Warped Convolutions: Efficient Invariance to Spatial Transformations

João F. Henriques [1] **Andrea Vedaldi** [1]

Abstract

Convolutional Neural Networks (CNNs) are extremely efficient, since they exploit the inherent translation-invariance of natural images. However, translation is just one of a myriad of useful spatial transformations. Can the same efficiency be attained when considering other spatial invariances? Such generalized convolutions have been considered in the past, but at a high computational cost. We present a construction that is simple and exact, yet has the same computational complexity that standard convolutions enjoy. It consists of a constant image warp followed by a simple convolution, which are standard blocks in deep learning toolboxes. With a carefully crafted warp, the resulting architecture can be made equivariant to a wide range of two-parameter spatial transformations. We show encouraging results in realistic scenarios, including the estimation of vehicle poses in the Google Earth dataset (rotation and scale), and face poses in Annotated Facial Landmarks in the Wild (3D rotations under perspective).

1. Introduction

A crucial aspect of current deep learning architectures is the encoding of invariances. This fact is epitomized in the success of convolutional neural networks (CNN), where *equivariance to image translation* is key: translating the input results in a translated output. When invariances are present in the data, encoding them explicitly in an architecture provides an important source of regularization, which reduces the amount of training data required for learning.

Invariances may also be used to improve the efficiency of implementations. For instance, a convolutional layer requires orders of magnitude less memory (by reusing filters across space) and less computation (due to their limited support) compared to a fully-connected layer. Its local and predictable memory access pattern also makes better use of modern hardware's caching mechanisms.

The success of CNNs indicates that translation invariance is an important property of images. However, this does not *explain why* translation equivariant operators work well for image understanding. The common interpretation is that such operators are matched to the statistics of natural images, which are well known to be translation invariant (Hyvärinen et al., 2009). However, natural image statistics are also (largely) invariant to other transformations such as isotropic scaling and rotation, which suggests that alternative neural network designs may also work well with images. Furthermore, in specific applications, invariances other than translation may be more appropriate.

Therefore, it is natural to consider generalizing convolutional architectures to other image transformations, and this has been the subject of extensive study (Kanazawa et al., 2014; Bruna et al., 2013; Cohen & Welling, 2016). Unfortunately these approaches do not possess the same memory and speed benefits that CNNs enjoy. The reason is that, ultimately, they have to transform (warp) an image or filter several times (Kanazawa et al., 2014; Marcos et al., 2016; Dieleman et al., 2015), incurring a high computational burden. Another approach is to consider a basis of filters (analogous to eigen-images) encoding the desired invariance (Cohen & Welling, 2014; Bruna et al., 2013; Cohen & Welling, 2016). Although they are able to handle transformations with many pose parameters, in practice most recent proposals are limited to very coarsely discretized transformations, such as horizontal/vertical flips and 90° rotations (Dieleman et al., 2015; Cohen & Welling, 2014).

In this work we consider generalizations of CNNs that overcome these disadvantages. Well known constructions in group theory enable the extension of convolution to general transformation groups (Folland, 1995). However, this generality usually comes at an increased computational cost or complexity. Here we show that, by making appropriate assumptions, we can design convolution operators that are equivariant to a large class of two-parameter transformations while reducing to a standard convolution in a warped image space. The fixed image warp can be

[1]Visual Geometry Group, University of Oxford, United Kingdom. Correspondence to: João F. Henriques <joao@robots.ox.ac.uk>.

Proceedings of the 34^{th} International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017.

implemented using bilinear resampling, a simple and fast operation that has been popularized by spatial transformer networks (Jaderberg et al., 2015; Heckbert, 1989) and is part of most deep learning toolboxes. Unlike previous proposals, the proposed *warped convolutions* can handle continuous transformations, such as fine rotation and scaling.

This makes generalized convolution easily implementable in neural networks, reusing fast convolution algorithms on GPU hardware, such as Winograd (Lavin, 2015) or the Fast Fourier Transform (Lyons, 2010).

2. Generalizing convolution

2.1. Discrete and continuous convolution

We start by looking at the basic building block of CNNs, i.e. the convolution operator. This operator computes the inner product of an image $I \in \mathbb{R}^{N\times N}$ with a translated version of the filter $F \in \mathbb{R}^{M\times M}$, producing a new image as output:

$$(I * F)(j) = \sum_k I(k)F(j-k), \tag{1}$$

where k, $j \in \mathbb{Z}^2$ are two-dimensional vectors of indexes, and the summation ranges inside the extents of both arrays. Over the next sections it will be more convenient to translate the image I instead of the filter F. This alternative form of eq. (1) is obtained by the substitution $k \leftarrow j + k$:

$$(I * F)(j) = \sum_k I(j+k)F(-k) \tag{2}$$

In the neural network literature, this is often written using the *cross-correlation convention* (Goodfellow et al., 2016), by considering the reflected filter $F_-(j) = F(-j)$:

$$(I * F)(j) = \sum_k I(j+k)F_-(k). \tag{3}$$

To handle continuous deformations of the input, it is more natural to express eq. (2) as an integral over continuous rather than discrete inputs:

$$(I * F)(y) = \int I(y+x)\,F(-x)\,dx, \tag{4}$$

where I, $F : \mathbb{R}^2 \to \mathbb{R}$ are functions of continuous inputs in $\mathbb{R}^2$. The real-valued 2D vectors $x, y \in \mathbb{R}^2$ now play the role of the indexes $k \in \mathbb{Z}^2$. Equation (4) reduces to the discrete case of eq. (1) if we define I and F as the sum of delta functions on grids (Dirac comb). Intermediate values can be obtained by interpolation, such as bilinear (which amounts to convolution of the delta functions with a triangle filter (Jaderberg et al., 2015)). Importantly, such continuous images can be deformed by a rich set of continuous transformations of the input coordinates, whereas strictly discrete operations would be more limiting.

2.2. Convolution on groups

The standard convolution operator of eq. (4) can be interpreted as applying the filter to translated versions of the image. We wish to replace translations with other image transformations g, belonging to a group G. In the context of machine learning models for images, this generalized (group) convolution can be understood to exhaustively search for a pattern at various poses $g \in G$ (e.g. rotation angles or scale factors) (Dieleman et al., 2015; Kanazawa et al., 2014).

Following standard derivations (Folland, 1995), the most common way of generalising convolution to transformations other than translations starts from the Haar integral. Given a measure μ over a group G, one can define the integral of a real function $\tilde{I} : G \to \mathbb{R}$, written:

$$\int_G \tilde{I}(g)\,d\mu(g).$$

The (left) Haar measure is the most natural choice for μ; it is the only measure (up to scaling factors) that is (left) invariant to group translation. In other words, μ satisfies the equation:

$$\forall h \in G: \quad \int_G \tilde{I}(hg)\,d\mu(g) = \int_G \tilde{I}(g)\,d\mu(g).$$

Mirroring eq. (4), the *group convolution* of two functions $\tilde{I}$ and $\tilde{F}$ is defined as[1]

$$(\tilde{I} \underset{G}{*} \tilde{F})(t) = \int \tilde{I}(tg)\tilde{F}(g^{-1})\,d\mu(g). \tag{5}$$

From the viewpoint of statistical learning, a key property of convolution is *equivariance*. Consider the (left) translation operator

$$L_h(\tilde{I}): \quad t \longmapsto \tilde{I}(h^{-1}t). \tag{6}$$

Then:

Lemma 1. *(Folland, 1995) Convolution is equivariant with group translations, in the sense that L_h commutes with $\underset{G}{*}$:*

$$L_h(\tilde{I} \underset{G}{*} \tilde{F})(t) = (L_h(\tilde{I}) \underset{G}{*} \tilde{F})(t).$$

[1] Due to the Haar invariance property, this definition is equivalent to the following one, also commonly found in the literature:

$$(\tilde{I} \underset{G}{*} \tilde{F})(t) = \int \tilde{I}(g)\tilde{F}(g^{-1}t)\,d\mu(g).$$

The equivalence mirrors the one between eq. (1) and eq. (2), and can be easily proved:

$$\begin{aligned}(\tilde{I} \underset{G}{*} \tilde{F})(t) &= \int \tilde{I}(tg)\tilde{F}((tg)^{-1}t)\,d\mu(g) \\ &= \int \tilde{I}(tg)\tilde{F}(g^{-1})\,d\mu(g).\end{aligned}$$

Proof. $L_h(\tilde{I} \underset{G}{*} \tilde{F})(t) = \int \tilde{I}((h^{-1}t)g)\tilde{F}(g^{-1})\,d\mu(g) = \int \tilde{I}(h^{-1}(tg))\tilde{F}(g^{-1})\,d\mu(g) = (L_h(\tilde{I}) \underset{G}{*} \tilde{F})(t)$. □

2.3. From groups to images

The functions $\tilde{I}$ and $\tilde{F}$ above have been defined on groups. In applications, however, we are interested in *images*, i.e. functions $I : \Omega \to \mathbb{R}$ defined on a subset Ω of the real plane $\mathbb{R}^2$.

The connection between the two function types is easy to establish. The assumption is that G acts[2] on the image domain Ω (i.e. G is a group of transformations of Ω). We can then define the $\tilde{\cdot}$ operator as:

$$\tilde{I}(g) = I(gx_0)$$

where $x_0 \in \Omega$ is an arbitrary pivot point. Note that the values of $\tilde{I}$ depend only on the values of I on the orbit of x_0, i.e. the set $Gx_0 = \{gx_0 : g \in G\}$. Therefore we typically set the domain Ω of I to be equal to Gx_0.

By this definition, left translation of $\tilde{I}$ by h corresponds to warping the image I by the transformation h:

$$L_h(\tilde{I})(t) = I((h^{-1}t)x_0) = I(h^{-1}(tx_0)) = \widetilde{(I \circ h^{-1})}(t).$$

We can then update eq. (5) to express group convolution as a function of images on the real plane:

$$(I \underset{G}{*} F)(t) = \int I(tgx_0)F(g^{-1}x_0)\,d\mu(g). \tag{7}$$

2.4. Standard convolutions with exponential maps

In the definitions of sections 2.2 and 2.3, while we could reduce the functions $\tilde{I}$ and $\tilde{F}$ to images I and F, the result of convolution is still a function defined over a group. One needs therefore to understand how to represent such functions and calculate the corresponding integrals.

We note here that the simplest case is when G is a Lie group parameterised by the exponential map $\exp : V \to G$, where V is a subset of $\mathbb{R}^P$, in such a way that $\exp$ is smooth, bijective and additive ($\exp(u+v) = \exp(u)\exp(v)$). Then:

$$\int_G \tilde{I}(g)\,d\mu(g) = \int_V \tilde{I}(\exp(u))\,du.$$

Hence we can define the warped image

$$I_{\mathrm{w}}(u) = \tilde{I}(\exp(u)) = I(\exp(u)x_0), \quad u \in V.$$

[2]This means that g defines a mapping $\Omega \mapsto \Omega$ and that the group multiplication hg corresponds to function composition $(hg)x = h(gx)$.

and group convolution reduces to the standard notion of convolution $\underset{V}{*}$ on V:

$$(\tilde{I} \underset{G}{*} \tilde{F})(\exp(v)) = \int_V I_{\mathrm{w}}(v+u)F_{\mathrm{w}}(-u)\,du. \tag{8}$$

We refer to this standard convolution in warped space (eq. (8)) as *warped convolution*.

Discussion. Note that the result is an image whose dimensionality P is that of the vector space V; in the following, we mainly work in the case $P = 2$. By far, the strongest requirement is that the map is additive: this is the same as requiring the transformation group G to be Abelian, in the sense that transformations commute ($hg = gh$). In section 5 we will show a variety of useful image transformations that respect this property. The advantage of introducing this restriction is that calculations simplify tremendously, ultimately enabling a simple and efficient implementation of the operator as discussed below.

3. Warped convolutions

Our main contribution is to note that certain group convolutions can be implemented efficiently by a *standard* convolution, by pre-warping the input image and filter appropriately. The warp is the same for any image, depending solely on the nature of the relevant transformations, and can be written in closed form. This result allows one to implement very efficient group convolutions using simple computational blocks, as shown in section 3.2.

3.1. Warped convolution layer

We can now reinterpret these results in terms of a new neural network convolutional layer. The input of the layer is an image I and the learnable parameters are the coefficients of the filter F_{w}. The output is a new "image" $C(I; F_{\mathrm{w}})(v)$ defined on the vector space V. This image is obtained by first warping I using the exponential map and then by convolving the result with F_{w} in the standard sense:

$$C(I; F_{\mathrm{w}})(v) = (I(\exp(\cdot)x_0) \underset{V}{*} F_{\mathrm{w}})(v). \tag{9}$$

The most important property of this layer is equivariance: if we warp the image by the transformation $h = \exp(u)$, then the convolution result translates by u:

$$C(I \circ h^{-1}; F_{\mathrm{w}}) = C(I; F_{\mathrm{w}})(v-u), \quad h = \exp(u).$$

Note that the output action equivalent to warping the input is simply to translate the result, as for standard convolution. The second most important property is that this operator can be implemented efficiently as the combination of warping and standard convolution.

3.2. Implementation and intuition

The warp (exponential map) that is applied to the input image eq. (9) can be implemented as follows. We start with an arbitrary pivot point x_0 in the image, and then sample all possible transformations of that point, $\{gx_0 : g \in G\}$. For discrete images, G will be implemented as a 2D grid of discrete transformations (e.g. rotations and scales at regular intervals), and $\{gx_0\}$ will be a 2D grid as well, referred to as the *warp grid*. Finally, sampling the input image at the points $\{gx_0\}$ (for example, by bilinear interpolation) yields the warped image.

An illustration is given in fig. 1, for various transformations (each one is discussed in more detail in section 5). The red dot shows the pivot point x_0, and the two arrows pointing away from it show the original axes of the sampled grid of parameters. The grids were generated by sampling the transformation parameters at regular intervals. Note that the warp grids are independent of the image contents – they can be computed once offline and then applied to any image.

The steps for implementing a warped convolution block are outlined in algorithm 1. The main advantage of implementing group convolution as warped convolution is that it replaces a large number of warping operations (one per group element) with a single warp.

4. Discussion

4.1. Interpretation as a filtering operator in image space

When $V \subset \mathbb{R}^2$, we can often interpret group convolution as an integration over image space, instead of over group elements. For this, we introduce the map $\eta(v) = \exp(v)x_0$ and assume that it is smooth, invertible, and surjective on the image domain Ω. Surjectivity means that G acts *transitively* on Ω, in the sense that every point $x \in \Omega$ can be reached from x_0 by a transformation g. Injectivity means that this transformation is unique.

Next, let $h = \exp(v)$ and note that $\exp(v+u) = \exp(v)\exp(u) = h\exp(u)$. Hence

$$(\tilde{I} \underset{G}{*} \tilde{F})(h) = \int_V I_{\mathrm{w}}(v+u)F_{\mathrm{w}}(-u)\,du = \int_V I(h\exp(u)x_0)F(\exp(-u)x_0)\,du.$$

Let the filter F_- be the reflection[3] of F around the pivot point x_0 by the group G, i.e. $F_-(gx_0) = F(g^{-1}x_0)$. It follows that:

$$(\tilde{I} \underset{G}{*} \tilde{F})(h) = \int_V I(h\exp(u)x_0)F_-(\exp(u)x_0)\,du.$$

We can now use the change of variable $u \leftarrow \eta^{-1}(x)$ to write

$$(\tilde{I} \underset{G}{*} \tilde{F})(h) = \int_V I(h\eta(u))F_-(\eta(u))\,du \tag{10}$$

$$= \int_\Omega I(hx)F_-(x)\left|\frac{d\eta^{-1}}{dx}\right|dx. \tag{11}$$

Thus we see that the group convolution amounts to applying a certain filter F_- to the warped image $I \circ h$. The filter elements are reweighed by the determinant of the Jacobian of η^{-1}, which accounts for the stretching and shrinking of space due to the non-linear map. In practice, both the reflection and Jacobian can be absorbed into a learned filter, making such calculations unnecessary. Nevertheless, they offer a complementary view of warped convolutions.

Algorithm 1 Warped Convolution

Grid generation (offline)

- Compute the 2D warp grid $w = g_u(x_0)$ by applying a two-parameter spatial transformation $g : \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^2$ to a single pivot point x_0, using a 2D grid of parameters $u = \{(u_1 + i\delta_1, u_2 + j\delta_2) : i = 0,\ldots,m, j = 0,\ldots,n\}$.

Warped convolution

1. Resample input image I using the warp grid w, by bilinear interpolation.
2. Convolve warped image I_{w} with a learned filter F_{w}.

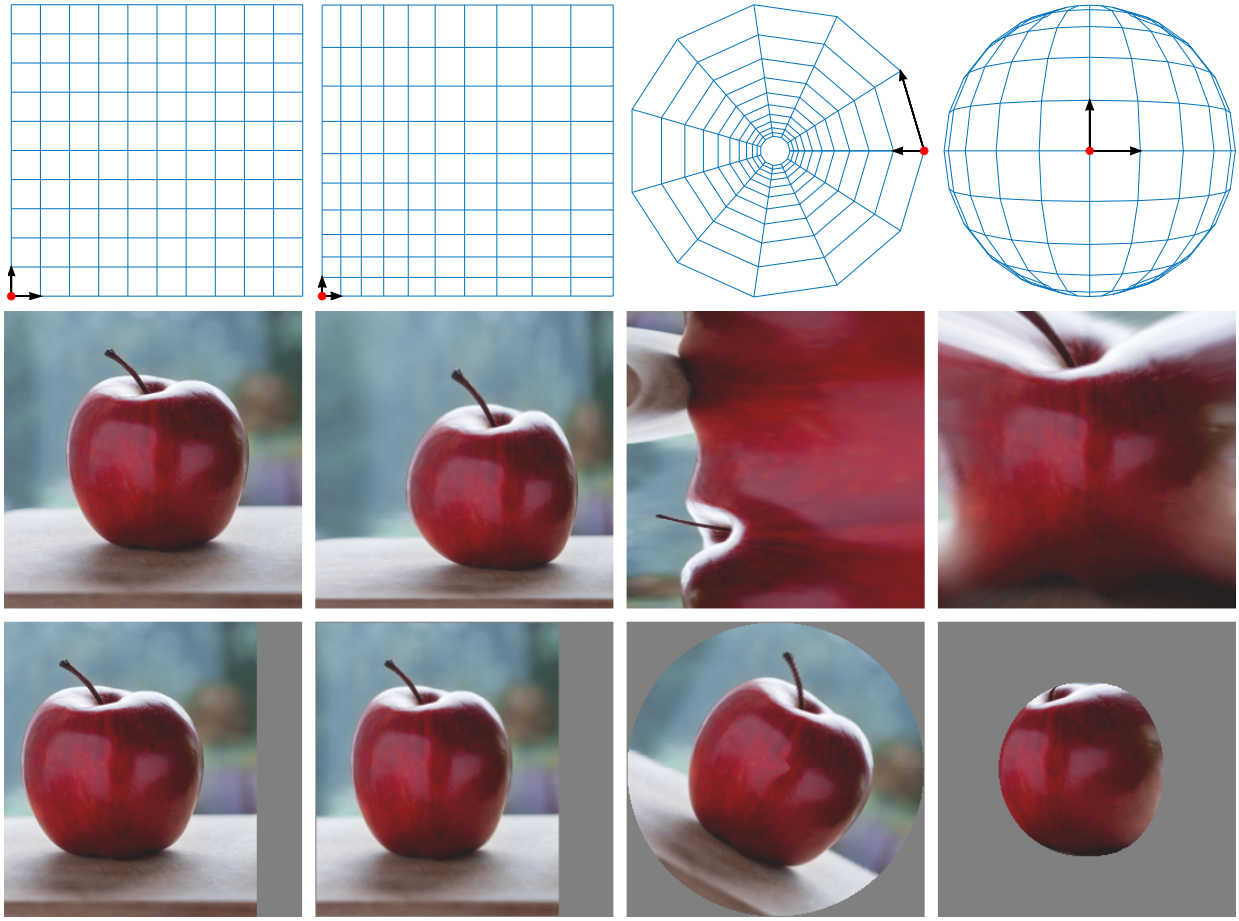
By eq. (8), and for appropriate transformations, these steps are equivalent to group convolution (which performs an exhaustive search across the pose-space of transformations), but at a much lower computational cost.

4.2. Efficiency vs. generality

By reducing to standard convolution, warped convolution allows one to take full advantage of modern convolution implementations (Lavin, 2015; Lyons, 2010), including those with lower computational complexity (e.g. FFT (Lyons, 2010)). However, while warped convolution works with an important class of transformations (including the ones considered in previous works Kanazawa et al. (2014); Cohen & Welling (2014); Marcos et al. (2016)), non-trivial restrictions are imposed on the transformation group: it must be Abelian and have only two parameters.

By contrast, the group-theoretic convolution operator of eq. (5) does not make (almost) any restriction on the

[3] This is well defined because η is invertible. In fact, if $x = gx_0 = \exp(u)x_0 = \eta(u)$, then $u = \eta^{-1}(x)$ and $F_-(x) = F(\eta(-\eta^{-1}(x)))$.



(a) translation (b) scale/aspect ratio (c) scale/rotation (d) 3D rotation (yaw/pitch)

Figure 1. First row: Sampling grids that define the warps associated with different spatial transformations. Second row: An example image (a) after warping with each grid (b-d). Third row: A small translation is applied to each warped image, which is then mapped back to the original space (by an inverse warp). Translation in one axis of the appropriate warped space is equivalent to (b) horizontal scaling; (c) planar rotation; (d) 3D rotation around the vertical axis.

transformation group. Unfortunately, it is in general significantly more difficult to compute efficiently than the special case we consider here. To understand some of the implementation challenges, consider specializing eq. (7) to a discrete group G such as a discrete set of planar rotations. In this case the Haar measure is trivial and equal to 1, and one has:

$$(I \underset{G}{*} F)(t) = \sum_{g \in G} I(tgx_0) F(g^{-1}x_0).$$

Direct computation of this equation has complexity $\mathcal{O}(|G|^2)$ where $|G|$ is the cardinality of the discrete group. Assuming that $|G|$ is in the order of $\mathcal{O}(N^2)$ where N is the resolution of the input image (as it would be for standard convolution), one would obtain a complexity of $\mathcal{O}(N^4)$. In practice, since usually the support of a $M \times M$ filter is much smaller than the image, this complexity might reduce to $\mathcal{O}(N^2 M^2)$, which is the complexity for the spatial domain implementation of convolution; however, compared to the standard case, this has two major disadvantages. First, the image is sampled in a spatially-varying manner, using bilinear or other interpolation, which foregoes the benefit of the regular, predictable, and local pattern of computations in standard convolution. This makes high-performance implementation of the naive algorithm difficult, particularly on GPUs. Secondly, it precludes the use of *faster* convolution routines such as Winograd's algorithm (Lavin, 2015) or the Fast Fourier Transform (Lyons, 2010), the later having lower computational complexity than exhaustive search ($\mathcal{O}(N^2 \log N)$). The development of analogues of the FFT for other general groups remains the subject of active research (Tygert, 2010; Li & Yang, 2016), which we sidestep by reusing highly optimized standard convolutions.

In practice, most recent works focus on very coarse transformations that do not change the filter support and can

be implemented strictly via permutations, like horizontal/vertical flips and 90° rotations (Dieleman et al., 2015; Cohen & Welling, 2014). Such difficulties may explain why group convolutions are not as widespread as CNNs.

5. Examples of spatial transformations

We now give some concrete examples of two-parameter spatial transformations that obey the conditions of section 2.4, and can be useful in practice.

5.1. Scale and aspect ratio

Visual object detection tasks require predicting the extent of an object as a bounding box. While the location can be found accurately by a standard CNN, which is equivariant to translation, the size prediction could similarly benefit from equivariance to horizontal and vertical scale (equivalently, scale and aspect ratio).

Such a spatial transformation, from which a warp can be constructed, is given by:

$$g_u(x) = \begin{bmatrix} x_1 s^{u_1} \\ x_2 s^{u_2} \end{bmatrix} \tag{12}$$

The s constant controls the total degree of scaling applied. Notice that the output must be exponential in the scale parameters u; this ensures the additive structure of the exponential map ($\exp(v+u) = \exp(v)\exp(u)$). The resulting warp grid can be visualized in fig. 1-b. In this case, the domain of the image must be $\Omega \in \mathbb{R}^2_+$, since a pivot x_0 in one quadrant cannot reach another quadrant by any amount of (positive) scaling.

5.2. Scale and rotation (log-polar warp)

Planar scale and rotation are perhaps the most obvious spatial transformations in images, and are a natural test case for works on spatial transformations (Kanazawa et al., 2014; Marcos et al., 2016). Rotating a point x by u_1 radians and scaling it by u_2, around the origin, can be performed with

$$g_u(x) = \begin{bmatrix} s^{u_2} \|x\| \cos(\text{atan}_2(x_2, x_1) + u_1) \\ s^{u_2} \|x\| \sin(\text{atan}_2(x_2, x_1) + u_1) \end{bmatrix}, \tag{13}$$

where atan_2 is the standard 4-quadrant inverse tangent function (`atan2`). The domain in this case must exclude the origin ($\Omega \in \mathbb{R}^2 \setminus \{0\}$), since a pivot $x_0 = 0$ cannot reach any other points in the image by rotation or scaling.

The resulting warp grid can be visualized in fig. 1-c. It is interesting to observe that it corresponds exactly to the log-polar domain, which is used in the signal processing literature to perform correlation across scale and rotation (Tzimiropoulos et al., 2010; Reddy & Chatterji, 1996). In fact, it was the source of inspiration for this work, which can be seen as a generalization of the log-polar domain to other spatial transformations.

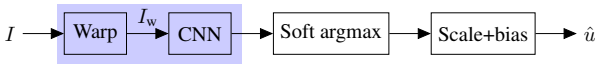


Figure 2. Equivariant pose estimation strategy used in the experiments (section 6). With an appropriate warp and a standard CNN, the shaded block becomes equivalent to a group-equivariant CNN, which performs exhaustive searches across pose-space instead of image-space.

5.3. 3D sphere rotation under perspective

We will now tackle a more difficult spatial transformation, in an attempt to demonstrate the generality of our result. The transformations we will consider are yaw and pitch rotations in 3D space, as seen by a perspective camera. In the experiments (section 6) we will show how to apply it to face pose estimation.

In order to maintain additivity, the rotated 3D points must remain on the surface of a sphere. We consider a simplified camera and world model, whose only hyperparameters are a focal length f, the radius of a sphere r, and its distance from the camera center d. The equations for the spatial transformation corresponding to yaw and pitch rotation under this model are in appendix A.

The corresponding warp grid can be seen in fig. 1-d. It can be observed that the grid corresponds to what we would expect of a 3D rendering of a sphere with a discrete mesh. An intuitive picture of the effect of the warp grid in such cases is that it wraps the 2D image around the surface of the 3D object, so that translation in the warped space corresponds to moving between vertexes of the 3D geometry.

6. Experiments

6.1. Architecture

As mentioned in section 2.2, group convolution can perform an exhaustive search for patterns across spatial transformations, by varying pose parameters. For tasks where invariance to that transformation is important, it is usual to pool the detection responses across all poses (Marcos et al., 2016; Kanazawa et al., 2014).

In the experiments, however, we will test the framework in pose prediction tasks. As such, we do not want to pool the detection responses (e.g. with a max operation) but rather find the pose with the strongest response (i.e., an argmax operation). To perform this operation in a differentiable

manner, we implement a soft argmax operation, defined as follows:

$$s_1(a) = \sum_{ij}^{mn} \frac{i}{m}\sigma_{ij}(a), \qquad s_2(a) = \sum_{ij}^{mn} \frac{j}{n}\sigma_{ij}(a), \quad (14)$$

where $\sigma(a) \in \mathbb{R}^{m\times n}$ is the softmax over all spatial locations, and $\sigma_{ij}(a)$ indexes the element at (i, j). The outputs are the two spatial coordinates of the maximum value, $s(a) \in \mathbb{R}^2$.

Our base architecture then consists of the following blocks, outlined in fig. 2. First, the input image is warped with a pre-generated grid, according to section 4.1. The warped image is then processed by a standard CNN, which is now equivariant to the spatial transformation that was used to generate the warp grid. A soft argmax (eq. (14)) then finds the maximum over pose-space. To ensure the pose prediction is well registered to the reference coordinate system, a learnable scale and bias are applied to the outputs; multiple predictions can be linearly combined into a single one at this stage. Training proceeds by minimizing the L^1 loss between the predicted pose and ground truth pose.

6.2. Google Earth

For the first task in our experiments, we will consider aerial photos of vehicles, which have been used in several works that deal with rotation invariance (Liu et al., 2014; Schmidt & Roth, 2012; Henriques et al., 2014).

Dataset. The Google Earth dataset (Heitz & Koller, 2008) contains bounding box annotations, supplemented with angle annotations from (Henriques et al., 2014), for 697 vehicles in 15 large images. We use the first 10 for training and the rest for validation. Going beyond these previous works, we focus on the estimation of both rotation and scale parameters. The object scale is taken to be the diagonal length of the bounding box. Due to its small size, we augment the dataset during training, by randomly rotating the images uniformly over 360° and scaling them by up to 20%.

Implementation. A 48×48 image is cropped around each vehicle, and then fed to a network for pose prediction. The proposed method, Warped CNN, follows the architecture of section 6.1 (visualized in fig. 2). The CNN block contains 3 convolutional layers with 3×3 filters, with 50, 20 and 50 output channels respectively. We use dilation factors of 2, 4 and 8 respectively (*à trous* convolution (Chen et al., 2015)), which increases the receptive field and resolution without adding paramenters. There is a batch normalization and ReLU layer after each convolution, and a 3×3 max-pooling operator (stride 2) after the second one. The CNN block outputs response maps over 2D pose-space, which in this case consists of rotation and scale. All networks are trained for 40 epochs using the ADAM solver (Kingma & Ba, 2015) and implemented in MatConvNet (Vedaldi & Lenc, 2015). Angular error is taken modulo 180° due to annotation ambiguity. We report the average validation error over 3 runs.

Table 1. Results of scale and rotation pose estimation of vehicles in the Google Earth dataset (errors in pixels and degrees, resp.).

	ROT. ERR.	SCALE ERR.
CNN+FC	22.54	5.04
CNN+SOFTARGMAX	9.36	4.87
WARPED CNN	8.29	4.79
(DIELEMAN ET AL., 2015)	31.11	4.29

Baselines and results. The results of the experiments are presented in table 1, which shows angular and scale error in the validation set. To verify whether the proposed warped convolution is indeed responsible for a boost in performance, rather than other architectural details, we compare it against a number of baselines with different components removed. The first baseline, CNN+softargmax, consists of the same architecture but without the warp (section 5.2). This is a standard CNN, with the soft argmax at the end. Since CNNs are equivariant to translation, rather than scale and rotation, we observe a drop in performance. For the second baseline, CNN+FC, we replace the soft argmax with a fully-connected (FC) layer, to allow a prediction that is not equivariant with translation. Due to the larger number of parameters, there is overfitting and a large drop in performance. We also compare against the method of (Dieleman et al., 2015), which applies the same CNN to 90° rotations and flips of the image, combining the result with a FC layer. Just like with CNN+FC, there is overfitting on this small dataset, which requires very fine angular predictions. The proposed Warped CNN achieves the best results, except for scale prediction where (Dieleman et al., 2015) performs better. Our method has the same runtime performance as the CNN baselines, since the cost of a single warp is negligible, however (Dieleman et al., 2015) is 16× slower, since it applies the same CNN to multiple transformed images.

6.3. Faces

We now turn to face pose estimation in unconstrained photos, which requires handling more complex 3D rotations under perspective.

Dataset. For this task we use the Annotated Facial Landmarks in the Wild (AFLW) dataset (Koestinger et al., 2011). It contains about 25K faces found in Flickr photos, and includes yaw (left-right) and pitch (up-down) an-

Table 2. Results of yaw and pitch pose estimation of faces on the AFLW dataset (error in degrees).

	YAW ERR.	PITCH ERR.
CNN+FC	12.56	6.59
STN (JADERBERG ET AL., 2015)	13.65	7.22
WARPED CNN	7.07	5.28

notations. We removed 933 faces with yaw larger than 90 degrees (i.e., facing away from the camera), resulting in a set of 24,384 samples. 20% of the faces were set aside for validation.

Implementation. The region in each face's bounding box is resized to a 64×64 image, which is then processed by the network. Recall that our simplified 3D model of yaw and pitch rotation (section 5.3) assumes a spherical geometry. Although a person's head roughly follows a spherical shape, the sample images are centered around the face, not the head. As such, we use an affine Spatial Transformer Network (STN) (Jaderberg et al., 2015) as a first step, to center the image correctly. Similarly, because the optimal camera parameters (f, r and d) are difficult to set by hand, we let the network learn them, by computing their derivatives numerically (which has a low overhead, since they are scalars). The rest of the network follows the same diagram as before (fig. 2). The main CNN has 4 convolutional layers, the first two with 5×5 filters, the others being 9×9. The numbers of output channels are 20, 50, 20 and 50, respectively. A 3×3 max-pooling with a stride of 2 is performed after the first layer, and there are ReLU non-linearities between the others. As for the STN, it has 3 convolutional layers (5×5), with 20, 50 and 6 output channels respectively, and 3×3 max-pooling (stride 2) between them. The remaining experimental settings are as in section 6.2.

Baselines and results. The angular error of the proposed equivariant pose estimation, Warped CNN, is shown in table 2, along with a number of baselines. The goal of these experiments is to demonstrate that it is possible to achieve equivariance to complex 3D rotations. To compare with non-equivariant models, we test two baselines with the same CNN architecture, where the softargmax is replaced with a fully-connected (FC) layer. We include the Spatial Transformer Network (Jaderberg et al., 2015) and CNN+FC, which is a standard CNN of equivalent (slightly larger) capacity. We observe that neither the FC or the STN components account for the performance of the warped convolution, which better exploits the natural 3D rotation equivariance of the data.

7. Conclusions

In this work we show that it is possible to reuse highly optimized convolutional blocks, which are equivariant to image translation, and coax them to exhibit equivariance to other operators, including 3D transformations. This is achieved by a simple warp of the input image, implemented with off-the-shelf components of deep networks, and can be used for image recognition tasks involving a large range of image transformations. Compared to other works, warped convolutions are simpler, relying on highly optimized convolution routines, and can flexibly handle many types of continuous transformations. Studying generalizations that support more than two parameters seems like a fruitful direction for future work. In addition to the practical aspects, our analysis offers some insights into the fundamental relationships between arbitrary image transformations and convolutional architectures.

A. Spatial transformation for 3D sphere rotation under perspective

Our simplified model consists of a perspective camera with focal length f and all other camera parameters equal to identity, at a distance d from a centered sphere of radius r (see fig. 1-d).

A 2D point x in image-space corresponds to the 3D point

$$p = (x_1, x_2, f). \tag{15}$$

Raycasting it along the z axis, it will intersect the sphere surface at the 3D point

$$q = \frac{p}{\|p\|}\left(k - \sqrt{k^2 - d^2 + r^2}\right),\ k = \frac{fd}{\|p\|}. \tag{16}$$

If the argument of the square-root is negative, the ray does not intersect the sphere and so the point transformation is undefined. This means that the domain of the image Ω should be restricted to the sphere region. In practice, in such cases we simply leave the point unmodified. Then, the yaw and pitch coordinates of the point q on the surface of the sphere are

$$\phi_1 = \cos^{-1}\left(-\frac{q_2}{r}\right),\ \phi_2 = \operatorname{atan}_2\left(-\frac{q_1}{d - q_3}\right). \tag{17}$$

These polar coordinates are now rotated by the spatial transformation parameters, $\phi' = \phi + u$. Converting them back to a 3D point q'

$$q' = (r \sin\phi'_1 \sin\phi'_2, -r\cos\phi'_1, r\sin\phi'_1\cos\phi'_2 - d) \tag{18}$$

Finally, projection of q' into image-space yields

$$g_u(x) = -\tfrac{f}{q'_3}\left(q'_1, q'_2\right). \tag{19}$$

References

Bruna, Joan, Szlam, Arthur, and LeCun, Yann. Learning stable group invariant representations with convolutional networks. *arXiv preprint arXiv:1301.3537*, 2013.

Chen, Liang-Chieh, Papandreou, George, Kokkinos, Iasonas, Murphy, Kevin, and Yuille, Alan L. Semantic image segmentation with deep convolutional nets and fully connected CRFs. In *Proceedings of the 2015 International Conference on Learning Representations*, 2015.

Cohen, Taco and Welling, Max. Learning the Irreducible Representations of Commutative Lie Groups. In *Proceedings of the 31st International Conference on Machine Learning*, 2014.

Cohen, Taco and Welling, Max. Group equivariant convolutional networks. In *Proceedings of the 33rd International Conference on Machine Learning*, 2016.

Dieleman, Sander, Willett, Kyle W, and Dambre, Joni. Rotation-invariant convolutional neural networks for galaxy morphology prediction. *Monthly notices of the royal astronomical society*, 450(2):1441–1459, 2015.

Folland, Gerald B. *A course in abstract harmonic analysis*. CRC Press, 1995.

Goodfellow, Ian, Bengio, Yoshua, and Courville, Aaron. *Deep learning*. MIT Press, 2016.

Heckbert, Paul S. Fundamentals of texture mapping and image warping. Master's thesis, University of California, Berkeley, 1989.

Heitz, Geremy and Koller, Daphne. Learning spatial context: Using stuff to find things. In *European Conference on Computer Vision*, pp. 30–43. Springer, 2008.

Henriques, J. F., Martins, P., Caseiro, R., and Batista, J. Fast training of pose detectors in the fourier domain. In *Advances in Neural Information Processing Systems*, 2014.

Hyvärinen, Aapo, Hurri, Jarmo, and Hoyer, Patrick O. *Natural Image Statistics: A Probabilistic Approach to Early Computational Vision.*, volume 39. Springer Science & Business Media, 2009.

Jaderberg, Max, Simonyan, Karen, Zisserman, Andrew, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pp. 2017–2025, 2015.

Kanazawa, Angjoo, Sharma, Abhishek, and Jacobs, David. Locally scale-invariant convolutional neural networks. *arXiv preprint arXiv:1412.5104*, 2014.

Kingma, Diederik P and Ba, Jimmy Lei. Adam: A method for stochastic optimization. In *Proceedings of the 2015 International Conference on Learning Representations*, 2015.

Koestinger, Martin, Wohlhart, Paul, Roth, Peter M., and Bischof, Horst. Annotated facial landmarks in the wild: A large-scale, real-world database for facial landmark localization. In *IEEE International Workshop on Benchmarking Facial Image Analysis Technologies*, 2011.

Lavin, Andrew. Fast algorithms for convolutional neural networks. *arXiv preprint arXiv:1509.09308*, 2015.

Li, Yingzhou and Yang, Haizhao. Interpolative butterfly factorization. *arXiv preprint arXiv:1605.03616*, 2016.

Liu, Kun, Skibbe, Henrik, Schmidt, Thorsten, Blein, Thomas, Palme, Klaus, Brox, Thomas, and Ronneberger, Olaf. Rotation-Invariant HOG Descriptors Using Fourier Analysis in Polar and Spherical Coordinates. *International Journal of Computer Vision*, 106(3):342–364, February 2014. ISSN 0920-5691, 1573-1405. doi: 10.1007/s11263-013-0634-z.

Lyons, Richard G. *Understanding digital signal processing*. Pearson Education, 2010.

Marcos, Diego, Volpi, Michele, and Tuia, Devis. Learning rotation invariant convolutional filters for texture classification. *arXiv preprint arXiv:1604.06720*, 2016.

Reddy, B. Srinivasa and Chatterji, Biswanath N. An FFT-based technique for translation, rotation, and scale-invariant image registration. *IEEE Transactions on Image Processing*, 5(8):1266–1271, 1996.

Schmidt, Uwe and Roth, Stefan. Learning rotation-aware features: From invariant priors to equivariant descriptors. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 2050–2057, 2012.

Tygert, Mark. Fast algorithms for spherical harmonic expansions, iii. *Journal of Computational Physics*, 229 (18):6181–6192, 2010.

Tzimiropoulos, Georgios, Argyriou, Vasileios, Zafeiriou, Stefanos, and Stathaki, Tania. Robust FFT-based scale-invariant image registration with image gradients. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(10):1899–1906, 2010.

Vedaldi, A. and Lenc, K. MatConvNet – Convolutional Neural Networks for MATLAB. In *Proceeding of the ACM Int. Conf. on Multimedia*, 2015.

Acknowledgements. This research was funded by ERC 677195-IDIU.