **Second**, the six point sets are volumetrically convolved with kernels to capture local patterns of the object. The generated features for each point set are then combined using compact bilinear pooling. **Third**, we use axial symmetry measurements to generate additional features. The features that represent each point set are then combined using compact bilinear pooling. **Fourth**, we feed features from second and third components of the overall architecture to two independent capsule networks and combine the outputs at decision level to obtain the final prediction. The overall architecture of the proposed scheme is shown in Fig. 2.

4.3 Optimum View Angles

We use three view angles to generate features for better representation of the object. First, we translate

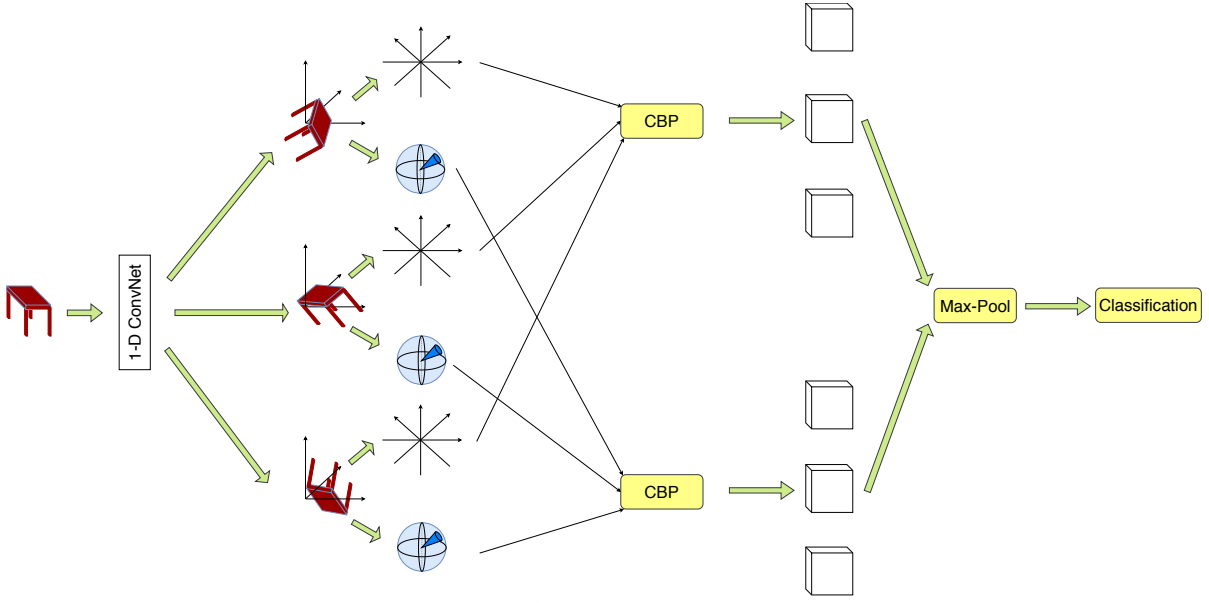


Figure 2: Experimental architecture: An object is first mapped to three view angles. For each angle, axial symmetry and volumetric convolution features are generated for P^+ and P^- . These two features are then separately combined using compact bilinear pooling. Finally, the features are fed to two individual capsule networks, and the decisions are max-pooled.

the center of mass of the set of (x, y, z) points to the origin. The goal of this step is to achieve a general translational invariance, which allows us to free the convolution operation from the burden of detecting translated local patterns. Subsequently, the point set is rearranged as an ordered set on x and z and a 1D convolution net is applied on y values of the points. Here, the objective is to capture local variations of points along the y axis, since later we analyze point sets P^+ and P^- independently. The trained filters can be assumed to capture properties similar to $\partial^n y / \partial x^n$ and $\partial^n y / \partial z^n$, where n is the order of derivative. The output of the 1D convolution net is rotation parameters represented by a 1×9 vector $\vec{r} = \{r_1, r_2, \dots, r_9\}$. Then, we compute $R_1 = R_x(r_1)R_y(r_2)R_z(r_3)$, $R_2 = R_x(r_4)R_y(r_5)R_z(r_6)$ and $R_3 = R_x(r_7)R_y(r_8)R_z(r_9)$ where R_1, R_2 and R_3 are the rotations that map the points to three different view angles.

After mapping the original point set to three view angles, we extract the P_k^+ and P_k^- point sets from each angle k that gives us six point sets. These sets are then fed to the volumetric convolution layer to obtain feature maps for each point set. We then measure the symmetry around four equi-angular axes using Eq. 14, and concatenate these measurement values to form a feature vector for the same point sets.

4.4 Feature Fusion using Compact Bilinear Pooling

Compact bilinear pooling (CBP) provides a compact representation of the full bilinear representation, but has the same discriminative power. The key advantage of compact bilinear pooling is the significantly reduced dimensionality of the pooled feature vector.

We first concatenate the obtained volumetric convolution features of the three angles, for P^+ and P^- separately to establish two feature vectors. These two features are then fused using compact bilinear pooling ((Gao et al., 2016)). The same approach is used to combine the axial symmetry features. These fused vectors are fed to two independent capsule nets.

Furthermore, we experiment with several other feature fusion techniques and present results in Sec. 5.2.

4.5 Capsule Network

Capsule Network (CapsNet) ((Sabour et al., 2017)) brings a new paradigm to deep learning by modeling input domain variations through vector based representations. CapsNets are inspired by so-called *inverse graphics*, i.e., the opposite operation of image rendering. Given a feature representation, CapsNets attempt to generate the corresponding geometrical representation. The motivation for using CapsNets in the network are twofold: 1) CapsNet promotes a dynamic ‘routing-by-agreement’ approach where only the features that

are in agreement with high-level detectors are routed forward. This property of CapsNets does not deteriorate extracted features and the final accuracy only depends on the richness of original shape features. It allows us to directly compare feature discriminability of spherical and volumetric convolution without any bias. For example, using multiple layers of volumetric or spherical convolution hampers a fair comparison since it can be argued that the optimum architecture may vary for two different operations. 2) CapsNet provides an ideal mechanism for disentangling 3D shape features through pose and view equivariance while maintaining an intrinsic co-ordinate frame where mutual relationships between object parts are preserved.

Inspired by these intuitions, we employ two independent CapsNets in our network for volumetric convolution features and axial symmetry features. In this layer, we rearrange the input feature vectors as two sets of primary capsules—for each capsule net—and use the dynamic routing technique proposed by (Sabour et al., 2017) to predict the classification results. The outputs are then combined using max-pooling, to obtain the final classification result. For volumetric convolution features, our architecture uses 1000 primary capsules with 10 dimensions each. For axial symmetry features, we use 2500 capsules, each with 10 dimensions. In both networks, decision layer consist of 12 dimensional capsules.

4.6 Hyperparameters

We use $n = 5$ to implement Eq. 13 and use a decaying learning rate $lr = 0.1 \times 0.9^{\frac{g_{step}}{3000}}$, where g_{step} is incremented by one per each iteration. For training, we use the Adam optimizer with $\beta_1 = 0.9, \beta_2 = 0.999, \epsilon = 1 \times 10^{-8}$ where parameters refer to the usual notation. All these values are chosen empirically. Since we have decomposed the theoretical derivations into sets of low-cost matrix multiplications, specifically aiming to reduce the computational complexity, the GPU implementation is highly efficient. For example, the model takes less than 15 minutes for an epoch during the training phase for ModelNet10, with a batchsize 2, on a single GTX 1080Ti GPU.

5 EXPERIMENTS

In this section, we discuss and evaluate the performance of the proposed approach. We first compare the accuracy of our model with relevant state-of-the-art work, and then present a thorough ablation study of our model, that highlights the importance of several

architectural aspects. We use ModelNet10 and ModelNet40 datasets in our experiments. Next, we evaluate the robustness of our approach against loss of information and finally show that the proposed approach for computing 3D Zernike moments produce richer representations of 3D shapes, compared to the conventional approach.

5.1 Comparison with the State-of-the-art

Table 1 illustrates the performance comparison of our model with state-of-the-art. The model attains an overall accuracy of 92.17% on ModelNet10 and 86.5% accuracy on ModelNet40, which is on par with state-of-the-art. We do not compare with other recent work, such as (Kanezaki et al., 2016), (Qi et al., 2016), (Sedaghat et al., 2016), (Wu et al., 2016), (Qi et al., 2016) and (Bai et al., 2016), (Maturana and Scherer, 2015) that show impressive performance on ModelNet10 and ModelNet40. These are not comparable with our proposed approach, as we propose a shallow, single model without any data augmentation, with a relatively low number of parameters. Furthermore, our model reports these results by using only a single volumetric convolution layer for learning features. Fig. 3 demonstrates effectiveness of our architecture by comparing accuracy against the number of trainable parameters in state-of-the-art models.

5.2 Ablation Study

Table 2 depicts the performance comparison between several variants of our model. To highlight the effectiveness of the learned optimum view points, we replace the optimum view point layer with three fixed orthogonal view points. This modification causes an accuracy drop of 6.57%, emphasizing that the optimum view points indeed depends on the shape. Another interesting—perhaps the most important—aspect to study is the performance of the proposed volumetric convolution against spherical convolution. To this end, we replace the volumetric convolution layer of our model with spherical convolution and compare the results. It can be seen that our volumetric convolution scheme outperforms spherical convolution by a significant margin of 12.56%, indicating that volumetric convolution captures shape properties more effectively.

Furthermore, using mean-pooling instead of max-pooling, at the decision layer drops the accuracy to 87.27%. We also evaluate performance of using a single capsule net. In this scenario, we combine axial symmetry features with volumetric convolution fea-

Table 1: Comparison with state-of-the-art methods on ModelNet10 and ModelNet40 datasets (ranked according to performance). Ours achieve a competitive performance with the least network depth.

Method	Trainable layers	# Params	M10	M40
SO-Net ((Li et al., 2018))	11FC	60M	95.7%	93.4%
Kd-Networks ((Klokov and Lempitsky, 2017))	15KD	-	94.0%	91.8%
VRN ((Brock et al., 2016))	45Conv	90M	93.11%	90.8%
Pairwise ((Johns et al., 2016))	23Conv	143M	92.8%	90.7%
MVCNN ((Su et al., 2015))	60Conv + 36FC	200M	-	90.1%
Ours	3Conv + 2Caps	4.4M	92.17%	86.5%
PointNet ((Qi et al., 2017))	2ST + 5Conv	80M	-	86.2%
ECC ((Simonovsky and Komodakis, 2017))	4Conv + 1FC	-	-	83.2%
DeepPano ((Shi et al., 2015))	4Conv + 3FC	-	85.45%	77.63%
3DShapeNets ((Wu et al., 2015))	4-3DConv + 2FC	38M	83.5%	77%
PointNet ((Garcia-Garcia et al., 2016))	2Conv + 2FC	80M	77.6%	-

Table 2: Ablation study of the proposed architecture on ModelNet10 dataset.

Method	Accuracy
Final Architecture (FA)	92.17%
FA + Orthogonal Rotation	85.60%
FA - VolCNN + SphCNN	79.53%
FA -MaxPool + MeanPool	87.27%
FA + Feature Fusion (Axial + Conv)	86.53%
Axial Symmetry Features	66.73%
VolConv Features	85.3%
SphConv Features	71.6%
FA - CapsNet + FC layers	87.3 %
FA - CBP + Feature concat	90.7%
FA - CBP + MaxPool	90.3%
FA - CBP + Average-pooling	85.3 %

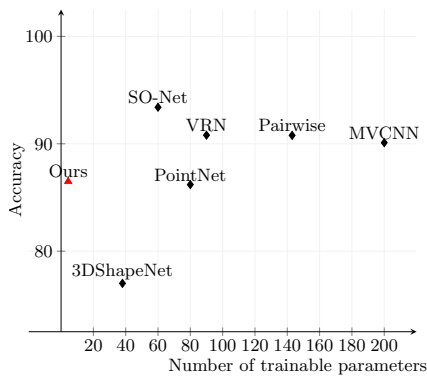


Figure 3: Accuracy vs number of trainable params (in millions) trend (ModelNet40).

tures using compact bilinear pooling (CBP), and feed it a single capsule network. This variant achieves an overall accuracy of 86.53%, is a 5.64% reduction in accuracy compared to the model with two capsule networks.

Moreover, we compare the performance of two feature categories—volumetric convolution features and axial symmetry features—individually. Axial symmetry features alone are able to obtain an accuracy of 66.73%, while volumetric convolution features reach a significant 85.3% accuracy. On the contrary, spherical convolution attains an accuracy of 71.6%, which again highlights the effectiveness of volumetric convolution.

Then we compare between different choices that can be applied to the experimental architecture. We first replace the capsule network with a fully connected layer and achieve an accuracy of 87.3%. This is perhaps because capsules are superior to a simple fully connected layer in modeling view-point invariant representations. Then we try different substitutions for compact bilinear pooling and achieve 90.7%, 90.3% and 85.3% accuracies respectively for feature concatenation, max-pooling and average-pooling. This justifies the choice of compact bilinear pooling as a feature fusion tool. However, it should be noted that these choices may differ depending on the architecture.

5.3 Rotation Parameters

The optimum view-angles for learning features in 3D space depend on the geometric features of a particular 3D object. Therefore, allowing the network to learn these optimum view-angles, in contrast to using fixed angles for every shape, improves the performance, as discussed in Section 5.2. Table 3 depicts the final rotation parameters learned by the network, which are then used for rotating the object. As it is clearly evident, each shape category corresponds to different rotation parameters, which empirically proves that the optimum view-angles indeed depend on the object class.

Table 3: Average rotation parameter values across classes of ModelNet10. The values are reformatted to be positive angles between 0 and 360.

Class	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9
Bathtub	319.2	100.5	57.8	185.2	223.4	98.3	350.6	167.4	14.2
Bed	264.3	196.3	103.7	208.5	186.2	194.4	267.9	246.3	81.2
Chair	198.6	91.2	243.7	47.4	161.2	87.9	240.5	47.3	203.4
Desk	88.4	80.2	130.9	206.6	86.5	112.8	291.7	233.2	351.4
Dresser	58.0	145.7	353.1	148.4	346.4	125.3	47.0	2.2	35.4
Monitor	218.9	279.0	58.1	10.4	30.3	331.4	90.7	285.6	346.1
Night stand	85.3	336.1	175.9	246.4	169.4	278.7	317.0	137.6	302.9
Sofa	306.1	86.9	109.2	311.1	22.5	321.4	96.9	47.0	76.2
Table	299.8	85.2	126.5	215.1	221.9	245.5	237.1	50.6	128.4
Toilet	277.0	325.3	215.5	255.6	192.2	19.8	278.4	193.4	348.2
Average	211.6	172.6	157.4	183.4	164.1	182.4	221.8	141.1	189.7

6 CONCLUSION

In this work, we present a novel experimental architecture, which can learn feature representations in \mathbb{B}^3 . We utilize the underlying theoretical foundations for volumetric convolution and demonstrate how it can be efficiently computed and implemented using low-cost matrix multiplications. Moreover, we show that our experimental architecture gives competitive results to state-of-the-art with a relatively shallow design, in 3D object recognition task. Finally, we empirically demonstrate that fusing learned and hand-crafted features results in improved performance, as they provide complementary information.

REFERENCES

- Bai, S., Bai, X., Zhou, Z., Zhang, Z., and Latecki, L. J. (2016). Gift: A real-time and scalable 3d shape search engine. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 5023–5032. IEEE.
- Boomsma, W. and Frellsen, J. (2017). Spherical convolutions and their application in molecular modelling. In *Advances in Neural Information Processing Systems*, pages 3436–3446.
- Brock, A., Lim, T., Ritchie, J. M., and Weston, N. (2016). Generative and discriminative voxel modeling with convolutional neural networks. *arXiv preprint arXiv:1608.04236*.
- Canterakis, N. (1999). 3d zernike moments and zernike affine invariants for 3d image analysis and recognition. In *In 11th Scandinavian Conf. on Image Analysis*. Citeseer.
- Cohen, T. S., Geiger, M., Koehler, J., and Welling, M. (2018). Spherical cnns. *International Conference on Learning representations (ICLR)*.
- Gao, Y., Beijbom, O., Zhang, N., and Darrell, T. (2016). Compact bilinear pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 317–326.
- Garcia-Garcia, A., Gomez-Donoso, F., Garcia-Rodriguez, J., Orts-Escolano, S., Cazorla, M., and Azorin-Lopez, J. (2016). Pointnet: A 3d convolutional neural network for real-time object class recognition. In *Neural Networks (IJCNN), 2016 International Joint Conference on*, pages 1578–1584. IEEE.
- He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.
- Johns, E., Leutenegger, S., and Davison, A. J. (2016). Pairwise decomposition of image sequences for active multi-view recognition. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, pages 3813–3822. IEEE.
- Kanezaki, A., Matsushita, Y., and Nishida, Y. (2016). Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints. *arXiv preprint arXiv:1603.06208*.
- Klokov, R. and Lempitsky, V. (2017). Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 863–872. IEEE.
- Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.
- Li, J., Chen, B. M., and Lee, G. H. (2018). So-net: Self-organizing network for point cloud analysis. *arXiv preprint arXiv:1803.04249*.
- Maturana, D. and Scherer, S. (2015). Voxnet: A 3d convolutional neural network for real-time object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 922–928. IEEE.
- Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017). Point-

net: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 1(2):4.

- Qi, C. R., Su, H., Nießner, M., Dai, A., Yan, M., and Guibas, L. J. (2016). Volumetric and multi-view cnns for object classification on 3d data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5648–5656.
- Ramasinghe, S., Khan, S., Barnes, N., and Gould, S. (2019). Representation learning on unit ball with 3d roto-translational equivariance. *arXiv preprint arXiv:1912.01454*.
- Sabour, S., Frosst, N., and Hinton, G. E. (2017). Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, pages 3859–3869.
- Sedaghat, N., Zolfaghari, M., Amiri, E., and Brox, T. (2016). Orientation-boosted voxel nets for 3d object recognition. *arXiv preprint arXiv:1604.03351*.
- Shi, B., Bai, S., Zhou, Z., and Bai, X. (2015). Deeppano: Deep panoramic representation for 3-d shape recognition. *IEEE Signal Processing Letters*, 22(12):2339–2343.
- Simonovsky, M. and Komodakis, N. (2017). Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proc. CVPR*.
- Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953.
- Weiler, M., Geiger, M., Welling, M., Boomsma, W., and Cohen, T. (2018). 3d steerable cnns: Learning rotationally equivariant features in volumetric data. *arXiv preprint arXiv:1807.02547*.
- Wu, J., Zhang, C., Xue, T., Freeman, B., and Tenenbaum, J. (2016). Learning a probabilistic latent space of object shapes via 3d generative-adversarial modeling. In *Advances in Neural Information Processing Systems*, pages 82–90.
- Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1912–1920.

APPENDIX

Convolution within Unit Sphere using 3D Zernike Polynomials

Theorem 1: Suppose $f, g : X \rightarrow \mathbb{R}^3$ are square integrable complex functions defined in \mathbb{B}^3 so that $\langle f, f \rangle < \infty$ and $\langle g, g \rangle < \infty$. Further, suppose g is symmetric around north pole and $\tau(\alpha, \beta) = R_y(\alpha)R_z(\beta)$

where $R \in SO(3)$. Then,

$$\begin{aligned} & \int_0^1 \int_0^{2\pi} \int_0^\pi f(\theta, \phi, r), \tau_{(\alpha, \beta)}(g(\theta, \phi, r)) \sin \phi d\phi d\theta dr \\ & \equiv \frac{4\pi}{3} \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n,l,0}(g) Y_{l,m}(\alpha, \beta), \end{aligned} \quad (17)$$

where $\Omega_{n,l,m}(f)$, $\Omega_{n,l,0}(g)$ and $Y_{l,m}(\theta, \phi)$ are $(n, l, m)^{th}$ 3D Zernike moment of f , $(n, l, 0)^{th}$ 3D Zernike moment of g , and spherical harmonics function respectively.

Proof: Since 3D Zernike polynomials are orthogonal and complete in \mathbb{B}^3 , an arbitrary function $f(r, \theta, \phi)$ in \mathbb{B}^3 can be approximated using Zernike polynomials as follows.

$$f(\theta, \phi, r) = \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) Z_{n,l,m}(\theta, \phi, r) \quad (18)$$

where $\Omega_{n,l,m}(f)$ could be obtained using,

$$\Omega_{n,l,m}(f) = \int_0^1 \int_0^{2\pi} \int_0^\pi f(\theta, \phi, r) Z_{n,l,m}^\dagger r^2 \sin \phi dr d\phi d\theta \quad (19)$$

where \dagger denotes the complex conjugate.

Leveraging this property (Eq. 18) of 3D Zernike polynomials Eq. 4 can be rewritten as,

$$\begin{aligned} f * g(\alpha, \beta) &= \left\langle \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) Z_{n,l,m}, \right. \\ & \left. \tau_{(\alpha, \beta)} \left(\sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m'=-l'}^l \Omega_{n',l',m'}(g) Z_{n',l',m'} \right) \right\rangle. \end{aligned} \quad (20)$$

But since $g(\theta, \phi, r)$ is symmetric around y, the rotation around y should not change the function. Which ensures,

$$g(r, \theta, \phi) = g(r, \theta - \alpha, \phi) \quad (21)$$

and hence,

$$\begin{aligned} & \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m'=-l'}^l \Omega_{n',l',m'}(g) R_{n',l'}(r) Y_{l',m'}(\theta, \phi) \\ & = \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m'=-l'}^l \Omega_{n',l',m'}(g) R_{n',l'}(r) Y_{l',m'}(\theta, \phi) e^{-im'\alpha}. \end{aligned} \quad (22)$$

This is true, if and only if $m' = 0$. Therefore, a symmetric function around y, defined inside the unit sphere can be rewritten as,

$$\sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \Omega_{n',l',0}(g) Z_{n',l',0} \quad (23)$$

which simplifies Eq. 20 to,

$$f * g(\alpha, \beta) = \left\langle \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) Z_{n,l,m}, \right. \\ \left. \tau_{(\alpha,\beta)} \left(\sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \Omega_{n',l',0}(g) Z_{n',l',0} \right) \right\rangle \quad (24)$$

Using the properties of inner product, Eq. 24 can be rearranged as,

$$f * g(\alpha, \beta) = \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n',l',0}(g) \\ \langle Z_{n,l,m}, \tau_{(\alpha,\beta)}(Z_{n',l',0}) \rangle. \quad (25)$$

Consider the term $\tau_{(\alpha,\beta)}(Z_{n',l',0})$. Then,

$$\tau_{(\alpha,\beta)}(Z_{n',l',0}(\theta, \phi, r)) = \tau_{(\alpha,\beta)}(R_{n',l'}(r) Y_{l',0}(\theta, \phi)) \\ = R_{n',l'}(r) \tau_{(\alpha,\beta)}(Y_{l',0}(\theta, \phi)) \\ = R_{n',l'}(r) \sum_{m''=-l'}^{l'} Y_{l',m''}(\theta, \phi) D_{m'',0}^{l'}(\alpha, \beta, \cdot), \quad (26)$$

where $D_{m,m'}^l$ is the Wigner-D matrix. But we know that $D_{m'',0}^{l'}(\theta, \phi) = Y_{l',m''}(\theta, \phi)$. Then Eq. 25 becomes,

$$f * g(\alpha, \beta) = \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{n'=0}^{\infty} \sum_{l'=0}^{n'} \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n',l',0}(g) \\ \sum_{m''=-l'}^{l'} Y_{l',m''}(\alpha, \beta) \langle Z_{n,l,m}, Z_{n',l',m''} \rangle, \quad (27)$$

$$f * g(\alpha, \beta) = \frac{4\pi}{3} \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n,l,0}(g) Y_{l,m}(\alpha, \beta), \quad (28)$$

Equivariance of Volumetric Convolution to 3D Rotation Group

Theorem 2: Suppose $f, g : \mathbb{B}^3 \rightarrow \mathbb{R}$ are square integrable complex functions defined in \mathbb{B}^3 such that $\langle f, f \rangle < \infty$ and $\langle g, g \rangle < \infty$. Also, let $\eta_{\alpha,\beta,\gamma}$ be a 3D rotation operator that can be decomposed into three Euler rotations $R_y(\alpha)R_z(\beta)R_y(\gamma)$ and $\tau_{\alpha,\beta}$ another rotation operator that can be decomposed into $R_y(\alpha)R_z(\beta)$. Suppose $\eta_{\alpha,\beta,\gamma}(g) = \tau_{\alpha,\beta}(g)$. Then,

$$\eta_{(\alpha,\beta,\gamma)}(f) * g(\theta, \phi) = \tau_{(\alpha,\beta)}(f * g)(\theta, \phi), \quad (29)$$

where $*$ is the volumetric convolution operator.

Proof: Since $\eta_{(\alpha,\beta,\gamma)} \in \mathbb{SO}(3)$, we know that $\eta_{(\alpha,\beta,\gamma)}(f(x)) = f(\eta_{(\alpha,\beta,\gamma)}^{-1}(x))$. Also we know that $\eta_{(\alpha,\beta,\gamma)} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ is an isometry. We define,

$$\langle \eta_{(\alpha,\beta,\gamma)} f, \eta_{(\alpha,\beta,\gamma)} g \rangle = \int_{B^3} f(\eta_{(\alpha,\beta,\gamma)}^{-1}(x)) g(\eta_{(\alpha,\beta,\gamma)}^{-1}(x)) dx. \quad (30)$$

Consider the Lebesgue measure $\lambda(B^3) = \int_{B^3} dx$. It can be proven that a Lebesgue measure is invariant under the isometries, which gives us $dx = d\eta_{(\alpha,\beta,\gamma)}(x) = d\eta_{(\alpha,\beta,\gamma)}^{-1}(x), \forall x \in B^3$. Therefore,

$$\langle \eta_{(\alpha,\beta,\gamma)} f, \eta_{(\alpha,\beta,\gamma)} g \rangle = \langle f, g \rangle \\ = \int_{S^3} f(\eta_{(\alpha,\beta,\gamma)}^{-1}(x)) g(\eta_{(\alpha,\beta,\gamma)}^{-1}(x)) d(\eta_{(\alpha,\beta,\gamma)}^{-1}x). \quad (31)$$

Let $f(\theta, \phi, r)$ and $g(\theta, \phi, r)$ be the object function and kernel function (symmetric around north pole) respectively. Then volumetric convolution is defined as

$$f * g(\theta, \phi) = \langle f, \tau_{(\theta,\phi)} g \rangle. \quad (32)$$

Applying the rotation $\eta_{(\alpha,\beta,\gamma)}$ to f , we get

$$\eta_{(\alpha,\beta,\gamma)}(f) * g(\theta, \phi) = \langle \eta_{(\alpha,\beta,\gamma)}(f), \tau_{(\theta,\phi)} g \rangle \quad (33)$$

Using the result in Eq. 31, we have

$$\eta_{(\alpha,\beta,\gamma)}(f) * g(\theta, \phi) = \langle f, \eta_{(\alpha,\beta,\gamma)}^{-1}(\tau_{(\theta,\phi)} g) \rangle. \quad (34)$$

However, since $\eta_{\alpha,\beta,\gamma}(g) = \tau_{\alpha,\beta}(g)$, we get

$$\eta_{(\alpha,\beta,\gamma)}(f) * g(\theta, \phi) = \langle f, \tau_{(\theta-\alpha,\phi-\beta)} g \rangle. \quad (35)$$

We know that,

$$f * g(\theta, \phi) = \langle f, \tau_{(\theta,\phi)} g \rangle \\ = \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n,l,0}(g) Y_{l,m}(\theta, \phi). \quad (36)$$

Then,

$$\eta_{(\alpha,\beta,\gamma)}(f) * g(\theta, \phi) = \langle f, \tau_{(\theta-\alpha,\phi-\beta)} g \rangle \\ = \sum_{n=0}^{\infty} \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m}(f) \Omega_{n,l,0}(g) Y_{l,m}(\theta - \alpha, \phi - \beta) \\ = (f * g)(\theta - \alpha, \phi - \beta) = \tau_{(\alpha,\beta)}(f * g)(\theta, \phi). \quad (37)$$

Hence, we achieve equivariance over 3D rotations. \square

Axial Symmetry Measure of a Function in \mathbb{B}^3 around an Arbitrary Axis

Proposition: Suppose $g : \mathbb{B}^3 \rightarrow \mathbb{R}^3$ is a square integrable complex function defined in \mathbb{B}^3 such that $\langle g, g \rangle < \infty$. Then, the power of projection of g in to $S = \{Z_i\}$ where S is the set of Zernike basis functions

that are symmetric around an axis towards (α, β) direction is given by

$$\|sym_{(\alpha, \beta)} [g(\theta, \phi, r)]\| = \sum_n \sum_{l=0}^n \left\| \sum_{m=-l}^l \Omega_{n,l,m} Y_{l,m}(\alpha, \beta) \right\|^2, \quad (38)$$

where α and β are azimuth and polar angles respectively.

Proof: The subset of complex functions which are symmetric around north pole is $S = \{Z_{n,l,0}\}$. Therefore, projection of the function into S gives

$$sym_y [g(\theta, \phi, r)] = \sum_n \sum_{l=0}^n \langle f, Z_{n,l,0} \rangle Z_{n,l,0}(\theta, \phi, r). \quad (39)$$

To obtain the symmetry function around any axis which is defined by (α, β) , we rotate the function by $(-\alpha, -\beta)$, project into S , and finally compute the power of the projection

$$sym_{(\alpha, \beta)} [g(\theta, \phi, r)] = \sum_{n,l} \langle \tau_{(-\alpha, -\beta)}(f), Z_{n,l,0} \rangle Z_{n,l,0}(\theta, \phi, r). \quad (40)$$

For any rotation operator τ , and for any two points defined on a complex Hilbert space, x and y ,

$$\langle \tau(x), \tau(y) \rangle_H = \langle x, y \rangle_H. \quad (41)$$

Applying this property to Eq. 40 gives

$$sym_{(\alpha, \beta)} [g(\theta, \phi, r)] = \sum_{n,l} \langle f, \tau_{(\alpha, \beta)}(Z_{n,l,0}) \rangle Z_{n,l,0}(\theta, \phi, r). \quad (42)$$

Using Eq. 18 we get

$$sym_{(\alpha, \beta)} [g(\theta, \phi, r)] = \sum_n \sum_{l=0}^n \left\langle \sum_{n'} \sum_{l'=0}^{n'} \sum_{m'=-l'}^{l'} \Omega_{n'l'm'} Z_{n'l,m'}, \tau_{(\alpha, \beta)}(Z_{n,l,0}) \right\rangle Z_{n,l,0}(\theta, \phi, r). \quad (43)$$

Using properties of inner product Eq. 43 further simplifies to

$$sym_{(\alpha, \beta)} [g(\theta, \phi, r)] = \sum_n \sum_{l=0}^n \sum_{n'} \sum_{l'=0}^{n'} \sum_{m'=-l'}^{l'} \Omega_{n'l'm'} \langle Z_{n'l,m'}, \tau_{(\alpha, \beta)}(Z_{n,l,0}) \rangle Z_{n,l,0}(\theta, \phi, r). \quad (44)$$

Using the same derivation as in Eq. 26,

$$sym_{(\alpha, \beta)} [g(\theta, \phi, r)] = \sum_n \sum_{l=0}^n \sum_{n'} \sum_{l'=0}^{n'} \sum_{m'=-l'}^{l'} \Omega_{n'l'm'} \sum_{m''=-l}^l Y_{l,m''}(\alpha, \beta) \langle Z_{n'l,m'}, Z_{n,l,m''} \rangle Z_{n,l,0}(\theta, \phi, r). \quad (45)$$

Since 3D Zernike Polynomials are orthogonal we get

$$sym_{(\alpha, \beta)} [g(\theta, \phi, r)] = \frac{4\pi}{3} \sum_n \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m} Y_{l,m}(\alpha, \beta) Z_{n,l,0}(\theta, \phi, r). \quad (46)$$

In signal theory the power of a function is taken as the integral of the squared function divided by the size of its domain. Following this we get

$$\|sym_{(\alpha, \beta)} [g(\theta, \phi, r)]\| = \left\langle \left(\sum_n \sum_{l=0}^n \sum_{m=-l}^l \Omega_{n,l,m} Y_{l,m}(\alpha, \beta) \right) Z_{n,l,0}(\theta, \phi, r), \left(\sum_{n'} \sum_{l'=0}^{n'} \sum_{m'=-l'}^{l'} \Omega_{n'l',m'} Y_{l',m'}(\alpha, \beta) Z_{n'l',0}(\theta, \phi, r) \right)^\dagger \right\rangle. \quad (47)$$

We drop the constants here since they do not depend on the frequency. Simplifying Eq. 47 gives

$$\|sym_{(\alpha, \beta)} [g(\theta, \phi, r)]\| = \sum_n \sum_{l=0}^n \sum_{m=-l}^l \sum_{m'=-l}^l \Omega_{n,l,m} Y_{l,m}(\alpha, \beta) \Omega_{n,l,m'} Y_{l',m'}(\alpha, \beta), \quad (48)$$

which leads to

$$\|sym_{(\alpha, \beta)} [g(\theta, \phi, r)]\| = \sum_n \sum_{l=0}^n \left\| \sum_{m=-l}^l \Omega_{n,l,m} Y_{l,m}(\alpha, \beta) \right\|^2. \quad (49)$$

which completes our proof. \square