WARPED CONVOLUTIONAL NETWORKS: BRIDGE HOMOGRAPHY TO $\mathfrak{sl}(3)$ ALGEBRA BY GROUP CONVOLUTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Homography has an essential relationship with the special linear group and the embedding Lie algebra structure. Although the Lie algebra representation is elegant, few researchers have established the connection between homography and algebra expression in neural networks. In this paper, we propose Warped Convolution Networks (WCN) to effectively learn and represent the homography by $SL(3)$ group and $\mathfrak{sl}(3)$ algebra with group convolution. To this end, six commutative subgroups within the $SL(3)$ group are composed to form a homography. For each subgroup, a warping function is proposed to bridge the Lie algebra structure to its corresponding parameters in homography. By taking advantage of the warped convolution, homography learning is formulated into several simple pseudo-translation regressions. By walking along the Lie topology, our proposed WCN is able to learn the features that are invariant to homography. Moreover, it can be easily plugged into other popular CNN-based methods. Extensive experiments on the POT benchmark, S-COCO-Proj, and MNIST-Proj dataset show that our proposed method is effective for planar object tracking, homography estimation, and classification.

1 INTRODUCTION

Group convolution attracts a lot of research attention due to its underlying group structure constrain upon the learning representation. The most successful case is Convolution Neural Networks (CNN) that is famous for the weak translation equivariance on representing visual object in image space. Essentially, the translation equivariance is achieved because of the constraint and intrinsic topological structure of discrete groups on the image. With a simple group structure, CNN has already been successfully and extensively used in a variety of tasks, including object detection (Dai et al., 2016), recognition (Zhou et al., 2014), tracking (Voigtlaender et al., 2020), and alignment (Ji & Telgarsky, 2020). To further exploit the representation capability, researchers try to extend the conventional convolution to group convolution (Macdonald et al., 2021; Cohen & Welling, 2016; Zhang, 2019; Sosnovik et al., 2020) with the diversity of group structures.

Among these group structures, special linear (SL) group and its embedding Lie algebra have great potential in visual representation since the corresponding homography describes the relation of two image planes for a 3D planar object with perspective transformation. Intuitively, a laptop is always a laptop wherever you are looking from. Every element in $SL(3)$ represents a homography of two different cameras shooting at a static 3D planar object in the scene. The corresponding Lie algebra space $\mathfrak{sl}(3)$ describes the changes of a camera's configuration, which means the local changes in Lie algebra coincide with the slight movement of viewpoint. Neural networks built on the space of $\mathfrak{sl}(3)$ could achieve the homography learning capability, which gives, to some extent, equivariance and invariance to the feature representation for visual objects. This property could benefit a number of applications, e.g. homography estimation (Japkowicz et al., 2017), planar object tracking (Zhan et al., 2022), feature representation (Jaderberg et al., 2015).

However, few researchers have investigated the relation between homography and the $\mathfrak{sl}(3)$ algebra. Some task-oriented works (Esteves et al., 2018; Ye et al., 2021; Finzi et al., 2020) show the preliminary results in the application but rarely establish the connection to the corresponding group. Benton et al. (2020) learn the invariance by parameterizing a distribution with augmentations and optimizing the training loss, simultaneously. Esteves et al. (2018) estimate the translation firstly and then classify

the image in the log-polar coordinates (Zwicke & Kiss, 1983), which is in fact a special case for the similarity group $Sim(2)$. Ye et al. (2021) enforce a $GL(n)$-invariance property with global statistics extracted from training data, in which the gradient descent optimization should maintain the group invariance under basis changes. Unfortunately, these methods either are only capable of dealing with several subgroups of $SL(3)$ and their corresponding transformations, or purely enforce the equivariance learning by tremendous data augmentation on the image domain. Dehmamy et al. (2021) introduce an unsupervised method for $\mathfrak{sl}(3)$ and Macdonald et al. (2021) achieve the SL(3) equivariant networks. Both methods have drawbacks due to either the inferior performance or heavy computational cost.

Our goal is to connect the representation learning of homography with $\mathfrak{sl}(3)$ algebra for neural networks in an efficient way. When the representation is based on Lie algebra, we could investigate the potential in the algebra space with its mathematical property. For instance, the feature representation is consistent with human's intuitive perception, as the transformation walks along the geodesic curve in algebra space. This allows the networks to have very robust feature representation for different transformations and have the capability to neglect the noise in training. Additionally, this is helpful for learning the implicit transformation from sinle image, which may facilitate the applications such as congealing and facade segmentation.

Inspired by the warped convolution (Henriques & Vedaldi, 2017), we construct six commutative subgroups within the $SL(3)$ group from the Lie algebra $\mathfrak{sl}(3)$ generators to achieve homography learning. For each subgroup, a warping function is proposed to bridge the Lie algebra structure to its corresponding parameters in homography. As the constructed subgroups are Abelian groups, the group convolution operation can be formulated into conventional convolution with a well-designed warping function. In this paper, we propose Warped Convolution Networks (WCN) to compose these subgroups to form an $SL(3)$ group convolution by predicting several pseudo-translation transformations. Our proposed WCN is able to handle noncommutative groups and learn the invariant features for homography. The main contribution can be summarized as follows:

- A general neural network for efficient Lie group convolution. Our proposed framework can deal with most of the Lie groups by easily combining different basis, which is able to learn the invariant features for different neural networks.
- A robust homography estimator based on WCN. To the best of our knowledge, it is the first work to directly estimate homography along with the $SL(3)$ group and its algebra topology.
- Extensive experimental evaluation of three datasets demonstrates that our proposed approach is effective for three computer vision tasks and offers more potential for robustly learning the large and implicit transformations.

2 RELATED WORK

In this section, we discuss the related prior studies, including the equivariant networks and transformation learning. Cohen & Welling (2016) present the fundamental work on equivariance of CNNs representations for the image transformations, where the underlying properties of symmetry groups Cohen & Welling (2015) are investigated. They replace the translational convolutions with group convolutions and propose Group equivariant convolutional networks (G-CNNs). For the continuous groups, Cohen & Welling (2016) discretize the group and employ the harmonic function as the irreducible representation of convolution (Weiler et al., 2018; Zhang, 2019; Sosnovik et al., 2020). Recently, Macdonald et al. (2021) change the convolution formula and modify the different layers of CNNs. They use the Schur-Poincaré formula for the derivative of exponential map, which enables to sample from Haar measure for arbitrary Lie groups. Dehmamy et al. (2021) propose an unsupervised learning approach to automatically learn the symmetry based on Lie algebra in CNNs. It performs inferior to the baseline when the symmetric group is already known. All these methods introduce the various architectures that are difficult to directly make use of the popular CNN backbones. Moreover, they can only deal with the classification problem.

Early work learns the transformation representation by an auto-encoder (Hinton et al., 2011). It attempts to build a generative model, where the target is a transformed input image. Lin et al. (2021) change the parameterization and project the distance onto $SO(3)$. Jaderberg et al. (2015) introduce a spatial transformation network to manipulate the data in the network without supervision on the transformation. All these methods have difficulty in estimating the transformations since the networks can only inference once for guessing and the parameters are entangled and highly coupled.

ESM (Benhimane & Malis, 2004) parameterizes the arguments as the Lie algebra basis to estimate the $SL(3)$ group. However, their parameters lose the interpretability in an image transformation. Henriques & Vedaldi (2017) employ the warp function on convolution and implement two-parameter group equivariance, since there are possibly utmost two independent dimensions in an image. Zhan et al. (2022) recently decompose the homography into two groups and estimate them in order, which loses the equivariance for the residual parameters. Besides, it requires an additional homography estimator to find the corners. Recently, deep learning-based approaches predict the homography mainly by estimating the corner offsets (Nguyen et al., 2018; Zhang et al., 2020a) or pixel flows (Zeng et al., 2018). They focus on the local movements in the image space, which are incapable of estimating the large transformation. Our proposed approach can be viewed as a general case of the warped convolution in 2D space, which is able to handle the most sophisticated 2D Lie group $SL(3)$.

3 METHOD

The main purpose of this work is to formulate a full homography on Lie subalgebras with several equivariant warped convolutions for 2D projective transformation. Since the warped convolution only implements two-parameter equivariance, a possible way is to combine the several warped convolutions. In general, the 2D projective transformation is an $SL(3, \mathbb{R})$ group having a few subgroups. Our proposed method divides this group into several one or two-parameter subgroups, whose Lie algebras are the subalgebras of $\mathfrak{sl}(3)$. As a result, the warped convolution can be employed to achieve the equivariance for each single or two-parameter transformation. Finally, they are combined to obtain the full transformation. In this section, we first introduce the fundamental of the warped convolution, and then describe our proposed method.

3.1 WARPED CONVOLUTION

The key to CNNs' equivariance is their convolution layers, in which the basic operation is the convolution of an image $I \in \mathbb{R}^{n \times n}$ and a convolution kernel $F \in \mathbb{R}^{n' \times n'}$. By employing the Dirac delta function on the image and kernel (Henriques & Vedaldi, 2017), the convolution formula can be treated as a special case of continuous function as follows,

$$(I * F)(\mathbf{v}) = \int I(\mathbf{v} + \mathbf{u})F(-\mathbf{u})d\mathbf{u}. \tag{1}$$

For the sake of convenience, we shift the image I instead of F in the convolution equations. To prove the equivariance, we define the transformation operator as $\pi_l : \mathbf{u} \longmapsto \mathbf{u} + \mathbf{l}$. Hence, the equivariance concerning the translation can be easily proved as: $(\pi_l(I) * F)(\mathbf{v}) = \int I(\mathbf{u} + (\mathbf{v} + \mathbf{l}))F(-\mathbf{u})d\mathbf{u} = (\pi_l(I * F))(\mathbf{v})$.

The standard convolution only takes into account the translation equivariance in the image. For the equivariance of other groups in the image domain, Henriques & Vedaldi (2017) suggest an intuitive solution that defines a function of a group action on the image as below,

$$(\tilde{g} * \tilde{h})(q) = \int_G g(pqx_0)h(p^{-1}x_0)d\xi(p). \tag{2}$$

Eq. 2 provides the convolution of two real functions $\tilde{g}$ and $\tilde{h}$. $g(px_0) = \tilde{g}$ and $h(px_0) = \tilde{h}$ are defined on a subset $\Omega \in \mathbb{R}^2$, where $x_0 \in \Omega$ is an arbitrary constant pivot point. Compared to the convolution on the image, the operation $\mathbf{u} + \mathbf{v}$ becomes pq for the function g and h, where $p, q \in G$. The integration is under the Haar measure ξ, which is the only measure invariant to the group transformation. Since Eq. 2 is still defined over the group, it needs to be further simplified. A simple approach is to define G as a Lie group, which can be projected onto the Lie algebra $\mathfrak{m}$. $\mathfrak{m}$ is a vector space tangent at identity ε of the group manifold $\mathcal{M}$, whose base elements $\mathbf{b}$ are easy to map to the Cartesian vector space $V \in \mathbb{R}^m$. The dimension m is the degrees of freedom for $\mathcal{M}$. This mapping allows us to estimate the Lie algebra on the real plane $\mathbb{R}^2$. Once the element of Lie algebra is obtained, it could be mapped to $\mathcal{M}$. Therefore, an exponential map $\exp : V \to G$ is employed to connect the Cartesian vector space with the Lie group manifold, where V is a subset of $\mathbb{R}^2$. We therefore have the warped image $g_w(\mathbf{u}) = g(\exp(\mathbf{u})x_0)$. Thus, Eq. 2 can be rewritten as:

$$(\tilde{g} * \tilde{h})(\exp(\mathbf{v})) = \int_E g_w(\mathbf{u} + \mathbf{v})h_w(-\mathbf{u})d\mathbf{u}. \tag{3}$$

Obviously, Eq. 3 has the same formulation as Eq. 1. This achieves the equivariance to the transformation belonging to the Lie group by performing a conventional convolution after warping the image which connects the warped convolution and group convolution.
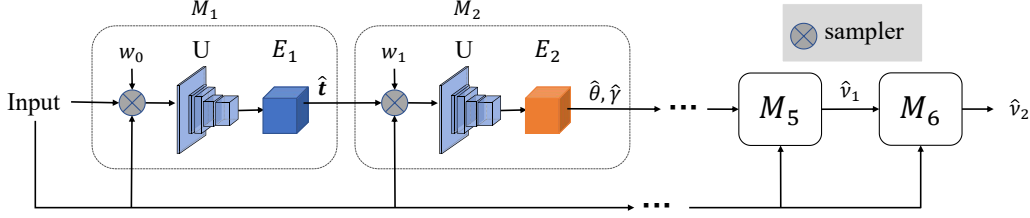
Figure 1: Overview of WCN. M_1 is a basic module of WCN, and w_i is the i_{th} warp function. The output(s) is the transformed version of the input image(s). The total number of inputs depends on the tasks. U is a CNN backbone, and E_1 is a translation estimator. The sampler is used to re-sample the input according to w and the previously estimated $\hat{t}$. We do not specify the detailed structure for its generality, however, details for specific tasks can be found in the Experiments and Appendix.

3.2 WARPED CONVOLUTIONAL NETWORKS

We employ the compositional method to estimate the Lie subalgebras and combine the subgroups in order. To warp the image by a series of functions, the $SL(3)$ generators need to be defined before the composition. In this paper, we choose the widely used 2D projective group decomposition (Harltey & Zisserman, 2003), whose corresponding eight-parameters Lie algebra is used to compose the Lie algebra $\mathfrak{sl}(3)$. Specifically, these generators are defined as follows,

$$\mathbf{A_1} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{A_2} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{A_3} = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{A_4} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \mathbf{A_5} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{A_6} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \mathbf{A_7} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \mathbf{A_8} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \quad (4)$$

For each generator, we construct a one-parameter group. The other dimension for g_w could be viewed as an identity transformation group, which is commutative to the one-parameter group. As a result, the equivariance is also valid in the case of one-parameter group. We choose the generators and compose them corresponding to two or one parameter group for warping, as long as they are commutative. In this paper, we propose to compose the generators of $\mathfrak{sl}(3)$ into six Lie subalgebras as $[b_1 A_1 + b_2 A_2, b_3 A_3 + b_4 A_4, b_5 A_5, b_6 A_6, b_7 A_7, b_8 A_8]$, where $[b_1, b_2, ..., b_8]$ are the elements of the generator coefficients vector $\mathbf{b}$. (More details can be found in Appendix B.) For better symbol presentation, we re-parameterize $\mathbf{b}$ into a homography-friendly format to link the Lie algebra with the homography. The resulting intermediate variables vector $\mathbf{x} = [t_1, t_2, \theta, \gamma, k_1, k_2, \nu_1, \nu_2] = [b_1, b_2, b_3, \exp(b_4), \exp(b_5), b_6, b_7, b_8]$. We will introduce the definition of $\mathbf{x}$ in the next sub-section. Therefore, the six Lie subalgebras corresponding subgroups parameterized by $\mathbf{x}$ are defined as follows:

$$\mathbf{H(x)} = \mathbf{H_t} \cdot \mathbf{H_s} \cdot \mathbf{H_{sc}} \cdot \mathbf{H_{sh}} \cdot \mathbf{H_{p1}} \cdot \mathbf{H_{p2}} \quad (5)$$

$$= \begin{bmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \gamma\cos\theta & -\gamma\sin\theta & 0 \\ \gamma\sin\theta & \gamma\cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} k_1 & 0 & 0 \\ 0 & 1/k_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & k_2 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \nu_1 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \nu_2 & 1 \end{bmatrix}$$

Based on the above equation, we propose Warped Convolutional Networks (WCN) to learn homography by six modules $[M_1, ..., M_6]$ as depicted in the Fig. 1. Each module M_i has three components, a shared backbone U, a translation estimator E_i and a warp function w_i. According to Eq. 3, recovering the Lie algebra parameters is equivalent to estimating a pseudo-translation in the subgroup to which the warp function transfers the image space. For each module, the input image is resampled with a special designed warp function w_i, fed to the backbone U and estimator E_i to obtain a pseudo-translation in the corresponding subalgebra. Note that we predict $\mathbf{b}$ essentially, and $\mathbf{x}$ is just a function of $\mathbf{b}$ for a convenient parameterization. E_i is different for each module to adapt to the different subalgebras. Finally, we obtain the output $\mathbf{x}$ and compose them to the transformation matrix as Eq. 4. Theoretically, the parameters of the proposed six groups may affect each other in the warped image domain. Fortunately, we found that the networks localize the object's position very well in the most of vision tasks, even with large deformations or distortions. We argue that the networks manage to learn the invariant feature for the target to compensate the interdependence. Intuitively, we transfer all 8 parameters of Lie algebra $\mathfrak{sl}(3)$ into 6 subalgebras that can be solved by pseudo-translation estimation. In the warped image domain, the pseudo-translation is more significant in contrast to other transformations. Thereby, we take advantage of this property to estimate the sub-algebra in each warped image. More details can be found in the Appendix.
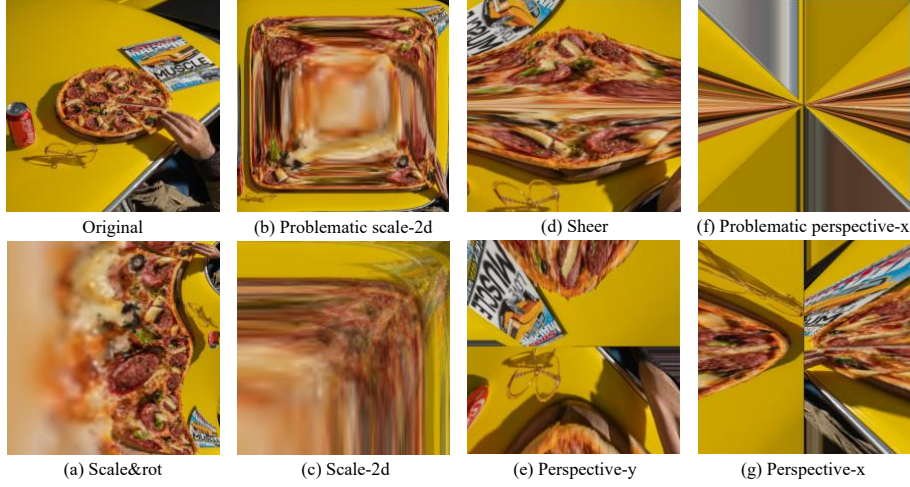
Figure 2: The examples after using the warping function for the original image.

3.3 WARP FUNCTIONS

As illustrated in Eq. 3, the key to recovering one or two-parameter transformation is to find a proper warp function so that the pseudo-translation shift in the warped image is equivalent to the linear changes of element on the corresponding Lie algebra. To this end, we define the warp function as $w(\mathbf{x}') = \mathbf{u}'$, where $\mathbf{x}' = (x_1', x_2')$ is the two intermediate parameters vector of the warp function for two-parameter group. For one-parameter group, x_2' is the identity Lie algebra parameter (x_ϵ). $\mathbf{u}' = (u_1', u_2')$ denotes the re-sampled point in the transformed I.

Scale and Rotation Generally, CNNs are equivariant to the translation that is preserved after feature extraction. As a result, w_0 is an identical function and we omit it in our implementation. For the scale and rotation groups, γ represents the uniform scale, and θ denotes the in-plane rotation. As described in (Henriques & Vedaldi, 2017; Zhan et al., 2022), the warping function w_1 for two parameters scale γ and rotation θ is defined as:

$$w_1(\gamma, \theta) = \mathbf{u}'^T = \begin{bmatrix} s^{\gamma'} \|\tau\| \cos(\arctan_2(\tau_2, \tau_1) + \theta) \\ s^{\gamma'} \|\tau\| \sin(\arctan_2(\tau_2, \tau_1) + \theta) \end{bmatrix} \tag{6}$$

where s determines the degree of scaling. $\tau = (\tau_1, \tau_2)$ is the pivot, and $\gamma = s^{\gamma'}$. The coordinates $\mathbf{u}' = (u_1', u_2')$ denote the re-sampled point in image I, and $\arctan_2$ represents the standard 4-quadrant inverse tangent function. We set $\tau = (0, 1)$ for convenience since τ can be any point except the origin. For the rest of the warp functions, τ can be used the same way, and we omit it from the warp function for simplicity. We have $\gamma = (s^{\gamma'} = e^{\gamma' \log s}) = e^{b_4}$. Let s be a constant. Thus, estimating γ' is equivalent to finding the Lie algebra element b_4. The range of the parameters should be consistent with the image size by scaling the coordinate in sampling. Therefore, we define the rescaled sampling function for the warped image $g_w^{n \times n}$ according to Eq. 6 as $(u_1', u_2')^T = [(\frac{n}{2})^{\frac{\mu_1}{n}} \cos(\frac{2\pi\mu_2}{n}), (\frac{n}{2})^{\frac{\mu_1}{n}} \sin(\frac{2\pi\mu_2}{n})]^T$. Given the warped image with the size of $n \times n$, the warped range is limited by a circle whose radius is $\frac{n}{2}$ in the original image. Finally, $\hat{\theta}$ and $\hat{\gamma}$ are recovered by estimating $(\hat{\theta}, \hat{\gamma}) = (\frac{2\pi\mu_2}{n}, (\frac{n}{2})^{\frac{\mu_1}{n}})$. Fig. 2 (a) shows the example warping functions for the scale and rotation.

Aspect Ratio For group $\mathbf{H}_s$, its corresponding element of Lie algebra is the aspect ratio b_5. Since there is a redundant dimension, we employ the warping function with both directions $k_x \in (0, +\infty)$ and $k_y \in (0, +\infty)$ in order to double-check the parameter k_1 through the supervision on $k_x \to k_1$ and $k_y \to \frac{1}{k_1}$.

$$w_2(k_x, k_y) = \left[s^{k_x'}, s^{k_y'} \right]^T, \tag{7}$$

where $(k_x, k_y) = (s^{k_x'}, s^{k_y'})$. Similarly, $k_1 = (s^{k_x'} = e^{k_x' \log s}) = e^{b_5}$. Estimating k_1 is actually to find the b_5 of $\mathfrak{sl}(3)$. In the image space, the range of parameters should be consistent with the image size by scaling the coordinate for sampling. The rescaled sampling function for scale estimation in

both directions from Eq. 7 can be derived as $\mathbf{u}'^T = [(\frac{n}{2})^{\frac{\mu_1}{n}}, (\frac{n}{2})^{\frac{\mu_2}{n}}]^T$. $\hat{k}_x$ and $\hat{k}_y$ are recovered by $(\hat{k}_x, \hat{k}_y) = ((\frac{n}{2})^{\frac{\hat{\mu}_1}{n}}, (\frac{n}{2})^{\frac{\hat{\mu}_2}{n}})$. Since the main task is usually related to an object, its center is treated as the origin of coordinates. In Eq. 7, $w_2 \in (0, +\infty)^2$. Thus, we overpass the other quadrants when $u_1 < 0$ or $u_2 < 0$. Fig. 2 (b) shows the resulting image. To account for the general case, we flip the other quadrant image to the positive quadrant and upsample it to the original image size of $n \times n$. This changes the size of the warped image $g_w^{n \times n \times 4}$. Fig. 2 (c) shows the example result of g_w. Under the proposed framework, the mapping function performs on the warped image $\mathcal{W}(I, [\hat{t}_1, \hat{t}_2, \hat{\gamma}, \hat{\theta}])$ according to the estimated parameters.

Shear Shear transformation, also known as shear mapping, displaces each point in a fixed direction. According to the following equation on point $\mathbf{u}$, it can be found that shear is caused by the translation of each row in the original image, in which the translation degree is uniformly increased along with the column value. $\mathbf{u}^*$ is the transformed points as below

$$\mathbf{u}^* = \mathbf{H}_{sh} \cdot \mathbf{u}^T = [u_1 + k_2 u_2, \quad u_2]^T \tag{8}$$

Inspired by the fact that the arc length of each concentric circle with the same angle increases by the radius uniformly, it is intuitive to project the lines onto a circle arc so that the sheer can be converted into rotation. Similar to the warping in Eq. 6, the rotation is eventually formulated into the translation estimation. The warping function for shear can be derived as follows:

$$w_3(k_2, k_\epsilon) = [k_2 k_\epsilon, \quad k_\epsilon]^T \tag{9}$$

where k_ϵ is the unchanged coordinate for one-parameter group. In the case of a real image, the rescaled sampling function for shearing from Eq. 9 becomes $\mathbf{u}'^T = \left[\frac{2}{n}(\mu_1 * \mu_2), \quad \mu_2\right]^T$. The estimated $\hat{k}_2$ is recovered by $\hat{k}_2 = \hat{\mu}_1$. Finally, the warping function performs on the warped image $\mathcal{W}(I, [\hat{t}_1, \hat{t}_2, \hat{\gamma}, \hat{\theta}, \hat{k}_1])$ according to the estimated parameters. Fig. 2 (d) shows an example of sheering in the horizontal direction.

Perspective The two elements ν_1 and ν_2 reflect the perspective distortion of an image, which is not the same as the previous one due to view change.

$$\mathbf{u}^* = \mathbf{H}_p \cdot \mathbf{u}^T = \mathbf{H}_{p2}\mathbf{H}_{p1} \cdot \mathbf{u}^T = \left[\frac{u_1}{\nu_1 u_1 + \nu_2 u_2 + 1}, \quad \frac{u_2}{\nu_1 u_1 + \nu_2 u_2 + 1}\right]^T \tag{10}$$

From the action of its group $\mathbf{H}_p$ in Eq. 10, it can be found that the slope of any point does not change after the transformation. ν_1 and ν_2 are entangled in Eq. 10. To make it clear, we design two one-parameter warp functions to account for the perspective changes of two groups $\mathbf{H}_{p1}$ and $\mathbf{H}_{p2}$.

$$w_4(\nu_1, k_\epsilon) = \left[\frac{1}{\nu_1}, \quad \frac{k_\epsilon}{\nu_1}\right]^T, w_5(k_\epsilon, \nu_2) = \left[\frac{k_\epsilon}{\nu_2}, \quad \frac{1}{\nu_2}\right]^T \tag{11}$$

As the same output size is required in sampling, the warp function in Eq. 11 for sampling can be derived as $\mathbf{u}'^T = \left[\frac{n}{2\mu_1}, \quad \frac{\mu_2 n}{2\mu_1}\right]^T$ and $\mathbf{u}'^T = \left[\frac{\mu_1 n}{2\mu_2}, \quad \frac{n}{2\mu_2}\right]^T$. $\hat{\nu}_1$ and $\hat{\nu}_2$ are recovered by $(\hat{\nu}_1, \hat{\nu}_2) = (\hat{\mu}_1, \hat{\mu}_2)$. As depicted in Fig. 2 (f), there exist serious distortions when this sampling function is used. The larger the radius is, the more sparse sampling points are. To tackle this issue, we select the patch near the center of the warped image. For transformation $\mathbf{H}_{p1}$, $w_4 = [\frac{\phi_2 n}{2(\mu_1 + sgn(\mu_1)\phi_1)}, \frac{u_2 n}{2(\mu_1 + sgn(\mu_1\phi_1))}]^T$, where sgn is the signum function. ϕ_1 and ϕ_2 are scaling factors. The above function acts on the warped image $\mathcal{W}(I, [\hat{t}_1, \hat{t}_2, \hat{\gamma}, \hat{\theta}, \hat{k}_1, \hat{k}_2])$ according to the estimated parameters. Fig. 2 (e,g) shows examples of the perspective warped image in two directions.

3.4 IMPLEMENTATION

The proposed WCN is designed to estimate the transformation with regard to the implicit or explicit reference image/object, which depends on the specific task. We give more details of structure in Section 4 for two tasks and more detailed structures can be found in Appendix B. The main idea is to recover the transformation parameters in the Lie subalgebras. To accomplish this goal, we add supervision to these elements through the robust loss function (i.e., smooth L1) defined in (Girshick, 2015). As for the classification problem, we adopt the cross-entropy loss function. Another problem is that there are no training datasets with these parameter labels. We thereby augment the training datasets with the objects' bounding boxes and class labels. The augmentation scheme simulates the estimation process completely. The translation estimator depends on the requirements of the specific tasks. For those tasks without the explicit template, we directly use convolutional layers and linear layers to predict the translation. For the tasks having the template, we simply apply the cross-correlation to predict translation as is widely used in object tracking.
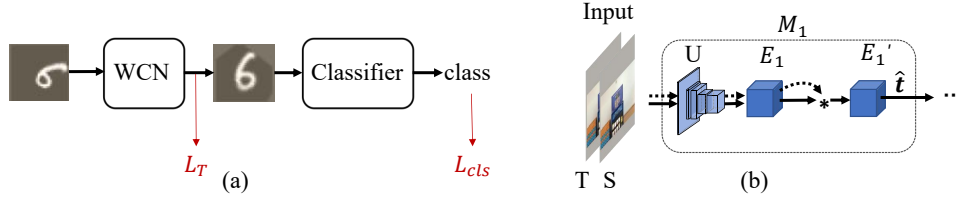
Figure 3: The pipeline of our method for MNIST-Proj digits recognition and planar object tracking tasks. (a): MNIST-Proj digits recognition. $\mathcal{L}_T$: transformation loss, $\mathcal{L}_{cls}$: classification loss. (b): Planar object tracking. This figure only shows changes in a basic component M_i, E_1 and E_1' are the convolution layers. Here, we use ResNet50 for backbone U. T is the template patch, and the dashed line denotes its data flow. Moreover, S is the search patch. and the solid line represents its data flow.

(a) Test digits samples   (b) Recovered digits(STN)   (c) Recovered digits(Ours)
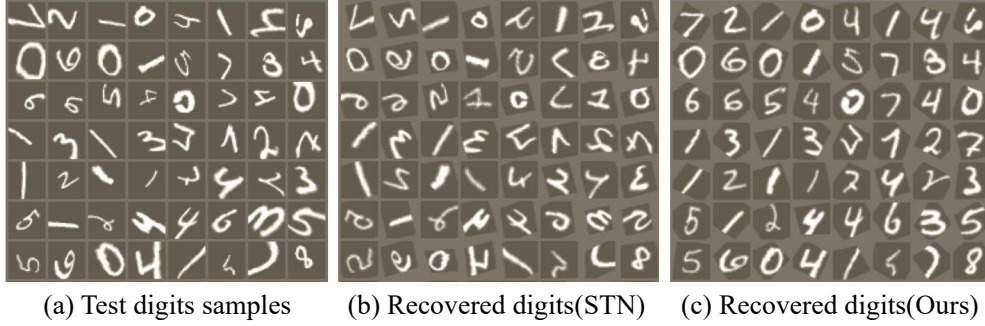
Figure 4: The visual results of the randomly selected MNIST testing after STN and WCN.

4   EXPERIMENTS

To demonstrate the effectiveness of our proposed approach, we evaluate the WCN framework on three different learning tasks including classification, planar object tracking, and homography estimation. All of them need to recover the underlying homography of the object. To this end, we have conducted the experiments on three datasets including MNIST-Proj, POT, and Synthetic COCO.

4.1   CLASSIFICATION ON MNIST-PROJ

MNIST handwriting dataset (LeCun & Cortes, 2005) is a small testbed for digits classification. We perform the experiment on it to show the effectiveness of WCN on the classification tasks. Specifically, we generate the MNIST-Proj dataset by augmenting the data in the training process with projective transformation. The testing dataset has 10,000 digits images and the size of samples is 28×28.

Usually, a homography recovery-based method requires a template reference. For the classification problems, there is no explicit reference object to learn. Inspired by the congealing tasks (Learned-Miller, 2006), we learn an implicit template pose, where the template is the upright digits in MNIST. As shown in Fig. 3, the pipeline consists of two components. We first recover the transformation of the digit, and then employ the classifier to predict its class label. We add the supervision both on estimating the transformation parameters using loss function $\mathcal{L}_T$ and image class with loss function $\mathcal{L}_{cls}$. Thus, the total loss is $\mathcal{L}_T + \lambda\mathcal{L}_{cls}$, where λ is the weight parameter to trade-off two terms.

Table 1: MNIST-Proj results.

Methods	Error (%)	Time(ms)
L-conv (Dehmamy et al. (2021))	19.16 ($\pm$1.84)	1.81
homConv (Macdonald et al. (2021))	14.72 ($\pm$0.72)	105.7
PDO-econv (Shen et al. (2020))	1.66 ($\pm$0.16)	**0.14**
LieConv(Finzi et al. (2020))	2.7 ($\pm$0.74)	\
PTN (Esteves et al. (2018))	2.45($\pm$0.66)	\
STN (Jaderberg et al. (2015))	0.79 ($\pm$0.07)	0.20
WCN (Ours)	**0.69** ($\pm$0.09)	0.42

In Table 1, our proposed framework outperforms six other methods. We use the official implementations from those methods, and all the methods are trained with perspective transform augmentation. Although L-conv (Dehmamy et al., 2021) is built based on Lie algebra, it performs inferior to the

other methods when the group is fixed. Especially, its results become worse when the center of the digits deviates from the image center. homConv (Macdonald et al., 2021) is theoretically equivariant to the $SL(3)$ group, nevertheless, it is not invariant at the feature level. This may lead to the difficulty in identifying the digit class. PDO-econv (Shen et al., 2020) and PTN (Esteves et al., 2018) handle the rotation well, yet we still attain a lower error rate.

4.2 PLANAR OBJECT TRACKING ON POT

POT (Liang et al., 2018) is a mainstream planar object tracking dataset that contains 210 videos of 30 planar objects sampled in the natural environment. We select the videos officially divided with perspective changes as our testing datasets to evaluate the performance in perspective transformation estimation. The state-of-the-art visual object trackers and planar object tracking approaches (Zhan et al., 2022; Zhang et al., 2020b) show that it is easy to predict the offset of an object through the cross-correlation(Bertinetto et al., 2016). Thus, we employ it as parameter estimator in our framework. As shown in Fig 3, there are two inputs for WCN, where the correlation is used to estimate the Lie algebra parameters. The only difference is the perspective ν_1 and ν_2 estimation, since the offset value in the warped image is much smaller than other parameters. We hence directly estimate the pseudo-translation, which shares the higher accuracy.

We choose three methods that directly estimate the Lie algebra elements for comparison. LDES (Li et al., 2019) takes advantage of the log-polar coordinate to estimate the similarity transform. HDN (Zhan et al., 2022) employs a deep network to estimate the similarity parameters, in which

Figure 5: Comparisons on POT. Precision (left), Success rate (right).

a corner offsets estimator is used to refine the corner. Thus, we divide them into two parts to compare the estimation performance fairly. ESM directly estimates 8 Lie algebra elements in images whose generators are different from ours. They compose these elements in one group, which does not share any equivariance in their structure.
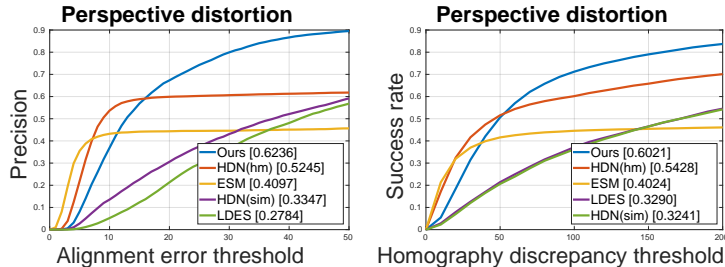
As defined in (Liang et al., 2018), we adopt two metrics for evaluation, including precision and homography success rate. Precision is defined as the percentage of frames whose alignment error is smaller than the given threshold. The alignment error is calculated as the average of four points L_2 distance between the predicted polygon and the ground truth label. Success rate describes the percentage of frames whose homography discrepancy score is less than a threshold. The leaderboard shows the average precision and success rate, which denote the average precision and success rate of all the error thresholds.

The experimental result is shown in Fig. 5. Compared with other trackers, our proposed method achieves a higher average precision and success rate. It has a much higher precision when the error is larger (up to 90%), which indicates the high robustness of our method with 2D perspective transformation. This supports the homography learning in the proposed method. Our estimation is a bit coarse as the proposed framework is not specially designed for the tracking task. More experiments and analyses are provided in Appendix C.

4.3 HOMOGRAPHY ESTIMATION ON S-COCO-PROJ

S-COCO-Proj is a large synthetic homography estimation dataset based on COCO14 (Lin et al., 2014). We augment the COCO14 with middle and large transform augmentation and mask the corners to test homography estimation performance with the middle and large transformation and the occlusion influence (see Appendix A for more details). This is because our method is robust rather than accurate due to several reasons like sampling density and the influence of different parameters as explained in Appendix D. Besides, S-COCO (DeTone et al., 2016) augments the data by moving the corners of the images, which mainly brings the perspective distortion and lacks other transformations like scale and rotation.

Table 2 exhibits the performance when using S-COCO-Proj. We compare our WCN with the recent methods PFNet (Zeng et al., 2018) in S-COCO-Proj. We adopt the standard MACE(Mean Average Corner Error) metric to evaluate the performance. Even with the tracking procedure and inference only once, our proposed approach outperforms the other four methods especially with large transformations. Content-Aware (Zhang et al., 2020a) is an unsupervised method, therefore performs worse with drastic transformations. biHomeE (Koguciuk et al., 2021) adopts the perceptual loss for unsupervised learning. It is still limited by the unchanged predicted parameters when using the previous estimation genre. PFNet (Zeng et al., 2018) and HomographyNet (DeTone et al., 2016) use the offsets of the local points to recover the homography. Therefore, their results are inferior to our WCN when there are occlusions on the corners. Moreover, we demonstrate the high accuracy performance of our proposed WCN as a robust homography representation for the SOTA method. The only difference of the settings is that we remove the mask occlusion (wo.mask) for the testing to show the up-boundary accuracy performance. It can be observed that the performance of Ours+PFNet is significantly better and more accurate for middle and large transformations than using them separately.

4.4 ABLATION STUDY

For fair evaluation, we compare our proposed approach with four baseline methods and compare them with the same backbone. As shown in Table 3, we use the mean error in the last five epochs to measure the performance. When equipped with deeper convolution layers (ResNet18), the CNNs are able to classify the digits well even with large transformations. To fairly compare with STN (Jaderberg et al., 2015), we use the same backbone for classification and achieve a lower error rate. With a five-layer CNN, our proposed approach outperforms STN by 1%. There is no significant gap between WCN and STN using a deeper CNN backbone. This is because there is little space for those hard cases. As shown in Fig. 4, our visual results of the recovered transformations are much better than STN's.

Table 2: S-COCO-Proj comparison.

Methods	MACE	
	Middle Aug.	Large Aug.
Content-Aware (Zhang et al. (2020a))	40.57	56.57
HomographyNet (DeTone et al. (2016))	19.17	35.59
PFNet (Zeng et al. (2018))	11.86	25.30
PFNet+biHomeE (Koguciuk et al. (2021))	12.62	33.12
WCN (Ours)	**10.23**	**17.73**
PFNet (wo.mask)	2.99	16.09
Ours (wo.mask)	7.39	12.82
Ours+PFNet (wo.mask)	1.66	3.21

To evaluate each warp function's contribution and the adaptability for different groups, we conduct the experiment on POT with different warped functions for transformation parameters. The results are shown in Table 4 which means when with more warped functions and parameters included the results are better. Besides, we can combine different warped functions freely.

Table 3: Ablation of backbone on MNIST-Proj.

Methods	Network	Error (%)
Naive	LeNet5*	11.48 ($\pm$1.42)
Navie	ResNet18	0.87 ($\pm$0.13)
STN	LeNet5*	4.00 ($\pm$0.35)
STN	ResNet18	0.79($\pm$0.07)
Ours	LeNet5*	3.05 ($\pm$0.33)
Ours	ResNet18	0.69 ($\pm$0.09)

Table 4: Ablation on warp functions

Transformation parameters	Precision (%)
$[t_1, t_2, \gamma, \theta, k_1, k_2, \nu_1, \nu_2]$	62.4%
$[t_1, t_2, \gamma, \theta, k_1, k_2]$	49.5%
$[t_1, t_2, \gamma, \theta, k_1]$	39.8%
$[t_1, t_2, \gamma, \theta]$	33.1%
$[t_1, t_2, \gamma]$	18.8%
$[t_1, t_2]$	13.6%

5 CONCLUSION

In this paper, we proposed Warped Convolution Networks (WCN) to effectively learn the homography by $SL(3)$ group and $\mathfrak{sl}(3)$ algebra with group convolution. Based on the warped convolution, our proposed WCN extended the capability of handling noncommutative groups and achieved to some extent equivariance. To this end, six commutative subgroups within the $SL(3)$ group along with their warping functions were composed to form a homography. By warping the corresponding space and coordinates, the group convolution was accomplished in a very efficient way. Extensive experiments showed that our proposed method is effective for representation learning. The proposed approach achieved the highest performance for both estimation task and classification problem compared to the direct methods on POT and STN on MNIST-Proj.

REPRODUCIBILITY STATEMENT

To ensure the reproducibility, We make efforts on several aspects and conclude here. To keep the generality, we do not specify the concrete structure of WCN. However, we do explain some details of it in Section 3.4. More details of architecture and implementation for three different tasks can be found in Appendix B. More importantly, we offer the source codes for MNIST-Proj classification and POT tracking problems in the supplemental materials. The training details and experimental setting are introduced in Appendix B.1 and C.1, including how to create the data for our experiment. Step-by-step instructions to reproduce the results are given in the "readme.md" file. For reproducibility, we set the random seed for homography estimation, and for MNIST-Proj testing results, we repeatedly test five times and give the error scope.

REFERENCES

Selim Benhimane and E. Malis. Real-time image-based tracking of planes using efficient second-order minimization. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1:943–948 vol.1, 2004.

Gregory Benton, Marc Finzi, Pavel Izmailov, and Andrew Gordon Wilson. Learning invariances in neural networks. *arXiv preprint arXiv:2010.11882*, 2020.

Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 850–865, 2016.

S. Taco Cohen and Max Welling. Transformation properties of learned visual representations. *International Conference on Learning Representations (ICLR)*, 2015.

T. Cohen and M. Welling. Group equivariant convolutional networks. In *Conference on International Conference on Machine Learning (ICML)*, 2016.

Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: Object detection via region-based fully convolutional networks. *Advances in neural information processing systems*, 29, 2016.

Nima Dehmamy, Robin Walters, Yanchen Liu, Dashun Wang, and Rose Yu. Automatic symmetry discovery with lie algebra convolutional network. In *NeurIPS*, 2021.

Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Deep image homography estimation, 2016.

Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. Polar transformer networks. In *International Conference on Learning Representations (ICLR)*, 2018.

Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. *arXiv preprint arXiv:2002.12880*, 2020.

Ross B. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pp. 1440–1448, 2015.

Andrew Harltey and Andrew Zisserman. *Multiple view geometry in computer vision*, chapter 2, pp. 42. Cambridge University Press, 2 edition, 2003.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

João F. Henriques and A. Vedaldi. Warped convolutions: Efficient invariance to spatial transformations. In *Conference on International Conference on Machine Learning (ICML)*, pp. 1461–1469, 2017.

Geoffrey E. Hinton, Alex Krizhevsky, and Sida Wang. Transforming auto-encoders. In *ICANN*, 2011.

Lianghua Huang, Xin Zhao, and Kaiqi Huang. GOT-10k: A large high-diversity benchmark for ggeneric object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. doi: 10.1109/tpami.2019.2957464.

Max Jaderberg, K. Simonyan, Andrew Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *Proceedings of the Neural Information Processing Systems (NeurIPS)*, pp. 6992–7003, 2015.

Nathalie Japkowicz, Farzan Erlik Nowruzi, and Robert Laganière. Homography estimation from image pairs with hierarchical convolutional networks. *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pp. 904–911, 2017.

Ziwei Ji and Matus Telgarsky. Directional convergence and alignment in deep learning. *ArXiv*, abs/2006.06657, 2020.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

Daniel Koguciuk, E. Arani, and Bahram Zonooz. Perceptual loss for robust unsupervised homography estimation. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 4269–4278, 2021.

Erik G. Learned-Miller. Data driven image models through continuous joint alignment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28:236–250, 2006.

Yann LeCun and Corinna Cortes. The mnist database of handwritten digits. 2005.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proc. IEEE*, 86:2278–2324, 1998.

Yang Li, Jianke Zhu, Steven C.H. Hoi, Wenjie Song, Zhefeng Wang, and Hantang Liu. Robust estimation of similarity transformation for visual object tracking. *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 33:8666–8673, 2019. ISSN 2374-3468. doi: 10.1609/aaai.v33i01.33018666.

Pengpeng Liang, Y. Wu, and H. Ling. Planar object tracking in the wild: A benchmark. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 651–658, 2018.

Feng Lin, Haohang Xu, Houqiang Li, Hongkai Xiong, and Guo-Jun Qi. Auto-encoding transformations in reparameterized lie groups for unsupervised learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2021.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 740–755, 2014.

Lachlan Macdonald, Sameera Ramasinghe, and Simon Lucey. Enabling equivariance for arbitrary lie groups. *ArXiv*, abs/2111.08251, 2021.

Ty Nguyen, Steven W. Chen, S. S. Shivakumar, C. J. Taylor, and Vijay Kumar. Unsupervised deep homography: A fast and robust homography estimation model. *IEEE Robotics and Automation Letters*, 3:2346–2353, 2018.

Zhengyang Shen, Lingshen He, Zhouchen Lin, and Jinwen Ma. Pdo-econvs: Partial differential operator based equivariant convolutions. In *ICML*, 2020.

Ivan Sosnovik, Michal Szmaja, and Arnold W. M. Smeulders. Scale-equivariant steerable networks. *ArXiv*, abs/1910.11093, 2020.

Paul Voigtlaender, Jonathon Luiten, Philip H. S. Torr, and B. Leibe. Siam r-cnn: Visual tracking by re-detection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6577–6587, 2020.

Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In *Proceedings of the Neural Information Processing Systems (NeurIPS)*, pp. 10381–10392, 2018.

Chengxi Ye, Xiong Zhou, Tristan McKinney, Yanfeng Liu, Qinggang Zhou, and Fedor Zhdanov. Exploiting invariance in training deep neural networks. *ArXiv*, abs/2103.16634, 2021.

Rui Zeng, Simon Denman, Sridha Sridharan, and Clinton Fookes. Rethinking planar homography estimation using perspective fields. In *ACCV*, 2018.

Xinrui Zhan, Yue Liu, Jianke Zhu, and Yang Li. Homography decomposition networks for planar object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume abs/2112.07909, 2022.

Jirong Zhang, Chuan Wang, Shuaicheng Liu, Lanpeng Jia, Jue Wang, and Ji Zhou. Content-aware unsupervised deep homography estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 653–669, 2020a.

Richard Zhang. Making convolutional networks shift-invariant again. *ArXiv*, abs/1904.11486, 2019.

Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 771–787, 2020b.

Bolei Zhou, Àgata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.

Philip E. Zwicke and Imre Kiss. A new implementation of the mellin transform and its application to radar classification of ships. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5:191–199, 1983.

Appendix

In this appendix, we first discuss the details of the warp function and analyze the influence of each parameter on the warped image. Then, the implementation details of our proposed method are provided, and more experimental details are introduced with additional results. Finally, we provide the proof of warping function property.

A WARP FUNCTIONS

For the warped convolution (Henriques & Vedaldi, 2017), the most ideal situation is that the group has commutative property with only two parameters. In this case, all the parameters are independent and the group convolution can be implemented as a warped function. However, it is impossible for both the affine group and projective group to have the same properties since they are not Abelian groups with more parameters. A transformation matrix can be represented as follows

$$\mathbf{H} = \exp(\mathbf{A}(\mathbf{b})) = \exp(\sum_{i=1}^{8} b_i \mathbf{A}_i), \tag{12}$$

where $\mathbf{A}_i$ is the generator of the Lie algebra $\mathbf{A}$. b_i is an element of the generator coefficients vector $\mathbf{b}$ in the real plane. For affine and projective group, it does not hold that $e^{\mathbf{A}(\mathbf{b})} e^{\mathbf{A}(\mathbf{a})} = e^{\mathbf{A}(\mathbf{a}+\mathbf{b})}$. Intuitively, this means the element of the Lie algebra loses its meaning in the image transform, while the corresponding one-parameter subalgebra still holds the property, e.g. rotation and scale. Besides, it does not satisfy the condition for Eq. (6) in the main paper. Therefore, no warping function can be found for both affine and projective groups directly.

It is worthy of discussing why not map the projective transformation onto 3D space to estimate the 6 independent parameters. The reason is that one cannot project the image into a particular camera view without the depth and camera intrinsic. Therefore, there is no way to warp the image like the log-polar coordinates for in-plane rotation (Esteves et al., 2018). In section 3.1 of the main paper, we follow the warped convolution (Henriques & Vedaldi, 2017) and decompose the homography into 6 subgroups that can be predicted independently by pseudo-translation estimation according to the equivariance.

However, there is little difference in estimating the two-dimension subgroup of $\mathbf{b}$ and predicting the translation. Given the object center as the origin, all transformations generated by the parameters of $\mathfrak{sl}(3)$ do not change the object's center. Unfortunately, this property does not hold for the warped image. The transformation of $\mathbf{x}$ in the warped image is different from the transformation in the original image. To analyze the influence of each parameter on the warping function, we draw the center offset of the warped image. One argument is the parameter η_1 of the warping function, and the other argument is the other parameter η_2 may influence the translation. Fig. 6 shows the example for warping function w_1. Fig. 6 (a,b,c,d) demonstrate the $k'_x, k'_2, \nu'_1, \nu'_2$ effect on the warped image center offsets compared with θ about warp function w_1. Fig. 6 (e,f,g,h) depict the $k'_x, k'_2, \nu'_1, \nu'_2$ effect on the warped image center in contrast to γ' for warp function w_1. We find the parameters of w_1 dominate the translation of the warped image center. This means that we can estimate γ' and θ in the warped image g_{w1} even with other existing transformations. The same conclusion is valid for other warp functions.

B IMPLEMENTATION DETAILS

We design two separate architectures with our proposed WCN for classification and planar object tracking, respectively. In this section, we introduce the implementation details of the two tasks with two datasets MNIST-Proj and POT accordingly.

Classification Two backbone networks are used for the classification task in MNIST-Proj. The first one is a modified LeNet-5 LeCun et al. (1998). As described in Table. 5, the localization stage is used to predict the pseudo-translation of the handwritten digits on a warped image compared to the implicit upright digits. Then, we resample the image according to the parameters and concatenate it with the original image as the input for the classification stage. To further examine the capability
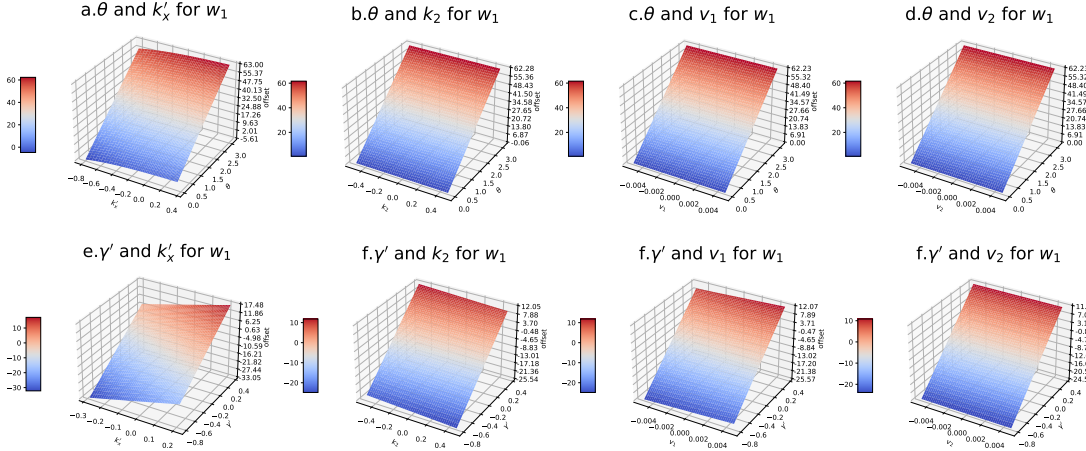
Figure 6: Analysis of the parameter influence to warping function. We use the offsets (pixels) to measure the influence of x' and other parameters.

of our method in the classification task, we implement another classifier to demonstrate the results with a deeper backbone ResNet-18 He et al. (2016). As listed in Table 6, we first use the ResNet-18 to extract the feature, then estimate the transformation parameters $\mathbf{x}'$ with several warp functions. Its localization network is the same as the Localization stage in Table 5 except that the input size is different. According to the estimated $\hat{\mathbf{x}}'$, the resampled image, and the original image are concatenated as the input of another Resnet-18 that uses a two-layer classifier to predict the class of the digits.

Planar Object Tracking   For the planar object tracking, we treat HDN (Zhan et al., 2022) as our baseline method, which has two warp functions to predict $\mathbf{b}$ in addition to $[t_1, t_2, \gamma, \theta]$. Besides, the perspective changes are small in the feature map. We thereby estimate p_1 and p_2 directly on the warped image according to the w_4 rather than using the correlation. The structure is similar to the homography estimator in HDN, yet we only estimate ν_1 and ν_2 directly.

Homography Estimation   For the homography estimation task, we simply apply the same tracking procedure for estimation. In the testing dataset of SCOCO, transformation is conducted by Eq. (8) in the paper. For middle augmentation, we set $\theta \in [0.6\text{rad}, 0.6\text{rad}]$, $\gamma \in [0.7, 1.3]$, $k_1 \in [-0.2, 0.2]$, $k_2 \in [-0.15, 0.15]$, $\nu_1 \in [-0.0001, 0.0001]$, and $\nu_2 \in [-0.0001, 0.0001]$. Large augmentation is with $\theta \in [0.8\text{rad}, 0.8\text{rad}]$, $\gamma \in [0.7, 1.3]$, $k_1 \in [-0.3, 0.3]$, $k_2 \in [-0.2, 0.2]$, $\nu_1 \in [-0.001, 0.001]$, and $\nu_2 \in [-0.001, 0.001]$.

B.1   TRAINING

Existing datasets lack the annotations of transformation parameters. Even with those provided, they need to be converted to $\mathbf{b}$ with matrix decomposition, which is not easy. Therefore, we augment the possible transformations according to Eq. (8) in the main paper and transform the images as the training data with the randomly sampled parameters. We augment the dataset MNIST LeCun & Cortes (2005) for classification and GOT-10K (Huang et al., 2019) and COCO-14 (Lin et al., 2014) for planar object tracking during the training, respectively.

For MINST, the model is trained firstly with the supervision on the predicted $\hat{\mathbf{x}}$ and predicted class for $N_e = 100$ epochs, which is retrained with only classification loss for N_e in MNIST-Proj. For transformation loss, $\mathcal{L}_T = \mathcal{L}_{sc} + \lambda_1(\mathcal{L}_t + \mathcal{L}_{k_1} + \mathcal{L}_{k_2} + \mathcal{L}_\nu)$, where $\mathcal{L}_{sc}$ is the loss of (γ, θ). $\mathcal{L}_t$ is the loss of translation, and $\mathcal{L}_{k_1}$ is the loss of k_x and k_y. $\mathcal{L}_{k_2}$ is the loss of k_2, and $\mathcal{L}_\nu$ is the loss of $\nu = (\nu_1, \nu_2)$. All transformation penalties make use of the robust loss function (i.e., smooth L1) defined in Girshick (2015). As for POT, we employ the same classification and offset loss as HDN (Zhan et al., 2022) for the newly added parameters in $\mathbf{x}$.

For MNIST-Proj classification task, we adopt Adam (Kingma & Ba, 2014) as the optimizer, where the batch size is set to 128. The learning rate starts from 0.001 and decays by a multiplicative factor

of 0.95 with an exponential learning scheduler. Similar to HDN, we trained the whole network for planar object tracking on GOT-10k (Huang et al., 2019) and COCO14 (Lin et al., 2014) for 30 epochs with 1 epoch warming up. The batch size is set to 28×4. Our model is trained in an end-to-end manner for 18 hours in our experimental settings.

For S-COCO-Proj homography estimation, we use the same training and testing procedure as in the POT, except that we remove the GOT-10k (Huang et al., 2019) from the training datasets. All the methods in the leaderboard are trained with the same augmented dataset with middle augmentation and mask the corner area with a circle mask with the radius of 60 pixels.

Table 5: Network details for MNIST-Proj. Conv (1,8,7) denotes a convolution layer with input channel=1, output channel=8, kernel size=7. MaxPool (2,2) represents the max-pooling layer with window=2, and stride=2. Linear (90,32) represents the fully connected layer with input size=90 and output size=32. C is the total number of channels for the output.

Stages	Operator	Output
Localization	Conv2d (1,8,7)	$C \times 8 \times 22 \times 22$
	MaxPool (2,2), ReLU	$C \times 8 \times 11 \times 11$
	Conv2d (8,10,5)	$C \times 10 \times 7 \times 7$
	MaxPool (2,2), ReLU	$C \times 10 \times 3 \times 3$
	Linear (90,32), ReLU	$C \times 32$
	Linear (32,2)	$C \times 2$
Classification	Conv2d (2,10,5)	$C \times 10 \times 24 \times 24$
	MaxPool (2,2), ReLU	$C \times 10 \times 12 \times 12$
	Conv2d (10,20,5)	$C \times 20 \times 8 \times 8$
	MaxPool (2,2), ReLU, Dropout	$C \times 20 \times 4 \times 4$
	Linear (90,32), ReLU	$C \times 50$
	Linear (50,10)	$C \times 10$

Table 6: Classification Networks details using ResNet18

Stages	Operator	Output
Classification	Linear (3136,128), Norm,ReLU	$C \times 128$
	Linear (128,10), LogSoftmax	$C \times 10$

C EXPERIMENTS

C.1 EXPERIMENTAL SETUP

We conducted all experiments on a PC with an intel E5-2678-v3 processor (2.5GHz), 32GB RAM, and an Nvidia GTX 2080Ti GPU. The proposed method is implemented in Pytorch.

For MNIST-Proj, the size of the input image is 28×28. For the hyperparameters of WCN in training, we set $\lambda = 2$, $\lambda_1 = 20$, $\gamma \in [1/1.4, 1.4]$, $\theta \in [-1.5, 1.5]$, $t \in [-28/8, 28/8]$, $k_1 \in [-1.3, 1.3]$, $k_2 \in [-0.02, 0.02]$, and $\nu \in [-0.02, 0.02]$. Due to the numbers 6 and 9 being identical with the rotation even from humans, we remove the number 9 from MNIST-Proj.

For POT, the size of input template T for our networks is 127×127, while search image I has the size of 255×255 to deal with the large homography changes. All the hyper-parameters are set empirically, and we do not use any re-initialization and failure detection scheme. For the hyper-parameters of HDN in training, we set $\gamma \in [1/1.38, 1.38]$, $\theta \in [-0.7, 0.7]$, $t \in [-32, 32]$, $k_1 \in [-0.1, 0.1]$, $k_2 \in [-0.015, 0.015]$ and $\nu \in [-0.0015, 0.0015]$.

Many parameters may influence the experimental results. We investigate the influence of the sampling circle radius. We fix it to be $n/2$, which is the half length of the side in the default setting. Theoretically, it is enough as long as the field covers the region of the original image. Actually, we tested 5 different radius $(0.5 \times n/2, 0.75 \times n/2, n/2, 1.25 \times n/2, 1.5 \times n/2)$ The results are quite similar, and the errors are within 0.1%.
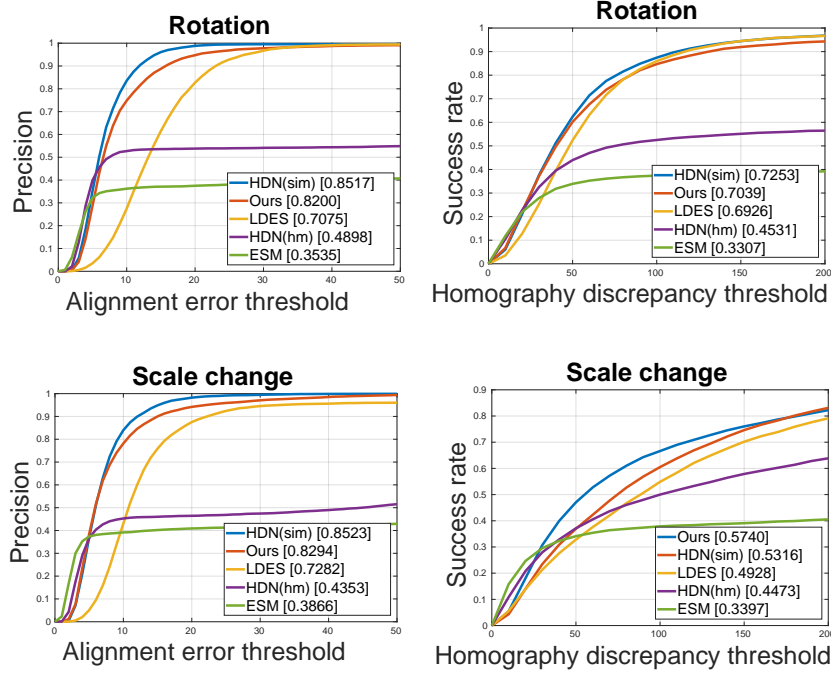
Figure 7: Results on rotation and scale sequences of POT.

C.2 MORE RESULTS ON POT

Apart from the perspective distortion in the main paper, we provide more results on other simple transformations, e.g. rotation and scale changes. Fig. 7 shows the precision and success rate of these two transformations. HDN (Zhan et al., 2022) has a similar structure for similarity estimation with two same warp function, which is similar to our presented WCN. Thereby, we have similar results on rotation and scale sequences. Our performance is inferior to HDN (Zhan et al., 2022) when the error threshold is small. When there are either rotation or scale changes for the object, our WCN estimates all eight parameters rather than similarity transformation compared to HDN. This may bring more estimation errors because the estimation is not accurate as explained in Sec. A. Furthermore, HDN uses more training data than WCN does. When the error threshold is large, their precision and success rate are close.

D LIMITATION AND FUTURE WORK

Whereas our proposed method is theoretically equivariant to the mentioned several groups, it is hard to theoretically prove why CNNs can robustly estimate a series of transformation parameters sequentially. Besides, a known problem of WCN is that the estimated offsets may be inaccurate due to the influence of other parameters in addition to $\mathbf{x}'$ and the small feature map size and error produced in different M_i. The warped image may thereby bring the error of the previously predicted parameters. Moreover, the sampling density also affects the estimation accuracy. Although no additional parameters will be added to the networks, the number of resampling times is the same as the number of inference times through the networks. It may be time-consuming if there are too many warping functions in a group. In our experiments, the proposed method is not robust to the challenging scenarios like partial occlusions and heavy blur, which can be solved by predicting either an extra occlusion map or a blur kernel. Furthermore, more applications such as AR, SLAM, recognition, congealing, and image stabilization can be benefited from our proposed method.

16

E PROOFS

We define the warp function w for different elements in $\mathbf{x}$, and let $\mathbf{b}'$ be the elements in $\mathbf{b}$ with regard to warp function $w(\mathbf{x}')$. Although the coordinates of the warped image are proportion to $\mathbf{b}'$, we still need to prove that the group action results on the sampled points $\mathbf{u}'$ in the original image are addictive about $\mathbf{b}'$. That is,

$$\mathbf{H}(\mathbf{x}'(\Delta \mathbf{b}')) \cdot \mathbf{u}'^T = \mathbf{H}(\mathbf{x}'(\Delta \mathbf{b}')) \cdot w(\mathbf{x}'(\mathbf{b}')) = w(\mathbf{x}'(\mathbf{b}' + \Delta \mathbf{b}')) \tag{13}$$

where $\mathbf{H}$ can be viewed as a function of $\mathbf{x}'$, and $\mathbf{x}'$ can be viewed as a function of $\mathbf{b}'$. $\boldsymbol{\Delta}\mathbf{b}'$ is the incremental value of $\mathbf{b}'$. Therefore, the warped function satisfies Eq. (6) in the main paper, which makes the convolution equivariant to $\mathbf{b}'$.

Scale and Rotation
As introduced in the main paper, the warp function for scale and rotation is :

$$w_1(\gamma, \theta) = (u_1', u_2')^T = \begin{bmatrix} s^{\gamma'} \cos(\theta) \\ s^{\gamma'} \sin(\theta) \end{bmatrix} \tag{14}$$

We have defined $\gamma = s^{\gamma'} = e^{b_4}$ and $\theta = b_3$, Therefore, the left of Eq. 13 can be rewritten as below:

$$\mathbf{H}_s(\mathbf{x}'((\Delta b_3, \Delta b_4))) \cdot \mathbf{u}'^T = \mathbf{H}_s \cdot w_1 = \begin{bmatrix} e^{\Delta b_4} \cos(\Delta b_3) u_1' - e^{\Delta b_4} \sin(\Delta b_3) u_2' \\ e^{\Delta b_4} \sin(\Delta b_3) u_1' + e^{\Delta b_4} \cos(\Delta b_3) u_2' \end{bmatrix} \tag{15}$$

$$= \begin{bmatrix} e^{b_4 + \Delta b_4} \cos(b_3 + \Delta b_3) \\ e^{b_4 + \Delta b_4} \sin(b_3 + \Delta b_3) \end{bmatrix} \tag{16}$$

As a result, the warp function w_1 supports the equivariance.

Aspect Ratio
The warp function for aspect ratio is defined as follows:

$$w_2(k_x, k_y) = \begin{bmatrix} s^{k_x'}, s^{k_y'} \end{bmatrix}^T, \tag{17}$$

For $\mathbf{H}_{sc}$, there is only one element. We thereby constrain $e^{b_5} = k_1 = k_x = 1/k_y = s^{k_x'}$. As a result, the left of Eq. 13 can be rewritten as below

$$\mathbf{H}_{sc}(\mathbf{x}'((\Delta k_x', \Delta k_y'))) \cdot \mathbf{u}'^T = \mathbf{H}_{sc} \cdot w_2 = \begin{bmatrix} e^{\Delta b_5} \cdot u_1' \\ e^{-\Delta b_5} \cdot u_2' \end{bmatrix} = \begin{bmatrix} e^{\Delta b_5 + b_5} \\ e^{-(\Delta b_5 + b_5)} \end{bmatrix} \tag{18}$$

Hence, we can prove the equivariance holds with the warp function w_2.

Shear
The warp function for Shear is defined as below:

$$w_3(k_2, k_\epsilon) = [k_2 k_\epsilon, \quad k_\epsilon]^T \tag{19}$$

We have defined the $k_2 = b_6$, The left of Eq. 13 can be rewritten as:

$$\mathbf{H}_{sh}(\mathbf{x}'((\Delta b_6, k_\epsilon))) \cdot \mathbf{u}'^T = H_{sh} \cdot w_3 = \begin{bmatrix} u_1' + \Delta b_6 u_2' \\ u_2' \end{bmatrix} = \begin{bmatrix} (b_6 + \Delta b_6) k_\epsilon \\ k_\epsilon \end{bmatrix} \tag{20}$$

Hence, the equivariance is tenable for $\mathbf{H}_{sh}$ with warp function w_3.

Perspective
The warp function for perspective can be derived as below:

$$w_4(\nu_1, k_\epsilon) = \begin{bmatrix} \frac{1}{\nu_1}, & \frac{k_\epsilon}{\nu_1} \end{bmatrix}^T, w_5(k_\epsilon, \nu_2) = \begin{bmatrix} \frac{k_\epsilon}{\nu_2}, & \frac{1}{\nu_2} \end{bmatrix}^T \tag{21}$$

We have defined the $\nu_1 = b_7$ and $\nu_2 = b_8$. Thus, the left of Eq. 13 with regard to ν_1 or ν_2 is rewritten as follows:

$$\mathbf{H}_{p1}(\mathbf{x}'((\Delta b_7, k_\epsilon))) \cdot \mathbf{u}'^T = \mathbf{H}_{p1} \cdot w_4 = \begin{bmatrix} \frac{u_1'}{u_1' \Delta b_7 + 1} \\ \frac{u_2'}{u_1' \Delta b_7 + 1} \end{bmatrix} = \begin{bmatrix} \frac{1}{\Delta b_7 + b_7} \\ \frac{k_\epsilon}{\Delta b_7 + b_7} \end{bmatrix} \tag{22}$$

$$\mathbf{H}_{p2}(\mathbf{x}'((k_\epsilon, \Delta b_8))) \cdot \mathbf{u}'^T = \mathbf{H}_{p2} \cdot w_5 = \begin{bmatrix} \frac{u_1'}{u_2' \Delta b_8 + 1} \\ \frac{u_2'}{u_2' \Delta b_8 + 1} \end{bmatrix} = \begin{bmatrix} \frac{k_\epsilon}{\Delta b_8 + b_8} \\ \frac{1}{\Delta b_8 + b_8} \end{bmatrix} \tag{23}$$

Hence, the equivariance holds for two perspective groups with warp function w_4 and w_5, respectively.