Supplementary Material

ECO: Efficient Convolution Operators for Tracking

Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, Michael Felsberg
Computer Vision Laboratory, Department of Electrical Engineering, Linköping University, Sweden
{martin.danelljan, goutam.bhat, fahad.khan, michael.felsberg}@liu.se

This supplementary document of [1] first provides the computational complexity analysis of the learning step in C-COT. Section 2 provides additional details of the numerical optimization procedure used for learning the factorized convolution operator (section 3.1 in the paper). We report relevant hardware specifications in section 3. Lastly, we provide additional results on the VOT2016 and OTB-2015 benchmarks in section 4 and 5, respectively.

1. Complexity Analysis of the Learning

Here, we derive the computational complexity of the learning step in the baseline C-COT [2]. The learning itself is completely dominated by the problem of solving the normal equations (eq. (5) in the paper),

$$\left(A^{\mathrm{H}}\Gamma A + W^{\mathrm{H}}W\right)\hat{\mathbf{f}} = A^{\mathrm{H}}\Gamma\hat{\mathbf{y}}\,. \tag{1}$$

Here, $^{\mathrm{H}}$ denotes the conjugate transpose. This linear system is iteratively solved using the Conjugate Gradient (CG) method [5, 6]. The dominating computation in CG is the evaluation of the left-hand side of (1), which is performed once per CG iteration. This computation is performed as

$$A^{\mathrm{H}}(\Gamma(A\hat{\mathbf{f}})) + W^{\mathrm{H}}(W\hat{\mathbf{f}}), \tag{2}$$

where the parentheses are used to indicate the order in which the operations are performed. Since the conjugate symmetry in the filter $\hat{\mathbf{f}}$ is preserved by the operations in (2), only half of the spectrum needs to be processed. We can therefore regard $\hat{\mathbf{f}}$ as a complex vector of $\sum_d K_d = D\bar{K}$ elements, where K_d is the bandwidth of channel d in the filter (see section 2 in the paper), $\bar{K} = \frac{1}{D}\sum_d K_d$ is the average number of Fourier coefficients per channel and D is the number of feature channels d.

The matrix A contains a diagonal block $A_{j,d}$ of size $K \times K_d$ for each sample $j \in \{1, \dots, M\}$ and channel $d \in \{1, \dots, D\}$. Here, we have defined $K = \max_d K_d$. The diagonal of $A_{j,d}$ consists of the elements $\{X_j^d[k]\hat{b}_d[k]\}_{k=0}^{K_d}$. As previously shown for the discrete DCF case [3], the matrix A can be permuted to a block diagonal matrix $\tilde{A} = \oplus_{k=0}^{K}\tilde{A}_k$, where $\tilde{A}_k$ contains the elements $(\tilde{A}_k)_{j,d} = X_j^d[k]\hat{b}_d[k]$. The operations $\hat{\mathbf{f}} \mapsto A\hat{\mathbf{f}}$ and $\hat{\mathbf{v}} \mapsto A^{\mathrm{H}}\hat{\mathbf{v}}$ can thus be implemented as block-wise dense matrix-vector multiplications, with a total of $\mathcal{O}(DM\bar{K})$ operations. Moreover, Γ is a diagonal matrix containing the weights α_j, giving $\mathcal{O}(M\bar{K})$ operations.

In the second term of (2), arising from the spatial regularization in the loss (eq. (3) in the paper), W and W^{H} are convolution matrices with the kernel $\hat{w}[k]$. These operations have a complexity of $\mathcal{O}(D\bar{K}K_w)$, where K_w are the number of non-zero coefficients in $\hat{w}$ (*i.e.* the size of the kernel). In practice however, the kernel $\hat{w}[k]$ is small (typically 5×5), having a lesser impact on the overall complexity. To simplify the complexity expression, we therefore disregard this term. By taking the number of CG iterations N_{CG} into account, we thus obtain the final expression $\mathcal{O}(N_{\mathrm{CG}}DM\bar{K})$ for the complexity of the learning step.

The preprocessing steps needed for the CG optimization only have a marginal impact on the overall learning time. The most significant of these being the Fast Fourier Transform (FFT) of the feature map, having a $\mathcal{O}(\sum_d N_d \log N_d)$ complexity, where N_d is the resolution of feature channel d. But since the FFT computations correspond to roughly 1% of the total time in C-COT, we exclude this part.

2. Factorized Convolution Operator Optimization

Here, we provide more details regarding the optimization procedure for learning the factorized convolution operator (section 3.1 in the paper). We consider the case of learning the factorized operator $S_{Pf}\{x\}$ (eq. (6) in the paper) based on a single sample (x, y),

$$E(f,P) = \|S_{f,P}\{x\} - y\|_{L^2}^2 + \sum_{c=1}^{C} \|wf^c\|_{L^2}^2 + \lambda \|P\|_F^2 \,. \tag{3}$$

The loss is obtained by employing the factorized operator $S_{f,P}\{x\}$ in the data term of the original loss (eq. (3) in the paper) and adding a regularization on the Frobenius norm $\|P\|_F^2$ of P.

By applying the Parseval's formula to the first two terms of (3) and utilizing the linearity and convolution properties of the Fourier series coefficients, we obtain the equivalent loss (same as eq. (7) in the paper),

$$E(f,P) = \left\| \hat{z}^{\mathrm{T}} P \hat{f} - \hat{y} \right\|_{\ell^2}^2 + \sum_{c=1}^{C} \left\| \hat{w} * \hat{f}^c \right\|_{\ell^2}^2 + \lambda \|P\|_F^2. \tag{4}$$

Here, we have defined the interpolated feature map as $z = J\{x\}$ to simplify notation. Note that the matrix-vector products in (4) are performed point-wise,

$$(\hat{z}^{\mathrm{T}} P \hat{f})[k] = \sum_{d=1}^{D} \sum_{c=1}^{C} \hat{z}^d[k] p_{d,c} \hat{f}^c[k] \,, \quad k \in \mathbb{Z} \,. \tag{5}$$

We use the Gauss-Newton method [5] to optimize the non-linear least squares problem (4). In each iteration i, the residual in the data-term is linearized by performing a first order Taylor expansion at the current estimate $(\hat{f}_i, P_i)$ (eq. (8) in the paper). This gives the following quadratic sub-problem (eq. (9) in the paper),

$$\tilde{E}(\hat{f}_{i,\Delta}, \Delta P) = \left\| \hat{z}^{\mathrm{T}} P_i \hat{f}_{i,\Delta} + (\hat{f}_i \otimes \hat{z})^{\mathrm{T}} \operatorname{vec}(\Delta P) - \hat{y} \right\|_{\ell^2}^2 + \sum_{c=1}^{C} \left\| \hat{w} * \hat{f}_{i,\Delta}^c \right\|_{\ell^2}^2 + \lambda \|P_i + \Delta P\|_F^2. \tag{6}$$

To derive a simple formula for the normal equations of (6), we first introduce some notation. Let $\hat{\mathbf{f}}$ be the vectorization of $\hat{f}_{i,\Delta}$, analogously to (1), and define $\mathbf{\Delta p} = \operatorname{vec}(\Delta P)$. Further, let $\mathbf{p}_c$ denote the cth column in P_i and set $\mathbf{p} = \operatorname{vec}(P_i)$. We then define the matrices,

$$A_P = \begin{bmatrix} \mathbf{0}_{K-K_1 \times 2K_1+1} & \cdots & \mathbf{0}_{K-K_C \times 2K_C+1} \\ \operatorname{diag}\begin{pmatrix} \hat{z}[-K_1]^{\mathrm{T}} \mathbf{p}_1 \\ \vdots \\ \hat{z}[K_1]^{\mathrm{T}} \mathbf{p}_1 \end{pmatrix} & \cdots & \operatorname{diag}\begin{pmatrix} \hat{z}[-K_C]^{\mathrm{T}} \mathbf{p}_C \\ \vdots \\ \hat{z}[K_C]^{\mathrm{T}} \mathbf{p}_C \end{pmatrix} \\ \mathbf{0}_{K-K_1 \times 2K_1+1} & \cdots & \mathbf{0}_{K-K_C \times 2K_C+1} \end{bmatrix}, \quad B_f = \begin{pmatrix} (\hat{f}_i \otimes \hat{z})[-K]^{\mathrm{T}} \\ \vdots \\ (\hat{f}_i \otimes \hat{z})[K]^{\mathrm{T}} \end{pmatrix}. \tag{7}$$

Here, A_P has a structure very similar to the matrix A in (1), but contains only a single training sample. Note that the diagonal blocks in A_P are padded with zero matrices $\mathbf{0}_{M \times N}$ along the columns to achieve the same number of $2K+1$ rows.

The Gauss-Newton sub-problem (6) can then be expressed as,

$$\tilde{E}(\hat{\mathbf{f}}, \mathbf{\Delta p}) = \left\| A_P \hat{\mathbf{f}} + B_f \mathbf{\Delta p} - \hat{\mathbf{y}} \right\|_2^2 + \left\| W \hat{\mathbf{f}} \right\|_2^2 + \lambda \left\| \mathbf{p} + \mathbf{\Delta p} \right\|_2^2 \,. \tag{8}$$

Here, the convolution matrix W and the vectorization $\hat{\mathbf{y}}$ are defined as in (1). The normal equations of (8) are obtained by setting the gradient to zero,

$$\begin{bmatrix} A_P^{\mathrm{H}} A_P + W^{\mathrm{H}} W & A_P^{\mathrm{H}} B_f \\ B_f^{\mathrm{H}} A_P & B_f^{\mathrm{H}} B_f + \lambda I \end{bmatrix} \begin{bmatrix} \hat{\mathbf{f}} \\ \mathbf{\Delta p} \end{bmatrix} = \begin{bmatrix} A_P^{\mathrm{H}} \hat{\mathbf{y}} \\ B_f^{\mathrm{H}} \hat{\mathbf{y}} - \lambda \mathbf{p} \end{bmatrix} . \tag{9}$$

We employ the Conjugate Gradient method to iteratively solve the sub-problem (9).

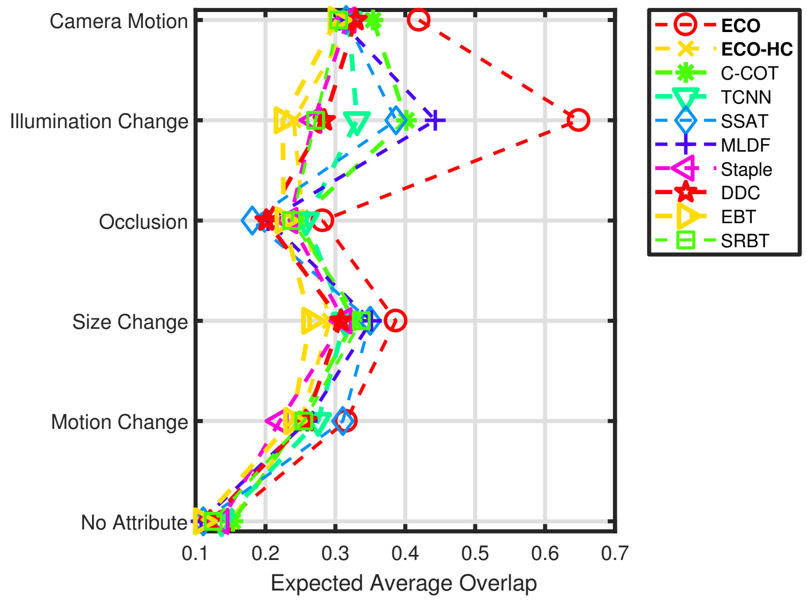

Figure 1. Expected Average Overlap (EAO) scores for each attribute on the VOT2016 dataset. Here, *No Attribute* denotes frames with no labeled attribute.

3. Hardware Specifications

Our tracker is implemented in Matlab and uses Matconvnet [7] for deep feature extraction. The frame-rate measurements of our CPU implementation were performed on a desktop computer with a 4-core Intel Core i7-6700 CPU at 3.4 GHz. The frame-rate measurements of our GPU implementation were performed on a Tesla K40 GPU.

4. Additional Results on VOT2016

Here, we provide further experimental evaluation on the VOT2016 dataset [4] with 60 videos. The videos and the evaluation toolkit can be obtained from `http://www.votchallenge.net/vot2016/`.

In the VOT2016 dataset, each frame is labeled with five different attributes: camera motion, illumination change, occlusion, size change and motion change. Figure 1 visualizes the EAO of each attribute individually. Our approach achieves the best results on all five attributes.

5. Additional Results on OTB-2015

Here, we report additional results on the OTB-2015 dataset [8] with 100 videos. The ground truth annotations and videos are available at `https://sites.google.com/site/benchmarkpami/`.

In the OTB-2015 dataset, each video is annotated with 11 different attributes: scale variation, background clutter, out-of-plane rotation, in-plane rotation, illumination variation, motion blur, fast motion, deformation, occlusion, out of view and low resolution. The success plots of all attributes are shown in figure 2. Our ECO tracker achieves the best performance on 8 out of 11 attributes. Further, our method improves over the baseline C-COT [2] on 9 out of 11 attributes. For a fair comparison, we employ the same combination of deep and hand-crafted features in the baseline C-COT and as in our ECO tracker on the OTB, TempleColor and UAV123 datasets (Conv1, Conv5 and HOG). Note that this set of features provides substantially improved performance in C-COT compared to the original results reported in [2], where only deep convolutional features are used.

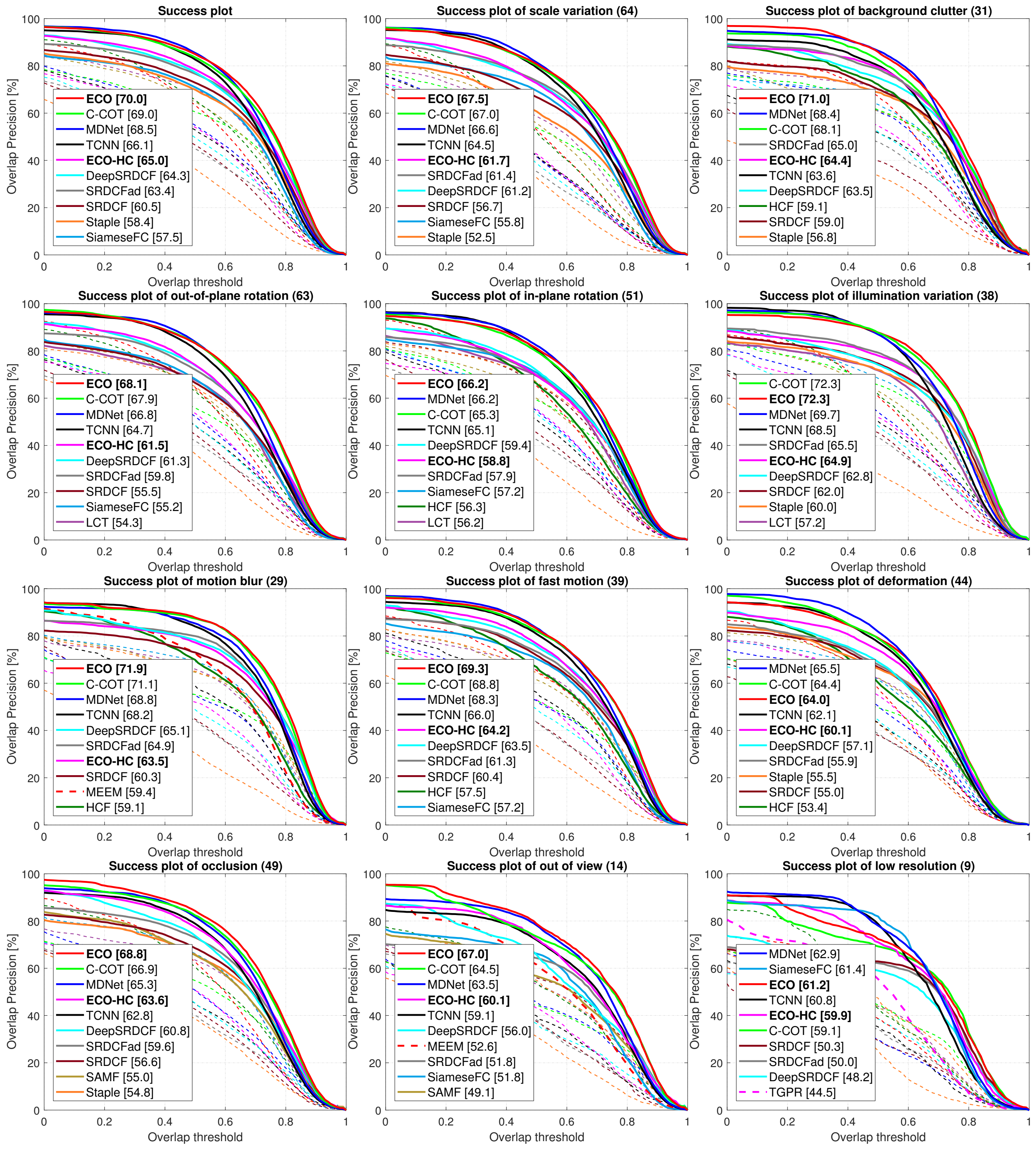

Figure 2. Success plots on the OTB-2015 dataset [8]. The total success plot (top-left) is displayed along with the plots for all 11 attributes. The title text indicate the name of the attribute and the number of videos associated with it. The area-under-the-curve scores for the top 10 trackers are shown in the legend.

References

[1] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017. 1

[2] M. Danelljan, A. Robinson, F. Shahbaz Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016. 1, 3

[3] H. K. Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. In *ICCV*, 2013. 1

[4] M. Kristan, A. Leonardis, J. Matas, R. Felsberg, Pflugfelder, M., L. Čehovin, G. Vojír, T.and Hger, and et al.˙The visual object tracking vot2016 challenge results. In *ECCV workshop*, 2016. 3

[5] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2nd edition, 2006. 1, 2

[6] J. R. Shewchuk. An introduction to the conjugate gradient method without the agonizing pain. Technical report, Pittsburgh, PA, USA, 1994. 1

[7] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. *CoRR*, abs/1412.4564, 2014. 3

[8] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *TPAMI*, 37(9):1834–1848, 2015. 3, 4