Deep Learning for 2D and 3D Rotatable Data: An Overview of Methods

Luca DELLA LIBERA, Vladmir GOLKOV*, Yue ZHU, Arman MIELKE and Daniel CREMERS

Computer Vision Group, Technical University of Munich, Germany
{lucadella, vladimir.golkov, yue.zhu, arman.mielke, cremers}@tum.de

Abstract

Convolutional networks are successful due to their equivariance/invariance under translations. However, rotatable data such as images, volumes, shapes, or point clouds require processing with equivariance/invariance under rotations in cases where the rotational orientation of the coordinate system does not affect the meaning of the data (e.g. object classification). On the other hand, estimation/processing of rotations is necessary in cases where rotations are important (e.g. motion estimation). There has been recent progress in methods and theory in all these regards. Here we provide an overview of existing methods, both for 2D and 3D rotations (and translations), and identify commonalities and links between them.

1 Introduction

Rotational and translational equivariance play an important role in image recognition tasks. Convolutional neural networks (CNNs) are translationally equivariant: the convolution of a translated image with a filter is equivalent to the convolution of the untranslated image with the same filter, followed by a translation. Unfortunately, standard CNNs do not have an analogous property for rotations.

A naive attempt to achieve rotational equivariance/invariance is data augmentation. Its major problem is that rotational equivariance in the data but not in the network architecture forces the network to learn each object orientation “from scratch” and hampers generalization. Methods that achieve rotational equivariance/invariance in more advanced architecture forces the network to learn each object orientation “from scratch” and hampers generalization. Methods that use rotations as input and/or as deep features have appeared recently.

In Sections 3–4 we present the main approaches used to achieve rotational equivariance. In Section 5.1 we categorize concrete methods that use those approaches to achieve equivariance/invariance. We also categorize methods that can return a rotation as output in Section 5.2 and methods that use rotations as input and/or deep features in Section 5.3. Finally, we draw conclusions in Section 6. The mathematical concepts (Section 2) serve as a foundation for the best (i.e. exact and most general) equivariant approach (Section 3.3).

2 Formal Definitions

2.1 Equivariance

Definition 1. A function \( f : \mathcal{X} \to \mathcal{Y} \) is \textit{equivariant} under a group \( G \) (with some group actions \( \pi \) and \( \psi \) of \( G \) that transform \( \mathcal{X} \) and \( \mathcal{Y} \), respectively) if

\[
\psi_g[f(x)] = f(\pi_g(x)) \quad \forall g \in G \quad \forall x \in \mathcal{X},
\]

where \( \pi_g \) is the action of \( g \) on \( \mathcal{X} \), i.e. a transformation (for example rotation) of the input of \( f \), and \( \psi_g \) is the action of \( g \) on \( \mathcal{Y} \), i.e. an “associated” (via the same group) but possibly different transformation (for example rotation of the image and of the feature space) of the output of \( f \). In other words, for each transformation \( \pi_g \) of \( g \) that modifies the input of \( f \), we know a transformation \( \psi_g \) that happens to the output of \( f \) (due to transforming the input by \( \pi_g \)), without the need to know the input \( x \). The usage of \( g \in G \) to associate \( \psi_g \) with \( \pi_g \) is important for correct composition of several transformations. For example, if \( \pi_g \) is a 180° rotation, i.e. \( \pi_g \pi_g \) is the identity mapping, then \( \psi_g \psi_g \) should also be the identity mapping.

If the content of a rectangular image is rotated (and/or translated), the “field of view” changes, i.e. features that used to be in the corners disappear and new features appear in the corners. This has caused some confusion as to how rectangular images can be processed in a rotation-equivariant way. The explanation is the following: An output value of the neural network is only affected if the change of features is within its receptive field and the network has not learned from data that such a feature change should be irrelevant for the output.

Similarly, if two input features are rotated relatively to each other, an output value changes only if both input features are relevant for the output.

Definition 2. A special case of equivariance is same-equivariance [Dieleman et al., 2016], when \( \psi = \pi \). In
some sources, same-equivariance is called equivariance, and what we call equivariance is called covariance.

**Definition 3.** A special case of equivariance is invariance, when \( \psi = I \), the identity.

**Definition 4.** Equivariance is **exact** if Eq. (1) holds strictly, approximate (for example approximated through learning) if Eq. (1) holds approximatively.

### 2.2 Steerability

**Definition 5.** A function \( f : \mathcal{X} \rightarrow \mathcal{Y} \) is steerable if rotated versions of \( f \) can be expressed using linear combinations of a fixed set of basis functions \( h_j \) for \( j = 1, \ldots, M \), that is:

\[
    f(\pi |x|) = \sum_{j=1}^{M} k_j(\pi) h_j(x),
\]

where \( \pi \) is a rotation and \( k_j \) are complex-valued rotation-dependent steering factors.

For example, if we consider a standard non-normalized 2D Gaussian \( G(x, y) = e^{-(x^2+y^2)} \), its first derivative \( G_x(x, y) = \frac{\partial G}{\partial x}(x, y) \) in the \( x \) direction can be steered at an arbitrary orientation \( \theta \) through a linear combination of \( G^\theta_x(x, y) = G_x(x, y) = -2xe^{-(x^2+y^2)} \) and \( G^\theta_y(x, y) = G_y(x, y) = -2ye^{-(x^2+y^2)} \):

\[
    G^\theta_x(x, y) = G_x(\pi^\theta [x, y]) = \cos(\theta)G^\theta_y(x, y) + \sin(\theta)G^{\theta 0}(x, y).
\]

A visualization of the case \( \theta = 30^\circ \) looks as follows:

\[
    G^{30^\circ}_x(x, y) \approx 0.87 \quad G^{30^\circ}_y(x, y) \approx 0.5.
\]

A useful consequence of steerability is that convolution of an image with basis filters \( h_j \) is rotationally equivariant. The mapping \( \psi \) in Eq. (1) in this case corresponds to a linear combination of the feature maps. As a side note, the mapping \( \psi \) can also be a certain kind of linear combinations even if the filters are not a basis of a steerable filter, see Section 3.2.2.

A **harmonic function** \( f \) is a twice continuously differentiable function that satisfies Laplace’s equation, i.e. \( \nabla^2 f = 0 \). Circular harmonics and spherical harmonics are defined on the circle and the sphere, respectively, and are similar to the Fourier series (i.e. sinusoids with different frequencies).

2D or 3D rotational equivariance can be hardwired in the network architecture by restricting the filters’ angular component to belong to the circular harmonic or spherical harmonic family, respectively. The proof utilizes steerability properties of such filter banks. The radial profile of these filters on the other hand can be learned. There are various techniques to parameterize the radial profile (with learnable parameters).

### 2.3 Group Convolution

**Definition 6.** The group convolution [Cohen and Welling, 2016; Esteves et al., 2018b] between a feature map \( F \) and a filter \( W \) is defined as

\[
    (F \ast_G W)(x) = \int_{g \in G} F(\phi_g[\eta]) W(\phi_g^{-1}[x]) \, dg, \tag{5}
\]

where \( G \) is a group, \( g \in G \) is a group element with group action \( \phi_g \), and \( \eta \) is typically a canonical element in the domain of \( F \) (e.g. the origin if the domain is \( \mathbb{R}^n \)). The group convolution can be shown [Esteves et al., 2018b; Kondor and Trivedi, 2018] to be equivariant. The ordinary convolution is a special case of the group convolution.

### 3 Approaches that Guarantee Exact Rotational Equivariance

In this section we list and briefly discuss the approaches used to achieve exact rotational equivariance/invariance. The state-of-the-art approach is described in Section 3.3.

#### 3.1 Hardwired Pose Normalization

A basic approach to address the problem of rotational invariance consists in trying to “erase” the effect of rotations by reverting the input to a canonical pose by hardwiring a reversal function such as PCA. Problems are: small noise can strongly influence the result especially for objects with symmetries; learned low-level feature detectors do not generalize to other orientations.

#### 3.2 Handcrafted Features

Extractors of simple rotationally invariant features can be handcrafted rather than learned. Examples include features based on distances between pairs of points (SE\((n)\)-invariant), and/or between each point and the origin (invariant under rotations around the origin). Handcrafted feature extractors are not trained, i.e. not optimal.

#### 3.3 General Linear Equivariant Mappings and Equivariant Nonlinearities

In many situations, the best approach are the most general methods that guarantee equivariance [Kondor and Trivedi, 2018; Weiler et al., 2018a; Cohen et al., 2019]. Several formulations are available.

When using so-called irreducible representations of the rotation group, the equivariant mapping corresponds to the usage of steerable filters (see Section 2.2). This enables equivariance under all infinitely many rotation angles, but pointwise nonlinearities such as ReLU are not equivariant in this basis. Only other, special nonlinearities are equivariant. (See Section 3.3 by [Cohen et al., 2019] for an overview of equivariant nonlinearities, equivariant batch normalization, and equivariant residual learning.)

On the other hand, when using regular representations, the mapping corresponds to group convolution, Eq. (5), i.e. the usage of a finite number of rotated versions of arbitrary filters. This enables equivariance only under a discrete subgroup (e.g. 45° rotations in a plane) of the rotation group, but pointwise nonlinearities like ReLU are equivariant.

For 2D rotations, results are best in practice when group convolutions with small rotation angles and pointwise nonlinearities are used [Weiler and Cesa, 2019].
For the most common purposes, this is arguably the best of all approaches: it guarantees exact equivariance (unlike Section 3.1) without an obligation to use untrainable feature extractors (unlike Section 3.2) or unstable pose normalization (unlike Section 3.1).

Note that the exactness of the equivariance is slightly reduced when data are discretized to a pixel/voxel grid. Deeper layers might further amplify the impact of the angle between the grid and the object on the features. A part of this “missing part of equivariance” can be learned from training data (see Section 4.1). If the trained network has such “partially learned” equivariance, it is not obvious how it achieves it, i.e., the group action $\psi$ of intermediate layers is not known, unlike in the case of exact equivariance.

4 Approaches to Learn Approximate Rotational Equivariance

Various approaches exist that facilitate the learning of approximated (inexact) rotational equivariance. Note that for datasets where exact equivariance is appropriate, approaches that provide exact equivariance (Section 3.3) usually work better.

4.1 Data Augmentation

Data augmentation (i.e., random rotations of samples during training) is the most naïve approach to deal with rotational equivariance. Such rotational equivariance in the training data but not in the network architecture forces the network to learn to recognize each orientation of each object part “from scratch” and hampers generalization.

4.2 Learned Pose Normalization

Instead of hardwiring a pose normalization function as described in Section 3.1, it is possible to force or encourage the network to learn a reversion function directly from the training data. As an example of the “encourage” case, in spatial transformer networks [Jaderberg et al., 2015], learning a pose normalization is a facilitated but not a guaranteed side effect of learning to classify.

4.3 Soft Constraints

Another approach to let the network learn rotational equivariance/invariance is to introduce additional soft constraints, which are typically expressed by auxiliary loss functions that are added to the main loss function. For example, a similarity loss [Coors et al., 2018] can be defined, which penalizes large distances between the predictions or feature embeddings of rotated copies of the input that are simultaneously fed into separate streams of a siamese network.

The advantage of this approach is the ease of implementation. Furthermore, it can be used in combination with other approaches that provide non-exact equivariance (e.g., pose normalization) in order to enhance it. The disadvantage is that equivariance/invariance is only approximative. The quality of the approximation depends on the loss formula, training data, network architecture and optimization algorithm.

4.4 Deformable Convolution

The deformable convolution [Dai et al., 2017] augments a CNN’s capability of modeling geometric transformations. Input-dependent offsets are added to the sampling locations of the standard convolution. The offsets are computed by applying an additional convolutional layer over the same input feature map. Bilinear image interpolation is used due to non-integer offsets. The advantage of this approach is that it can learn to handle very general transformations such as rotation, scaling, and deformation, if training data encourage this. The disadvantage is that there is no guarantee of equivariance.

5 Overview of Methods

In this section we list and categorize deep learning methods for handling rotatable data and rotations. This includes methods that are equivariant under rotations of the input, methods that output a rotation, and methods that use rotations as inputs and/or deep features.

5.1 Equivariance under Rotations of the Input

Methods that are equivariant under rotations of the input are categorized in Table 1 (for 2D rotations) and Table 2 (for 3D rotations) according to the following criteria:

- **Input:**
  - Pixel grid: a grid representation of 2D image data
  - Voxel grid: a grid representation of 3D volumetric data
  - Point cloud: a set of 3D point coordinates
  - Spherical signal: a function defined on the sphere
  - Polygon mesh: a collection of vertices, edges, and faces that describes a surface consisting of polygons
  - dMRI (6D): six-dimensional diffusion-weighted magnetic resonance images [Müller et al., 2021]

- **Approach:** see Definition 4 and Sections 3–4
- **Property:** equivariance (Definition 1) or invariance (Definition 3)
- **Group:**
  - SO(2): the group of 2D rotations
  - SE(2): the group of 2D rigid-body motions
  - SO(3): the group of 3D rotations
  - SE(3): the group of 3D rigid-body motions
- **Cardinality:** continuous (entire group) or discretized to specific angles

5.2 Rotations as Output

Examples of deep learning methods that output a (3D) rotation are categorized in Table 3 and Table 4 according to the following characteristics:

- **Input to the network that outputs the rotation, and according rotation-prediction task:**
  - Image. The task is to estimate the orientation of a depicted object relative to the camera.
- Cropped stereo image: A pair of images is taken at the same time from two cameras that are close together and point in the same direction. The images are cropped in a predefined fashion. Each pair of cropped images constitutes one input. The position of the cameras relative to each other is fixed. The task is to estimate the orientation of a depicted object relative to the cameras.
- Volumetric data. The task is to estimate the orientation of an object relative to volume coordinates.
- Slice of volumetric data. The task is to estimate the orientation of a 2D slice relative to an entire (predefined) 3D object.
- Video: Two or three or more images (video frames). The task is to estimate the relative rotation and translation of the camera between (not necessarily consecutive) frames.

**Number of objects/rotations:** Describes how many rotations the network outputs.

- One rigid object: One rotation is estimated that is associated with a rigid object. The visible “object” is the entire scene in cases where camera motion relative to a static scene is estimated. Other small moving objects can be additionally accounted for (for example to refine the optical-flow estimation), but their rotation is not estimated.
- Hierarchy of object parts: Relative rotations between the objects and their parts are estimated, with several hierarchy levels, i.e. an “object” consisting of parts can itself be one of several parts of a “higher-level” object. Among the methods listed in Table 1, only capsule networks belong to this category.

**Specialization:**

- Specialized on one object: The network can only process one type of object on which it was trained.
- Specialized on multiple objects: The network can process an object from an arbitrarily large but fixed set of object types on which it was trained.
- Generalizing to new objects: The network can generalize to unseen types of objects.

**Group:** SO(3) or SE(3)

**Representation (embedding) of the rotation(s):**

- Rotation matrix
- Quaternion
- Euler angles
- Axis-angle representation
- Discrete bins: Rotations are grouped into a finite set of bins.
- Transformation matrix
- 3D coordinates of four keypoints on the object (predefined object-specifically, e.g. four of its corners)
- Eight corners and centroid of 3D bounding box projected into 2D image space

### Table 1: Methods with equivariance under 2D rotations. The terminology is summarized in Section 5.1. The input to each method is an image. In Polar Transformer Networks, the image is transformed to a circular signal in an intermediate layer. Methods with identical cell entries differ in terms of details. Potential weaknesses are highlighted. General Steerable CNNs and their implementation in the eCnn library are the “best” in that they provide various hyperparameter choices, with the other exact methods being special cases thereof (see Section 3.3).

| Method                                      | Approach                  | Property               | Group       | Cardinality |
|---------------------------------------------|---------------------------|------------------------|-------------|-------------|
| Many                                        | Learned                   | *                      | *           | *           |
| Spatial Transformer Networks [Jaderberg et al., 2015] | Learned (Data augmentation) | Invariance             | SE(2)       | Continuous  |
| Cyclic Symmetry in CNNs [Dieleman et al., 2016] | Exact                    | Equi-var.              | SE(2)       | Discretized (90° angles) |
| Group Equivariant CNNs [Cohen and Welling, 2016] | Exact                    | Equi-var.              | SE(2)       | Discretized (90° angles) |
| Harmonic Networks [Worrall et al., 2017]     | Exact                    | Equi-var.              | SE(2)       | Continuous  |
| Vector Field Networks [Marcos et al., 2017]  | Exact                    | Equi-var.              | SE(2)       | Discretized (any angle) |
| Oriented Response Networks [Zhou et al., 2017b] | Exact                    | Equi-var.              | SE(2)       | Discretized (any angle) |
| Deformable CNNs [Dai et al., 2017]           | Learned (Deformable convolution) | Equi-var.              | SE(2)       | Continuous  |
| Polar Transformer Networks [Esteves et al., 2018b] | Learned (learned pose normalization) | Equi-variance           | SE(2)       | Continuous  |
| Steerable Filter CNNs [Weiler et al., 2018b] | Exact                    | Equi-var.              | SE(2)       | Discretized (any angle) |
| Learning invariance with weak supervision [Coors et al., 2018] | Learned (soft constraints) | Invariance             | SE(2)       | Continuous  |
| Roto-Translation Covariant CNNs [Bekkers et al., 2018] | Exact                    | In-var.                | SE(2)       | Discretized (any angle) |
| RotDCF: Decomposition of Convolutional Filters [Cheng et al., 2019] | Exact                    | Equi-variance           | SE(2)       | Discretized (any angle) |
| RfCNN [Chidester et al., 2018]               | Exact                    | In-var.                | SO(2)       | Discretized (any angle) |
| Siamese Equivariant Embedding [Véges et al., 2019] | Learned (soft constraints) | Equi-var.              | SO(2)       | Continuous  |
| CNN model of primary visual cortex [Ecker et al., 2019] | Exact                    | Equi-variance           | SE(2)       | Discretized (any angle) |
| General Steerable CNNs [Weiler and Cesa, 2019] | Exact                    | Equi-var.              | SE(2)       | Continuous  |

- Learned representation: The latent space (e.g. of an autoencoder) is used to represent the rotation. There are several interesting aspects at play:
  - Learned representations allow for ambiguity: If
Table 2: Methods with equivariance under 3D rotations. The terminology is summarized in Section 5.1. Potential weaknesses are highlighted. Tensor Field Networks are the “best” for point clouds in that they provide continuous exact SE(3)-equivariance. Similarly, 3D Steerable CNNs are the “best” neural networks for voxel grids. On the other hand, CubeNet and 3D G-CNNs offer only discrete rotations but are compatible with nonlinearities such as ReLU. The exact methods for 3D data are available via the e3nn library.

Table 3: Examples of deep learning methods that can output a 3D rotation, where a ground truth rotation is used for training. The terminology is summarized in Section 5.2. Losses that lack rotational invariance are highlighted.

1 Initially squared $L^2$ distance in embedding space (fast to compute); then geodesic distance for rotation and squared $L^2$ distance for translation.
2 Each method from the SSD-AF family uses a different embedding: discrete bins, four keypoint locations in 3D space, quaternion, Euler angles.
| Method                              | Input | Specialization | Group | Embedding | Loss function |
|------------------------------------|-------|----------------|-------|-----------|---------------|
| Capsule Networks [Sabour et al., 2017] | Image | Generalizing to new objects | SE(3) | Transformation matrices | Object classification |
| Spatial Transformer Networks [Jaderberg et al., 2015] | Volume | Generalizing to new objects | SE(3) | Transformation matrix | Object classification |
| Unsupervised learning of depth and ego-motion [Zhou et al., 2017a] | Video (three consecutive frames) | Generalizing to new objects | SE(3) | Euler angles | View warping |
| Learning implicit representations of 3D object orientations from RGB [Sundermeyer et al., 2018] | Image | One object | SO(3) | Learned representation | Auto-encoder |
| GeoNet [Yin and Shi, 2018] | Video (several consecutive frames) | Generalizing to new objects | SE(3) | Euler angles | View warping |

1. Poses of lowest-level object parts: learned representation; part-to-object pose transformations: transformation matrices (as trainable parameters).

Table 4: Examples of deep learning methods that can output a 3D rotation, where a ground truth rotation is not necessary for training. The terminology is summarized in Section 5.2. Losses are highlighted for which a good loss value does not guarantee a good prediction of rotations.

results in the same loss value, regardless of the ground truth rotation and of the direction into which the prediction is biased.

- $L^p$ distance in embedding space: This loss value is fast to compute but not rotationally invariant, i.e., an error of $10^p$ yields different loss values depending on the ground truth and on the prediction. Due to this “unfairness” or “arbitrarity”, such losses are highlighted in the table.

- Rotations are estimated in an intermediate layer and used in subsequent layers for a “higher-level” goal of a larger system. Ground truth rotations are not required. These methods are listed in Table 4.

- Object classification: Rotation prediction is trained as part of a larger system for object classification. The estimated rotation is used to rotate the input or feature map (in spatial transformer networks) or predicted poses (in capsule networks) as an intermediate processing step. It is assumed that learning to rotate to a canonical pose (in spatial transformer networks) or to let object parts vote about the overall object pose (in capsule networks) is beneficial for object classification. The estimation of rotation is incidental and encouraged by the overall setup. However, its approximate correctness is not necessary for perfect object classification. Therefore, the “predicted rotation” can be very wrong, and due to this danger this loss is highlighted in the table.

- View warping: At least two video frames are used to estimate the scene geometry (depth maps) and camera motion between the views (rotation, translation). These estimates are used to warp one view (image, and possibly depth map) to resemble another view. The loss measures this resemblance. This is a form of self-supervised learning: ground truth geometry and motion are not given, but are estimated such that they cause warping that is consistent with the input images. The rotation estimation can be expected to be good, because it is necessary for good view synthesis.

- Autoencoder reconstruction loss: The network is trained to reconstruct its input (a view of the object) after passing it through a lower-dimensional latent space. The output target has a neutral image background and lacks other objects that were visible in the input image. This allows the network to learn to discard the information about the background and other objects before the bottleneck layer. If the network is specialized on one object, then maintaining in the latent space only the information about the object pose is sufficient for such reconstruction. If additionally the latent space is sufficiently low-dimensional, then the learning is encouraged to be economic about the amount of information encoded in the latent space, i.e., to encode nothing but the pose.

Estimation of 2D rotations is simpler in terms of representation. Predicting the sine and cosine of the rotation angle (and normalizing the predicted vector to length 1), because otherwise the predicted sine and cosine might slightly contradict each other, or be beyond $[-1, 1]$, is better than predicting the angle, because the latter requires learning a function that has a jump (from 360° to 0°), which is not easy for (non-generative) neural networks.

5.3 Rotations as Input or as Deep Features

Other uses of rotations in deep learning are to take rotations as input, or to restrict deep features to belong to $SO(n)$ (without requiring them to directly approximate rotations present in the data). For example, [Huang et al., 2017] use rotation matrices as inputs and as deep features. They restrict deep features to $SO(3)$ by using layers that map from $SO(3)$ to $SO(3)$.

6 Conclusions

Among the methods for equivariance/invariance, the most successful ones are based on the exact and most general approach (Section 3.3). They are very effective in 3D input domains as well. With emerging theory [Kondor and Trivedi, 2018, Cohen et al., 2019] for exact equivariance and with emerging approaches, it appears to be the perfect time to use the methods in various application domains and to tune them. Existing pipelines that do not have (exact) equivariance yet and for
example rely on data augmentation are likely to benefit from incorporating exact-equivariance approaches.

Acknowledgments

We thank Erik Bekkers, Maurice Weiler, Gabriele Cesa, Antonij Golkov, Taco Cohen, Christine Allen-Blanchette, Qadeer Khan, Philip Müller, Philip Häusser, and Remco Duits for valuable discussions. This manuscript was supported by the ERC Consolidator Grant “3DReloaded”, the Munich Center for Machine Learning (Grant No. 01IS18036B), and the BMBF project MLwin.

References

[Bekkers et al., 2018] E. J. Bekkers, M. W. Lafarge, M. Veta, K. A. J. Eppehof, J. P. W. Pluim, and R. Duits. Rotation-translation covariant convolutional networks for medical image analysis. In MICCAI, pages 440–448, 2018.

[Cheng et al., 2019] X. Cheng, Q. Qiu, R. Calderbank, and G. Sapiro. RotDCF: decomposition of convolutional filters for rotation-equivariant deep networks. In ICLR, 2019.

[Chidester et al., 2018] B. Chidester, M. N. Do, and J. Ma. Rotation equivariance and invariance in convolutional neural networks. arXiv:1805.12301, 2018.

[Cohen and Welling, 2016] T. S. Cohen and M. Welling. Group equivariant convolutional networks for medical image analysis. In MICCAI, pages 440–448, 2016.

[Cohen et al., 2018] T. S. Cohen, M. Geiger, J. Köhler, and M. Welling. Spherical CNNs. In ICLR, 2018.

[Cohen et al., 2019] T. S. Cohen, M. Geiger, and M. Weiler. A general theory of equivariant CNNs on homogeneous spaces. In NeurIPS, pages 9142–9153, 2019.

[Coors et al., 2018] B. Coors, A. Condurache, A. Mertins, and A. Geiger. Learning transformation invariant representations with weak supervision. In VISAPP, pages 64–72, 2018.

[Dai et al., 2017] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei. Deformable convolutional networks. In ICCV, pages 764–773, 2017.

[Han et al., 2021] P. de Haan, M. Weiler, T. Cohen, and M. Welling. Gauge equivariant mesh CNNs: Anisotropic convolutions on geometric graphs. In ICLR, 2021.

[Deng et al., 2018] H. Deng, T. Birdal, and S. Ilic. PF-FoldNet: unsupervised learning of rotation invariant 3D local descriptors. In ECCV, pages 620–638, 2018.

[Dieleman et al., 2016] S. Dieleman, J. De Fauw, and K. Kavukcuoglu. Exploiting cyclic symmetry in convolutional neural networks. In ICML, pages 1889–1898, 2016.

[Ecker et al., 2019] A. S. Ecker, F. H. Sinz, E. Froudarakis, P. G. Fahey, S. A. Cadena, E. Y. Walker, E. Cobos, J. Reimer, A. S. Tolias, and M. Bethge. A rotation-equivariant convolutional neural network model of primary visual cortex. In ICLR, 2019.

[Esteves et al., 2018a] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis. Learning SO(3) equivariant representations with spherical CNNs. In ECCV, pages 54–70, 2018.

[Esteves et al., 2018b] C. Esteves, C. Allen-Blanchette, X. Zhou, and K. Daniilidis. Polar transformer networks. In ICLR, 2018.

[Geiger et al., 2020] M. Geiger, T. Smidt, Alby M., B. K. Miller, W. Boomsma, B. Dice, K. Lapcheykys, M. Weiler, M. Tyszkievicz, S. Batzner, M. Uhrin, J. Frellsen, N. Jung, S. Sanborn, J. Rackers, and M. Bailey. Euclidean neural networks: e3nn, 2020.

[Georgakis et al., 2019] G. Georgakis, S. Karanam, Z. Wu, and J. Kosecka. Learning local RGB-to-CAD correspondences for object pose estimation. In ICCV, pages 8967–8976, 2019.

[Huang et al., 2017] Z. Huang, C. Wan, T. Probst, and L. V. Gool. Deep learning on Lie groups for skeleton-based action recognition. In CVPR, pages 1243–1252, 2017.

[Jaderberg et al., 2015] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In NeurIPS, pages 2017–2025, 2015.

[Kendall et al., 2015] A. Kendall, M. K. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In ICCV, pages 2938–2946, 2015.

[Kondor and Trivedi, 2018] R. Kondor and S. Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. In ICML, pages 2747–2755, 2018.

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

[Mahendran et al., 2017] S. Mahendran, H. Ali, and R. Vidal. 3D pose regression using convolutional neural networks. In CVPR Workshops, pages 494–495, 2017.

[ Marcos et al., 2017] D. Marcos, M. Volpi, N. Komodakis, and D. Tuia. Rotation equivariant vector field networks. In ICCV, pages 5058–5067, 2017.

[Melekhov et al., 2017] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Relative camera pose estimation using convolutional neural networks. In ACTIVS, pages 675–687, 2017.

[Mohseni Salehi et al., 2019] S. S. Mohseni Salehi, S. Khan, D. Erdogmus, and A. Gholipour. Real-time deep pose estimation with geodesic loss for image-to-template rigid registration. IEEE Trans Med Imaging, pages 470–481, 2019.

[Müller et al., 2021] P. Müller, V. Golkov, V. Tomassini, and D. Cremers. Rotation-equivariant deep learning for diffusion MRI. arXiv preprint, 2021.

[Pandey et al., 2018] R. Pandey, P. Pidhynskiy, S. Yang, and C. Kaeser-Chen. Efficient 6-DoF tracking of hand-held objects from an egocentric viewpoint. In ECCV, pages 426–441, 2018.
[Sabour et al., 2017] S. Sabour, N. Frosst, and G. E. Hinton. Dynamic routing between capsules. In NeurIPS, pages 3856–66, 2017.

[Sundermeyer et al., 2018] M. Sundermeyer, E. Y. Puang, Z.-C. Marton, M. Durner, and R. Triebel. Learning implicit representations of 3D object orientations from RGB. In ICRA, 2018.

[Tekin et al., 2018] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6D object pose prediction. In CVPR, pages 292–301, 2018.

[Thomas et al., 2018] N. Thomas, T. Smidt, S. M. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley. Tensor field networks: rotation- and translation-equivariant neural networks for 3D point clouds. arXiv:1802.08219, 2018.

[Végés et al., 2019] M. Végés, V. Varga, and A. Lörincz. 3D human pose estimation with siamese equivariant embedding. Neurocomputing, pages 194 – 201, 2019.

[Weiler and Cesa, 2019] M. Weiler and G. Cesa. General E(2)-equivariant steerable CNNs. In NeurIPS, pages 14334–45, 2019.

[Weiler et al., 2018a] M. Weiler, M. Geiger, M. Welling, W. Boomsma, and T. Cohen. 3D steerable CNNs: learning rotationally equivariant features in volumetric data. In NeurIPS, pages 10381–10392, 2018.

[Weiler et al., 2018b] M. Weiler, F. A. Hamprecht, and M. Storath. Learning steerable filters for rotation equivariant CNNs. In CVPR, pages 849–858, 2018.

[Winkels and Cohen, 2018] M. Winkels and T. S. Cohen. 3D G-CNNs for pulmonary nodule detection. In MIDL, 2018.

[Worrall and Brostow, 2018] D. Worrall and G. Brostow. CubeNet: equivariance to 3D rotation and translation. In ECCV, pages 567–584, 2018.

[Worrall et al., 2017] D. E. Worrall, S. J. Garbin, D. Turmukhambetov, and G. J. Brostow. Harmonic networks: deep translation and rotation equivariance. In CVPR, pages 7168–7177, 2017.

[Yin and Shi, 2018] Z. Yin and J. Shi. GeoNet: Unsupervised learning of dense depth, optical flow and camera pose. In CVPR, pages 1983–1992, 2018.

[Zhou et al., 2017a] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and egomotion from video. In CVPR, pages 6612–6619, 2017.

[Zhou et al., 2017b] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao. Oriented response networks. In CVPR, pages 4961–70, 2017.