

# Graph-Based Classification of Omnidirectional Images

Renata Khasanova  
 EPFL  
 Switzerland

renata.khasanova@epfl.ch

Pascal Frossard  
 EPFL  
 Switzerland

pascal.frossard@epfl.ch

## Abstract

Omnidirectional cameras are widely used in such areas as robotics and virtual reality as they provide a wide field of view. Their images are often processed with classical methods, which might unfortunately lead to non-optimal solutions as these methods are designed for planar images that have different geometrical properties than omnidirectional ones. In this paper we study image classification task by taking into account the specific geometry of omnidirectional cameras with graph-based representations. In particular, we extend deep learning architectures to data on graphs; we propose a principled way of graph construction such that convolutional filters respond similarly for the same pattern on different positions of the image regardless of lens distortions. Our experiments show that the proposed method outperforms current techniques for the omnidirectional image classification problem.

## 1 Introduction

Omnidirectional cameras are very attractive for various applications in robotics [1, 2] and computer vision [3, 4] thanks to their wide viewing angle. Despite this advantage, working with the raw images, taken by such cameras is difficult because of severe distortion effects introduced by the camera geometry or lens optics, which has a significant impact on local image statistics. Therefore, all methods that aim at solving different computer vision tasks (e.g. detection, points matching, classification) on the images from the omnidirectional cameras need to find a way of compensating for this distortion. The natural way to do it is to apply calibration techniques [5, 6, 7] to undistort images and then use standard computer vision algorithms. However, undistorting the full omnidirectional images is a complex problems by itself, and distorting parts of the image requires a priori information about the camera and the image model. Further, undistorting real images may suffer from interpo-

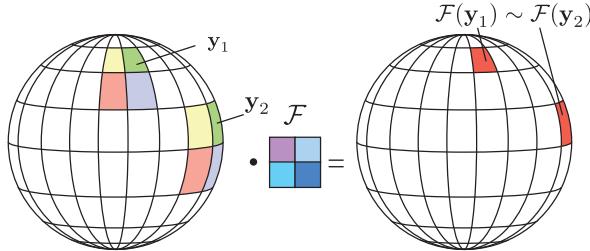


Figure 1. The proposed graph construction method makes response of the filter similar regardless of different position of the pattern on an image from an omnidirectional camera.

tion artifacts. A different, ‘naive’, approach is to apply standard techniques directly to raw (distorted) images. However, algorithms proposed for the planar images lead to non-optimal solutions when applied to distorted images.

One example of such standard techniques are the Convolutional Neural Networks (ConvNets) [8], which are primarily designed for regular domains [9]. They have achieved remarkable success in various areas of computer vision [10, 11, 12]. The drawback of this solution is that ConvNets require a lot of training data for omnidirectional image classification task, as the same object will not have the same local statistics, for different image locations, which results in different filter responses. Therefore, the dataset should include images where same objects are seen in different parts of the image in order to reach invariance to distortions.

In this work, we propose to design a solution for image classification that inherently takes into account the camera geometry. Developing such a technique based on the classic ConvNets is, however, complicated due to the two main reasons. First as we mentioned before the features, extracted by the network, need to be invariant to positions of objects in the scene and different orientations with respect to the omnidirectional camera. Second, it is challenging to incorporate lens geometry knowledge in the structure of convolutional filters. Luckily graph-based deep learning techniques have been recently introduced [13, 14, 15] that allow applying deep learning techniques to irregularly structured data. Our work is inspired by [15] where the authors use

graphs to create isometry invariant features of images in Euclidean space. This tackles the first of the aforementioned challenges, however, the same object seen at different positions of an omnidirectional image still remains different from the network point of view. To mitigate this issue, we propose to incorporate the knowledge about the geometry of the omnidirectional camera lens into the signal representation, namely in the structure of the graph (see Fig. 1). In summary we therefore propose the following contributions:

- a principled way of graph construction based on geometry of omnidirectional images;
- graph-based deep learning architecture for the omnidirectional image classification task.

The reminder of the paper is organized as follows. We first discuss the related work in Section 2. Further, in Section 3 we briefly introduce the TIGraNet architecture [15], as it is tightly related to our approach, and then we describe our graph construction method that can be efficiently used by TIGraNet. Finally, we show the result of our experiments in Section 5 and we conclude in Section 6.

## 2 Related work

To the best of our knowledge, image classification methods designed specifically for the omnidirectional camera do not exist. Therefore, in this section we review methods designed for wide-angle cameras for different computer vision applications. Then we discuss recent classification approaches based on graphs as we believe that graph signal processing provides with powerful tools to deal with images that have an irregular structure.

### 2.1 Wide-angle view cameras

A broad variety of computer vision tasks benefit from having wide-angle cameras. For example, images from fisheye [16], which can reach field of view (FOV) of more than  $180^\circ$ , or omnidirectional cameras, that provide  $360^\circ$  FOV [17, 18] are widely used in virtual reality and robotics [17, 19] applications. Despite practical their benefits these images are challenging to process due to the fact that most of the approaches are developed for planar images and suffer from distortion effects when applied to images from wide-angle view cameras [16].

There exist different ways of acquiring an omnidirectional image. First such an image can be built based on a set of multiple images, taken either by a camera that is rotated around its center of projection, or by multiple calibrated cameras. Rotating camera systems, however cannot be applied to dynamic scenes, while multi-camera systems suffer from calibration difficulties. Alternatively, one can obtain

an omnidirectional image from dioptric or catadioptric cameras [20]. Most of the existing catadioptric cameras have the following mirror types: elliptic, parabolic and hyperbolic. The authors in [21] show that such mirror types allow for a bijective mapping of the lens surface to a sphere, which simplifies processing of omnidirectional images. In our paper we work with this spherical representation of catadioptric omnidirectional cameras. The analysis of images from wide-angle cameras remains however an open problem.

For example, the standard approaches for interest point matching propose affine-invariant descriptors such as SIFT [22], GIST [23]. However, designing descriptors that preserve invariance to geometric distortions for wide-angle camera's images is challenging. One of the attempts to achieve such invariance is proposed by [24], where the authors extend the GIST descriptor to omnidirectional images by exploiting their circular nature. Instead of using hand-crafted descriptors, the authors in [25] suggest to learn them from the data by creating a similarity preserving hashing function. Further, inspired by the aforementioned method, the work in [4] proposes to learn descriptors for images from omnidirectional cameras using a siamese neural network [26]. While this method is not using specific geometry of the lens, it significantly outperforms state-of-the-art as it encodes transformations that are present in the omnidirectional images. However, the method requires carefully constructed training dataset to learn all possible variations of the data.

Contrary to the previous approaches, the methods in [3, 27, 28] design a scale invariant SIFT descriptor for the wide-angle cameras based on the result of the work in [21] that introduced a bijection mapping between omnidirectional images and a spherical surface. In particular, the method in [3] maps images to a unit sphere, and those in [27] propose two SIFT-based algorithms, which work in spherical coordinates. The first approach (local spherical) matches points between two omnidirectional images, while the second one (local planar) works between spherical and planar images. Finally, the authors in [28] adapt a Harris interest point detector [29] to spherical nature of images from omnidirectional cameras. All the aforementioned works are designed for interest point matching task. In our work we use the similar idea of mapping omnidirectional images to the spherical surface [21] for omnidirectional image classification problem.

Omnidirectional cameras have also been widely used in other computer vision and robotics tasks. For example, the authors in [30] propose a segmentation method for catadioptric cameras. They derive explicit expression for edge detection and smoothing differential operators and design a new energy functional to solve segmentation problem. The work in [31] then develops a stereo-based depth estimation approach from multiple cameras. Further, the authors in

[20] extend previous geometry-based calibration approach to compute depth and disparity maps from images captured by a pair of omnidirectional cameras. They also suggest an efficient way of sparse 3D scene representation. The works in [17, 18] then use omnidirectional cameras for robot self-localisation and reliable estimation of the 3D map of the environment (SLAM). The authors in [32] propose a motion estimation method for catadioptric omnidirectional images by the evaluation of the correlation between them from arbitrary viewpoints. Finally, the work in [33] utilizes geometry of the omnidirectional camera to adapt the quantization tables of ordinary block-based transform codecs for panoramic images computed by equirectangular projection.

In summary, processing images from omnidirectional cameras becomes an important topic in the computer vision community. However, most of the existing solutions rely on methods developed for planar images. In this paper we are particularly interested in image classification tasks and propose a solution based on a combination of powerful deep learning architecture and camera lens geometry.

## 2.2 Deep learning on graphs

We briefly review here classification methods based on deep learning algorithms (DLA) for graph data as DLA has proven its efficiency in many computer vision tasks and graphs allow extending these methods to irregularly structured data, such as omnidirectional images. A more complete review can be found in [34].

First, the authors in [35, 36] propose a new deep learning architecture that uses filters in spectral domain to work with irregular data, where they add a smoothing constraint to avoid overfitting. These methods have high computational complexity as they require eigendecomposition as a preprocessing step. To reduce this complexity, the authors in [14] propose using Chebyshev polynomials, which can be efficiently computed in an iterative manner and allow for fast filter generation. The work in [37] uses similar polynomial filters of degree 1, which allow training deeper and more efficient models for semi-supervised learning tasks without increasing the complexity.

Finally, the recent method in [15] introduces graph-based global isometry invariant features. Their approach is developed for regular planar images. We, on the other hand, propose to design a graph signal representation that decreases feature sensitivity to different types of geometric distortion introduced by omnidirectional cameras.

## 3 Graph convolutional network

In this paper we construct a system to classify images from omnidirectional camera based on a deep learning architecture. In particular, we extend the network from [15] to pro-

cess images with geometric distortion. In this section we briefly review the main components of this approach. The system in [15] takes as input images that are represented as signals on a grid graph and gives classification labels as output. Briefly this approach proposes a network of alternatively stacked spectral convolutional and dynamic pooling layers, which creates features that are equivariant to the isometric transformation. Further, the output of the last layer is processed by a statistical layer, which makes the equivariant representation of data invariant to isometric transformations. Finally, the resulting feature vector is fed to a number of fully-connected layers and a softmax layer, which outputs the probability distribution that the signal belongs to each of the given classes.

We extend this transformation-invariant classification algorithm to omnidirectional images by incorporating the knowledge about the camera lens geometry in the graph structure. We assume that the image projection model is known and propose representing images as signals  $\mathbf{y}$  on the irregular grid graph  $G$ . More formally, the graph is a set of nodes, edges and weights. Thus, each graph signal  $\mathbf{y} : \{\mathbf{y}(v_i)\}$  is defined on nodes  $v_i, i \in [1..N]$  of  $G$ . We denote by  $\mathbf{A}$  an adjacency matrix of  $G$ , which shows weighted connection between vertices, and by  $\mathbf{D}$  a diagonal degree matrix with  $D_{ii} = \sum_{j=1}^N A_{ij}$ . This allows us to define Laplacian matrix<sup>1</sup> as follows:

$$\mathbf{L} = \mathbf{D} - \mathbf{A}. \quad (1)$$

The Laplacian matrix is an operator, that is widely used in graph signal processing [38], because it allows to define a graph Fourier transform to perform analysis of graph signals. The transformed signal reads:

$$\hat{\mathbf{y}}(\lambda_l) := \langle \mathbf{y}, \mathbf{u}_l \rangle, \quad (2)$$

where  $\lambda_l$  is an eigenvalue of  $\mathbf{L}$  and  $\mathbf{u}_l$  is the associated eigenvector. It gives a spectral representation of the signal  $\mathbf{y}$  and  $\lambda_l, l \in [0..N-1]$  provide a similar notion of frequencies as in classical Fourier analysis. Thus, the filtering of a graph signal  $\mathbf{y}$  can be defined on graphs in the spectral domain:

$$\mathcal{F}(\mathbf{y}(v_i)) = \sum_{l=0}^{N-1} \hat{f}(\lambda_l) \hat{\mathbf{y}}(\lambda_l) \mathbf{u}_l(v_i), \quad (3)$$

where  $\mathcal{F}(\mathbf{y}(v_i))$  is the filtered signal value on the node  $v_i$  and  $\hat{f}(\lambda_l)$  is a graph filter. Graph filters can be constructed as polynomial function, namely  $\hat{f}(\lambda_l) = \sum_{m=0}^M \alpha_m \lambda_l^m$ , where  $M$  is the degree of the polynomial and  $\alpha_m$  are the

<sup>1</sup>We use non-normalized version of Laplacian matrix which is different from [15] to simplify the derivation. The same result can be obtained with the normalized version of  $\mathbf{L}$ .

parameters. Such filters can advantageously be applied directly in vertex domain to avoid computationally expensive eigendecomposition [38]:

$$\mathcal{F}(\mathbf{y}) = \left[ \sum_{m=0}^M \alpha_m \mathbf{L}^m \right]^T \mathbf{y}, \quad (4)$$

where the filter has the form  $\mathcal{F} = \sum_{m=0}^M \alpha_m \mathbf{L}^m$ . The spectral convolutional layer in the deep learning system of [15] consists of  $J$  such filters  $\mathcal{F}_j, j \in [1..J]$ . Each column  $i$  of  $\mathcal{F}_j$  represents localization of the filter on the node  $v_i$ . These nodes are chosen by preceding pooling layer, which selects nodes with maximum response. Finally, a statistical layer collects global multi-scale statistic of input feature maps, which results in rotation and translation invariant features. Please refer to [15] for more details about the TIGraNet architecture.

In the next section, we show how this architecture can be extended for omnidirectional images. In particular, we discuss how to compute the weights  $\mathbf{A} : \{w_{ij}\}$  between the nodes according to the lens geometry in order to build a proper Laplacian matrix for spectral convolutional filters.

## 4 Graph-based representation

### 4.1 Image model

An omnidirectional image is typically represented as a spherical one (see Fig. 2), where each point  $\mathbf{X}_k$  from 3D space is projected to the points  $\mathbf{x}_k$  on the spherical surface  $\mathbf{S}$  with radius  $r$ , which we set to  $r = 1$  without loss of generality. The point  $\mathbf{x}_k$  is then uniquely defined by its longitude  $\theta_k \in [-\pi, \pi]$  and latitude  $\phi_k \in [-\frac{\pi}{2}, \frac{\pi}{2}]$  and its coordinates can be written as:

$$\mathbf{x}_k : \begin{bmatrix} \cos \theta_k \cos \phi_k \\ \sin \theta_k \cos \phi_k \\ \sin \phi_k \end{bmatrix}, k \in [1..N]. \quad (5)$$

We consider objects on a plane that is tangent to the sphere  $\mathbf{S}$ . We denote by  $\mathbf{X}_{k,i}$  a 3D space point on the plane  $\mathbf{T}_i$  tangent to the sphere at  $(\phi_i, \theta_i)$ . The point  $\mathbf{X}_{k,i}$  is defined by the coordinates  $(x_{k,i}, y_{k,i})$  on the plane  $\mathbf{T}_i$ . We, further, denote by  $\mathbf{x}_{k,i} : (\phi_k, \theta_k)$  the points on the surface of the sphere that are obtained by connecting its center and the point  $\mathbf{X}_{k,i}$  on  $\mathbf{T}_i$ . We can find coordinates of each point  $\mathbf{X}_{k,i}$  on  $\mathbf{T}_i$  by using the gnomonic projection, which generally provides a good model for omnidirectional images [39]:

$$\begin{aligned} x_{k,i} &= \frac{\cos \phi_k \sin(\theta_k - \theta_i)}{\cos c}, \\ y_{k,i} &= \frac{\cos \phi_i \sin \phi_k - \sin \phi_i \cos \phi_k \cos(\theta_k - \theta_i)}{\cos c}, \end{aligned} \quad (6)$$

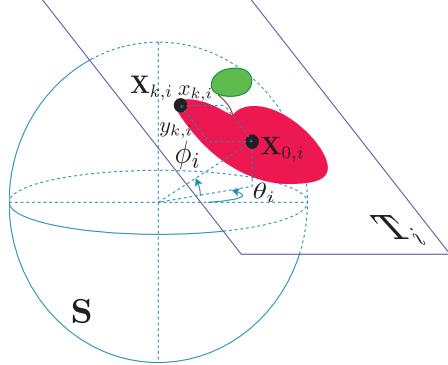


Figure 2. Example of the gnomonic projection. An object from tangent plane  $\mathbf{T}_i$  is projected to the sphere at tangency point  $\mathbf{X}_{0,i}$ , which is defined by spherical coordinates  $\phi_i, \theta_i$ . The point  $\mathbf{X}_{k,i}$  is defined by coordinates  $(x_{k,i}, y_{k,i})$  on the plane.

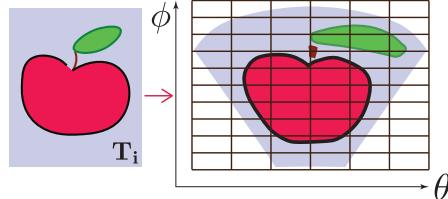


Figure 3. Example of the equirectangular representation of the image. On the left, the figure depicts the original image on the tangent plane  $\mathbf{T}_i$ , on the right, projected to the points of the sphere. To build an equirectangular image the values points on the discrete regular grid are often approximated from the values of projected points by interpolation.

where  $c$  is the angular distance between the point  $(x_{k,i}, y_{k,i})$  and the center of projection  $\mathbf{X}_{0,i}$  and is defined as follows:

$$\begin{aligned} \cos c &= \sin \phi_i \sin \phi_k + \cos \phi_i \cos \phi_k \cos(\theta_k - \theta_i), \\ c &= \tan^{-1} \left( \sqrt{x_{k,i}^2 + y_{k,i}^2} \right). \end{aligned} \quad (7)$$

Fig. 2 illustrates an example of this gnomonic projection.

In order to easily process the signal defined on the spherical surface, it is typically projected to an equirectangular image (see Fig. 3). The latter represents the signal on the regular grid with step sizes  $\Delta\theta$  and  $\Delta\phi$  for angles  $\theta$  and  $\phi$  respectively. In this paper we work with these equirectangular images and assume that the object, which we are classifying, is lying on a plane  $\mathbf{T}_i$  tangent to the sphere  $\mathbf{S}$  at the point  $(\phi_i, \theta_i)$ . Our work could however be adapted to other projection models, such as [40]. Finally, each point on the equirectangular image is considered as a vertex  $v_i$  in our graph representation. The graph then connects nearest neighbors of the equirectangular image  $\mathbf{y}(v_i) = \mathbf{y}(\phi_i, \theta_i)$

### 4.2 Weight design

Our goal is to develop a transformation invariant system, which can recognize the same object on different planes  $\mathbf{T}_i$

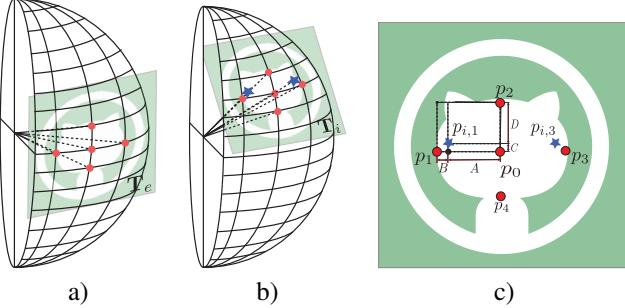


Figure 4. a) We choose pattern  $p_0, \dots, p_4$  from an object on tangent plane  $\mathbf{T}_e$  at equator ( $\phi_e = 0, \theta_e = 0$ ) (red points) and then, b) move this object on the sphere by moving the tangent plane  $\mathbf{T}_i$  to point  $(\phi_i, \theta_i)$ . c) Thus, the filter localized at tangency point  $(\phi_i, \theta_i)$  uses values  $p_{i,1}, p_{i,3}$  (blue points) which we can obtain by interpolation.

that are tangent to  $\mathbf{S}$  at different points  $(\phi_i, \theta_i)$  without any extra training. The challenge of building such a system is to design a proper graph signal representation that allow compensating for the distortion effects that appear on different elevations of  $\mathbf{S}$ . In order to properly define the structure, namely to compute the weights that satisfy the above condition we analyze, how a pattern projected a plane  $\mathbf{T}_e$  at equator ( $\phi_e = 0, \theta_e = 0$ ) varies on  $\mathbf{S}$  with respect to the same pattern projected onto another plane  $\mathbf{T}_i$  tangent to the sphere at  $(\phi_i, \theta_i)$ . We use this result to minimize the difference between filter responses of two projected pattern versions. Generally, the weight choice depends on distances  $d_{ij}$  between neighboring nodes of graph  $w_{ij} = g(d_{ij})$ . In this section we show that the function  $g(d_{ij}) = \frac{1}{d_{ij}}$  satisfies the above invariance condition.

**Pattern choice.** For simplicity we consider a 5-point pattern  $\{p_0, \dots, p_4\}$  on a tangent plane, which is depicted by the Fig. 4:

$$p_j := \mathbf{X}_{j,e}, \quad \forall j \in [0..4], \quad (8)$$

where  $\mathbf{X}_{j,e}$  are the points on the plane  $\mathbf{T}_e$  tangent to an equator point  $\phi_e = 0, \theta_e = 0$  and  $\mathbf{X}_{0,e} = \mathbf{x}_{0,e}$  is the tangency point. Further, pattern points  $\{p_0, \dots, p_4\}$  are also chosen in such a way that they are projected to the following locations on the sphere  $\mathbf{S}$ :

$$\begin{aligned} p_0 &\mapsto (0, 0) \\ p_2, p_4 &\mapsto (0 \pm \Delta\phi, 0) \\ p_1, p_3 &\mapsto (0, 0 \pm \Delta\theta) \end{aligned} \quad (9)$$

These essentially correspond to the pixel locations of the equirectangular representation of the spherical surface introduced in Section 4.1. The chosen pattern has the following coordinates on the tangent plane at equator  $\mathbf{T}_e$ :

$$\begin{aligned} \mathbf{X}_{0,e} &= (0, 0) \\ \mathbf{X}_{2,e}, \mathbf{X}_{4,e} &= (0, \pm \tan \Delta\phi) \\ \mathbf{X}_{1,e}, \mathbf{X}_{3,e} &= (\pm \tan \Delta\theta, 0) \end{aligned} \quad (10)$$

**Filter response.** Our objective is to design a graph, which can encode the geometry of an omnidirectional camera in the final feature representation of an image. Ideally, the same object at different positions on the sphere should have the same feature response (see Fig. 1) or equivalently they should generate the same response to given filters. Therefore, we choose the graph construction, or equivalently the weights of the graph in such a way that the difference between the responses of a filter applied to gnomonic projection of the same pattern on different tangent planes  $\mathbf{T}_i$  is minimized. We consider a graph where each node is connected with 4 of its nearest neighbours and take as an example the polynomial spectral filter  $\mathcal{F} = \mathbf{L}$  of degree 1 ( $\alpha_{0,j} = 0, \alpha_{1,j} = 1$ ), we can compute the filter response according to the Eq. (1) and Eq. (4):

$$\mathcal{F}(\mathbf{y}(v_i)) = D_{ii}\mathbf{y}(v_i) - \sum_{j \in \mathcal{E}} A_{ij}\mathbf{y}(v_j), \quad (11)$$

at the vertex  $p_0$ , one can write in particular:

$$\begin{aligned} \mathcal{F}(\mathbf{y}(p_0)) = & 2(w_V + w_H)\mathbf{y}(p_0) - w_V(\mathbf{y}(p_2) + \mathbf{y}(p_4)) \\ & - w_H(\mathbf{y}(p_1) + \mathbf{y}(p_3)), \end{aligned} \quad (12)$$

where  $w_V, w_H$  are the weight of the ‘vertical’ and ‘horizontal’ edges of the graph. For the graph nodes representing points  $p_l : (\theta_l, \phi_l)$  and  $p_m : (\theta_m, \phi_m)$  we refer to edges as ‘vertical’ or ‘horizontal’ if  $\theta_l = \theta_m, \phi_l \neq \phi_m$  or  $\theta_l \neq \theta_m, \phi_l = \phi_m$  correspondingly.

We now calculate the filter response  $\mathcal{F}(\mathbf{y}(p_{0,i}))$  for a point  $p_0$  on the tangent plane  $\mathbf{T}_i$  and compare the result with Eq. (12). For simplicity, we assume that we shift the position of the tangent plane by an integer number of pixel positions on the spherical surface, namely  $\phi_i, \theta_i$  corresponds to a node of the graph given by the equirectangular image.

According to the gnomonic projection (Eq. (6)), the locations of  $p_{k,i}, k = [0, \dots, 4]$  defined on the surface of  $\mathbf{S}$  as  $(\phi_i, \theta_i), (\phi_i \pm \Delta\phi, \theta_i), (\phi_i, \theta_i \pm \Delta\theta)$  correspond to the following positions  $\mathbf{x}_{k,i} = (x_{k,i}, y_{k,i})$  on the tangent plane  $\mathbf{T}_i$ :

$$\begin{aligned} \mathbf{x}_{0,i} &= (0, 0) \\ \mathbf{x}_{2,i}, \mathbf{x}_{4,i} &= (0, \pm \tan \Delta\phi) \\ \mathbf{x}_{1,i}, \mathbf{x}_{3,i} &= \left( \pm \frac{\cos \phi_i \sin \Delta\theta}{\sin^2 \phi_i + \cos^2 \phi_i \cos \Delta\theta}, \frac{\sin \phi_i \cos \phi_i (1 - \cos \Delta\theta)}{\sin^2 \phi_i + \cos^2 \phi_i \cos \Delta\theta} \right) \end{aligned} \quad (13)$$

The tangent plane’s positions of points  $\mathbf{x}_{0,i}, \mathbf{x}_{2,i}, \mathbf{x}_{4,i}$  are independent of  $(\phi_i, \theta_i)$ , therefore their values remain the same as those of  $p_0, p_2$  and  $p_4$  respectively. However, the positions of points  $\mathbf{x}_{1,i}, \mathbf{x}_{3,i}$  depend on  $(\phi_i, \theta_i)$ , so that we need to interpolate the values of the pattern signal at the vertices  $p_{i,1}$  and  $p_{i,3}$  (see Fig. 4).

We can approximate the values at  $p_{i,1}$  and  $p_{i,3}$  using the bilinear interpolation method [41]. We denote by  $A, B, C, D$  the distances between the corresponding points

of the pattern  $\mathbf{T}_i$ , as shown in Fig. 4 (c). We can then express  $p_{i,1}$  and  $p_{i,3}$  as:

$$\begin{aligned}\mathbf{y}(p_{i,1}) &= E^{-1}(AD\mathbf{y}(p_1) + BD\mathbf{y}(p_0) + CB\mathbf{y}(p_2)), \\ \mathbf{y}(p_{i,3}) &= E^{-1}(AD\mathbf{y}(p_3) + BD\mathbf{y}(p_0) + CB\mathbf{y}(p_2)),\end{aligned}\quad (14)$$

where, using Eq (13),

$$\begin{aligned}E &= (C+D)(A+B), \\ A+B &= \tan \Delta\theta, \\ C+D &= \tan \Delta\phi\end{aligned}\quad . \quad (15)$$

Using Eq. (14) we can then write the expression for the filter response  $\mathcal{F}(\mathbf{y}(p_{0,i}))$ , as follows:

$$\begin{aligned}\mathcal{F}(\mathbf{y}(p_{0,i})) &= 2(w_{i,V} + w_{i,H})\mathbf{y}(p_0) \\ &\quad - w_{i,V}(\mathbf{y}(p_2) + \mathbf{y}(p_4)) \\ &\quad - w_{i,H}(\mathbf{y}(p_{i,1}) + \mathbf{y}(p_{i,3})),\end{aligned}\quad (16)$$

where  $w_{i,H}$  and  $w_{i,V}$  are the weights of the ‘horizontal’ and ‘vertical’ edges of the graph at points with the elevation  $\phi_i$ .

**Objective function.** We now want the filter responses  $p_0$  and  $p_{0,i}$  in (Eq. (12) and Eq. (16)) to be close to each other in order to build translation-invariant features. Therefore, we need to find weights  $w_H, w_V, w_{i,H}$  and  $w_{i,V}$  such that the following distance is minimized:

$$|\mathcal{F}(\mathbf{y}(p_{0,e})) - \mathcal{F}(\mathbf{y}(p_{0,i}))|. \quad (17)$$

Additionally, as we want to build a unique graph independently of the tangency point of  $\mathbf{T}_i$  and  $\mathbf{S}$ , we have additional constraint of  $w_V = w_{i,V}$ . The latter is important, as from Eq. (13) we can see that ‘vertical’ (or elevation) distances are not affected by translation of the tangent plane.

We assume that the camera has a good resolution, which leads to  $\Delta\theta \simeq 0$ . Therefore, based on Eq. (17), we can derive the following:

$$\begin{cases} w_H \simeq \left( \frac{\cos \phi_i \cos \Delta\theta}{\sin^2 \phi_i + \cos^2 \phi_i \cos \Delta\theta} \right) w_{i,H} = w_{i,H} \cos \phi_i, \\ \cos \Delta\theta \simeq 1. \end{cases} \quad (18)$$

Therefore, under our assumptions, we can conclude that the difference between filter responses, defined by Eq. (17) is minimized if the following condition is valid:

$$w_{i,H} = w_H (\cos \phi_i)^{-1}, \quad (19)$$

where  $w_H$  is the weight of the edge between points on the equator of the sphere  $\mathbf{S}$ .

Now, we can use this result to choose a proper function  $g(d_{ij})$  to define the weights  $w_{i,H}$  based on the Euclidean distances between two neighboring points  $(\mathbf{x}_i, \mathbf{x}_j)$  on the sphere  $\mathbf{S}$ . For the case when  $\phi_i = \phi_j = \phi_*$ ,  $\theta_i \neq \theta_j$  the Euclidean distance can be expressed as follows:

$$d_{ij}^2 = r^2(1 - \cos \Delta\theta)(1 + \cos 2\phi_*) = r^2 \cos^2 \phi_*. \quad (20)$$

For simplicity let us denote  $d_{ij} = d_{\phi_*}$ , where  $\phi_*$  is the elevation of the points  $\mathbf{x}_i, \mathbf{x}_j$ . Using these notations, we can compute the proportion between distances  $d_{\phi_e}$  and  $d_{\phi_*}$ , which are the distances between neighboring points at equator  $\phi_e = 0$  and elevations  $\phi_*$  respectively. It reads:

$$\frac{d_{\phi_*}}{d_{\phi_e}} = \frac{\cos \phi_*}{\cos \phi_e}. \quad (21)$$

Given Eq. (19), we can rewrite Eq. (21) for elevation  $\phi_* = \phi_i$  as:

$$\cos \phi_i = \frac{d_{\phi_i}}{d_{\phi_e}} = \frac{w_H}{w_{i,H}}. \quad (22)$$

As we can see, the distance between neighboring points on different elevation levels  $\phi_i$ , is proportional to  $\cos \phi_i$ . Given Eq. (17), we can see that making weights inversely proportional to Euclidean distance allows to minimize difference between filter responses. Therefore, we propose using  $w_{i,H}$  as:

$$w_{i,H} = \frac{1}{d_{\phi_i}}. \quad (23)$$

This formula can also be used to compute the weights for vertical edges, as the distance  $d$  between any pair neighboring points  $(\mathbf{x}_i, \mathbf{x}_j)$ , for which  $\theta_i = \theta_j$  and  $\phi_i \neq \phi_j$  is constant. This nicely fits with our assumption that the weights of ‘vertical’ edges should not depend on the tangency point of plane  $\mathbf{T}_i$  and sphere  $\mathbf{S}$ .

Thus, summing it up we choose the weights  $w_{ij}$  of a graph based on the Euclidean distance between pixels on spherical surface  $d_{ij}$  as follows:

$$w_{ij} = \frac{1}{d_{ij}}. \quad (24)$$

The graph representation finally forms the set of signals  $\mathbf{y}$  that are fed into the network architecture defined in Section 3.

## 5 Experiments

In this section we present our experiments. We first describe the datasets that we use for evaluation of our algorithm. We then compare our method to state-of-the-art algorithms.

We have used the following two datasets for the evaluation of our approach.

**MNIST-012** is based on a fraction of the popular MNIST dataset [42] that consists of 1100 images of size  $28 \times 28$ , subdivided in three different digit classes: ‘0’, ‘1’ and ‘2’. We then randomly split these images into training, validation and test sets of 600, 200 and 300 images respectively. In order to make this data suitable for our task we project them to the sphere at a point  $(\phi_i, \theta_i)$ , as depicted by Fig. 2. To evaluate accuracy with the change of  $(\phi_i, \theta_i)$ , for

each image we randomly sample from 9 different positions:  $\phi_i \in \{0, 1/8, 1/4\}$ ,  $\theta_i \in \{\pm 1/8, 0\}$ . Finally we compute equirectangular images (see Fig. 2) from these projections, as defined in Section 4.1 and use the resulting images to analyze the performance of our method.

**ETH-80** is a modified version of the dataset introduced in [43]. It comprises 3280 images of size  $128 \times 128$  that features 80 different objects from 8 classes, each seen from 41 different viewpoints. We further resize them to  $50 \times 50$  and randomly split these images into 2300 and 650 training and test images, respectively. We use the remaining 330 ones for validation. Finally, we follow the similar procedure to project them onto the sphere and create equirectangular images as we do for MNIST-012 dataset.

For our first set of experiments we train the network in [15] with the following parameters. We use two spectral convolutional layers with 10 and 20 filters correspondingly, with global pooling which selects  $P_1$  and  $P_2$  nodes, where the parameters  $P_1 = 2000$  and  $P_2 = 200$  for MNIST-012 dataset and  $P_1 = 2000$ ,  $P_2 = 700$  for ETH-80 dataset. We then use a statistical layer with  $12 \times 2$  statistics and three fully-connected layers with ReLU and 500, 300, 100 neurons correspondingly.

We have evaluated our approaches with respect to baseline methods in terms of classification accuracy. The **MNIST-012** dataset is then primarily used for the analysis of both the architecture and the graph construction approach. We then report the final comparisons to state-of-the-art approaches on the **ETH-80** dataset.

First of all, we visually show that feature maps on the last convolutional layer of our network are similar for different positions, namely for different tangent planes  $\mathbf{T}_i$  with the same object. Fig. 5 and 6 depict some feature maps of images from MNIST-012 and ETH-80 correspondingly.

The first column of each figure shows original equirectangular images of the same object projected to different elevations  $\phi_i = [0, 1/8, 1/4]$  and the rest visualize feature maps produced by two randomly selected filters. We can see, that the feature maps stay similar independently of the distortion of the corresponding input image. We believe that this, further, leads to closer feature representations, which is essential for good classification.

We recall that the goal of our new method is to construct a graph to process images from omnidirectional camera and use it to create similar feature vectors for the same object for different positions  $(\phi_i, \theta_i)$  of the tangent plane. To justify the advantage of proposed approach we design the following experiment. First of all, we randomly select three images of digits ‘2’, ‘1’ and ‘0’ from the test set of MNIST-012. We then project each of these images to 9 positions on the sphere  $\phi_i \in \{0, 1/8, 1/4\}$ ,  $\theta_i \in \{\pm 1/8, 0\}$ . We then evaluate Euclidean distances between the features that are given by the statistical layer of the network for all pairs of

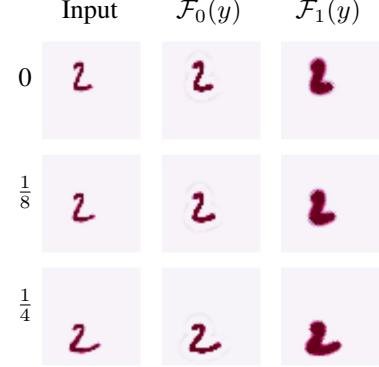


Figure 5. Example of the feature maps of the last spectral convolutional layer extracted from equirectangular images. The first column corresponds to the original images created for the same object, which is projected from different tangent planes  $\mathbf{T}_i$ , with  $\phi_i \in \{0, \frac{1}{8}, \frac{1}{4}\}$ ; the last two columns show the feature maps given by two randomly selected filters.

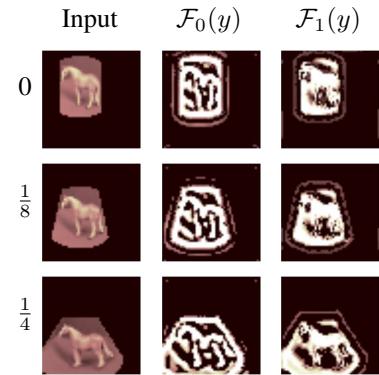


Figure 6. Example of feature maps of equirectangular images from **ETH-80** datasets. Here, we randomly select an input image from the test set and project it on three elevation  $\phi_i \in \{0, 1/8, 1/4\}$  and two spectral filters, which are named  $\mathcal{F}_1$  and  $\mathcal{F}_2$ . The figure illustrates resulting feature maps given by the selected filters (second and third columns) and input images (first column).

these 27 images. Fig. 7 presents the resulting  $[27 \times 27]$  matrix of this experiment for a grid graph and for the proposed graph representation, which captures the lens geometry. Ideally we expect that images with the same digit give the same feature vector regardless of different elevations  $\phi_i$  of the tangent plane. This essentially means that cells of the distance matrix should have low value on the  $[9 \times 9]$  diagonal sub-matrices, which correspond to the same object, and high values on the rest of the matrix elements. Fig. 7 shows that our method gives more similar features for the same object compared to the approach based on a grid graph. This suggests that building graph based on image geometry, as described in Section 4.2, makes features less sensitive to image distortions. This consequently simplifies the learning process of the algorithm.

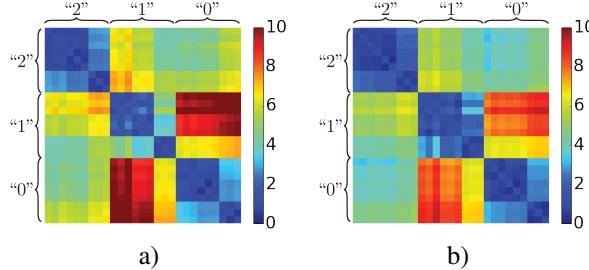


Figure 7. Illustration of the Euclidean distances between the features given by the networks of a) [15] and b) our geometry-aware graph. This figure depicts resulting matrix  $[27 \times 27]$  for the images from 3 classes and 9 different positions, where axes correspond to the image indexes. Each diagonal  $[9 \times 9]$  sub-matrix corresponds to the same object (digits “2”, “1”, “0”), the lowest value (blue) corresponds to the most similar features and the highest value (red) to the least similar (best seen in color).

method	graph type	# Parameters	<b>ETH-80</b>
Accuracy (%)			
<i>classic Deep Learning:</i>			
FC Nets	–	1.4M	71.3
STN [44]	–	1.1M	73.1
ConvNets [8]	–	1.1M	76.7
<i>graph-based DLA:</i>			
ChebNet [14]	grid	3.8M	72.9
TIGraNet [15]	grid	0.4M	74.2
ChebNet [14]	geometry	3.8M	78.6
Ours	geometry	0.4M	<b>80.7</b>

Table 1. Comparison to the state-of-the-art methods on the **ETH-80** datasets. We select the architecture of different methods to feature similar number of convolutional filters and neurons in the fully-connected layers.

We further evaluated our approach with respect to the state-of-the-art methods on ETH-80 dataset. The competing deep learning approaches can be divided in classical and graph-based methods. Among the former ones we use Fully-connected Networks (FCN), Convolutional Network (ConvNets) [8] and Spatial Transformer Networks (STN) [44]. STN has an additional to ConvNets layer which is able to learn specific transformation of a given input image. Among the graph-based methods, we choose ChebNet [14] and TIGraNet [15] for our experiments. ChebNet is a network designed based on Chebyshev polynomial filters. TIGraNet is a method invariant to isometric transformation of the input signal. The architectures are selected such that the number of parameters in convolutional and fully-connected layers roughly match each other across different techniques. More precisely, all networks have 2 convolutional layers with 10 and 20 filters, correspondingly, and 3 fully-connected layers with 300, 200 and 100 neurons. Filter size of the convolutional layer in classical ar-

chitectures is  $5 \times 5$ . For ChebNet we try polynomials of degree 5 and 10 and pick the latter one as it produces better results. For TIGraNet we use polynomial filters of degree 5. The results of this experiment are presented in Table 1.

Table 1 further shows that ConvNet [8] outperforms TIGraNet [15]. This likely happens as [15] gathers global statistics and loses the information about the location of the particular object. This information, however, is crucial for the network to adapt to different distortions on omnidirectional images. We can see that the introduced graph construction method helps to create similar feature representations for the same object at different elevations, which results into different distortion effects but similar feature response. Therefore, the object looks similar for the network and global statistics become more meaningful compared to a method based on the regular grid graph [15].

Further, we can see that the proposed graph construction method allows to improve accuracy of both graph-based algorithms: ChebNet-geometry outperforms ChebNet-grid, and proposed algorithm based on TIGraNet outperforms the same method on the grid-graph [15]. Finally, we also notice that our geometry-based method performs better than ChebNet-geometry on the ETH-80 task due to the isometric transformation invariant features; these are an advantage for the image classification problems, where images are captured from different viewpoints.

Thus, we can conclude that our algorithm produces similar filter responses for the same object at different positions. This, in combination with global graph-based statistics, leads to the better classification accuracy.

## 6 Conclusion

In this paper we propose a novel image classification method based on deep neural network that is specifically designed for omnidirectional cameras, which introduce severe geometric distortion effects. Our graph construction method allows learning filters that respond similarly to the same object seen at different elevations on the equirectangular image. We evaluated our method on challenging datasets and prove its effectiveness in comparison to state-of-the-art approaches that are agnostic to the geometry of the images.

Our discussion in this paper was limited to specific type of the mapping projection. However, the proposed solution has a potential to be extend to more general geometries of the camera lenses.

## 7 Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the GPU card used for this research.

## References

- [1] D. Scaramuzza, *Omnidirectional Vision: From Calibration to Robot Motion Estimation*. ETH, 2008. [1](#)
- [2] M. Blösch, S. Weiss, D. Scaramuzza, and R. Siegwart, “Vision based MAV navigation in unknown and unstructured environments,” in *International Conference on Robotics and Automation*, pp. 21–28, 2010. [1](#)
- [3] P. Hansen, P. Corke, W. W. Boles, and K. Daniilidis, “Scale invariant feature matching with wide angle images,” in *International Conference on Intelligent Robots and Systems*, pp. 1689–1694, 2007. [1, 2](#)
- [4] J. Masci, D. Migliore, M. M. Bronstein, and J. Schmidhuber, “Descriptor Learning for Omnidirectional Image Matching,” in *Registration and Recognition in Images and Videos*, pp. 49–62, 2014. [1, 2](#)
- [5] D. Scaramuzza, A. Martinelli, and R. Siegwart, “A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion,” in *International Conference on Computer Vision Systems*, p. 45, 2006. [1](#)
- [6] C. Hughes, P. Denny, E. Jones, and M. Glavin, “Accuracy of fish-eye lens models,” *Applied Optics*, vol. 49, no. 17, p. 3338, 2010. [1](#)
- [7] A. W. Fitzgibbon, “Simultaneous linear estimation of multiple view geometry and lens distortion,” in *Conference on Computer Vision and Pattern Recognition*, pp. 125–132, 2001. [1](#)
- [8] Y. Boureau, J. Ponce, and Y. LeCun, “A Theoretical Analysis of Feature Pooling in Visual Recognition,” in *International Conference on Machine Learning*, pp. 111–118, 2010. [1, 8](#)
- [9] Y. LeCun and Y. Bengio, “The handbook of brain theory and neural networks,” ch. Convolutional Networks for Images, Speech, and Time Series, pp. 255–258, MIT Press, 1998. [1](#)
- [10] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, “Going deeper with convolutions,” in *Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015. [1](#)
- [11] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” *arXiv Preprint*, vol. abs/1409.1556, 2014. [1](#)
- [12] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka, “3D Bounding Box Estimation Using Deep Learning and Geometry,” *arXiv Preprint*, vol. abs/1612.00496, 2016. [1](#)
- [13] D. Boscaini, J. Masci, S. Melzi, M. M. Bronstein, U. Castellani, and P. Vandergheynst, “Learning class-specific descriptors for deformable shapes using localized spectral convolutional networks,” *Computer Graphics Forum*, vol. 34, no. 5, pp. 13–23, 2015. [1](#)
- [14] M. Defferrard, X. Bresson, and P. Vandergheynst, “Convolutional Neural Networks on Graphs with Fast Localized Spectral Filtering,” in *Advances in Neural Information Processing Systems*, pp. 3837–3845, 2016. [1, 3, 8](#)
- [15] R. Khasanova and P. Frossard, “Graph-based Isometry Invariant Representation Learning,” in *International Conference on Machine Learning*, 2017. [1, 2, 3, 4, 7, 8](#)
- [16] L. Posada, K. K. Narayanan, F. Hoffmann, and T. Bertram, “Semantic classification of scenes and places with omnidirectional vision,” in *European Conference on Mobile Robots*, pp. 113–118, 2013. [2](#)
- [17] T. Goedemé, T. Tuytelaars, L. V. Gool, G. Vanacker, and M. Nuttin, “Omnidirectional sparse visual path following with occlusion-robust feature tracking,” in *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras*, 2005. [2, 3](#)
- [18] D. Scaramuzza and R. Siegwart, “Appearance-Guided Monocular Omnidirectional Visual Odometry for Outdoor Ground Vehicles,” *Transactions on Robotics*, vol. 24, no. 5, pp. 1015–1026, 2008. [2, 3](#)
- [19] L. B. Marinho, J. S. Almeida, J. W. M. Souza, V. H. C. de Albuquerque, and P. P. R. Filho, “A novel mobile robot localization approach based on topological maps using classification with reject option in omnidirectional images,” *Expert Systems with Applications*, vol. 72, pp. 1–17, 2017. [2](#)
- [20] I. Tasic and P. Frossard, “Spherical Imaging in Omnidirectional Camera Networks,” in *Multi-Camera Networks: Concepts and Applications*, Elsevier, 2009. [2, 3](#)
- [21] C. Geyer and K. Daniilidis, “Catadioptric Projective Geometry,” *International Journal of Computer Vision*, vol. 45, no. 3, pp. 223–243, 2001. [2](#)
- [22] D. G. Lowe, “Object Recognition from Local Scale-Invariant Features,” in *International Conference on Computer Vision*, pp. 1150–1157, 1999. [2](#)
- [23] A. Torralba, “Contextual Priming for Object Detection,” *International Journal of Computer Vision*, vol. 53, no. 2, pp. 169–191, 2003. [2](#)
- [24] A. Rituerto, A. C. Murillo, and J. J. Guerrero, “Semantic labeling for indoor topological mapping using a wearable catadioptric system,” *Robotics and Autonomous Systems*, vol. 62, no. 5, pp. 685–695, 2014. [2](#)
- [25] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, “LDAHash: Improved Matching with Smaller Descriptors,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 1, pp. 66–78, 2012. [2](#)
- [26] R. Hadsell, S. Chopra, and Y. LeCun, “Dimensionality Reduction by Learning an Invariant Mapping,” in *Conference on Computer Vision and Pattern Recognition*, pp. 1735–1742, 2006. [2](#)
- [27] J. Cruz-Mota, I. Bogdanova, B. Paquier, M. Bierlaire, and J. Thiran, “Scale Invariant Feature Transform on the Sphere: Theory and Applications,” *International Journal of Computer Vision*, vol. 98, no. 2, pp. 217–241, 2012. [2](#)
- [28] H. Hadj-Abdelkader, E. Malis, and P. Rives, “Spherical image processing for accurate visual odometry with omnidirectional cameras,” *Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras*, 2008. [2](#)
- [29] C. Harris and M. Stephens, “A Combined Corner and Edge Detector,” in *Alvey Vision Conference*, pp. 1–6, 1988. [2](#)

[30] I. Bogdanova, X. Bresson, J. Thiran, and P. Vandergheynst, “Scale Space Analysis and Active Contours for Omnidirectional Images,” *IEEE Transactions on Image Processing*, vol. 16, no. 7, pp. 1888–1901, 2007. [2](#)

[31] Z. Arican and P. Frossard, “Dense disparity estimation from omnidirectional images,” in *International Conference on Advanced Video and Signal Based Surveillance*, pp. 399–404, 2007. [2](#)

[32] I. Totic, I. Bogdanova, P. Frossard, and P. Vandergheynst, “Multiresolution motion estimation for omnidirectional images,” in *European Signal Processing Conference*, pp. 1–4, 2005. [3](#)

[33] F. De Simone, P. Frossard, P. Wilkins, N. Birkbeck, and A. C. Kokaram, “Geometry-driven quantization for omnidirectional image coding,” in *Picture Coding Symposium*, pp. 1–5, 2016. [3](#)

[34] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, “Geometric deep learning: going beyond Euclidean data,” *Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017. [3](#)

[35] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, “Spectral Networks and Locally Connected Networks on Graphs,” *arXiv Preprint*, vol. abs/1312.6203, 2013. [3](#)

[36] M. Henaff, J. Bruna, and Y. LeCun, “Deep Convolutional Networks on Graph-Structured Data,” *arXiv Preprint*, vol. abs/1506.05163, 2015. [3](#)

[37] T. N. Kipf and M. Welling, “Semi-Supervised Classification with Graph Convolutional Networks,” *arXiv Preprint*, vol. abs/1609.02907, 2016. [3](#)

[38] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains,” *Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013. [3](#), [4](#)

[39] F. Pearson, *Map Projections Theory and Applications*. Taylor & Francis, 1990. [4](#)

[40] K. Miyamoto, “Fish Eye Lens,” *Journal of the Optical Society of America*, vol. 54, no. 8, p. 1060, 1964. [4](#)

[41] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, 3 ed., 2007. [5](#)

[42] Y. LeCun and C. Cortes, “MNIST handwritten digit database,” 2010. [6](#)

[43] B. Leibe and B. Schiele, “Analyzing Appearance and Contour Based Methods for Object Categorization,” in *Conference on Computer Vision and Pattern Recognition*, pp. 409–415, 2003. [7](#)

[44] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, “Spatial Transformer Networks,” in *Advances in Neural Information Processing Systems*, pp. 2017–2025, 2015. [8](#)