Scale Equivariance Improves Siamese Tracking
Supplementary Material

Ivan Sosnovik* Artem Moskalev * Arnold Smeulders
UvA-Bosch Delta Lab
University of Amsterdam, Netherlands
{i.sosnovik, a.moskalev, a.w.m.smeulders}@uva.nl

1. Proofs

1.1. Convolution is all you need

In the paper we consider trackers of the following form

$$h(z, x) = \phi_X(x) \star \phi_Z(z) \tag{1}$$

where ϕ_X and ϕ_Z are parameterized with feed-forward neural networks.

Theorem 1. *A function given by Equation 1 is equivariant under a transformation L from group G if and only if ϕ_X and ϕ_Z are constructed from G-equivariant convolutional layers and $\star$ is the G-convolution.*

Proof. Let us fix $z = z_0$ and introduce a function $h_X = h(x, z_0) = \phi_X(x) \star \phi_Z(z_0)$. This function is a feed-forward neural network. All its layers but the last one are contained in ϕ_X and the last layer is a convolution with $\phi_Z(z_0)$. According to [2] a feed-forward neural network is equivariant under transformations from G if and only if it is constructed from G-equivariant convolutional layers. Thus, the function h_X is equivariant under transformations from G if and only if

- The function ϕ_X is constructed from G-equivariant convolutional layers
- The convolution $\star$ is the G-convolution

If we then fix $x = x_0$, we can show that a function $h_Z = h(x_0, z) = \phi_X(x_0) \star \phi_Z(z)$ is equivariant under transformations from G if and only if

- The function ϕ_Z is constructed from G-equivariant convolutional layers
- The convolution $\star$ is the G-convolution

The function h is equivariant under G if and only if both the function h_X and the function h_Z are equivariant. □

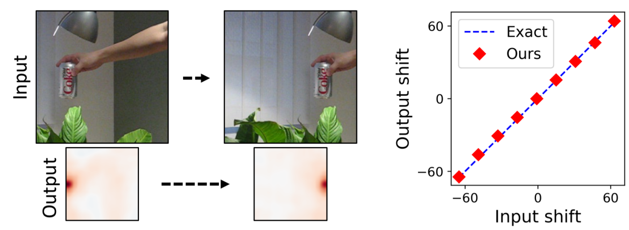

Figure 1: Left: two samples from the simulated sequence. The input image is a translated and cropped version of the source image. The output is the heatmap produced by the proposed model. The red color represents the place where the object is detected. Right: correspondence between the input and the output shifts.

1.2. Non-parametric scale-convolution

Given two functions f_1, f_2 of scale and translation the non-paramteric scale convolution is defined as follows:

$$[f_1 \star_H f_2](s, t) = L_{s^{-1}}[L_s[f_1] \star f_2](t) \tag{2}$$

Lemma 1. *A function given by Equation 2 is equivariant under scale-translation.*

Proof. A function given by Equation 2 is equivariant under scale transformations of f_1, indeed

$$\begin{aligned} [L_{\hat{s}}[f_1] \star_H f_2](s, t) &= L_{s^{-1}}[L_{s\hat{s}}[f_1] \star f_2](t) \\ &= L_{\hat{s}} L_{(s\hat{s})^{-1}}[L_{s\hat{s}}[f_1] \star f_2](t) \\ &= L_{\hat{s}}[f_1 \star_H f_2](s\hat{s}, t) \end{aligned} \tag{3}$$

For a pair of scale and translation $s, \hat{t}$ we have the following property of the joint transformation $L_s T_{\hat{t}} = T_{\hat{t}s} L_s$ from [3], where $T_{\hat{t}}$ is the translation operator defined as

*equal contribution

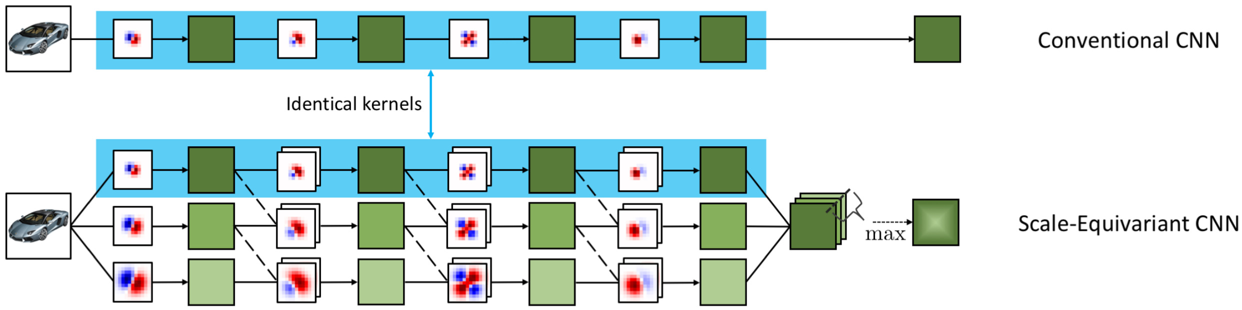

Figure 2: The visualization of the weight initialization scheme from a pretrained model. Dashed connections are initialized with 0.

$T_{\hat{t}}[f](t) = f(t - \hat{t})$. Now we can show the following:

$$\begin{aligned}
[T_{\hat{t}}[f_1] \star_H f_2](s,t) &= L_{s^{-1}}[L_s[T_{\hat{t}}[f_1]] \star f_2](t) \\
&= L_{s^{-1}}[T_{\hat{t}s} L_s[f_1] \star f_2](t) \\
&= L_{s^{-1}} T_{\hat{t}s}[L_s[f_1] \star f_2](t) \\
&= T_{\hat{t}} L_{s^{-1}}[L_s[f_1] \star f_2](t) \\
&= T_{\hat{t}}[f_1 \star_H f_2](t)
\end{aligned} \tag{4}$$

Therefore, a function given by Equation 2 is also equivariant under translations of f_1. The equivariance of the function with respect to a joint transformation follows from the equivariance to each of the transformations separately [3].

We proved the equivariance with respect to f_1. The proof with respect to f_2 is analogous. $\square$

2. Weight initialization

The proposed weight initialization scheme from a pretrained model is depicted in Figure 2.

3. Experiments

3.1. Padding

We conduct an experiment to verify that the proposed padding technique does not violate translation equivariance of convolutional trackers. We choose an image and select a sequence of translated and cropped windows inside of it. We process this sequence with a deep model that consists of the proposed convolutional layers and follows the inference procedure described in [4]. We derive the predicted location of the object and compare its value to the input shift. Figure 1 demonstrates that the input and the output translations have nearly identical values.

3.2. Translating-Scaling MNIST

For both T-MNIST and S-MNIST, we use architectures described in Table 1. 2D BatchNorm and ReLU are inserted after each of the convolutional layers except the last one.

Stage	SiamFC	SE-SiamFC
Conv1	$[3 \times 3, 96, s = 2]$	
Conv2	$[3 \times 3, 128, s = 2]$	
Conv3	$[3 \times 3, 256, s = 2]$	
Conv4	$[3 \times 3, 256, s = 1]$	
Connect.	Cross-correlation	Non-parametric scale-convolution
# Params	999 K	999 K

Table 1: Architectures used in T/S-MNIST experiment. All convolutions in SE-SiamFC are scale-convolutions.

We do not use max pooling to preserve strict translation-equivariance.

We train both models for 50 epochs using SGD with a mini-batch of 8 images and exponentially decay the learning rate from 10^{-2} to 10^{-5}. We set the momentum to 0.9 and the weight decay to 0.5^{-4}. A binary cross-entropy loss as in [1] is used. The inference algorithm is the same for both SiamFC and SE-SiamFC and follows the original implementation [1].

3.3. OTB and VOT

For OTB and VOT experiments we used architectures described in Table 2. We use the baseline [4] with Cropping Inside Residual (CIR) units. SE-SiamFC is constructed directly from the baseline as described in the paper. In Table 2 the kernel size refers to the smallest scale $\sigma = 1$ in the network. The sizes of the kernels, which correspond to bigger scales are 9×9 for Conv1 and 5×5 for other layers. Figure 3 gives a qualitative comparison of the proposed method and the baseline.

<table>
<tr><th>Stage</th><th>SiamFC+</th><th>SE-SiamFC</th></tr>
<tr><td>Conv1</td><td>$\left[7 \times 7, 64, s = 2\right]$</td><td>$\left[7 \times 7, 64, s = 2\right]$</td></tr>
<tr><td rowspan="2">Conv2</td><td colspan="2">max pool $\left[2 \times 2, s = 2\right]$</td></tr>
<tr><td>$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$</td><td>$\begin{bmatrix} 1 \times 1, 64, i = 2 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$</td></tr>
<tr><td>Conv3</td><td>$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 3$</td><td>$\begin{bmatrix} 1 \times 1, 128, sp \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 3$</td></tr>
<tr><td>Connect.</td><td>Cross-correlation</td><td>Non-parametric scale-convolution</td></tr>
<tr><td># Params</td><td>1.44 M</td><td>1.45 M</td></tr>
</table>

Table 2: Architectures used in OTB/VOT experiments. All convolutions in SE-SiamFC are scale-convolutions. *s* refers to stride, *sp* denotes scale pooling, *i* — is the size of the kernel in a scale dimension.

Acknowledgments

We thank Thomas Andy Keller, Konrad Groh, Zenglin Shi and Deepak Gupta for valuable comments, discussions and help with the project. We appreciate the help of Zhipeng Zhang and Houwen Peng in reproducing the experiments from [4].

References

[1] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016.

[2] Risi Kondor and Shubhendu Trivedi. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *arXiv preprint arXiv:1802.03690*, 2018.

[3] Ivan Sosnovik, Michał Szmaja, and Arnold Smeulders. Scale-equivariant steerable networks. *arXiv preprint arXiv:1910.11093*, 2019.

[4] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4591–4600, 2019.

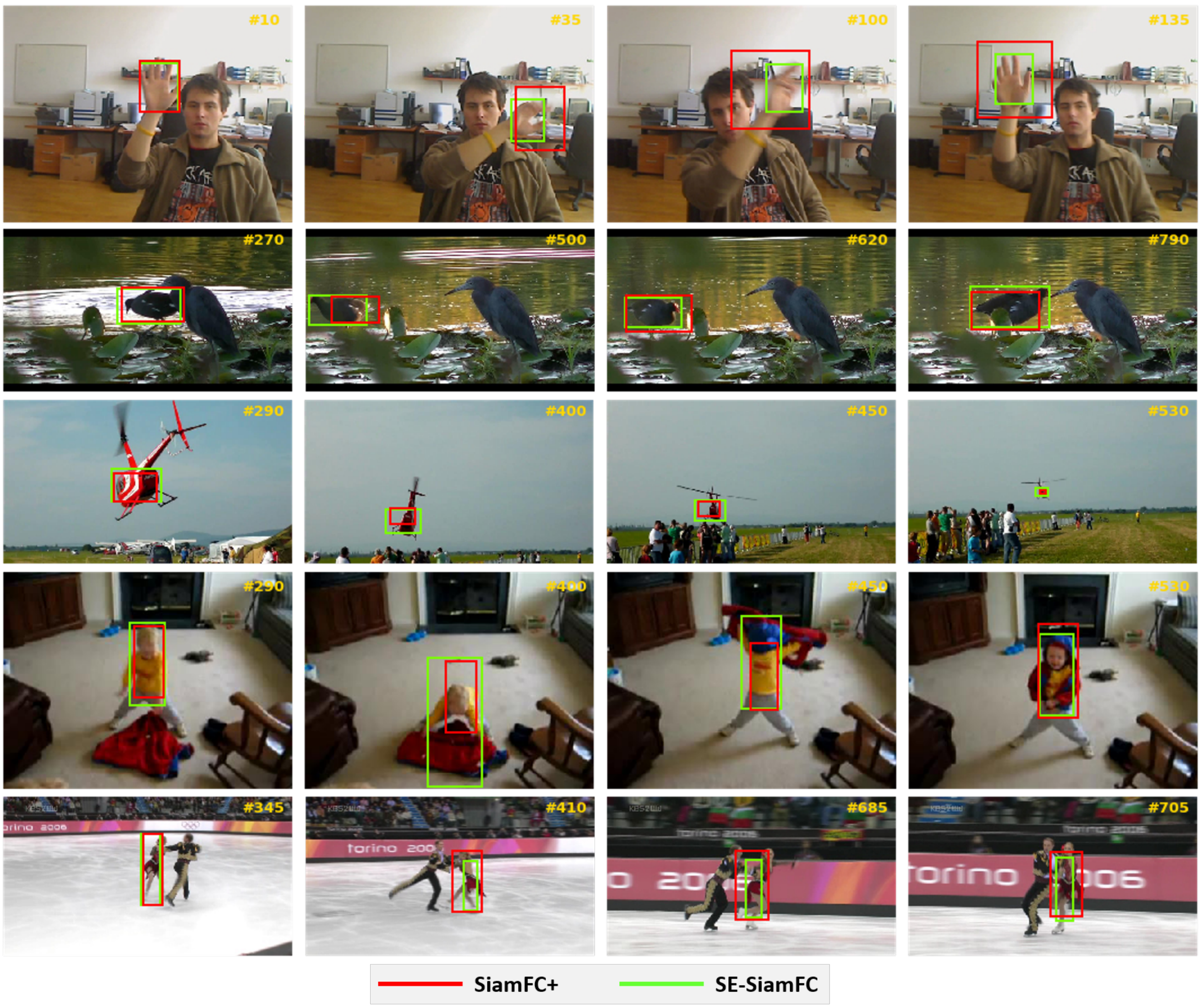

Figure 3: Qualitative comparison of SE-SiamFC with SiamFC+ on VOT2016/2017 sequences.