WARPED CONVOLUTIONS: EFFICIENT INVARIANCE TO SPATIAL TRANSFORMATIONS

João F. Henriques & Andrea Vedaldi
Visual Geometry Group
University of Oxford
{joao,vedaldi}@robots.ox.ac.uk

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 2-parameters 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 allows to reduce 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 and also less computation compared to an equivalent fully-connected layer.

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), which requires more storage than a convolutional filter.

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 propose a generalization of CNNs that overcomes these disadvantages. Our main result shows that a linear layer with equivariance w.r.t. a large class of 2-parameters transformations can always be implemented efficiently, using a standard convolution in a warped image space. The image warp can be implemented using bilinear resampling, a simple and fast operation that has been popularized by spatial transformer networks (Jaderberg et al., 2015), 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, including using fast convolution algorithms on GPU hardware, such as Winograd (Lavin, 2015) or the Fast Fourier Transform (Lyons, 2010). We present these notions in the simplest possible way (sections 2 to 4), but we note that they can be derived in broader generality from well know concepts of group theory (section 4.2).

2 GENERALIZING CONVOLUTION

2.1 CONVOLUTIONS OF CONTINUOUS IMAGES

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 $I \in \mathbb{R}^{m \times n}$ with a translated version of the filter $F \in \mathbb{R}^{r \times s}$, producing a new image as output:

$$H_j = \sum_k I_k F_{k+j}, \tag{1}$$

where $k,\ j \in \mathbb{Z}^2$ are two-dimensional vectors of indexes, and the summation ranges inside the extents of both arrays.[1] To handle continuous deformations of the input, it is more natural to express eq. 1 as an integral over continuous rather than discrete inputs:

$$H(u; I) = \int I(x)\, F(x + u)\, dx, \tag{2}$$

where $I(x)$ and $F(x)$ are continuous functions over a bounded 2D region $\Omega \subset \mathbb{R}^2$, that is: $I,\ F : \Omega \to \mathbb{R}$. The real-valued 2D vectors $x \in \Omega$ now play the role of the indexes $k \in \mathbb{Z}^2$. Equation 2 reduces to the discrete case of eq. 1 if we define $I(x)$ and $F(x)$ as the sum of delta functions on grids. 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 very rich continuous transformations of the input coordinates, whereas strictly discrete operations would be more limiting.

Over the next sections it will be more convenient to translate the image I instead of the filter F. This alternative form of eq. 2 is obtained by replacing $x + u \to x$:

$$H(u; I) = \int I(x - u)\, F(x)\, dx. \tag{3}$$

2.2 BEYOND IMAGE TRANSLATIONS

The standard convolution operator of eq. 3 can be interpreted as applying the filter to translated versions of the image. Translations can be replaced by other transformations as follows (Henriques et al., 2014):

$$H(t; I) = \int I(t(x))\, F(x)\, dx, \quad t \in G \tag{4}$$

where G is a set of transformation functions $t : \Omega \to \Omega$ (assumed to be invertible). Intuitively, this *generalized convolution* performs an exhaustive search for a pattern, at many different poses (Henriques et al., 2014; Kanazawa et al., 2014). The interest in this definition lies in the fact that it makes convolution *equivariant* (Lenc & Vedaldi, 2015):

[1] Note that eq. 1 defines cross-correlation in the signal processing literature, but here we follow the convention used in machine learning and call it convolution. We also ignore the possibility that the input image has more than one channel, and that convolution layers in CNN involve banks of filters instead of single ones. All such details are immaterial to our discussion.

Lemma 1 (Equivariance)**.** *Consider the generalized convolution operator $H(t; I)$ of eq.* 4*. Generalized convolution "commutes" with any transformation $q \in G$ of the image:*

$$H(t; I \circ q) = H(q \circ t; I).$$

Proof. One has immediately $H(t; I \circ q) = \int I(q(t(x)))\, F(x)\, dx = H(q \circ t; I)$. □

A notable case is when transformations have an additive parametrization $t : \Omega \times \mathbb{R}^2 \to \Omega$, with $(x, u) \mapsto t_u(x)$ and $t_u \circ t_v = t_{u+v}$. In this case, the equivariance relation can be written as

$$H(u; I \circ t_v) = H(v + u; I). \tag{5}$$

In particular, standard convolution is obtained when $t_u(x) = x - u$ is the translation operator. In this case, the lemma above simply states that any translation of the input of the convolution results in a corresponding translation of the output.

In section 5, we will look in more detail at a few concrete examples of transformations other than translations. Although we will not do so explicitly, in this construction it is also possible to let one or more dimensions of the parameter space $\mathbb{R}^2$ be given modulus a period Q, in the sense of replacing $\mathbb{R}$ with $\mathbb{R}/\mathbb{Z}(Q)$; the latter is required to parameterise transformations such as rotation.

3 Computational efficiency

Unfortunately, what eq. 4 gains us in generality, it loses in both performance and ease of implementation. Most works in computer vision that looked at filtering under generalized transformations (e.g. scale pyramids (Kanazawa et al., 2014) or rotated filter banks (Marcos et al., 2016; Cohen & Welling, 2014; 2016; Henriques et al., 2014)) compute eq. 4 directly by evaluating a large number of transformations $t \in G$. This entails warping (transforming) either the image or the filter once per transformation t, which can be expensive.

Opting to transform the filter instead of the image can be advantageous, since it is smaller in size. On the other hand, the filter and its domain then become spatially-varying, which foregoes the benefit of the regular, predictable, and local pattern of computations in standard convolution. It precludes the use of fast convolution routines such as Winograd's algorithm (Lavin, 2015), or the Fast Fourier Transform (Lyons, 2010), which has lower computational complexity than exhaustive search (eq. 3).

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 explain why generalized convolutions are not as widespread as CNNs.

In section 4 we will show that, for an important class of transformations, including the ones considered in previous works (such as Kanazawa et al. (2014); Cohen & Welling (2014); Marcos et al. (2016)) it is possible to perform generalized convolution by composing a single warp with a standard convolution, instead of several warps. Thus, we are able to take full advantage of modern convolution implementations (Lavin, 2015; Lyons, 2010), including those with lower computational complexity.

4 Main result

Our main contribution is to show that the generalized convolution operator of eq. 4 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, given in theorem 1, allows us to implement very efficient generalized convolutions using simple computational blocks, as shown in section 4.1. We name this method *warped convolution*.

The strongest assumption is that transformations must have an additive parametrization. By this, we mean that there exists a bijection $t_u : \Omega \to \Omega$ such that, for any $u, v \in \mathbb{R}^2$, parameters compose additively $t_u \circ t_v = t_{u+v}$. The second assumption is that there exists a pivot point $x_0 \in \Omega$ such

that $u \mapsto t_u(x_0)$ defines a bijection $\mathbb{R}^2 \to \Omega$ from the parameter space to the real plane. The latter requirements means that any point $x \in \Omega$ can be "reached" by transforming x_0 under a suitable t_u. We then have that:

Theorem 1. *Consider the generalized convolution of eq.* 4*. Assume that the transformation is additive ($t_u \circ t_v = t_{u+v}$). Assume also that, for a fixed pivot point x_0, the function $u \mapsto t_u(x_0)$ is bijective. Then we can rewrite generalized convolution (eq.* 4*) as the standard convolution*

$$H(u; I) = \int \hat{I}(u+v)\hat{F}(v)\, dv, \tag{6}$$

where $\hat{I}$ and $\hat{F}$ are the warped image and filter given by:

$$\hat{I}(u) = I(t_u(x_0)), \qquad \hat{F}(u) = F(t_u(x_0)) \left| \frac{dt_u(x_0)}{du} \right|. \tag{7}$$

Proof. We use the variable substitution $x = t_v(x_0)$ in eq. 4. Then:

$$\begin{aligned} H(u; I) &= \int I(t_u(x))\, F(x)\, dx \\ &= \int I(t_u(t_v(x_0)))\, F(t_v(x_0)) \left| \frac{dt_v(x_0)}{dv} \right| dv \\ &= \int \underbrace{I(t_{u+v}(x_0))}_{\hat{I}(u+v)} \underbrace{F(t_v(x_0)) \left| \frac{dt_v(x_0)}{dv} \right|}_{\hat{F}(v)} dv \\ &= \int \hat{I}(u+v)\hat{F}(v)\, dv. \end{aligned}$$

□

The warp that is applied to both inputs in eq. 7 can be interpreted as follows. We start with an arbitrary pivot point x_0 in the image and them sample other points by repeatedly applying the transformation $t_u(x_0)$ to the pivot (by varying u). When discretized, this sampling is performed over a 2D grid of parameters u. Finally, sampling the input at these points (for example, by bilinear interpolation) yields the warped input.

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 two directions of increasing u values (recall that transformation parameters are two-dimensional). The grids were generated by sampling u 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 last factor in eq. 7 is the determinant of the Jacobian of the image transformation t. It rescales the image values to account for the stretching and shrinking of space due to non-linear warps. It can also be computed offline, and its application amounts to an element-wise product by a constant array. A generalization using group theory is discussed in section 4.2.

4.1 Practical considerations

There are a few interesting aspects that simplify the use of theorem 1 in practice.

First, since in most applications the filter F is learned, we are free to ignore the constant warp and Jacobian in eq. 7 (which amounts to a simple reparametrization), and learn $\hat{F}$ directly. In practice, this means that we warp only the input image I to obtain $\hat{I}$, and then perform a standard convolution with a filter $\hat{F}$. The learned warped filter $\hat{F}$ has a one-to-one correspondence to an image-space filter F by means of eq. 7, although there is no real need to build the latter explicitly.

Second, we can choose either one or two spatial transformations for the generalized convolution (e.g. scale and rotation, simultaneously). The reason is that the input image is 2D, so the parameter-space after warping is also 2D. The choice is not arbitrary though: the two transformations must commute, in order to respect additivity. This will be the case of the pairs we study in section 5.

Algorithm 1 Warped convolution.

Grid generation (offline)

- Apply the spatial transformation t repeatedly to a pivot point x_0, using a 2D grid of parameters $u = \{(u_1 + i\delta_1, u_2 + j\delta_2) : i = 0, \dots, m, j = 0, \dots, n\}$, obtaining the 2D warp grid $t_u(x_0)$.

Warped convolution

1. Resample input image I using the warp grid $t_u(x_0)$, by bilinear interpolation.
2. Convolve the warped image $\hat{I}$ with filter $\hat{F}$.

By theorem 1, these steps are equivalent to a generalized convolution, which performs an exhaustive search across the pose-space of transformation t, but at a much lower computational cost.

4.2 Relationship to group theory

This section relates our results, which have been presented using a simple formalism and in a restricted setting, to a more general approach based on group theory (Folland, 1995).

To this end, let G be a group of transformations. Under very mild conditions (the group has to be locally compact and Hausdorff), there exists a unique measure on the group, the Haar measure, which is invariant to the group action, in the sense that, given a measurable function $\tilde{I} : G \to \mathbb{R}$, then $\int \tilde{I}(g'g)\, dg = \int \tilde{I}(g)\, dg$. Using this measure, one can define generalized convolution as $(\tilde{I} * \tilde{F})(t) = \int_G \tilde{I}(tg)\tilde{F}(g^{-1})\, dg$. This resembles our definition (4), although image and filter are defined on the group G instead of the spatial domain $\mathbb{R}^2$. Lemma 1 translates immediately to this case (Folland, 1995).

In order to extend Theorem 1, we need to make this general but abstract construction concrete. Here one assumes that the group acts transitively on a subset $X \subset \mathbb{R}^2$ (which means that any point $x \in X$ can be written as $x = gx_0$, for a fixed point $x_0 \in X$ and a suitable transformation $g \in G$). Then one can define the image as $\tilde{I}(g) = I(g(x_0))$, where I is a function of the spatial domain X instead of the group G, and likewise for the filter. Next, it is necessary to explicitly calculate the integral over G. If the group is an Abelian (commutative) Lie group, then one can show that there exists a map $\exp : V \to G$, the exponential map, defined on a vector space V. Under commutativity, this map is also additive, in the sense that $\exp(u)\exp(v) = \exp(u + v)$. The structure of V depends on the specific group, but under such restrictive conditions, it is a torus, which allows the calculation of $\int \tilde{I}(g)\, dg$ as $\int \tilde{I}(\exp(u)x_0)\, du$.

Finally, in order to swap integration over the group parameters with integration over space, one assumes that $x = \exp(u)x_0$ defines a smooth bijection $V \to X$, so that it is possible to use the change of variable $u \to u(x)$ where $\exp(u(x))x_0 = x$. This allows writing the integral as $\int \tilde{I}(\exp(u)x_0)\, du = \int I(x)\, |du/dx|\, dx$. Note that this Jacobian is the inverse of the one found in (1) due to the fact that we started by defining our convolution using I instead of $\tilde{I}$.

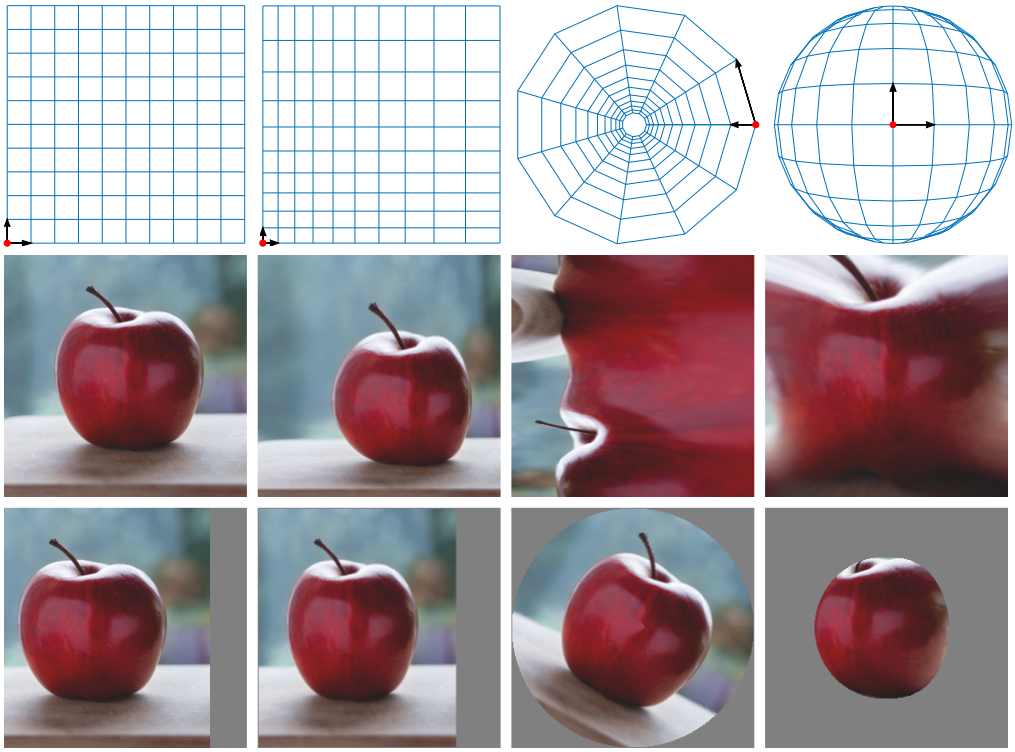
5 Examples of spatial transformations

We now give some concrete examples of pairs of spatial transformations that obey the conditions of theorem 1, and can be useful in practice.

5.1 Scale and aspect ratio

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:



(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.

$$t_u(x) = \begin{bmatrix} x_1 s^{u_1} \\ x_2 s^{u_2} \end{bmatrix} \tag{8}$$

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 required by theorem 1: $t_u(t_v(x)) = t_{u+v}(x)$. 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

$$t_u(x) = \begin{bmatrix} s^{u_2} \|x\| \cos(\text{atan}_2(x_2, x_1) + u_1) \\ s^{u_2} \|x\| \sin(\text{atan}_2(x_2, x_1) + u_1) \end{bmatrix}, \tag{9}$$

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.

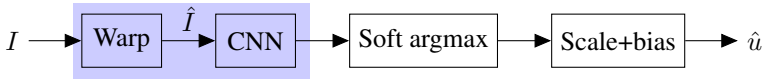


Figure 2: Equivariant pose estimation strategy used in the experiments (section 6). With an appropriate warp and a standard CNN, the shaded block becomes equivalent to a generalized CNN (by theorem 1), which performs exhaustive searches across pose-space instead of image-space.

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.

5.3 3D SPHERE ROTATION UNDER PERSPECTIVE

We will now tackle a more difficult spatial transformation, in an attempt to demonstrate the generality of theorem 1. 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, generalized convolution performs 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:

$$s_1(a) = \sum_{ij}^{mn} \frac{i}{m} \sigma_{ij}(a), \qquad s_2(a) = \sum_{ij}^{mn} \frac{j}{n} \sigma_{ij}(a), \tag{10}$$

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. 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. 10) 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

	CNN+FC	CNN+softargmax	Warped CNN
Rotation error (degrees)	28.87	30.6	26.44
Scale error (px)	17.51	5.783	5.4

Table 1: Results of scale and rotation pose estimation of vehicles in the Google Earth dataset.

Figure 3: Example pose estimates (rotation and scale) on the Google Earth dataset (Section 6.2).

to the outputs. 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.

Implementation. A 48×48 image around each vehicle is cropped and downscaled by 50%, 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 5×5 filters, with 20, 50 and 1 output channels respectively. Recall that the output of the CNN block is a single-channel response map over 2D pose-space, which in this case consists of rotation and scale. Between the convolutional layers there are 3×3 max-pooling operators, with a stride of 2, and a ReLU before the last layer. All networks are trained for 20 epochs with SGD, using hyperparameters chosen by cross-validation.

Baselines and results. The results of the experiments are presented in table 1, which shows angular and scale error in the validation set. Qualitative results are shown in fig. 3. 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 layer, to allow a prediction that is not equivariant with translation. The FC layer improves the angular error, but not the scale error. The proposed Warped CNN has a similar (slightly lower) capacity to the CNN+FC baseline, but we see it achieve better performance, since its architectural equivariance seems to be better matched to the data distribution.

	CNN+FC	STN+FC	STN+softargmax	Warped CNN
Yaw err. (deg.)	13.87	16.92	15.01	10.65
Pitch err. (deg.)	7.23	10.17	6.88	6.351

Table 2: Results of yaw and pitch pose estimation of faces on the AFLW dataset.

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) annotations. 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 1, 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.

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. Qualitative results are shown in fig. 4. The goal of these experiments is to demonstrate that it is possible to achieve equivariance to complex 3D rotations. We also wish to disentangle the performance benefits of the warped convolution from the other architectural aspects. The first baseline, STN+softargmax, is the same as the proposed method, but without the warp. The large performance drop indicates that the spherical model incorporates important domain knowledge, which is ignored by a translation-equivariant STN. To allow non-equivariant models, we also test two other baselines where the softargmax is replaced with a fully-connected (FC) layer. The STN+FC includes an affine Spatial Transformer, while the CNN+FC does not, corresponding to a standard CNN of equivalent capacity. We observe that neither the FC or the STN components can account up 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.

Figure 4: Example pose estimates (yaw and pitch) on the AFLW dataset (Section 6.3).

REFERENCES

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

Taco Cohen and Max Welling. Learning the Irreducible Representations of Commutative Lie Groups. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 2014.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*, 2016.

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

Gerald B Folland. A course in abstract harmonic analysis. 1995.

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

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

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

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

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

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

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

Karel Lenc and Andrea Vedaldi. Understanding image representations by measuring their equivariance and equivalence. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 991–999, 2015.

Kun Liu, Henrik Skibbe, Thorsten Schmidt, Thomas Blein, Klaus Palme, Thomas Brox, and Olaf Ronneberger. 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.

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

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

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

Uwe Schmidt and Stefan Roth. 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.

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

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

$$p = (x_1, x_2, f). \tag{11}$$

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

$$q = \frac{p}{\|p\|}\left(k - \sqrt{k^2 - d^2 + r^2}\right),\ k = \frac{fd}{\|p\|}. \tag{12}$$

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

$$\phi_1 = \cos^{-1}\left(-\frac{q_2}{r}\right),\ \phi_2 = \operatorname{atan}_2\left(-\frac{q_1}{d - q_3}\right). \tag{13}$$

These polar coordinates are now rotated by the spatial transformation parameters, $\phi' = \phi + u$.

Converting the polar coordinates back to a 3D point q'

$$q' = \begin{bmatrix} r\sin\phi'_1 \sin\phi'_2 \\ -r\cos\phi'_1 \\ r\sin\phi'_1 \cos\phi'_2 - d \end{bmatrix}. \tag{14}$$

Finally, projection of q' into image-space yields

$$t_u(x) = -\frac{f}{q'_3}\begin{bmatrix} q'_1 \\ q'_2 \end{bmatrix}. \tag{15}$$