Fast Spatially Regularized Correlation Filter Tracker

Alan Lukežič, Luka Čehovin Zajc, Matej Kristan

Faculty of Computer and Information Science, University of Ljubljana
{*alan.lukezic, luka.cehovin, matej.kristan*}*@fri.uni-lj.si*

Abstract

Discriminative correlation filters (DCF) have attracted significant attention of the tracking community. Standard formulation of the DCF affords a closed form solution, but is not robust and constrained to learning and detection using a relatively small search region. Spatial regularization was proposed to address learning from larger regions. But this prohibits a closed form solution and leads to an iterative optimization with significant computational load, resulting in slow model learning and tracking. We propose to reformulate the spatially regularized filter cost function such that it offers an efficient optimization. This significantly speeds up the tracker (approximately 14 times) and results in real-time tracking at the same or better accuracy.

1 Introduction

Visual object tracking is a task of continuous target localization given an initial position in first frame of the video sequence. Significant advancements have been made in the field over the recent years, largely due to a number publicly available benchmarks and steps towards standardized performance evaluation [1, 2, 3, 4].

Most recent developments have focused on discriminative correlation filters (DCF), which were primarily proposed for object detection [5]. The idea of learning discriminative correlation filters for tracking has been promoted by the seminal paper of Bolme et al. [6]. Further improvements by other authors included correlation filter with kernels [7], multiple-channel formulation [8] and scale estimation with DCF [9, 10].

In DCF tracking the target is localized using a filter which is learned on a pre-defined (Gaussian) response on the training image. The standard formulation of DCF uses circular correlation which allows to implement filter learning efficiently by Fast Fourier transform (FFT) but requires the filter and the patch size to be equal which limits the detection range. Due to the circularity, the filter is trained on many examples that contain unrealistic, wrapped-around circularly shifted versions of the target. These windowing problems were recently addressed by [11, 12] using the boundary constraints and by Danneljan et al. [13] who introduce spatial regularization to penalize filter values outside the object boundaries. All these approaches train from image patches larger than the object and thus increase the detection range.

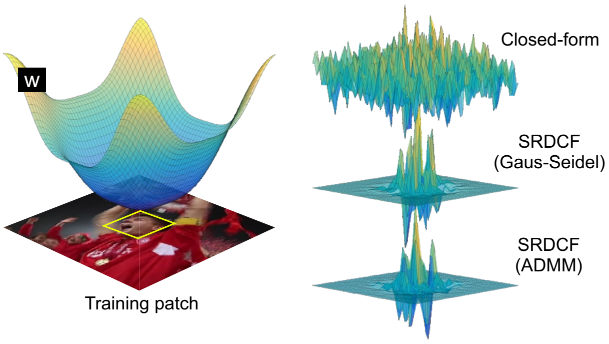

Figure 1: Training patch and spatial weight function are shown on the left. Yellow rectangle represents target region. Three filters are shown on the right (top to bottom): filter calculated with a closed form solution (Equation 1), and two filters obtained with iterative optimization methods Gauss-Seidel and ADMM optimization.

Spatially regularized discriminative correlation filter (SRDCF) [13] shows a great potential and trackers using this method [14, 15] achieve excellent performance on recent benchmarks. But the spatial regularization prevents using a closed form solution as in standard formulation [6] and iterative optimization has to be used. This optimization is based on Gauss-Seidel steps which simultaneously solves a large number of equations and leads to a very slow solution. Despite excellent performance, SRDCF is of limited use in practical application since it runs far below real-time.

We propose to reformulate the spatially regularized filter cost function such that it affords an efficient optimization by using alternating direction method of multipliers (ADMM) [16]. Since all steps in optimization are performed pixel-wise, this significantly speeds up the tracker (approximately 14 times) and results in real-time tracking of approximately 29fps at the same or better accuracy.

2 Discriminative Correlation Filters

In the standard DCF formulation [6], the filter $\mathbf{h}$ is trained to output a desired response $\mathbf{g}$ when correlated with a training sample $\mathbf{f}$, i.e., $\mathbf{g} = \sum_{i=1}^{d} \mathbf{h}_i * \mathbf{f}_i$, where $*$ denotes a circular correlation and the training sample $\mathbf{f}$ consists of d-channel feature maps, therefore $\mathbf{h}$ contains d channels. The following cost is minimized in filter learning:

$$\epsilon(\mathbf{h}) = || \sum_{i=1}^{d} \mathbf{f}_i * \mathbf{h}_i - \mathbf{g}||^2 + \lambda \sum_{i=1}^{d} ||\mathbf{h}_i||^2 \qquad (1)$$

An efficient closed-form solution for (1) is obtained via the Fourier domain:

$$\hat{\mathbf{h}}_i = \frac{\hat{\mathbf{f}}_i \odot \overline{\hat{\mathbf{g}}}}{\sum_{i=1}^{d} \overline{\hat{\mathbf{f}}}_i \odot \hat{\mathbf{f}}_i + \lambda}, \qquad (2)$$

where $\hat{(\ )}$ represents Fourier transform, i.e., $\hat{a} = \mathcal{F}(a)$, the symbol $\overline{(\ )}$ is complex-conjugate operation and $\odot$ is a Hadamard product. Target is localized by finding position of the maximum peak in the correlation response $\mathbf{r}$:

$$\mathbf{r} = \mathcal{F}^{-1}\Big(\sum_{i=1}^{d} \hat{\mathbf{h}}_i \odot \hat{\mathbf{f}}_i \Big), \qquad (3)$$

where $\mathcal{F}^{-1}$ represents an inverse Fourier transformation.

2.1 Spatially Regularized DCF

The SRDCF [13] tracker introduces a spatial weight function $\mathbf{w}$ which penalizes large filter values further away from the target center. It avoids the windowing problem due to the circular correlation in Fourier domain and it reduces impact of the background in the filter. The visualization of the penalty function $\mathbf{w}$ and the resulting filter is shown in the Figure 1. Penalty function is incorporated in a standard DCF formulation (1) by adding a spatial regularization term, i.e.,

$$\epsilon(\mathbf{h}) = || \sum_{i=1}^{d} \mathbf{f}_i * \mathbf{h}_i - \mathbf{g}||^2 + \lambda_1 \sum_{i=1}^{d} ||\mathbf{w}\mathbf{h}_i||^2 + \lambda_0 \sum_{i=1}^{d} ||\mathbf{h}_i||^2. \qquad (4)$$

An iterative Gauss-Seidel optimization is used to solve a linear system of equations resulting from (4). The size of the linear problem is $dMN \times dMN$, where M, N, and d are width, height and number of feature channels, respectively, resulting in slow computation despite carrying it out in Fourier domain. We refer the reader to [13] for complete derivation of the optimization.

3 Proposed Method

We propose to reformulate the cost function (4), which can be decomposed into independent per-pixel (or per-frequency in the relevant Fourier-transformed terms) subproblems,

$$\epsilon(\hat{\mathbf{h}}) = \sum_{j=1}^{N} \Big(||\hat{\mathbf{f}}_j^T \overline{\hat{\mathbf{h}}}_j - \hat{\mathbf{g}}_j||^2 + \lambda_1 ||\mathbf{w}_j \mathbf{h}_j||^2 + \lambda_0 ||\hat{\mathbf{h}}_j||^2 \Big), \qquad (5)$$

where N is number of pixels. Note that $\mathbf{h}_j$ and $\mathbf{f}_j$ are d-dimensional column vectors and $\mathbf{g}_j$ and $\mathbf{w}_j$ are scalars on j-th pixel. The subscript j is omitted in the following due to the compactness of the notation. An auxiliary variable $\boldsymbol{\omega}$ is introduced and the cost function (5) is augmented with the constraint $\boldsymbol{\omega} \equiv \mathbf{h}$, leading to an augmented Lagrangian formulation

$$\begin{aligned} \mathcal{L}(\hat{\mathbf{h}}, \mathbf{l}, \boldsymbol{\omega}) = {} & ||\hat{\mathbf{f}}^T \overline{\hat{\mathbf{h}}} - \hat{\mathbf{g}}||^2 + \lambda_1 ||\mathbf{w}\boldsymbol{\omega}||^2 + \qquad (6) \\ & + \lambda_0 ||\hat{\mathbf{h}}||^2 + \mathbf{l}^T(\mathbf{h} - \boldsymbol{\omega}) + \frac{\mu}{2}||\mathbf{h} - \boldsymbol{\omega}||^2, \end{aligned}$$

where $\mathbf{l}$ is a Lagrange multiplier and $\mu > 0$ is a parameter.

The cost function (6) is solved following the alternate direction method of multipliers (ADMM) [16] by alternatingly solving two subproblems:

$$\nabla_{\overline{\hat{\mathbf{h}}}} \mathcal{L} \equiv 0 \qquad (7)$$

$$\nabla_{\boldsymbol{\omega}} \mathcal{L} \equiv 0. \qquad (8)$$

The first derivative in (7) is equivalent to $\nabla_{\overline{\hat{\mathbf{h}}}} \mathcal{L} = \hat{\mathbf{f}}\hat{\mathbf{f}}^H \hat{\mathbf{h}} - \hat{\mathbf{f}}\overline{\hat{\mathbf{g}}} + \lambda_0 \hat{\mathbf{h}} + \frac{\mu}{2}\hat{\mathbf{h}} - \frac{\mu}{2}\hat{\boldsymbol{\omega}}$, where $(\cdot)^H$ is a Hermitian transpose. Setting the derivative to zero yields the following solution

$$\hat{\mathbf{h}} = (\hat{\mathbf{f}}\hat{\mathbf{f}}^H + \lambda_2)^{-1}(\hat{\mathbf{f}}\overline{\hat{\mathbf{g}}} + \frac{\mu}{2}\hat{\boldsymbol{\omega}}), \qquad (9)$$

where $\lambda_2 = \lambda_0 + \frac{\mu}{2}$.

The first part of (9) represents a matrix inverse calculated for each pixel, which is computationally demanding. This can be avoided by using a Sherman-Morrison-Woodbury formula [17] leading to

$$\Big(\hat{\mathbf{f}}\hat{\mathbf{f}}^H + \lambda_2\Big)^{-1} = \frac{1}{\lambda_2}\Big(\mathbf{I} - \frac{\hat{\mathbf{f}}\hat{\mathbf{f}}^H}{\hat{\mathbf{f}}^H\hat{\mathbf{f}} + \lambda_2}\Big), \qquad (10)$$

which simplifies (9) into

$$\hat{\mathbf{h}} = \frac{\hat{\mathbf{f}}\overline{\hat{\mathbf{g}}} - \frac{\mu}{2\lambda_2}\hat{\mathbf{f}}\hat{\mathbf{f}}^H\hat{\boldsymbol{\omega}}}{\hat{\mathbf{f}}^H\hat{\mathbf{f}} + \lambda_2} + \frac{\mu}{2\lambda_2}\hat{\boldsymbol{\omega}}. \qquad (11)$$

The derivative in (8) yields the following solution: $\nabla_{\boldsymbol{\omega}} \mathcal{L} = 2\lambda_1 \mathbf{w}\boldsymbol{\omega} - \mathbf{l} - \mu\mathbf{h} - \frac{\mu}{2}\boldsymbol{\omega}$. Setting it to zero, leads to

$$\boldsymbol{\omega} = \frac{\mathbf{l} + \mu\mathbf{h}}{2\lambda_1\mathbf{w} - \frac{\mu}{2}}. \qquad (12)$$

Since $\hat{\mathbf{h}}$, $\boldsymbol{\omega}$ and $\mathbf{l}$ are used in (11) and (12) the final solution for filter $\mathbf{h}$ is calculated iteratively, i.e.,

$$\begin{aligned} \hat{\mathbf{h}}^{(k)} &= \frac{\hat{\mathbf{f}}\overline{\hat{\mathbf{g}}} - \frac{\mu}{2\lambda_2}\hat{\mathbf{f}}\hat{\mathbf{f}}^H\hat{\boldsymbol{\omega}}^{(k-1)}}{\hat{\mathbf{f}}^H\hat{\mathbf{f}} + \lambda_2} + \frac{\mu}{2\lambda_2}\hat{\boldsymbol{\omega}}^{(k-1)}, (13) \\ \boldsymbol{\omega}^{(k)} &= \frac{\mathbf{l}^{(k-1)} + \mu\mathbf{h}^{(k)}}{2\lambda_1\mathbf{w} - \frac{\mu}{2}}, \qquad (14) \\ \mathbf{l}^{(k)} &= \mu(\mathbf{h}^{(k)} - \boldsymbol{\omega}^{(k)}), \qquad (15) \end{aligned}$$

where k represents the iteration index.

4 Tracking with augmented SRDCF

A tracking iteration in each frame consists of two main steps: target localization and update of the visual model. Both steps are described in the following.

Target localization. When a new frame arrives at time step t, an image patch four times larger than target size is extracted from the image, centered on the target position from previous time step p_{t-1}. A 41-channel feature map $\mathbf{f}_t$ is calculated from the extracted image patch, corresponding to 31 HoG [18] and 10 colornames [8] channels. Circular correlation between the feature channels $\mathbf{f}_t$ and filter from previous time-step $\mathbf{h}_{t-1}$ is calculated in the Fourier domain (3). Position of the correlation output maximum $\mathbf{p}_t$ presents the new target position. Change of target size is estimated using the approach from [10].

Update. Target appearance is expected to change during tracking due to the numerous effects like, rotation, deformation or scale change of the target. The visual model is therefore updated after the target is localized. An image patch four times larger than target size is extracted from image centered at $\mathbf{p}_t$ and a feature map $\mathbf{f}_t$ is calculated. A new filter $\tilde{\mathbf{h}}_t$ is calculated using method described in Section 3. Filter from the previous time-step is denoted as $\mathbf{h}_{t-1}$ and it is combined with $\tilde{\mathbf{h}}_t$ using a moving average, resulting in an updated filter

$$\hat{\mathbf{h}}_t = (1-\eta)\hat{\mathbf{h}}_{t-1} + \eta\hat{\tilde{\mathbf{h}}}_t. \qquad (16)$$

This filter is used in the next time-step to localize the target. Parameter $\eta = 0.01$ is a learning rate which controls the impact of the new filter in $\hat{\mathbf{h}}_t$. The update is performed in the Fourier domain for efficiency.

5 Experiments

The proposed tracker, the augmented SRDCF (SRDCF$_{\mathrm{A}}$) is compared to the original SRDCF [13] on VOT16 [2] (Section 5.1) and on OTB100 [4] (Section 5.2). Speed comparison is given in Section 5.3

5.1 VOT16 dataset

The VOT16 [2] dataset consists of 60 sequences which are chosen so that the dataset is moderately large, but rich in attributes relevant for tracking. A standard reset-based experiment [1] is performed by running a tracker on each sequence until overlap of the predicted target position with the ground-truth is zero. This event is called failure and after that the tracker is reinitialized. Tracking performance is measured by accuracy, robustness and expected average overlap (EAO). Robustness measures average number of failures and accuracy represents average overlap. Expected average overlap combines both measures into the single number and it can be interpreted as expected overlap at the typical sequence length in short-term tracking scenario.

In the Table 1 tracking performance of SRDCF$_{\mathrm{A}}$ is comparable to the original SRDCF. The SRDCF$_{\mathrm{A}}$ has 6% less failures, a lower average overlap (4%), but a 4% better expected average overlap. The similar performance indicates that the filter calculated with ADMM optimization is comparable to the filter calculated with Gauss-Seidel optimization for tracking tasks.

5.2 OTB100 dataset

The OTB100 [4] dataset consists of 100 sequences. A tracker is in each sequence initialized on the first frame and run to the end, which is called a no-reset experiment. Tracking performance is measured by the area-under-the-curve (AUC) on the success and precision plots, shown in Figure 2. The success plot shows overlap threshold values on x and the proportion of frames with the overlap between the predicted and ground truth bounding boxes as greater than a threshold on y. The precision plot shows a similar statistics computed from the center error.

Table 1: Accuracy (Acc.), robustness (Rob.) and expected average overlap (EAO) for SRDCF and SRDCF$_{\mathrm{A}}$ on the VOT16 dataset. The sign ↑ denotes *high is better* and ↓ denotes *low is better*.

Tracker	**Acc.** ↑	**Rob.** ↓	**EAO** ↑
SRDCF$_{\mathrm{A}}$	0.51	1.42	0.2520
SRDCF	0.53	1.52	0.2431

Results are shown in the Figure 2. The difference between the original SRDCF and the SRDCF$_{\mathrm{A}}$ is negligible (approximately 1% better AUC on both graphs). This further supports the equivalence of the learned filters by the slow Gauss-Seidel and our fast ADMM formulation from tracking performance perspective.

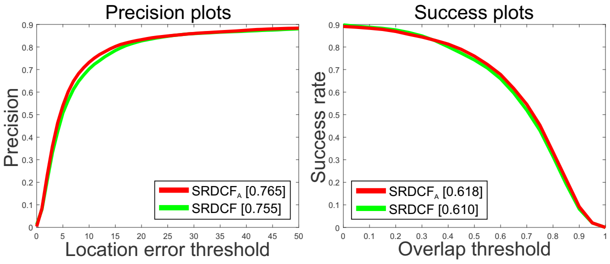

Figure 2: Tracking performance on the OTB100 [4] dataset. The area-under-the-curve (AUC) is shown in brackets in the legend.

5.3 Speed comparison

Our main motivation behind reformulation of the SRDCF cost function (6) is computational efficiency. The original tracker runs on average at 2 frames-per-second which is far from real-time and it presents a serious limitation for practical tracker application. In this section we compare speed measurements of the original SRDCF and the proposed SRDCF$_{\mathrm{A}}$ which uses the ADMM optimization. Both trackers are run on the 60 sequences of VOT16 [2] dataset using a no-reset experiment and tracking speed is averaged over all sequences.

In each frame time needed to process the whole frame is measured including target localization and filter calculation stage, without image loading from disk. Additionally, filter calculation step is reported separately, to demonstrate the difference between the both optimization methods. Since original SRDCF requires a significant pre-computation overhead performed only in the initialization frame, time needed to process the first frame is reported separately. All experiments were conducted on the same desktop Intel i7 6700 CPU, 3.4 GHz and 16GB RAM computer.

Results of the speed comparison are presented in the Table 2. Time for the initialization required by the SRDCF tracker is almost 2 seconds while ADMM-based SRDCF$_\mathrm{A}$ takes only 47 milliseconds, which is more than 40-times faster. Considering only filter calculation step, the Gauss-Seidel optimization takes in average 329 milliseconds, while ADMM optimization needs only 9 milliseconds. Average per-frame speed of the original SRDCF is 2FPS (480ms per frame), while SRDCF$_\mathrm{A}$ achieves over 14-times faster performance running at 29FPS (34ms per frame).

Table 2: The average times needed for tracker initialization and filter calculation are shown in the first two columns. Average tracking speed is presented in milliseconds and in frames-per-second (third and fourth column).

Tracker	**Init.** [ms]	**Filter** [ms]	**Average** [ms]	**Average** [FPS]
SRDCF$_\mathrm{A}$	47	9	34	29
SRDCF	1814	329	480	2

6 Conclusion

In this paper an optimization method for spatially regularized discriminative correlation filter based on the ADMM method [16] is derived. The optimization is used to minimize the cost function from the popular SRDCF [13] tracker. We show experimentally that tracking performance of our version of SRDCF using ADMM optimization method is slightly better than the original method. In addition, the speed of the proposed tracker is more than fourteen times faster than original version, running in real-time at 29 frames-per-second. In our future work we plan to test different spatial weight functions and to incorporate learning from multiple training samples.

Acknowledgements

This work was supported in part by the following research programs and projects funded by the Slovenian research agency ARRS: project ViAMaRo (J2-8175) and research program P2-0214.

References

[1] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Cehovin, “A novel performance evaluation methodology for single-target trackers,” *IEEE Trans. Pattern Anal. Mach. Intell.*, 2016.

[2] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Čehovin, T. Vojir, G. Häger, A. Lukežič, and G. et al. Fernandez, “The visual object tracking vot2016 challenge results,” in *Proc. European Conf. Computer Vision*, 2016.

[3] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Hager, A. Lukezic, A. Eldesokey, and G. Fernandez, “The visual object tracking vot2017 challenge results,” in *The IEEE International Conference on Computer Vision (ICCV)*, 2017.

[4] Y. Wu, J. Lim, and M. H. Yang, “Object tracking benchmark,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sept 2015.

[5] C. F. Hester and D. Casasent, “Multivariant technique for multiclass pattern recognition,” *Applied Optics*, vol. 19, no. 11, pp. 1758–1761, 1980.

[6] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, “Visual object tracking using adaptive correlation filters,” in *Comp. Vis. Patt. Recognition*. IEEE, 2010, pp. 2544–2550.

[7] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filters,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, 2015.

[8] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer, “Adaptive color attributes for real-time visual tracking,” in *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, 2014, pp. 1090–1097.

[9] Y. Li and J. Zhu, “A scale adaptive kernel correlation filter tracker with feature integration,” in *Proc. European Conf. Computer Vision*, 2014, pp. 254–265.

[10] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, “Discriminative scale space tracking,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1561–1575, 2017.

[11] H. Kiani Galoogahi, T. Sim, and S. Lucey, “Correlation filters with limited boundaries,” in *Comp. Vis. Patt. Recognition*, 2015, pp. 4630–4638.

[12] A. Lukežič, T. Vojíř, L. Čehovin Zajc, J. Matas, and M. Kristan, “Discriminative correlation filter with channel and spatial reliability,” in *Comp. Vis. Patt. Recognition*, 2017, pp. 6309–6318.

[13] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, “Learning spatially regularized correlation filters for visual tracking,” in *Int. Conf. Computer Vision*, 2015, pp. 4310–4318.

[14] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, “Beyond correlation filters: learning continuous convolution operators for visual tracking,” in *Proc. European Conf. Computer Vision*. Springer, 2016, pp. 472–488.

[15] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg, “Eco: Efficient convolution operators for tracking,” in *Comp. Vis. Patt. Recognition*, 2017, pp. 6638–6646.

[16] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, “Distributed optimization and statistical learning via the alternating direction method of multipliers,” *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[17] R. A. Horn and C. R. Johnson, *Matrix Analysis*, 2nd ed. New York, NY, USA: Cambridge University Press, 2012.

[18] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” in *Comp. Vis. Patt. Recognition*, vol. 1, June 2005, pp. 886–893.