PDO-s3DCNNs: Partial Differential Operator Based Steerable 3D CNNs

Anonymous Authors[1]

Abstract

Steerable models can provide very general and flexible equivariance by formulating equivariance requirements in the language of representation theory and feature fields, which has been recognized to be effective for many vision tasks. However, deriving steerable models for 3D rotations is much more difficult than that in the 2D case, due to more complicated mathematics of 3D rotations. In this work, we employ partial differential operators (PDOs) to model 3D filters, and derive general steerable 3D CNNs, which are called PDO-s3DCNNs. We prove that the equivariant filters are subject to linear constraints, which can be solved efficiently under various conditions. As far as we know, PDO-s3DCNNs are the most general steerable CNNs for 3D rotations, in the sense that they cover all common subgroups of $SO(3)$ and their representations, while existing methods can only be applied to specific groups and representations. Extensive experiments show that our models can preserve equivariance well in the discrete domain, and outperform previous works on SHREC'17 retrieval and ISBI 2012 segmentation tasks with a low network complexity.

1. Introduction

In the past few years, convolutional neural networks (CNNs) have dominated the computer vision field on various tasks. Compared with fully-connected (FC) neural networks, one main advantage of CNNs is that they are inherently translation equivariant: shifting an input and then feeding it through a CNN is the same as feeding the original input and then shifting the resulted feature maps. However, conventional CNNs are not naturally equivariant to other transformations, such as rotations.

Consequently, many works focus on incorporating more equivariance into CNNs. Firstly adopted in 2D images, Cohen & Welling (2016) proposed group equivariant CNNs (G-CNNs) by rotating filters, which is equivariant over the cyclic group C_4 or the dihedral group D_4. Then some methods (Hoogeboom et al., 2018; Weiler et al., 2018b) are successively proposed to exploit larger groups. However, the equivariance achieved by these methods is very inflexible, because the feature maps inherently can only transform in one single pattern (which can be characterized by regular representation in the language of representation theory) as inputs rotate. But in many vision tasks, it would be better if the transformation behavior of feature maps could be flexibly defined as needed. For example, if the vectors in feature maps are used to predict the object orientations, they should also rotate as the inputs rotate.

In order to utilize more flexible and general equivariance, Cohen & Welling (2017) proposed a theory called steerable CNNs to describe equivariant models in the language of representation theory and feature fields. To be specific, the feature spaces are defined as the spaces of feature fields, characterized by a group representation which determines ("steers") their transformation behavior under the transformations of the input. After the feature fields or representations are specified, equivariant filters are obtained by directly solving the equivariance constraints. Steerable CNNs include G-CNNs as special cases when employing regular features. Weiler & Cesa (2019) further extended the theory of steerable CNNs from C_4 to the Euclidean group $E(2)$ and its subgroups, proposing E2CNNs. E2CNNs are the most general steerable CNNs for 2D rotations so far, as they can deal with all common subgroups of $SO(2)$ and their representations. However, E2CNN *cannot* be extended to the 3D case trivially, since the mathematics of rotation equivariance in 3D is much more complicated than in 2D because 3D rotations do *not* commute (Thomas et al., 2018).

Actually, equivariance is more important for the 3D case because 3D rotations are inevitable – even though a 3D object is up-right, rotations may still occur around the vertical axis, not to mention that some 3D data even have no inherent orientations, such as molecular data (Anderson et al., 2019). In addition, there are relatively few works on 3D CNNs, due to their computation and storage complexity. Thus exploiting equivariance to reduce both computation and storage is critical for applications of 3D CNNs. N-body networks

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

(Kondor, 2018), Tensor Field Networks (TFNs) (Thomas et al., 2018) and SE3CNNs (Weiler et al., 2018a) succeeded in solving the equivariance constraints when defining the feature spaces via irreducible representations (irreps) of $SO(3)$. However, the constraints for discrete subgroups are still not solved. CubeNets (Worrall & Brostow, 2018) exploit equivariance over discrete subgroups by applying G-CNNs to 3D data, i.e., rotating 3D filters. However, viewed as steerable CNNs, CubeNets can only accommodate regular representations of the cubic group $\mathcal{O}$ and its subgroups, noting that there are only cubic rotational symmetries on regular 3D grids. They are not applicable to larger groups (e.g., the icosahedral group $\mathcal{I}$ and $SO(3)$) and other feature fields (e.g., quotient features). In all, existing methods can only deal with specific groups and representations, while the theory, analogous to E2CNN in the 2D case, that can accommodate all common subgroups of $SO(3)$ and their representations is still missing.

Besides theoretical meaning, we would like to emphasize that although the equivariance over a continuous group covers the equivariance over its discrete subgroups, empirical results (Weiler et al., 2018a) show that the models equivariant over discrete subgroups perform even better[1]. This phenomenon explains why we still need to investigate the equivariance over discrete subgroups in practice even though some works have achieved $SO(3)$-equivariance.

In this work, we are devoted to proposing general steerable CNNs for 3D rotations and filling in the deficiency of existing works, and the comparison between our theory and other 3D steerable CNNs is summarized in Table 1. Specifically, Shen et al. (2020; 2021) showed that partial differential operators (PDOs) are very effective for designing equivariant CNNs, which are convenient for mathematical derivation in the continuous domain. Following them, we model 3D filters using PDOs, hence our models are called PDO-s3DCNNs. We emphasize that our work is *not* a trivial extension of PDO-eConv (Shen et al., 2020) to the 3D case, because PDO-eConv is limited to the *regular features* of *discrete subgroups*, and cannot address other feature fields and continuous groups whereas ours can. In our theory, we prove that given any specific rotation group and feature field, the equivariance requirement for a PDO-based filter can be deduced to a linear constraint for its coefficients. Then we illustrate how to solve this constraint efficiently under various conditions. In the implementation, it is easy to discretize our derived PDO-based equivariant filters on volumetric inputs using Finite Difference (FD) or Gaussian discretization. Experiments verify that our methods can preserve equivariance well in the discrete domain, and outperform their counterparts, including SE3CNN (Weiler et al., 2018a) and CubeNet (Worrall & Brostow, 2018), on SHREC'17 retrieval and ISBI 2012 segmentation tasks, respectively, with a low network complexity.

[1]Similar results also occur in our experiments, and we argue that it is due to the amenable normalizations and nonlinearities, as discussed in Sections 5.2 and 6.2.

In summary, our contributions are as follows:

- We propose general steerable 3D CNNs for rotations, which cover both the continuous group $SO(3)$ and its discrete subgroups. In contrast, existing works cover either $SO(3)$ or its discrete subgroups, but not both.
- It is the first time that PDOs are utilized to design equivariant 3D CNNs, and also the first time that the quotient feature is investigated in the 3D case. Note that the quotient feature is a very flexible feature field, including the regular feature as a special case.
- We employ more discretization schemes than that in PDO-eConvs for implementation, and show that Gaussian discretization can preserve equivariance for large groups in the discrete domain much better than FD.
- In experiments, our models perform significantly better than their counterparts SE3CNN on SHREC'17 retrieval task, and CubeNet on ISBI 2012 segmentation task, respectively.

2. Related Work

2.1. Equivariant 2D CNNs

Lenc & Vedaldi (2015) observed that CNNs spontaneously learn representations equivariant to some transformations, indicating that equivariance is a good inductive bias for CNNs. Cohen & Welling (2016) succeeded in incorporating rotation equivariance into neural networks by group equivariant correlation, and proposed G-CNNs. However, this method can only deal with the 4-fold rotational symmetry. Thus some follow-up works (Zhou et al., 2017; Marcos et al., 2017; Hoogeboom et al., 2018; Weiler et al., 2018b; Shen et al., 2020) focused on exploiting larger discrete groups. Also, there are some works (Worrall et al., 2017; Esteves et al., 2018b; Finzi et al., 2020) managing to achieve rotation equivariance over the continuous group $SO(2)$.

The above methods mostly achieve equivariance by using group equivariant correlation. To exploit more general equivariance, Cohen & Welling (2017) proposed steerable CNNs, where feature spaces are characterized as feature fields. Then the equivariant filters are obtained by solving equivariance constraints directly. E2CNNs (Weiler & Cesa, 2019) are the most general steerable CNNs for 2D rotations, because they involve both the continuous group $SO(2)$ and its discrete subgroups into a unified framework. It motivates us to develop equally general steerable 3D CNNs for rotations, as such theory is still missing in the 3D case.

Table 1. The comparison between PDO-s3DCNNs and other 3D steerable models. Ours can accommodate all common subgroups of $SO(3)$ and feature fields, while others can only address specific groups and feature fields.

	Group $\mathcal{G}$			Feature field			
Method	$\mathcal{G} \leq \mathcal{O}$	$\mathcal{I}$	$SO(3)$	Regular	Quotient	Irreducible	Data type
N-body networks (Kondor, 2018)			✓			✓	graphs
TFN (Thomas et al., 2018)			✓			✓	point clouds
CubeNets (Worrall & Brostow, 2018)	✓			✓			voxels
SE3CNNs (Weiler et al., 2018a)			✓			✓	voxels
SE(3)-Transformers (Fuchs et al., 2020)			✓			✓	point clouds/graphs
PDO-s3DCNNs (Ours)	✓	✓	✓	✓	✓	✓	voxels

2.2. Equivariant 3D Models

Some methods have achieved equivariance over 3D rotations by solving equivariant constraints, and they are successfully applied to graphs (Kondor, 2018), point clouds (Thomas et al., 2018) and volumetric data (Weiler et al., 2018a), respectively. Based on TFN (Thomas et al., 2018), Fuchs et al. (2020) proposed SE(3)-Transformer by designing equivariant self-attention. However, these methods can only deal with the continuous group $SO(3)$. Lang & Weiler (2020) provided a general characterization of equivariant filters for any compact group. Even so, the case for discrete subgroups of $SO(3)$ is still not solved for practical use due to the complexity of 3D rotations (Reisert & Burkhardt, 2009). There are also some works exploiting the equivariance over discrete subgroups. Worrall & Brostow (2018) proposed CubeNets by applying the idea of G-CNNs, which is equivariant over the cubic group and its subgroups. A similar idea is applied to the medical image analysis (Winkels & Cohen, 2019). However, these methods cannot exploit larger groups because there are no more rotational symmetries on regular 3D grids, while our method can break this limit. Chen et al. (2021) achieved approximate $SO(3)$-equivariance by discretizing the continuous group $SO(3)$ and sampling over point clouds, and their equivariance is very weak.

Besides, some methods achieve rotation equivariance on other data types. Specifically, some projected 3D objects to the sphere and designed rotation equivariant spherical CNNs (Cohen et al., 2018; Esteves et al., 2018a; Kondor et al., 2018; Esteves et al., 2020). Esteves et al. (2019) applied the idea of G-CNNs to process multiple views of 3D inputs. Unfortunately, these methods lose translation equivariance. By contrast, our method is translation equivariant because PDOs are naturally translation equivariant.

3. PDO-s3DCNNs

3.1. Prior Knowledge

Equivariance in 3D Equivariance indicates that the outputs of a mapping transform in a predictable way with the transformations of the inputs. To be specific, let Ψ be a filter, which could be represented by a layer of neural network from the input feature space to the output feature space, and $\mathcal{G}$ is a transformation group. Ψ is called group equivariant over $\mathcal{G}$ if it satisfies that $\forall g \in \mathcal{G}$,

$$\pi'(g)\left[\Psi\left[f\right]\right] = \Psi\left[\pi(g)[f]\right], \tag{1}$$

where f can be represented as a stack of feature maps f_k (for $k = 1, 2, \cdots, K$) in deep learning. For ease of derivation, we further suppose each f_k to be a smooth function over $\mathbb{R}^3$, then we have that $f \in C^\infty(\mathbb{R}^3, \mathbb{R}^K)$. $\pi(g)$ and $\pi'(g)$ are called *group actions*, where $\pi(g)$ describes how the transformation g acts on inputs, and $\pi'(g)$ allows us to "steer" the resulted feature $\Psi[f]$ without referring to the input f. In addition, since we hope that two transformations $g, h \in \mathcal{G}$ acting on the feature maps successively is equivalent to the composition of transformations $gh \in \mathcal{G}$ acting on the feature maps directly, we require that $\pi(g)\pi(h) = \pi(gh)$. The same is the case with $\pi'(g)$.

In this work, we focus on the equivariance over rotations, so $\mathcal{G}$ is taken as a rotation group in the sequel. Since we employ PDOs to design equivariant 3D CNNs, our models are naturally translation equivariant.

Feature Fields and Group Representations Now we examine rotation group actions $\pi(g)$. The corresponding feature f which is transformed according to g is called a *feature field*[2]. For example, for input 3D data, the action can be naturally formulated as

$$[\pi(g)f](x) = f(g^{-1}x), \tag{2}$$

and the corresponding feature is called a *scalar feature field*. If the output is also taken as a scalar feature, the filter Ψ should be isotropic and restricts the capacity of neural networks (Cohen et al., 2019).

In order to address this problem, we consider the following

[2] Without ambiguity, we may omit "field" sometimes for ease of presentation.

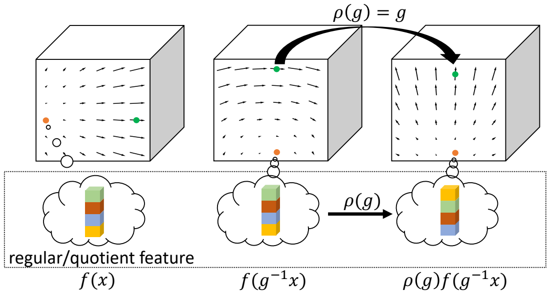

Figure 1. The schema of a rotation g acting on a ρ-field. If $\rho(g) = (1)$, each feature vector is moved to its new position; if $\rho(g) = g$, each feature vector additionally rotates according to g; if $\rho(g)$ is a regular or quotient representation, the values of each feature vector are permutated.

much more general feature field:

$$[\pi(g)f](x) = \rho(g) f\left(g^{-1}x\right), \tag{3}$$

where $\rho(g)$ is a $K \times K$ matrix and determines how a rotation g acts on the feature, and the corresponding feature field is called a ρ-field. $\pi(g)\pi(h) = \pi(gh)$ indicates that $\rho(g)\rho(h) = \rho(gh)$, so $\rho(g)$ is essentially a *group representation*[3] of the group $\mathcal{G}$. The simplest examples are the trivial representation $\rho(g) = (1)$, which is a one-dimensional identical matrix and exactly corresponds to the scalar feature field. When $\rho(g) = g$, where g can be parameterized as a 3×3 rotation matrix, each feature vector $f(x) \in \mathbb{R}^3$ should additionally rotate according to g. In this way, if a feature vector is used to predict the object orientation, it can transform properly as the inputs rotate. The schema of a rotation g acting on a ρ-field is shown in Figure 1.

As will be shown in Section 4, some common feature fields, e.g., regular, quotient, and irreducible features, which have been investigated in E2CNNs (Weiler & Cesa, 2019), are re-defined in the 3D case and can be easily addressed by our theory. By contrast, previous works on steerable 3D CNNs can only deal with one of these features. For example, CubeNets (Worrall & Brostow, 2018) and SE3CNNs (Weiler et al., 2018a) can only accommodate regular and irreducible features, respectively. Also, it is the first time that quotient features are investigated in the 3D case, which increases the flexibility of equivariant CNNs and allows us to decouple the computational cost from the group size.

3.2. PDO-based Equivariant 3D Filters

Shen et al. (2020; 2021) have shown that PDOs are very effective for deriving equivariant models, as they are convenient for mathematical derivation. Following them, we employ a combination of PDOs to define a 3D filter on the input function $f \in C^\infty(\mathbb{R}^3, \mathbb{R}^K)$:

$$\begin{aligned}\Psi[f] = \Big(& A_0 + A_1\partial_{x_1} + A_2\partial_{x_2} + A_3\partial_{x_3} + A_{11}\partial_{x_1^2} \\ & + A_{12}\partial_{x_1x_2} + A_{13}\partial_{x_1x_3} + A_{22}\partial_{x_2^2} + A_{23}\partial_{x_2x_3} \\ & + A_{33}\partial_{x_3^2}\Big)[f],\end{aligned} \tag{4}$$

where the coefficients $A_i \in \mathbb{R}^{K' \times K}$, and K' is the number of channels of the output feature. We employ the PDOs up to the second-order, and higher-order PDOs can be obtained by stacking multiple Ψ's. Then the requirement that Ψ is equivariant can be deduced to a linear constraint for the coefficients. Supposing that the input and output features are ρ- and ρ'-field, respectively, we have the following theorem[4].

Theorem 3.1. *Ψ is equivariant over $\mathcal{G}$, if and only if its coefficients satisfy the following linear constraints: $\forall g \in \mathcal{G}$,*

$$\begin{cases} \rho'(g)B_0 = B_0\rho(g), \\ \rho'(g)B_1 = B_1\left(g \otimes \rho(g)\right), \\ \rho'(g)B_2 = B_2\left(P\left(g \otimes g\right)P^\dagger \otimes \rho(g)\right), \end{cases} \tag{5}$$

i.e.,

$$\begin{cases} \left(I_K \otimes \rho'(g) - \rho(g)^T \otimes I_{K'}\right) vec(B_0) = 0, \\ \left(I_{3K} \otimes \rho'(g) - (g \otimes \rho(g))^T \otimes I_{K'}\right) vec(B_1) = 0, \\ \left(I_{6K} \otimes \rho'(g) - \left(P(g \otimes g)P^\dagger \otimes \rho(g)\right)^T \otimes I_{K'}\right) vec(B_2) = 0, \end{cases} \tag{6}$$

where

$$\begin{cases} B_0 = A_0, \\ B_1 = [A_1, A_2, A_3], \\ B_2 = [A_{11}, A_{12}, A_{13}, A_{22}, A_{23}, A_{33}], \end{cases}$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$P^\dagger$ is the Moore-Penrose inverse of P, I_K is a K-order identity matrix, $\otimes$ denotes the Kronecker product, the superscript "T" denotes the transpose operator, and $vec(B)$ is the vectorization operator that stacks the columns of B into a vector.

Proof sketch. We substitute Eqn. (4) into Eqn. (1), and use the coefficient comparison method to deduce the linear constraints.

That is to say, if the coefficients of PDO-based filter Ψ satisfy the constraints (5) or (6) for any $g \in \mathcal{G}$, then Ψ is equivariant over $\mathcal{G}$. Since Ψ is naturally translation equivariant, it

[3] This paper involves some terminologies in the group theory, and readers may refer to Appendix A and (Artin, 2011; Serre, 1977) for more details if interested.

[4] All the detailed proofs can be found in Appendices B and D.

is easy to verify that Ψ is equivariant over all combinations of rotations and translations (Thomas et al., 2018; Weiler et al., 2018a). Finally, according to the working spaces, we stack multiple layers Ψ's properly, inserted by nonlinearities that do not disturb the equivariance. As a result, we can get PDO-based steerable 3D CNNs, called PDO-s3DCNNs.

4. 3D Rotation Groups and Feature Fields

In this section, we show how to apply our theory to various 3D rotation groups and their representations. We provide some examples for ease of understanding. We emphasize that our theory can be applied to any representations, including irreps of discrete subgroups, not limited to these given ones. In all, common 3D rotation groups include the continuous group $SO(3)$ and its discrete subgroups.

4.1. Discrete Subgroups

Classification of Discrete Subgroups Up to conjugacy, any finite discrete 3D rotation group is one of the following groups (Artin, 2011): 1) C_N: the cyclic group of rotations by multiples of $2\pi/N$ about an axis; 2) D_N: the dihedral group of symmetries of a regular N-gon; 3) $\mathcal{T}$: the tetrahedral group of 12 rotational symmetries of a tetrahedron; 4) $\mathcal{O}$: the octahedral/cubic group of 24 rotational symmetries of an octahedron/cube; 5) $\mathcal{I}$: the icosahedral group of 60 rotational symmetries of an icosahedron.

Particularly, $D_2 = \mathcal{V}$ is also called Klein's four-group, which is the smallest non-cyclic group. The schema of groups $\mathcal{V}, \mathcal{T}, \mathcal{O}$, and $\mathcal{I}$ and their generators are shown in Appendix C. Particularly, two rotation groups being conjugate means that they are the symmetry groups of one polyhedron and its rotated version, respectively. We treat the groups in each conjugate class without any distinction, because 3D data always assume no preferred rotation transformations.

Regular and Quotient Features The most important group representation of a finite group is called *regular representation*, and the corresponding feature is called *regular feature*. To be specific, each feature vector $f(x)$ is $|\mathcal{G}|$-dimensional and indexed by $\mathcal{G}$, where $|\mathcal{G}|$ denotes the size of $\mathcal{G}$, i.e., the number of elements in $\mathcal{G}$. A transformation $g \in \mathcal{G}$ acts on $f(x) \in \mathbb{R}^{|\mathcal{G}|}$ in the way that it permutes the value $f_{\tilde{g}}(x)$ to $f_{g\tilde{g}}(x)$ for any $\tilde{g} \in \mathcal{G}$. It is easy to derive the regular representation $\rho(g)$ by referring to the Cayley Table.

Quotient features are closely related to regular features. Specifically, given a group $\mathcal{G}$ and its subgroup $\mathcal{H} \leq \mathcal{G}$, the left cosets $g\mathcal{H}$ of $\mathcal{H}$ partition $\mathcal{G}$ properly, where $g \in \mathcal{G}$, and we denote the set of left cosets as $\mathcal{G}/\mathcal{H}$. As for $\mathcal{H}$-quotient features, each feature vector $f(x) \in \mathbb{R}^{|\mathcal{G}|/|\mathcal{H}|}$ is indexed by $\mathcal{G}/\mathcal{H}$. A transformation $g \in \mathcal{G}$ acts on $f(x)$ in the way that it permutes the value $f_{\tilde{g}\mathcal{H}}(x)$ to $f_{g\tilde{g}\mathcal{H}}(x)$, and the corresponding group representation $\rho(g)$ can be obtained similarly. If $\mathcal{H} = \{e\}$, where e is the identity element of $\mathcal{G}$, then an $\mathcal{H}$-quotient feature is equivalent to a regular feature, indicating that the regular feature is essentially a special case of the quotient feature.

Table 2. $(n_{B_0}, n_{B_1}, n_{B_2})$ denote the dimensions of solution spaces of B_0, B_1 and B_2. $\rho(g)$ and $\rho'(g)$ denote the group representations of the input and the output feature fields, respectively.

$\rho(g)$ \ $\rho'(g)$	Trivial	$\mathcal{T}$-quotient	$\mathcal{V}$-quotient	Regular
Trivial	$(1,0,1)$	$(1,0,1)$	$(1,0,3)$	$(1,3,6)$
$\mathcal{T}$-quotient	$(1,0,1)$	$(2,0,2)$	$(2,0,6)$	$(2,6,12)$
$\mathcal{V}$-quotient	$(1,0,3)$	$(2,0,6)$	$(6,0,18)$	$(6,18,36)$
Regular	$(1,3,6)$	$(2,6,12)$	$(6,18,36)$	$(24,72,144)$

An advantage of quotient features over regular features is that the number of channels required for each feature is reduced by $|\mathcal{H}|$ times. For example, for the octahedral group $\mathcal{O}$, a regular feature contains 24 channels. Noting that $\mathcal{V}$ and $\mathcal{T}$ are both subgroups of $\mathcal{O}$, we can employ $\mathcal{V}$- and $\mathcal{T}$-quotient features, then each feature only contains 6 and 2 channels, respectively. We emphasize that although it seems that $\mathcal{O}/\mathcal{V} \cong D_3$ and $\mathcal{O}/\mathcal{T} \cong C_2$, where $\cong$ denotes the group isomorphism, employing quotient features does not indicate that the equivariance is reduced to the quotient space, because we still require the equivariance constraints to be satisfied for any g in $\mathcal{G}$.

Equivariant Kernels over Discrete Subgroups As derived in Section 3.2, given the input and the output feature fields, solving an equivariant kernel is no more difficult than solving the homogeneous linear equations, Eqn. (6). However, Eqn. (6) still needs to be satisfied for any $g \in \mathcal{G}$, and it is time-consuming to solve if $\mathcal{G}$ is very large. Fortunately, it is easy to verify that as long as Eqn. (1) or (5) is satisfied for the generators of $\mathcal{G}$, Ψ can be an equivariant kernel over $\mathcal{G}$ (see Theorem B.2 in Appendix). In this way, the bases of the coefficients B_0, B_1 and B_2 can be efficiently computed offline using singular value decomposition. For example, when $\mathcal{G} = \mathcal{O}$, the dimension of each solution space is given in Table 2. Then we parameterize B_0, B_1 and B_2 as linear combinations of their bases, and the coefficients are learnable. Finally, they are substituted into Eqn. (4) to obtain a parameterized equivariant kernel Ψ, which is learnable.

4.2. Continuous Group $SO(3)$

As for the $SO(3)$ group, we cannot leverage regular or quotient features, because $SO(3)$ is an infinite group. Following SE3CNN (Weiler et al., 2018a), we use irreducible features instead, which are realized by irreps. Specifically, any representation of $SO(3)$ can be decomposed into irreps of dimension $2l+1$, for $l = 0, 1, 2, \cdots, \infty$. The irrep acting on the l-order irreducible features $f(x) \in \mathbb{R}^{2l+1}$ is known as the Wigner-D matrix of order l, denoted as $D^l(\rho)$.

Particularly, $D^0(\rho) = (1)$, which exactly corresponds to the trivial representation. Then we solve the linear equations (6) given the input and the output features. However, the difficulty lies in that Eqn. (6) essentially contains infinite equations as $SO(3)$ contains infinite elements, which is impossible to solve directly. Fortunately, we have the following observation, and then the equivariant filters can be solved easily.

Theorem 4.1. *If Eqn. (1) or (5) is satisfied for $g_1 = Z(1)$ and $g_2 = Y(1)$, where*

$$Z(\alpha) = \begin{bmatrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}, Y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix},$$

then $\forall g \in SO(3)$, Eqn. (1) is satisfied.

Proof intuition. Any elements in $SO(3)$ can be approximately generated by $Z(1)$ and $Y(1)$.

5. Implementation

We have introduced equivariant 3D filters in the continuous domain. They are easy to implement in the discrete domain by discretizing PDOs using the given discrete data.

5.1. Discretization

FD This is a similar discretization method to that adopted in PDO-eConvs (Shen et al., 2020). For the volumetric data/feature $\boldsymbol{F}$, it can be viewed as a sample from a continuous function f defined on $\mathbb{R}^3$. Then PDOs acting on f can be discretized as convolutional kernels acting on $\boldsymbol{F}$ using FD. Formally, we have that $\forall i \in S_x$,

$$\partial_i[f] = u_i * \boldsymbol{F} + O\left(\rho^2\right),$$

where $S_x = \{x_1, x_2, x_3, x_1^2, x_2^2, x_3^2, x_1x_2, x_1x_3, x_2x_3\}$, u_i's are the FD schemes of PDOs (see Appendix D.1), $*$ denotes the convolution operation, and ρ is the diameter of a 3D grid. Since Ψ is essentially a combination of PDOs, it can be discretized as a convolutional filter $\hat{\Psi}$. Particularly, we observe that all the PDOs up to the second-order can be approximated using $3 \times 3 \times 3$ convolutional filters with a quadratic precision, so $\hat{\Psi}$ is also of the size $3 \times 3 \times 3$, and the equivariance error resulted from FD discretization is of the quadratic order.

Gaussian Discretization We can also employ the derivatives of Gaussian function for estimation (Jenner & Weiler, 2021): given points $x^n \in \mathbb{R}^3$, $\forall i \in S_x$,

$$\partial_i[f](x) \approx \sum_{n=1}^{N} \partial_i G(x^n; \sigma) f(x + x^n),$$

where $G(x;\sigma)$ is a Gaussian kernel with standard deviation σ around 0.

Extension to Point Clouds Since PDOs can also be approximated based on irregular point clouds (Liang & Zhao, 2013), our models can also operate on point clouds. In practice, we operate on the volumetric data because they are more general than point cloud data: point clouds can always be voxelized, but not vice versa, such as medical images. Also, regular grids can be processed more efficiently on current hardware.

5.2. Common Deep Learning Techniques

Nonlinearities and Batch Normalization (BN) In equivariant models, inserted nonlinearities and BN should be compatible with employed features to preserve equivariance. For trivial, regular, and quotient features, pointwise nonlinearities, e.g., ReLU, are admissible because their representations are realized by permutation matrices. As for BN, it should be implemented with a single scale and a single bias per feature, instead of per channel. In the continuous case, as the irreducible features are not realized by permutation matrices, pointwise nonlinearities and standard BN are not admissible. We instead use the scale-BN and scale-like nonlinearities, e.g., gated nonlinearities, to preserve equivariance (see Appendix D.2). However, compared with standard BN, scale-BN cannot correct activations to be zero-mean, and thus is not as beneficial as standard BN for internal covariate shift reduction and optimization (Bjorck et al., 2018). Besides, as proven in (Finzi et al., 2021), scale-like nonlinearities are not sufficient for universality, and can limit the model performance. These disadvantages also account for why most related works (Weiler & Cesa, 2019; Jenner & Weiler, 2021) employ regular or quotient representations of discrete subgroups for implementation instead of irreps.

Composition of Basic Feature Fields Analogy to multiple channels, we can also put multiple basic feature fields together, e.g., regular, quotient and irreducible ones, to acquire more general feature fields (see Appendix D.3).

6. Experiments

We perform extensive experiments to evaluate the performance of our models. The experimental details for each task are provided in Appendix E.

6.1. 3D Tetris (Testing 3D Equivariance)

We confirm the equivariance of our models on 3D Tetris, which is a common dataset for testing 3D rotation equivariance (Thomas et al., 2018; Weiler et al., 2018a). We feed the dataset in a single orientation into the network during training, and then test the network with rotated shapes.

Firstly, we discretize PDOs using FD, and test three $\mathcal{O}$-steerable models with regular, $\mathcal{V}$- and $\mathcal{T}$-quotient features,

Table 3. The test accuracy of the $\mathcal{O}$- and $SO(3)$- steerable CNNs discretized by FD on 3D Tetris with cubic rotations.

Group	Feature field	Test acc. (%)	# Params	Time
$\mathcal{O}$	Regular	100.0 ± 0.0	31k	14.3s
$\mathcal{O}$	$\mathcal{V}$-quotient	100.0 ± 0.0	5.5k	2.3s
$\mathcal{O}$	$\mathcal{T}$-quotient	100.0 ± 0.0	2.2k	1.3s
$SO(3)$	Irreducible	100.0 ± 0.0	22.8k	66.7s

Table 4. The test accuracy of the $SO(3)$-steerable CNNs on 3D Tetris with arbitrary rotations.

Discretization	Kernel size	Test acc. (%)	Time
FD	$3 \times 3 \times 3$	18.20 ± 3.13	66.7s
Gaussian	$3 \times 3 \times 3$	29.99 ± 4.98	67.3s
Gaussian	$5 \times 5 \times 5$	99.04 ± 0.14	109.5s

on 3D Tetris with cubic rotations. As shown in Table 3, all the models achieve 100% test accuracies. Compared with regular features, quotient features have fewer channels, and thus they have a lower computational cost. Although we use a small dataset here, the complexity comparison can be scaled to larger datasets and models. Besides, the $SO(3)$-steerable CNN also achieves 100% test accuracy under this setting. Theoretically, our $\mathcal{G}$-steerable CNNs with FD discretization can achieve exactly equivariance over the cubic group $\mathcal{O}$ in the discrete domain if $\mathcal{G}$ includes $\mathcal{O}$ as a subgroup (see Theorem D.1 in Appendix).

We then test our $SO(3)$-steerable model on the shapes with arbitrary rotations, and FD cannot perform well (see Table 4), because inputs are assumed to be smooth while real data are always non-smooth especially on the edge. Considering that Gaussian functions can help smooth features, we then employ Gaussian discretization and find that it performs very well when using a kernel size of $5 \times 5 \times 5$. Also, we provide the equivariance error analysis in Appendix E.1.

6.2. SHREC'17 Retrieval

The SHREC'17 retrieval task (Savva et al., 2017) contains 51,162 models of 3D shapes belonging to 55 classes. This dataset is divided into 35,764 training samples, 5,133 validation samples, and 10,265 test samples. We focus on the "perturbed" version where models are arbitrarily rotated. The retrieval performance is given by the average value of the mean average precisions (mAP) of micro-average version and macro-average version, denoted as the *score*. We convert these 3D models into voxels of size $64 \times 64 \times 64$.

Firstly, we investigate the impact of the equivariance group. We evaluate the performance of $\mathcal{V}$-, $\mathcal{T}$-, $\mathcal{O}$-, and $\mathcal{I}$-steerable CNNs with regular features. The $\mathcal{I}$-steerable one is discretized using Gaussian discretization as its equivariance exceeds the symmetries of regular 3D grids, and others using FD. We adjust the feature numbers so that these models use comparable numbers of parameters (about 0.15M) for a fair comparison. As shown in Table 5, the $\mathcal{O}$-steerable model performs better than $\mathcal{V}$- and $\mathcal{T}$-steerable ones, as it can exploit more symmetries. However, for the $\mathcal{I}$-steerable CNN, we find it too large (each feature contains 60 channels) and slow to converge during training, and it performs inferiorly to the $\mathcal{O}$-steerable one. We then evaluate $SO(3)$-steerable models with irreducible features. When using FD for discretization, it performs significantly inferiorly to the $\mathcal{O}$-steerable one even though it can also achieve exact equivariance over $\mathcal{O}$. A similar phenomenon has also occurred in E2CNNs (Weiler & Cesa, 2019) in the 2D case, where C_N-steerable models significantly outperform $SO(2)$-steerable models. We argue that this is because irreducible features are not amenable to conventional BN and pointwise nonlinearities, which degrades the model performance in practice. When using Gaussian discretization, the performance gets better due to the improved equivariance, but the disadvantage still exists (58.3 vs. 58.6). In addition, the larger kernel size ($5 \times 5 \times 5$) results in a larger computational cost.

Table 5. The retrieval performance of $\mathcal{V}$-, $\mathcal{T}$-, $\mathcal{O}$-, $\mathcal{I}$- and $SO(3)$-steerable CNNs, tested on SHREC'17.

Group	Discretization	Feature field	Score
$\mathcal{V}$	FD	Regular	52.7
$\mathcal{T}$	FD	Regular	57.6
$\mathcal{O}$	FD	Regular	**58.6**
$\mathcal{I}$	Gaussian	Regular	55.5
$SO(3)$	FD	Irreducible	57.4
$SO(3)$	Gaussian	Irreducible	58.3

Then we compare our methods to various equivariant 3D models, as shown in Table 6. Our $SO(3)$-steerable model performs much better than its counterpart SE3CNN using comparable numbers of parameters, showing great expressive ability of our PDO-based filters. Compared with some recent point clouds based rotation equivariant models (KIM et al., 2020; Chen et al., 2021; Li et al., 2021), our method performs better, as volumetric data are more regular and can provide more structured and compact information. Also, our method shows great parameter efficiency (0.15M vs. 2.9M+). In addition, our method outperforms some equivariant spherical CNNs (Cohen et al., 2018; Kondor et al., 2018; Esteves et al., 2018a; Banerjee et al., 2020; Cobb et al., 2020), where 3D shapes are projected to a sphere to achieve rotation equivariance. Spherical models perform worse as they are not translation equivariant while ours are.

In addition, we examine the effectiveness of quotient features. The $\mathcal{O}$-steerable model using $\mathcal{V}$-quotient features performs inferiorly to the model using regular features (55.5

Table 6. SHREC' 17 perturbed dataset results. Mixed features mean that the features are composed of regular and $\mathcal{V}$-quotient features. Score is the average value of mAP of micro-average version and macro-average version.

Method	Score	micro			macro			#Params	Input
		P@N	R@N	mAP	P@N	R@N	mAP		
EPN (Chen et al., 2021)	55.3	68.1	68.1	64.5	46.2	54.8	46.1	7.9M	point clouds
RI-GCN (KIM et al., 2020)	56.2	69.1	68.0	64.5	47.4	57.0	47.8	4.4M	point clouds
(Li et al., 2021) (with TTA)	56.5	69.4	69.4	65.8	48.1	56.0	47.2	2.9M	point clouds
S2CNN (Cohen et al., 2018)	-	70.1	71.1	67.6	-	-	-	1.4M	spherical
FFS2CNN (Kondor et al., 2018)	-	70.7	72.2	68.3	-	-	-	-	spherical
VolterraNet (Banerjee et al., 2020)	-	71.0	70.0	67.0	-	-	-	0.4M	spherical
(Esteves et al., 2018a)	56.5	71.7	**73.7**	68.5	45.0	55.0	44.4	0.5M	spherical
(Cobb et al., 2020)	-	71.9	71.0	67.9	-	-	-	0.25M	spherical
SE3CNN (Weiler et al., 2018a)	55.5	70.4	70.6	66.1	49.0	54.9	44.9	0.14M	voxels
Ours ($SO(3)$)	58.3	73.1	73.4	69.3	**52.5**	55.4	47.3	0.15M	voxels
Ours ($\mathcal{O}$ with regular features)	58.6	72.9	73.0	68.8	51.9	57.7	48.3	0.15M	voxels
Ours ($\mathcal{O}$ with $\mathcal{V}$-quotient features)	55.5	69.2	69.6	65.0	48.0	56.3	46.0	0.15M	voxels
Ours ($\mathcal{O}$ with mixed features)	**59.1**	**73.2**	73.3	**69.3**	51.7	**57.8**	**48.8**	0.15M	voxels

Table 7. The performance on the ISBI 2012 Challenge evaluated and compared by V_{rand}, which are explained in (Arganda-Carreras et al., 2015). Larger is better.

Method	V_{rand}
U-Net (Ronneberger et al., 2015)	0.96210
CENet (Zhou et al., 2022)	0.96960
FusionNet (Quan et al., 2016)	0.97804
CubeNet (Worrall & Brostow, 2018)	0.98018
SFCNN (Weiler et al., 2018b)	0.98680
PDO-s3DCNN ($\mathcal{V}$)	0.98415
PDO-s3DCNN ($\mathcal{O}$)	**0.98727**

vs. 58.6), mainly because its filters essentially do not employ the first order PDOs. As shown in Table 2, when the input and the output feature fields are both quotient features, the coefficients of the first order PDOs, B_1, can only be $\mathbf{0}$. An ablation study to justify the choice of PDOs is in Appendix E.2. When we employ a feature field composed of regular and quotient features, the performance can be drastically improved (from 55.5 to 59.1), indicating that quotient features can be effective supplements for the regular features.

6.3. ISBI 2012 Challenge

ISBI 2012 Challenge (Arganda-Carreras et al., 2015) is a volumetric boundary segmentation benchmark, and the target is to segment Drosophila ventral nerve cords from a serial-section transmission electron microscopy image. This dataset is suitable for evaluating a rotation-equivariant model, because biomedical images always have no inherent orientations. The full training image is $512 \times 512 \times 30$ voxels in shape, and each voxel is $4 \times 4 \times 50$ nm^3. The setting is the same for test images. Following the official ranking list, we compare different models using the metrics V_{rand}, which is explained in (Arganda-Carreras et al., 2015). Larger is better.

Firstly, we evaluate the $\mathcal{V}$-steerable model. We replace the equivariant filters in CubeNet (Worrall & Brostow, 2018) by ours with regular features. We keep the architecture and the numbers of features the same as that in CubeNet to maintain a fair comparison. As shown in Table 7, our model performs better than CubeNet with a much lower network complexity (4.4M vs. 11.9M), noting that on average each learnable filter of our model contains 10 parameters, while that of CubeNet contains 27, which shows a great storage advantage. We then employ an $\mathcal{O}$-steerable model (use comparable numbers of parameters with the $\mathcal{V}$-steerable one) and achieve a better result, because it can exploit more symmetries. Particularly, CubeNet cannot exploit $\mathcal{O}$-equivariance, because the voxels here are not cubic. Our model even outperforms SFCNN (Weiler et al., 2018b) (0.98727 vs. 0.98680), which additionally uses a task-specific lifting multi-cut (Beier et al., 2016) post-processing to improve performance, while we do not use any post-processing.

7. Conclusions

We employ PDOs to establish general 3D steerable CNNs, which cover both the continuous group $SO(3)$ and its discrete subgroups. Our theory can also deal with various feature fields easily, such as regular, quotient, and irreducible features. Experiments verify that our methods can preserve equivariance well in the discrete domain and perform better than previous works with a low network complexity.

Actually, besides volumetric data, our method can also operate on the point cloud data. In addition, our theory can be applied to arbitrary representations, while we only work on several given representations in practice. We will explore more possibilities in future work.

References

Anderson, B., Hy, T. S., and Kondor, R. Cormorant: Covariant molecular neural networks. In *NeurIPS*, pp. 14537–14546, 2019.

Arganda-Carreras, I., Turaga, S. C., Berger, D. R., Cireşan, D., Giusti, A., Gambardella, L. M., Schmidhuber, J., Laptev, D., Dwivedi, S., Buhmann, J. M., et al. Crowdsourcing the creation of image segmentation algorithms for connectomics. *Frontiers in Neuroanatomy*, 9:142, 2015.

Artin, M. *Algebra*. Pearson Prentice Hall, 2011. ISBN 9780132413770. URL https://books.google.com/books?id=S6GSAgAAQBAJ.

Banerjee, M., Chakraborty, R., Bouza, J., and Vemuri, B. C. Volterranet: A higher order convolutionalnetwork with group equivariance forhomogeneous manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

Beier, T., Andres, B., Köthe, U., and Hamprecht, F. A. An efficient fusion move algorithm for the minimum cost lifted multicut problem. In *ECCV*, pp. 715–730. Springer, 2016.

Bjorck, J., Gomes, C., Selman, B., and Weinberger, K. Q. Understanding batch normalization. In *NeurIPS*, pp. 7705–7716, 2018.

Chen, H., Liu, S., Chen, W., Li, H., and Hill, R. Equivariant point network for 3d point cloud analysis. In *CVPR*, pp. 14514–14523, 2021.

Cobb, O. J., Wallis, C. G., Mavor-Parker, A. N., Marignier, A., Price, M. A., d'Avezac, M., and McEwen, J. D. Efficient generalized spherical CNNs. In *ICLR*, 2020.

Cohen, T. and Welling, M. Group equivariant convolutional networks. In *ICML*, pp. 2990–2999, 2016.

Cohen, T., Weiler, M., Kicanaoglu, B., and Welling, M. Gauge equivariant convolutional networks and the icosahedral CNN. In *ICML*, pp. 1321–1330, 2019.

Cohen, T. S. and Welling, M. Steerable CNNs. In *ICLR*, 2017.

Cohen, T. S., Geiger, M., Köhler, J., and Welling, M. Spherical CNNs. In *ICLR*, 2018.

Esteves, C., Allen-Blanchette, C., Makadia, A., and Daniilidis, K. Learning SO(3) equivariant representations with spherical CNNs. In *ECCV*, pp. 52–68, 2018a.

Esteves, C., Allenblanchette, C., Zhou, X., and Daniilidis, K. Polar transformer networks. In *ICLR*, 2018b.

Esteves, C., Xu, Y., Allen-Blanchette, C., and Daniilidis, K. Equivariant multi-view networks. In *ICCV*, pp. 1568–1577, 2019.

Esteves, C., Makadia, A., and Daniilidis, K. Spin-weighted spherical cnns. *NeurIPS*, 33, 2020.

Finzi, M., Stanton, S., Izmailov, P., and Wilson, A. Generalizing convolutional networks for equivariance to Lie groups on arbitrary continuous data. In *ICML*, 2020.

Finzi, M., Welling, M., and Wilson, A. G. A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In *ICML*, 2021.

Fuchs, F., Worrall, D., Fischer, V., and Welling, M. SE(3)-Transformers: 3D roto-translation equivariant attention networks. *NeurIPS*, 33, 2020.

Glorot, X. and Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256. JMLR Workshop and Conference Proceedings, 2010.

Hoogeboom, E., Peters, J. W., Cohen, T. S., and Welling, M. HexaConv. In *ICLR*, 2018.

Jenner, E. and Weiler, M. Steerable partial differential operators for equivariant neural networks. *arXiv preprint arXiv:2106.10163*, 2021.

KIM, S., Park, J., and Han, B. Rotation-invariant local-to-global representation learning for 3d point cloud. *NeurIPS*, 33, 2020.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Kondor, R. N-body networks: a covariant hierarchical neural network architecture for learning atomic potentials. *arXiv preprint arXiv:1803.01588*, 2018.

Kondor, R., Lin, Z., and Trivedi, S. Clebsch–Gordan Nets: a fully fourier space spherical convolutional neural network. *NeurIPS*, 31:10117–10126, 2018.

Lang, L. and Weiler, M. A wigner-eckart theorem for group equivariant convolution kernels. In *ICLR*, 2020.

Lenc, K. and Vedaldi, A. Understanding image representations by measuring their equivariance and equivalence. In *CVPR*, pp. 991–999, 2015.

Li, F., Fujiwara, K., Okura, F., and Matsushita, Y. A closer look at rotation-invariant deep point cloud analysis. In *ICCV*, pp. 16218–16227, 2021.

Liang, J. and Zhao, H. Solving partial differential equations on point clouds. *SIAM Journal on Scientific Computing*, 35(3):A1461–A1486, 2013.

Marcos, D., Volpi, M., Komodakis, N., and Tuia, D. Rotation equivariant vector field networks. In *ICCV*, pp. 5048–5057, 2017.

Quan, T. M., Hildebrand, D. G., and Jeong, W.-K. FusionNet: A deep fully residual convolutional neural network for image segmentation in connectomics. *arXiv preprint arXiv:1612.05360*, 2016.

Reisert, M. and Burkhardt, H. Spherical tensor calculus for local adaptive filtering. In *Tensors in Image Processing and Computer Vision*, pp. 153–178. Springer, 2009.

Ronneberger, O., Fischer, P., and Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pp. 234–241. Springer, 2015.

Savva, M., Yu, F., Su, H., Kanezaki, A., Furuya, T., Ohbuchi, R., Zhou, Z., Yu, R., Bai, S., Bai, X., et al. Large-scale 3D shape retrieval from ShapeNet Core55: SHREC'17 track. In *Proceedings of the Workshop on 3D Object Retrieval*, pp. 39–50. Eurographics Association, 2017.

Serre, J.-P. *Linear representations of finite groups*, volume 42. Springer, 1977.

Shen, Z., He, L., Lin, Z., and Ma, J. PDO-eConvs: Partial differential operator based equivariant convolutions. In *ICML*, 2020.

Shen, Z., Shen, T., Lin, Z., and Ma, J. PDO-eS2CNNs: Partial differential operator based equivariant spherical CNNs. In *AAAI*, 2021.

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

Weiler, M. and Cesa, G. General E(2)-equivariant steerable CNNs. In *NeurIPS*, pp. 14334–14345, 2019.

Weiler, M., Geiger, M., Welling, M., Boomsma, W., and Cohen, T. S. 3D steerable CNNs: Learning rotationally equivariant features in volumetric data. In *NeurIPS*, pp. 10381–10392, 2018a.

Weiler, M., Hamprecht, F. A., and Storath, M. Learning steerable filters for rotation equivariant CNNs. In *CVPR*, pp. 849–858, 2018b.

Winkels, M. and Cohen, T. S. Pulmonary nodule detection in CT scans with equivariant CNNs. *Medical image analysis*, 55:15–26, 2019.

Worrall, D. and Brostow, G. CubeNet: Equivariance to 3D rotation and translation. In *ECCV*, pp. 567–584, 2018.

Worrall, D. E., Garbin, S. J., Turmukhambetov, D., and Brostow, G. J. Harmonic networks: Deep translation and rotation equivariance. In *CVPR*, pp. 5028–5037, 2017.

Zhou, Q., Wu, X., Zhang, S., Kang, B., Ge, Z., and Latecki, L. J. Contextual ensemble network for semantic segmentation. *Pattern Recognition*, 122:108290, 2022.

Zhou, Y., Ye, Q., Qiu, Q., and Jiao, J. Oriented response networks. In *CVPR*, pp. 519–528, 2017.

A. Groups and Group Representations

Groups

- Closure: $\forall g_1, g_2 \in \mathcal{G}, g_1 \circ g_2 \in \mathcal{G}$;
- Associativity: $\forall g_1, g_2, g_3 \in \mathcal{G}, (g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$;
- Identity: There exists an identity element $e \in \mathcal{G}$ such that $e \circ g = g \circ e = g$ for all $g \in \mathcal{G}$;
- Inverses: $\forall g \in \mathcal{G}$, there exists a $g^{-1} \in \mathcal{G}$, such that $g^{-1} \circ g = g \circ g^{-1} = e$.

In practice, we always omit writing the binary composition operator $\circ$, so we would write gh instead of $g \circ h$. Groups can be finite or infinite.

Group Representations A group representation $\rho : \mathcal{G} \to GL(N)$ is a mapping from a group $\mathcal{G}$ to the set of $N \times N$ invertible matrices $GL(N)$. Critically, ρ satisfies the following property:

$$\forall g_1, g_2 \in \mathcal{G}, \quad \rho(g_1 g_2) = \rho(g_1)\,\rho(g_2).$$

Group Isomorphism Given two groups $\mathcal{G}_1$ and $\mathcal{G}_2$, the two groups are isomorphic if there exists an isomorphism from one to the other, denoted as $\mathcal{G}_1 \cong \mathcal{G}_2$. Spelled out, this means that there is a bijective function $\sigma : \mathcal{G}_1 \to \mathcal{G}_2$ such that

$$\forall u, v \in \mathcal{G}_1, \quad \sigma(uv) = \sigma(u)\sigma(v).$$

B. The Proofs of Theorems

We define a convolution on the input function $f \in C^\infty(\mathbb{R}^3, \mathbb{R}^K)$:

$$\Psi[f] = \Big(A_0 + A_1\partial_{x_1} + A_2\partial_{x_2} + A_3\partial_{x_3} + A_{11}\partial_{x_1^2} + A_{12}\partial_{x_1x_2} + A_{13}\partial_{x_1x_3} + A_{22}\partial_{x_2^2} + A_{23}\partial_{x_2x_3} + A_{33}\partial_{x_3^2}\Big)[f], \tag{7}$$

where the coefficients $A_i \in \mathbb{R}^{K' \times K}$. In order that Ψ is an equivariant mapping over $\mathcal{G}$, it should satisfy the equivariance condition: $\forall g \in \mathcal{G}$,

$$\pi'(g)\left[\Psi[f]\right] = \Psi\left[\pi(g)[f]\right], \tag{8}$$

where

$$\begin{cases} [\pi(g)f](x) = \rho(g) f\left(g^{-1}x\right), \\ \left[\pi'(g)f\right](x) = \rho'(g) f\left(g^{-1}x\right). \end{cases}$$

To solve this equivariance requirement, we have the following theorem.

Theorem B.1. *Ψ is equivariant over $\mathcal{G}$, if and only if its coefficients satisfy the following linear constraints: $\forall g \in \mathcal{G}$,*

$$\begin{cases} \rho'(g)B_0 = B_0\rho(g), \\ \rho'(g)B_1 = B_1\left(g \otimes \rho(g)\right), \\ \rho'(g)B_2 = B_2\left(P\left(g \otimes g\right)P^\dagger \otimes \rho(g)\right), \end{cases} \tag{9}$$

i.e.,

$$\begin{cases} \left(I_K \otimes \rho'(g) - \rho(g)^T \otimes I_{K'}\right) vec(B_0) = 0, \\ \left(I_{3K} \otimes \rho'(g) - (g \otimes \rho(g))^T \otimes I_{K'}\right) vec(B_1) = 0, \\ \left(I_{6K} \otimes \rho'(g) - \left(P(g \otimes g)P^\dagger \otimes \rho(g)\right)^T \otimes I_{K'}\right) vec(B_2) = 0, \end{cases} \tag{10}$$

where

$$\begin{cases} B_0 = A_0, \\ B_1 = \left[A_1, A_2, A_3\right], \\ B_2 = \left[A_{11}, A_{12}, A_{13}, A_{22}, A_{23}, A_{33}\right], \end{cases}$$

$$P = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 0 & 0 & 1/2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

$P^\dagger$ is the Moore-Penrose inverse of P, I_K is a K-order identity matrix, $\otimes$ denotes the Kronecker product, the superscript "T" denotes the transpose operator, and $vec(B)$ is the vectorization operator that stacks the columns of B into a vector.

Proof. we substitute (7) into the equivariance condition Eqn. (8) and get that for any $g \in \mathcal{G}$,

$$\begin{aligned} &\rho'(g)\Big(A_0 + A_1\partial_{x_1} + A_2\partial_{x_2} + A_3\partial_{x_3} + A_{11}\partial_{x_1^2} \\ &+ A_{12}\partial_{x_1x_2} + A_{13}\partial_{x_1x_3} + A_{22}\partial_{x_2^2} + A_{23}\partial_{x_2x_3} \\ &+ A_{33}\partial_{x_3^2}\Big)[f]\left(g^{-1}x\right) \\ =&\Big(A_0 + A_1\partial_{x_1} + A_2\partial_{x_2} + A_3\partial_{x_3} + A_{11}\partial_{x_1^2} \\ &+ A_{12}\partial_{x_1x_2} + A_{13}\partial_{x_1x_3} + A_{22}\partial_{x_2^2} + A_{23}\partial_{x_2x_3} \\ &+ A_{33}\partial_{x_3^2}\Big)\left[\rho(g)f\left(g^{-1}x\right)\right]. \end{aligned} \tag{11}$$

We denote that $\forall i = 1, 2, \cdots, K$,

$$\nabla^T[f_i] = \left[\partial_{x_1}[f_i], \partial_{x_2}[f_i], \partial_{x_3}[f_i]\right],$$

$$\nabla^2[f_i] = \begin{bmatrix} \partial_{x_1^2}[f_i] & \partial_{x_1x_2}[f_i] & \partial_{x_1x_3}[f_i] \\ \partial_{x_1x_2}[f_i] & \partial_{x_2^2}[f_i] & \partial_{x_2x_3}[f_i] \\ \partial_{x_1x_3}[f_i] & \partial_{x_2x_3}[f_i] & \partial_{x_3^2}[f_i] \end{bmatrix}.$$

Then the left hand side of Eqn. (11) can be written as

$$\rho'(g)\left(B_0 f\left(g^{-1}x\right) + B_1 V_1 + \bar{B}_2\bar{V}_2\right), \tag{12}$$

where

$$\bar{B}_2 = \left[A_{11}, \frac{A_{12}}{2}, \frac{A_{13}}{2}, \frac{A_{12}}{2}, A_{22}, \frac{A_{23}}{2}, \frac{A_{13}}{2}, \frac{A_{23}}{2}, A_{33}\right],$$

$$\begin{aligned}V_1 =& vec\begin{bmatrix} \nabla^T\left[f_1\right]\left(g^{-1}x\right) \\ \nabla^T\left[f_2\right]\left(g^{-1}x\right) \\ \cdots \\ \nabla^T\left[f_K\right]\left(g^{-1}x\right) \end{bmatrix} \\ =& vec\left[\partial_{x_1}[f]\left(g^{-1}x\right), \partial_{x_2}[f]\left(g^{-1}x\right), \partial_{x_3}[f]\left(g^{-1}x\right)\right],\end{aligned}$$

and

$$\begin{aligned}\bar{V}_2 =& vec\begin{bmatrix} vec^T\left(\nabla^2\left[f_1\right]\left(g^{-1}x\right)\right) \\ vec^T\left(\nabla^2\left[f_2\right]\left(g^{-1}x\right)\right) \\ \cdots \\ vec^T\left(\nabla^2\left[f_K\right]\left(g^{-1}x\right)\right) \end{bmatrix} \\ =& vec\Big[\partial_{x_1^2}[f]\left(g^{-1}x\right), \partial_{x_1x_2}[f]\left(g^{-1}x\right), \partial_{x_1x_3}[f]\left(g^{-1}x\right), \\ & \partial_{x_1x_2}[f]\left(g^{-1}x\right), \partial_{x_2^2}[f]\left(g^{-1}x\right), \partial_{x_2x_3}[f]\left(g^{-1}x\right), \\ & \partial_{x_1x_3}[f]\left(g^{-1}x\right), \partial_{x_2x_3}[f]\left(g^{-1}x\right), \partial_{x_3^2}[f]\left(g^{-1}x\right)\Big].\end{aligned}$$

In addition, we have

$$\bar{B}_2 = B_2\left(P \otimes I_K\right).$$

Then, we denote that

$$\begin{aligned}V_2 =& vec\Big[\partial_{x_1^2}[f]\left(g^{-1}x\right), \partial_{x_1x_2}[f]\left(g^{-1}x\right), \partial_{x_1x_3}[f]\left(g^{-1}x\right), \\ & \partial_{x_2^2}[f]\left(g^{-1}x\right), \partial_{x_2x_3}[f]\left(g^{-1}x\right), \partial_{x_3^2}[f]\left(g^{-1}x\right)\Big],\end{aligned}$$

and we have

$$\bar{V}_2 = \left(P^\dagger \otimes I_K\right)V_2,$$

where

$$P^\dagger = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix},$$

and it is easy to verify that $P^\dagger$ is exactly the Moore-Penrose inverse of P. As a result, Eqn. (12) can be further written as

$$\begin{aligned}&\rho'(g)\left(B_0 f\left(g^{-1}x\right) + B_1V_1 + \bar{B}_2\bar{V}_2\right) \\ =&\rho'(g)\left(B_0 f\left(g^{-1}x\right) + B_1V_1 + B_2\left(P\otimes I_K\right)\left(P^\dagger\otimes I_K\right)V_2\right) \\ =&\rho'(g)\left(B_0 f\left(g^{-1}x\right) + B_1V_1 + B_2V_2\right).\end{aligned}$$

Also, the right hand side of (11) can be written as [5]

$$\begin{aligned}&\Big(A_0 + A_1\partial_{x_1} + A_2\partial_{x_2} + A_3\partial_{x_3} + A_{11}\partial_{x_1^2} \\ &+A_{12}\partial_{x_1x_2} + A_{13}\partial_{x_1x_3} + A_{22}\partial_{x_2^2} + A_{23}\partial_{x_2x_3} \\ &+A_{33}\partial_{x_3^2}\Big)\left[\rho(g)f\left(g^{-1}x\right)\right] \\ =&\Big(A_0\rho(g) + A_1\rho(g)\partial_{x_1} + A_2\rho(g)\partial_{x_2} + A_3\rho(g)\partial_{x_3} \\ &+A_{11}\rho(g)\partial_{x_1^2} + A_{12}\rho(g)\partial_{x_1x_2} + A_{13}\rho(g)\partial_{x_1x_3} \\ &+A_{22}\rho(g)\partial_{x_2^2} + A_{23}\rho(g)\partial_{x_2x_3} + A_{33}\rho(g)\partial_{x_3^2}\Big)\left[f\left(g^{-1}x\right)\right] \\ =&B_0\rho(g)f\left(g^{-1}x\right) + B_1\left(I_3\otimes\rho(g)\right)vec\begin{bmatrix} \nabla^T\left[f_1\left(g^{-1}x\right)\right] \\ \nabla^T\left[f_2\left(g^{-1}x\right)\right] \\ \cdots \\ \nabla^T\left[f_K\left(g^{-1}x\right)\right] \end{bmatrix} \\ &+\bar{B}_2\left(I_9\otimes\rho(g)\right)vec\begin{bmatrix} vec^T\left(\nabla^2\left[f_1\left(g^{-1}x\right)\right]\right) \\ vec^T\left(\nabla^2\left[f_2\left(g^{-1}x\right)\right]\right) \\ \cdots \\ vec^T\left(\nabla^2\left[f_K\left(g^{-1}x\right)\right]\right) \end{bmatrix} \\ =&B_0\rho(g)f\left(g^{-1}x\right) + B_1\left(I_3\otimes\rho(g)\right)vec\begin{bmatrix} \nabla^T\left[f_1\right]\left(g^{-1}x\right)g^{-1} \\ \nabla^T\left[f_2\right]\left(g^{-1}x\right)g^{-1} \\ \cdots \\ \nabla^T\left[f_K\right]\left(g^{-1}x\right)g^{-1} \end{bmatrix} \\ &+\bar{B}_2\left(I_9\otimes\rho(g)\right)vec\begin{bmatrix} vec^T\left(g\nabla^2\left[f_1\right]\left(g^{-1}x\right)g^{-1}\right) \\ vec^T\left(g\nabla^2\left[f_2\right]\left(g^{-1}x\right)g^{-1}\right) \\ \cdots \\ vec^T\left(g\nabla^2\left[f_K\right]\left(g^{-1}x\right)g^{-1}\right) \end{bmatrix} \\ =&B_0\rho(g)f\left(g^{-1}x\right) + B_1\left(I_3\otimes\rho(g)\right)vec\left[I_K\begin{pmatrix} \nabla^T\left[f_1\right]\left(g^{-1}x\right) \\ \nabla^T\left[f_2\right]\left(g^{-1}x\right) \\ \cdots \\ \nabla^T\left[f_K\right]\left(g^{-1}x\right) \end{pmatrix}g^{-1}\right] \\ &+\bar{B}_2(I_9\otimes\rho(g))vec\begin{bmatrix} ((g\otimes g)vec(\nabla^2[f_1](g^{-1}x)))^T \\ ((g\otimes g)vec(\nabla^2[f_2](g^{-1}x)))^T \\ \cdots \\ ((g\otimes g)vec(\nabla^2[f_K](g^{-1}x)))^T \end{bmatrix} \\ =&B_0\rho(g)f(g^{-1}x) + B_1(I_3\otimes\rho(g))(g\otimes I_K)vec\begin{bmatrix} \nabla^T[f_1](g^{-1}x) \\ \nabla^T[f_2](g^{-1}x) \\ \cdots \\ \nabla^T[f_K](g^{-1}x) \end{bmatrix} \\ &+\bar{B}_2(I_9\otimes\rho(g))vec\left[I_K\begin{pmatrix} vec^T\left(\nabla^2[f_1]\left(g^{-1}x\right)\right) \\ vec^T\left(\nabla^2[f_2]\left(g^{-1}x\right)\right) \\ \cdots \\ vec^T\left(\nabla^2[f_K]\left(g^{-1}x\right)\right) \end{pmatrix}(g\otimes g)^T\right] \\ =&B_0\rho(g)f(g^{-1}x) + B_1(I_3\otimes\rho(g))(g\otimes I_K)vec\begin{bmatrix} \nabla^T[f_1]\left(g^{-1}x\right) \\ \nabla^T[f_2]\left(g^{-1}x\right) \\ \cdots \\ \nabla^T[f_K]\left(g^{-1}x\right) \end{bmatrix} \\ &+\bar{B}_2(I_9\otimes\rho(g))(g\otimes g\otimes I_K)vec\begin{bmatrix} vec^T\left(\nabla^2[f_1]\left(g^{-1}x\right)\right) \\ vec^T\left(\nabla^2[f_2]\left(g^{-1}x\right)\right) \\ \cdots \\ vec^T\left(\nabla^2[f_K]\left(g^{-1}x\right)\right) \end{bmatrix} \\ =&B_0\rho(g)f\left(g^{-1}x\right) + B_1\left(g\otimes\rho(g)\right)V_1 \\ &+B_2\left(P\otimes I_K\right)\left(I_9\otimes\rho(g)\right)\left(g\otimes g\otimes I_K\right)\left(P^\dagger\otimes I_K\right)V_2 \\ =&B_0\rho(g)f\left(g^{-1}x\right) + B_1\left(g\otimes\rho(g)\right)V_1 + B_2\left(P\left(g\otimes g\right)P^\dagger\otimes\rho(g)\right)V_2.\end{aligned}$$

[5] In the following derivation, we utilize two important properties of Kronecker product: (1) $vec(MXN) = \left(N^T\otimes M\right)vec(X)$; (2) $\left(M_1\otimes N_1\right)\left(M_2\otimes N_2\right) = \left(M_1M_2\otimes N_1N_2\right)$. Besides, $g^{-1} = g^T$, because g is essentially an orthogonal matrix.

As a result, Eqn. (11) can be deduced to

$$\begin{aligned}&\rho'(g)B_0 f\left(g^{-1}x\right)+\rho'(g)B_1V_1+\rho'(g)B_2V_2\\=&B_0\rho(g)f\left(g^{-1}x\right)+B_1\left(g\otimes\rho(g)\right)V_1+B_2\left(P\left(g\otimes g\right)P^{\dagger}\otimes\rho(g)\right)V_2\end{aligned}$$

Finally, we can use the coefficient comparison method to derive that Eqn. (11) is satisfied if and only if

$$\begin{cases}\rho'(g)B_0=B_0\rho(g),\\ \rho'(g)B_1=B_1\left(g\otimes\rho(g)\right),\\ \rho'(g)B_2=B_2\left(P\left(g\otimes g\right)P^{\dagger}\otimes\rho(g)\right),\end{cases}$$

i.e.,

$$\begin{cases}\left(I_K\otimes\rho'(g)-\rho(g)^T\otimes I_{K'}\right)vec\left(B_0\right)=0,\\ \left(I_{3K}\otimes\rho'(g)-\left(g\otimes\rho(g)\right)^T\otimes I_{K'}\right)vec\left(B_1\right)=0,\\ \left(I_{6K}\otimes\rho'(g)-\left(P(g\otimes g)P^{\dagger}\otimes\rho(g)\right)^T\otimes I_{K'}\right)vec\left(B_2\right)=0,\end{cases}$$

□

Theorem B.2. *When $\mathcal{G}$ is a discrete group, if Eqn. (8) or (9) is satisfied for the generators of $\mathcal{G}$, then $\forall g\in\mathcal{G}$, Eqn. (8) is satisfied.*

Proof. Firstly, for any given $g\in\mathcal{G}$, it is easy to deduce that Eqn. (8) is equivalent to Eqn. (9) from Theorem B.1. Suppose that g_1 and g_2 are the generators of $\mathcal{G}$, we have

$$\begin{aligned}\pi'\left(g_1\right)[\Psi[f]]=&\Psi\left[\pi\left(g_1\right)[f]\right],\\ \pi'\left(g_2\right)[\Psi[f]]=&\Psi\left[\pi\left(g_2\right)[f]\right].\end{aligned}$$

As a result, $\forall m\in\mathbb{Z}$,

$$\begin{aligned}\pi'\left(g_1^m\right)[\Psi[f]]=&\pi'\left(g_1\right)^m[\Psi[f]]\\=&\pi'\left(g_1\right)^{m-1}\pi'\left(g_1\right)[\Psi[f]]\\=&\pi'\left(g_1\right)^{m-1}\Psi\left[\pi(g_1)[f]\right],\\=&\Psi\left[\pi(g_1)^m[f]\right],\\=&\Psi\left[\pi\left(g_1^m\right)[f]\right].\end{aligned}$$

Analogously, we have

$$\pi'\left(g_2^m\right)[\Psi[f]]=\Psi\left[\pi\left(g_2^m\right)[f]\right].$$

Actually, any $g\in\mathcal{G}$ can be generated by g_1 and g_2, i.e., it can be written as $g=g_1^{m_1}g_2^{m_2}\cdots g_1^{m_{l-1}}g_2^{m_l}$, where $m_1,m_2,\cdots,m_l\in\mathbb{Z}$. Consequently, we have

$$\begin{aligned}&\pi'(g)[\Psi[f]]\\=&\pi'\left(g_1^{m_1}g_2^{m_2}\cdots g_1^{m_{l-1}}g_2^{m_l}\right)[\Psi[f]]\\=&\pi'\left(g_1^{m_1}\right)\pi'\left(g_2^{m_2}\right)\cdots\pi'\left(g_1^{m_{l-1}}\right)\pi'\left(g_2^{m_l}\right)[\Psi[f]]\\=&\Psi\left[\pi\left(g_1^{m_1}\right)\pi\left(g_2^{m_2}\right)\cdots\pi\left(g_1^{m_{l-1}}\right)\pi\left(g_2^{m_l}\right)[f]\right]\\=&\Psi\left[\pi\left(g_1^{m_1}g_2^{m_2}\cdots g_1^{m_{l-1}}g_2^{m_l}\right)[f]\right]\\=&\Psi[\pi(g)[f]].\end{aligned}$$

□

Theorem B.3. *When $\mathcal{G}=SO(3)$, if Eqn. (8) or (9) is satisfied for the approximated generator $g_1=Z(1)$ and $g_2=Y(1)$, where*

$$Z(\alpha)=\begin{bmatrix}\cos\alpha&-\sin\alpha&0\\ \sin\alpha&\cos\alpha&0\\0&0&1\end{bmatrix},\ Y(\beta)=\begin{bmatrix}\cos\beta&0&\sin\beta\\0&1&0\\-\sin\beta&0&\cos\beta\end{bmatrix},$$

then $\forall g\in\mathcal{G}$, Eqn. (8) is satisfied.

Proof. $\forall\alpha\in[0,2\pi)$, there exists a sequence of positive integers $\{n_k\in\mathbb{Z}^+\}$, such that

$$\lim_{k\to+\infty}Z\left(n_k\right)=Z(\alpha).$$

As a result, since Eqn. (8) (or Eqn.(9)) is satisfied for g_1, i.e.,

$$\pi'\left(g_1\right)[\Psi[f]]=\Psi\left[\pi(g_1)[f]\right],$$

and $Z\left(n_k\right)=g_1^{n_k}$, we have that

$$\begin{aligned}\pi'\left(Z\left(n_k\right)\right)[\Psi[f]]=&\pi'\left(g_1^{n_k}\right)[\Psi[f]]\\=&\pi'\left(g_1\right)^{n_k}[\Psi[f]]\\=&\pi'\left(g_1\right)^{n_k-1}\pi'\left(g_1\right)[\Psi[f]]\\=&\pi'\left(g_1\right)^{n_k-1}\Psi\left[\pi(g_1)[f]\right],\\=&\Psi\left[\pi\left(g_1^{n_k}\right)[f]\right]\\=&\Psi\left[\pi\left(Z(n_k)\right)[f]\right].\end{aligned}$$

As a result, we have that

$$\begin{aligned}\pi'\left(Z(\alpha)\right)[\Psi[f]]=&\lim_{k\to+\infty}\pi'\left(Z(n_k)\right)[\Psi[f]]\\=&\lim_{k\to+\infty}\Psi[\pi\left(Z(n_k)\right)[f]]\\=&\Psi\left[\pi\left(Z(\alpha)\right)[f]\right].\end{aligned}$$

Analogously, we can prove that $\forall\beta\in[0,\pi]$,

$$\pi'\left(Y(\beta)\right)[\Psi[f]]=\Psi\left[\pi\left(Y(\beta)\right)[f]\right].$$

Actually, any $g\in SO(3)$ can be decomposed using the ZYZ Euler parameterization, i.e.,

$$g=Z\left(\alpha_g\right)Y\left(\beta_g\right)Z\left(\gamma_g\right),$$

where $\alpha_g,\gamma_g\in[0,2\pi)$ and $\beta_g\in[0,\pi]$. Consequently, it is easy to deduce that

$$\begin{aligned}\pi'(g)[\Psi[f]]=&\pi'(Z\left(\alpha_g\right))\pi'\left(Y(\beta_g)\right)\pi'\left(Z(\gamma_g)\right)[\Psi[f]]\\=&\Psi\left[\pi\left(Z\left(\alpha_g\right)\right)\pi\left(Y\left(\beta_g\right)\right)\pi\left(Z\left(\gamma_g\right)\right)[f]\right]\\=&\Psi\left[\pi\left(Z\left(\alpha_g\right)Y\left(\beta_g\right)Z\left(\gamma_g\right)\right)[f]\right]\\=&\Psi\left[\pi(g)[f]\right].\end{aligned}$$

□

C. The Schema and the Generators of Different Groups

See Figure 2 and Table 8.

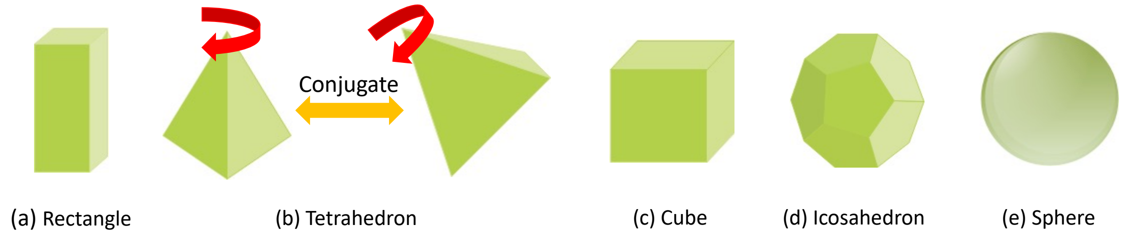

Figure 2. 3D rotation groups $\mathcal{V}, \mathcal{T}, \mathcal{O}, \mathcal{I}$ and $SO(3)$ are composed of all the rotational symmetries of (a) rectangle, (b) tetrahedron, (c) cube, (d) icosahedron, and (e) sphere, respectively. (b) Two tetrahedron groups are conjugate, as they are composed of the rotational symmetries of a tetrahedron and a rotated tetrahedron, respectively.

Table 8. The generators and sizes of discrete groups $\mathcal{V}, \mathcal{T}, \mathcal{O}$ and $\mathcal{I}$, where $\phi = (1+\sqrt{5})/2$.

Group	Generators	Size
$\mathcal{V}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}$	4
$\mathcal{T}$	$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	12
$\mathcal{O}$	$\begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix}$	24
$\mathcal{I}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} \frac{1-\phi}{2} & \frac{\phi}{2} & -\frac{1}{2} \\ -\frac{\phi}{2} & -\frac{1}{2} & \frac{1-\phi}{2} \\ -\frac{1}{2} & \frac{\phi-1}{2} & \frac{\phi}{2} \end{bmatrix}$	60

D. Implementation

D.1. Finite Difference (FD)

The volumetric data/feature $\boldsymbol{F}$ can be viewed as a sample from a continuous function f defined on $\mathbb{R}^3$. The grid size along x_1- , x_2- and x_3-axis are h_1, h_2 and h_3, respectively, and the grid diameter $\rho = \sqrt{h_1^2 + h_2^2 + h_3^2}$. When using FD for discretization, we have that $\forall i \in S_x$,

$$\partial_i[f] = u_i * \boldsymbol{F} + O\left(\rho^2\right),$$

where $S_x = \{x_1, x_2, x_3, x_1^2, x_2^2, x_3^2, x_1x_2, x_1x_3, x_2x_3\}$, and

$$\begin{cases} u_{x_1}(:,0,0) = \frac{1}{h_1}\begin{bmatrix} -1/2 & 0 & 1/2 \end{bmatrix}, \\ u_{x_2}(0,:,0) = \frac{1}{h_2}\begin{bmatrix} -1/2 & 0 & 1/2 \end{bmatrix}, \\ u_{x_3}(0,0,:) = \frac{1}{h_3}\begin{bmatrix} -1/2 & 0 & 1/2 \end{bmatrix}, \\ u_{x_1^2}(:,0,0) = \frac{1}{h_1^2}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}, \\ u_{x_2^2}(0,:,0) = \frac{1}{h_2^2}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}, \\ u_{x_3^2}(0,:,0) = \frac{1}{h_3^2}\begin{bmatrix} 1 & -2 & 1 \end{bmatrix}, \\ u_{x_1x_2}(:,:,0) = \frac{1}{h_1h_2}\begin{bmatrix} -1/4 & 0 & 1/4 \\ 0 & 0 & 0 \\ 1/4 & 0 & -1/4 \end{bmatrix}. \\ u_{x_1x_3}(:,0,:) = \frac{1}{h_1h_3}\begin{bmatrix} -1/4 & 0 & 1/4 \\ 0 & 0 & 0 \\ 1/4 & 0 & -1/4 \end{bmatrix}, \\ u_{x_2x_3}(0,:,:) = \frac{1}{h_2h_3}\begin{bmatrix} -1/4 & 0 & 1/4 \\ 0 & 0 & 0 \\ 1/4 & 0 & -1/4 \end{bmatrix}. \end{cases}$$

We only show some elements of each convolution filter for ease of presentation, and other elements are all zeros.

Particularly, for regular 3D grids, if we discretize a $\mathcal{G}$-steerable mapping Ψ using FD, where $\mathcal{G}$ includes the cubic group $\mathcal{O}$ as a subset, i.e., $\mathcal{O} \leq \mathcal{G}$, then discretized $\hat{\Psi}$ can preserve exact equivariance over $\mathcal{O}$ in the discrete domain. Formally, we have the following theorem.

Theorem D.1. *If Ψ is equivariant over $\mathcal{G}$, where $\mathcal{O} \leq \mathcal{G}$, then $\hat{\Psi}$ is exactly equivariant over $\mathcal{O}$, i.e., $\forall g \in \mathcal{O}$, we have*

$$\pi'(g)\left[\hat{\Psi}[\boldsymbol{F}]\right] = \hat{\Psi}\left[\pi(g)\boldsymbol{F}\right], \tag{13}$$

where the volumetric data/feature $\boldsymbol{F}$ is defined on regular

3D grids, $h_1 = h_2 = h_3$*, and*

$$\left\{\begin{array}{l}[\pi(g)\boldsymbol{F}](x)=\rho(g)\boldsymbol{F}\left(g^{-1}x\right), \\ \left[\pi'(g)\boldsymbol{F}\right](x)=\rho'(g)\boldsymbol{F}\left(g^{-1}x\right).\end{array}\right.$$

Proof. The proof is mostly following the proof of Theorem B.1, and most symbols and procedures are similar for ease of understanding. Specifically, Eqn. (13) can be rewritten as $\forall g \in \mathcal{O}$,

$$\begin{aligned}&\rho'(g)\Big(A_0+A_1u_{x_1}+A_2u_{x_2}+A_3u_{x_3}+A_{11}u_{x_1^2}\\&+A_{12}u_{x_1x_2}+A_{13}u_{x_1x_3}+A_{22}u_{x_2^2}+A_{23}u_{x_2x_3}\\&+A_{33}u_{x_3^2}\Big)*\boldsymbol{F}\left(g^{-1}x\right)\\=&\Big(A_0+A_1u_{x_1}+A_2u_{x_2}+A_3u_{x_3}+A_{11}u_{x_1^2}\\&+A_{12}u_{x_1x_2}+A_{13}u_{x_1x_3}+A_{22}u_{x_2^2}+A_{23}u_{x_2x_3}\\&+A_{33}u_{x_3^2}\Big)*\left[\rho(g)\boldsymbol{F}\left(g^{-1}x\right)\right].\end{aligned}\tag{14}$$

We denote that $\forall i = 1, 2, \cdots, K$,

$$\begin{aligned}\hat{\nabla}^T\left[\boldsymbol{F}_i\right]&=\left[u_{x_1}*\boldsymbol{F}_i,u_{x_2}*\boldsymbol{F}_i,u_{x_3}*\boldsymbol{F}_i\right],\\ \hat{\nabla}^2\left[\boldsymbol{F}_i\right]&=\begin{bmatrix}u_{x_1^2}*\boldsymbol{F}_i & u_{x_1x_2}*\boldsymbol{F}_i & u_{x_1x_3}*\boldsymbol{F}_i\\ u_{x_1x_2}*\boldsymbol{F}_i & u_{x_2^2}*\boldsymbol{F}_i & u_{x_2x_3}*\boldsymbol{F}_i\\ u_{x_1x_3}*\boldsymbol{F}_i & u_{x_2x_3}*\boldsymbol{F}_i & u_{x_3^2}*\boldsymbol{F}_i\end{bmatrix}.\end{aligned}$$

Then the left hand side of Eqn. (14) can be written as

$$\rho'(g)\left(B_0\boldsymbol{F}\left(g^{-1}x\right)+B_1\hat{V}_1+\bar{B}_2\hat{\bar{V}}_2\right),\tag{15}$$

where

$$\begin{aligned}\hat{V}_1=&vec\begin{bmatrix}\hat{\nabla}^T\left[\boldsymbol{F}_1\right]\left(g^{-1}x\right)\\ \hat{\nabla}^T\left[\boldsymbol{F}_2\right]\left(g^{-1}x\right)\\ \cdots\\ \hat{\nabla}^T\left[\boldsymbol{F}_K\right]\left(g^{-1}x\right)\end{bmatrix}\\=&vec\left[u_{x_1}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_3}*\boldsymbol{F}\left(g^{-1}x\right)\right],\end{aligned}$$

and

$$\begin{aligned}\hat{\bar{V}}_2=&vec\begin{bmatrix}vec^T\left(\hat{\nabla}^2\left[\boldsymbol{F}_1\right]\left(g^{-1}x\right)\right)\\ vec^T\left(\hat{\nabla}^2\left[\boldsymbol{F}_2\right]\left(g^{-1}x\right)\right)\\ \cdots\\ vec^T\left(\hat{\nabla}^2\left[\boldsymbol{F}_K\right]\left(g^{-1}x\right)\right)\end{bmatrix}\\=&vec\Big[u_{x_1^2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_1x_2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_1x_3}*\boldsymbol{F}\left(g^{-1}x\right),\\&u_{x_1x_2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_2^2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_2x_3}*\boldsymbol{F}\left(g^{-1}x\right),\\&u_{x_1x_3}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_2x_3}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_3^2}*\boldsymbol{F}\left(g^{-1}x\right)\Big].\end{aligned}$$

We further denote that

$$\begin{aligned}\hat{V}_2=&vec\Big[u_{x_1^2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_1x_2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_1x_3}*\boldsymbol{F}\left(g^{-1}x\right),\\&u_{x_2^2}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_2x_3}*\boldsymbol{F}\left(g^{-1}x\right),u_{x_3^2}*\boldsymbol{F}\left(g^{-1}x\right)\Big],\end{aligned}$$

and we have

$$\hat{\bar{V}}_2=\left(P^\dagger\otimes I_K\right)\hat{V}_2,$$

As a result, the left hand size of Eqn. (14) can be written as

$$\begin{aligned}&\rho'(g)\left(B_0\boldsymbol{F}\left(g^{-1}x\right)+B_1\hat{V}_1+\bar{B}_2\hat{\bar{V}}_2\right)\\=&\rho'(g)\left(B_0\boldsymbol{F}\left(g^{-1}x\right)+B_1\hat{V}_1+B_2\left(P\otimes I_K\right)\left(P^\dagger\otimes I_K\right)\hat{V}_2\right)\\=&\rho'(g)\left(B_0\boldsymbol{F}\left(g^{-1}x\right)+B_1\hat{V}_1+B_2\hat{V}_2\right).\end{aligned}$$

Also, the right hand side of (14) can be written as

$$\begin{aligned}&\Big(A_0+A_1u_{x_1}+A_2u_{x_2}+A_3u_{x_3}+A_{11}u_{x_1^2}\\&+A_{12}u_{x_1x_2}+A_{13}u_{x_1x_3}+A_{22}u_{x_2^2}+A_{23}u_{x_2x_3}\\&+A_{33}u_{x_3^2}\Big)*\left[\rho(g)\boldsymbol{F}\left(g^{-1}x\right)\right]\\=&\left(A_0\rho(g)+A_1\rho(g)u_{x_1}+A_2\rho(g)u_{x_2}+A_3\rho(g)u_{x_3}\right.\\&+A_{11}\rho(g)u_{x_1^2}+A_{12}\rho(g)u_{x_1x_2}+A_{13}\rho(g)u_{x_1x_3}\\&\left.+A_{22}\rho(g)u_{x_2^2}+A_{23}\rho(g)u_{x_2x_3}+A_{33}\rho(g)u_{x_3^2}\right)*\left[\boldsymbol{F}\left(g^{-1}x\right)\right]\\=&B_0\rho(g)\boldsymbol{F}\left(g^{-1}x\right)+B_1\left(I_3\otimes\rho(g)\right)vec\begin{bmatrix}\hat{\nabla}^T\left[\boldsymbol{F}_1\left(g^{-1}x\right)\right]\\ \hat{\nabla}^T\left[\boldsymbol{F}_2\left(g^{-1}x\right)\right]\\ \cdots\\ \hat{\nabla}^T\left[\boldsymbol{F}_K\left(g^{-1}x\right)\right]\end{bmatrix}\\&+\bar{B}_2\left(I_9\otimes\rho(g)\right)vec\begin{bmatrix}vec^T\left(\hat{\nabla}^2\left[\boldsymbol{F}_1\left(g^{-1}x\right)\right]\right)\\ vec^T\left(\hat{\nabla}^2\left[\boldsymbol{F}_2\left(g^{-1}x\right)\right]\right)\\ \cdots\\ vec^T\left(\hat{\nabla}^2\left[\boldsymbol{F}_K\left(g^{-1}x\right)\right]\right)\end{bmatrix}\end{aligned}$$

As for volumetric input $\boldsymbol{F}_i$ defined on regular 3D grids with $h_1 = h_2 = h_3$, it can be verified that $\forall g \in \mathcal{O}$,

$$\begin{aligned}\hat{\nabla}^T\left[\boldsymbol{F}_i\left(g^{-1}x\right)\right]=&\hat{\nabla}^T\left[\boldsymbol{F}_i\right]\left(g^{-1}x\right)g^{-1},\\ \hat{\nabla}^2\left[\boldsymbol{F}_i\left(g^{-1}x\right)\right]=&g\hat{\nabla}^2\left[\boldsymbol{F}_i\right]\left(g^{-1}x\right)g^{-1}.\end{aligned}$$

Then similar to that in the proof of Theorem B.1, the previous equation can be further written as

$$\begin{aligned}&B_0\rho(g)\boldsymbol{F}\left(g^{-1}x\right)+B_1\left(I_3\otimes\rho(g)\right)vec\begin{bmatrix}\hat{\nabla}^T\left[\boldsymbol{F}_1\right]\left(g^{-1}x\right)g^{-1}\\ \hat{\nabla}^T\left[\boldsymbol{F}_2\right]\left(g^{-1}x\right)g^{-1}\\ \cdots\\ \hat{\nabla}^T\left[\boldsymbol{F}_K\right]\left(g^{-1}x\right)g^{-1}\end{bmatrix}\\&+\bar{B}_2\left(I_9\otimes\rho(g)\right)vec\begin{bmatrix}vec^T\left(g\hat{\nabla}^2\left[\boldsymbol{F}_1\right]\left(g^{-1}x\right)g^{-1}\right)\\ vec^T\left(g\hat{\nabla}^2\left[\boldsymbol{F}_2\right]\left(g^{-1}x\right)g^{-1}\right)\\ \cdots\\ vec^T\left(g\hat{\nabla}^2\left[\boldsymbol{F}_K\right]\left(g^{-1}x\right)g^{-1}\right)\end{bmatrix}\\=&B_0\rho(g)\boldsymbol{F}\left(g^{-1}x\right)+B_1\left(g\otimes\rho(g)\right)\hat{V}_1+B_2\left(P\left(g\otimes g\right)P^\dagger\otimes\rho(g)\right)\hat{V}_2.\end{aligned}$$

As a result, Eqn. (13) is equivalent to that $\forall g \in \mathcal{O}$,

$$\begin{aligned}&\rho'(g)B_0\boldsymbol{F}\left(g^{-1}x\right)+\rho'(g)B_1\hat{V}_1+\rho'(g)B_2\hat{V}_2\\=&B_0\rho(g)\boldsymbol{F}\left(g^{-1}x\right)+B_1\left(g\otimes\rho(g)\right)\hat{V}_1+B_2\left(P\left(g\otimes g\right)P^\dagger\otimes\rho(g)\right)\hat{V}_2.\end{aligned}$$

Since Ψ is equivariant over $\mathcal{O}$, Eqn. (9) is satisfied for any g in $\mathcal{O}$. Finally, Eqn. (13) is satisfied. $\square$

D.2. Nonlinearities and Batch Normalization (BN)

Scale-like Nonlinearities As irreducible representations (irreps) we employ are all unitary representations which preserve the norm of features, i.e., they satisfy that $\|\rho(g)f(x)\|_2 = \|f(x)\|_2$. As a result, nonlinearities which solely act on the norm of feature vectors but preserve their orientation are admissible. They can in general be decomposed in $\sigma_{\text{norm}}: f(x) \mapsto \eta(\|f(x)\|_2)\frac{f(x)}{\|f(x)\|_2}$, where η is a nonlinear function. For instance, Norm-ReLUs are defined by using $\eta(\|f(x)\|_2) = ReLU(\|f(x)\|_2 - b)$ where b is a learnable bias. Gated nonlinearity scales the norm of a feature field by learned sigmoid gates $\frac{1}{1+e^{-s(x)}}$, parameterized by a scalar feature field s. In all, existing nonlinearities act by scaling the feature vectors, where the scales are acquired by different methods that do not disturb equivariance. We employ gated nonlinearity for implementation as many works have shown that it works in practice better than other options (Weiler et al., 2018a; Weiler & Cesa, 2019).

Scale-BN Non-trivial irreducible feature fields are normalized with the average of their norms:

$$f(x) \mapsto f(x)\left(\frac{1}{|\mathcal{B}|V}\sum_{j\in\mathcal{B}}\int \|f(x)\|_2^2 dx\right)^{-1/2},$$

where $\mathcal{B}$ is the batch size and V is the size of domain. As a result, this kind of BN can only scale the feature vectors and cannot correct activations to zero-mean.

D.3. Composition of Basic Feature Fields

We can easily employ one single basic feature field per layer, such as the regular, quotient, and irreducible feature field. Actually, we can put multiple basic feature fields together and acquire much more general feature fields, and the constraints for coefficients can be solved efficiently. Formally, we have the following theorem.

Theorem D.2. *If $\rho(g) = \bigoplus_{i=1}^t \rho_i(g)$ and $\rho'(g) = \bigoplus_{i=1}^s \rho'_i(g)$, where $\rho_i(g)$ are $\rho'_i(g)$ are both basic group representations, and*

$$\bigoplus_{i=1}^t \rho_i(g) = \begin{bmatrix} \rho_1(g) & & \\ & \ddots & \\ & & \rho_t(g) \end{bmatrix},$$

then Eqn. (9) is equivalent to

$$\begin{cases} \rho'_i(g)C_0^{ij} = C_0^{ij}\rho_j(g), \\ \rho'_i(g)C_1^{ij} = C_1^{ij}\left(g\otimes\rho_j(g)\right), \\ \rho'_i(g)C_2^{ij} = C_2^{ij}\left(P\left(g\otimes g\right)P^\dagger\otimes\rho_j(g)\right), \end{cases}$$

where

$$\begin{cases} C_0^{ij} = A_0^{ij}, \\ C_1^{ij} = \left[A_1^{ij}, A_2^{ij}, A_3^{ij}\right], \\ C_2^{ij} = \left[A_{11}^{ij}, A_{12}^{ij}, A_{13}^{ij}, A_{22}^{ij}, A_{23}^{ij}, A_{33}^{ij}\right], \end{cases} \tag{16}$$

and A_k^{ij} is the submatrix of A_k, i.e.,

$$A_k = \begin{bmatrix} A_k^{11} & \cdots & A_k^{1t} \\ \vdots & & \vdots \\ A_k^{s1} & \cdots & A_k^{st} \end{bmatrix}.$$

Proof. The constraint for $B_0 = A_0$ in (9) can be written as

$$\begin{bmatrix} \rho'_1(g) & & \\ & \ddots & \\ & & \rho'_s(g) \end{bmatrix}\begin{bmatrix} A_0^{11} & \cdots & A_0^{1t} \\ \vdots & & \vdots \\ A_0^{s1} & \cdots & A_0^{st} \end{bmatrix} = \begin{bmatrix} A_0^{11} & \cdots & A_0^{1t} \\ \vdots & & \vdots \\ A_0^{s1} & \cdots & A_0^{st} \end{bmatrix}\begin{bmatrix} \rho_1(g) & & \\ & \ddots & \\ & & \rho_t(g) \end{bmatrix},$$

i.e., $\forall i = 1, 2, \cdots, s$ and $\forall j = 1, 2, \cdots, t$,

$$\rho'_i(g)C_0^{ij} = C_0^{ij}\rho_j(g),$$

where $C_0^{ij} = A_0^{ij}$. As for the constraint for $B_1 = [A_1, A_2, A_3]$, we have

$$\rho'(g)\left[A_1, A_2, A_3\right] = \left[A_1, A_2, A_3\right]\left(g\otimes\rho(g)\right),$$

i.e.,

$$\begin{aligned} &\rho'(g)[A_1, A_2, A_3] \\ =&[A_1, A_2, A_3]\begin{bmatrix} g_{11}\rho(g) & g_{12}\rho(g) & g_{13}\rho(g) \\ g_{21}\rho(g) & g_{22}\rho(g) & g_{23}\rho(g) \\ g_{31}\rho(g) & g_{32}\rho(g) & g_{33}\rho(g) \end{bmatrix}, \end{aligned}$$

i.e.,

$$\begin{aligned} &[\rho'(g)A_1, \rho'(g)A_2, \rho'(g)A_3] \\ =&[g_{11}A_1\rho(g) + g_{21}A_2\rho(g) + g_{31}A_3\rho(g), \\ &g_{12}A_1\rho(g) + g_{22}A_2\rho(g) + g_{32}A_3\rho(g), \\ &g_{13}A_1\rho(g) + g_{23}A_2\rho(g) + g_{33}A_3\rho(g)]. \end{aligned} \tag{17}$$

Eqn. (17) can be further expanded as

$$\begin{aligned} &\left[\left(\rho'_i(g)A_1^{ij}\right)_{ij}, \left(\rho'_i(g)A_2^{ij}\right)_{ij}, \left(\rho'_i(g)A_3^{ij}\right)_{ij}\right] \\ =&\left[\left(g_{11}A_1^{ij}\rho_j(g) + g_{21}A_2^{ij}\rho_j(g) + g_{31}A_3^{ij}\rho_j(g)\right)_{ij}, \right. \\ &\left(g_{12}A_1^{ij}\rho_j(g) + g_{22}A_2^{ij}\rho_j(g) + g_{32}A_3^{ij}\rho_j(g)\right)_{ij}, \\ &\left.\left(g_{13}A_1^{ij}\rho_j(g) + g_{23}A_2^{ij}\rho_j(g) + g_{33}A_3^{ij}\rho_j(g)\right)_{ij}\right], \end{aligned}$$

i.e., $\forall i = 1, 2, \cdots, s$ and $\forall j = 1, 2, \cdots, t$,

$$\begin{aligned}
&\left[\rho'_i(g)A_1^{ij}, \rho'_i(g)A_2^{ij}, \rho'_i(g)A_3^{ij}\right] \\
&= \Big[g_{11}A_1^{ij}\rho_j(g) + g_{21}A_2^{ij}\rho_j(g) + g_{31}A_3^{ij}\rho_j(g), \\
&\quad g_{12}A_1^{ij}\rho_j(g) + g_{22}A_2^{ij}\rho_j(g) + g_{32}A_3^{ij}\rho_j(g), \\
&\quad g_{13}A_1^{ij}\rho_j(g) + g_{23}A_2^{ij}\rho_j(g) + g_{33}A_3^{ij}\rho_j(g)\Big],
\end{aligned}$$

and this can be reversely written as

$$\rho'_i(g)\left[A_1^{ij}, A_2^{ij}, A_3^{ij}\right] = \left[A_1^{ij}, A_2^{ij}, A_3^{ij}\right](g \otimes \rho_j(g)),$$

i.e.,

$$\rho'_i(g)C_1^{ij} = C_1^{ij}(g \otimes \rho_j(g)),$$

where $C_1^{ij} = \left[A_1^{ij}, A_2^{ij}, A_3^{ij}\right]$.

Analogously, the constraints for B_2 is equivalent to that

$$\rho'_i(g)C_2^{ij} = C_2^{ij}\left(P(g \otimes g)P^\dagger \otimes \rho_j(g)\right),$$

where $C_2^{ij} = \left[A_{11}^{ij}, A_{12}^{ij}, A_{13}^{ij}, A_{22}^{ij}, A_{23}^{ij}, A_{33}^{ij}\right]$. □

In this way, we can firstly compute C_0^{ij}, C_1^{ij} and C_2^{ij} according to the basic group representations $\rho'_i(g)$ and $\rho_j(g)$, then A_k^{ij} are obtained according to Eqn. (16). Finally, the coefficients A_k are obtained by placing A_k^{ij} properly.

E. Experimental Details

We provide the experimental details in this section. Our experiments are implemented using Pytorch, and each one is run using one single Tesla-V100 GPU.

E.1. 3D Tetris (Testing 3D equivariance)

This dataset contains 8 shapes, and we convert each Tetris block into $40 \times 40 \times 40$ voxels for preprocessing. The $\mathcal{O}$-steerable models consist of 3 convolutional layers. The first two layers are both composed of 10 basic features, and the last layer is composed of 64 scalar features. The architecture of the $SO(3)$-steerable model is in Table 9, which shows the numbers and sizes of feature fields. We use the average pooling for downsampling and a global average pooling after the final convolution layers to yield an invariant representation. Our models are trained using the Adam algorithm (Kingma & Ba, 2015). We use the generalized He's weight initialization scheme introduced in (Weiler et al., 2018b) for the convolutional layers and the Xavier optimizer (Glorot & Bengio, 2010) for the fully-connected (FC) layers. We train for 200 epochs with an initial learning rate of 0.01 and an exponential decay of 0.98 after 50 epochs.

Table 9. The architecture of the $SO(3)$-steerable model with irreducible features.

	$D^0(\rho)$	$D^1(\rho)$	$D^2(\rho)$	Size
Input	1	0	0	40^3
Layer1	4	4	4	40^3
Layer2	16	16	16	20^3
Layer3	32	16	16	10^3
Layer4	128	0	0	10^3
Output	8	0	0	1

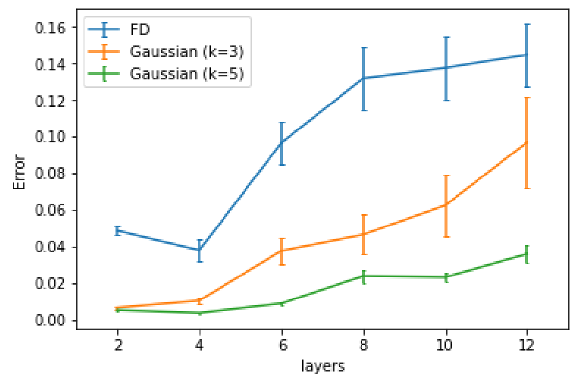

Figure 3. The $SO(3)$-equivariance errors from FD and Gaussian discretizations, respectively.

For equivariance error analysis, we randomly sample $n = 100$ rotations g_i and shapes $\boldsymbol{I}_i$, and compute equivariance error $\frac{1}{n}\sum_{i=1}^{n}\|\Phi\left[\pi(g_i)[\boldsymbol{I}_i]\right] - \Phi[\boldsymbol{I}_i]\|_2/\|\Phi[\boldsymbol{I}_i]\|_2$, where Φ is a composition of $SO(3)$-steerable layers following by an invariant layer with trivial output features. Each intermediate feature fieds are $\bigoplus_{i=0}^{2} D_i^l(g)$, and the models are tested after weights being initialized without training. The standard deviation σ for Gaussian discretization is taken as half the radius of the kernel size. That is, $\sigma = 0.5$ for the kernel size of $3 \times 3 \times 3$ ($k = 3$), and $\sigma = 1.0$ for $5 \times 5 \times 5$ ($k = 5$). As shown in Figure 3, Gaussian discretization results in a much lower error for $SO(3)$-equivariance.

E.2. SHREC'17 Retrieval Challenge

We choose models and hyperparameters with the lowest validation error during training. The architectures of steerable models for the SHREC'17 retrieval task are shown in Table 11-19. Our models are trained using the Adam optimizer (Kingma & Ba, 2015). We train for 2000 epochs with a batch size of 32. The initial learning rate is set to 0.01 and is divided by 10 at 700 and 1,400 epochs. The filters from Gaussian discretization are of the size $5 \times 5 \times 5$, and $\sigma = 1.0$.

Ablation Study We conduct an ablation study to justify the choice of PDOs in Eqn. (7). We choose several com-

Table 10. The results when we use different combinations of PDOs. I, ∇ and ∇^2 denote the identity operator, the first and the second order PDOs, respectively.

Convolution kernel	Score
$I + \nabla$	56.2
$I + \nabla^2$ ($\mathcal{V}$-quotient features)	55.5
$I + \nabla^2$ (regular features)	57.3
$I + \nabla + \nabla^2$	**58.6**

binations of PDOs and report the final scores based on the $\mathcal{O}$-steerable models with regular features. For all settings, we use a model with about 0.15M parameters for evaluation on the SHREC'17 retrieval task. Results for this ablation study are listed in Table 10. Our choice of PDO combinations in Eqn. (7) achieves the best performance, indicating the necessity of employing all the PDOs up to the second-order.

Interestingly, we observe that when we do not use the first-order PDOs, the model with regular features still outperforms that with quotient features (57.3 vs. 55.5). We argue that it is because that the regular features encode 24 rotation symmetries using 24 channels, while $\mathcal{V}$-quotient features only use 6, which may result in a low representation capacity.

E.3. ISBI 2012 Challenge

The full training image is $512 \times 512 \times 30$ voxels in shape. The setting is the same for test images. We take the first 25 slices of the training image as our training set, and the last 5 as the validation set. For the training set, we extract random $128 \times 128 \times 5$ voxel patches from the training volume, and reflection pad 48 voxels in the x-y plane. Following (Quan et al., 2016), we apply a random elastic distortion and rotation in the x-y plane and add Gaussian noise ($\sigma = 0.1$) for data augmentation. Finally, following CubeNet (Worrall & Brostow, 2018), we pass these patches through an equivariant FusionNet (Quan et al., 2016). We choose hyperparameters with the highest validation score during training. Our models are trained using the Adam optimizer (Kingma & Ba, 2015). We train for $4,000$ iterations with a batch size of 4. We use an initial learning rate of 0.001 and an exponential decay of 0.99.

Table 11. The architecture of the $\mathcal{V}$-steerable model with regular features.

	Scalar	Regular	Size
Input	1	0	64^3
Layer1	0	8	32^3
Layer2	0	8	32^3
Layer3	0	12	16^3
Layer4	0	12	16^3
Layer5	0	16	8^3
Layer6	0	16	8^3
Layer7	0	16	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 12. The architecture of the $\mathcal{T}$-steerable model with regular features.

	Scalar	Regular	Size
Input	1	0	64^3
Layer1	0	4	32^3
Layer2	0	4	32^3
Layer3	0	8	16^3
Layer4	0	8	16^3
Layer5	0	12	8^3
Layer6	0	12	8^3
Layer7	0	12	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 13. The architecture of the $\mathcal{O}$-steerable model with regular features.

	Scalar	Regular	Size
Input	1	0	64^3
Layer1	0	4	32^3
Layer2	0	4	32^3
Layer3	0	6	16^3
Layer4	0	6	16^3
Layer5	0	8	8^3
Layer6	0	10	8^3
Layer7	0	10	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 14. The architecture of the $\mathcal{I}$-steerable model with regular features.

	Scalar	Regular	Size
Input	1	0	64^3
Layer1	0	3	32^3
Layer2	0	3	32^3
Layer3	0	5	16^3
Layer4	0	5	16^3
Layer5	0	6	8^3
Layer6	0	6	8^3
Layer7	0	6	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 15. The architecture of the $SO(3)$-steerable model with irreducible features.

	$D^0(\rho)$	$D^1(\rho)$	$D^2(\rho)$	Size
Input	1	0	0	64^3
Layer1	8	4	2	32^3
Layer2	8	4	2	32^3
Layer3	16	8	4	16^3
Layer4	16	8	4	16^3
Layer5	32	16	8	8^3
Layer6	32	16	8	8^3
Layer7	64	32	16	8^3
Layer8	512	0	0	8^3
Output	55	0	0	1

Table 16. The architecture of the $\mathcal{O}$-steerable model with $\mathcal{V}$-quotient features.

	Scalar	$\mathcal{V}$-quotient	Size
Input	1	0	64^3
Layer1	0	8	32^3
Layer2	0	8	32^3
Layer3	0	16	16^3
Layer4	0	16	16^3
Layer5	0	28	8^3
Layer6	0	28	8^3
Layer7	0	30	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 17. The architecture of the $\mathcal{O}$-steerable model with the features composed of regular and $\mathcal{V}$-quotient features.

	Scalar	$\mathcal{V}$-quotient	Size
Input	1	0	64^3
Layer1	0	8	32^3
Layer2	0	8	32^3
Layer3	0	16	16^3
Layer4	0	16	16^3
Layer5	0	28	8^3
Layer6	0	28	8^3
Layer7	0	30	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 18. The architecture of the $\mathcal{O}$-steerable model using regular features without the first-order PDOs.

	Scalar	Regular	Size
Input	1	0	64^3
Layer1	0	4	32^3
Layer2	0	6	32^3
Layer3	0	8	16^3
Layer4	0	8	16^3
Layer5	0	10	8^3
Layer6	0	12	8^3
Layer7	0	12	8^3
Layer8	512	0	8^3
Output	55	0	1

Table 19. The architecture of the $\mathcal{O}$-steerable model using regular features without the second-order PDOs.

	Scalar	Regular	Size
Input	1	0	64^3
Layer1	0	6	32^3
Layer2	0	6	32^3
Layer3	0	10	16^3
Layer4	0	10	16^3
Layer5	0	16	8^3
Layer6	0	16	8^3
Layer7	0	18	8^3
Layer8	512	0	8^3
Output	55	0	1