The Importance of Estimating Object Extent when Tracking with Correlation Filters

Luca Bertinetto, Jack Valmadre, Ondrej Miksik, Stuart Golodetz, Philip H.S. Torr
University of Oxford
{name.surname}@eng.ox.ac.uk

***Abstract*— Trackers based on correlation filters have achieved excellent performance in single-target model-free tracking. Several versions exist, incorporating multi-channel features, kernels and scale adaptation. However, they all suffer from a well-known limitation of the tracking-by-detection paradigm: the update step is performed at the last estimated position of the object, so that small inaccuracies accumulate and often lead to the loss of the target. In this paper, we demonstrate that it is possible to partially correct this behaviour using a simple refinement of the estimated position based on a pixel-wise object likelihood. This makes these trackers significantly more robust to shape change, because the model update uses a non-drifted version of the target. Our experiments on the popular VOT2014 and VOT2015 benchmarks demonstrate that our modification is able to considerably improve both the robustness and the accuracy of state-of-the-art correlation trackers.**

I. Introduction

We consider the widely-adopted scenario of short-term, single-object tracking, in which the target is only specified in the first frame (using a rectangle). *Short-term* implies that re-detection should not be necessary. The key challenge of tracking an unfamiliar object in video is to be robust to changes in its appearance. The task of tracking unfamiliar objects, for which training examples are not available in advance, is interesting because in many situations it is not feasible to obtain such a dataset. It is advantageous for the algorithm to perform above real-time for computationally intensive applications such as robotics, surveillance, video processing and augmented reality.

Since an object's appearance can vary significantly during a video, it is not generally effective to estimate its model from the first frame alone and use this single, fixed model to locate the object in all other frames. Most state-of-the-art algorithms therefore employ model adaptation to take advantage of information present in later frames. The simplest, most widespread approach is to treat the tracker's predictions in new frames as training data with which to update the model. In this way, the estimated position in the previous frame is considered ground truth for the update of the model. The danger of learning from predictions is that small errors often accumulate and cause model drift. This is particularly likely to happen when the appearance of the object changes.

In this paper, we demonstrate how the robustness of state-of-the-art trackers based on correlation filters ([15], [7], [4]) can be dramatically improved by addressing this problem. Namely, we use a simple likelihood map to correct the prediction of the model learnt with correlation filters and therefore subsequently update the model with a non-drifted template.

Our experiments on the popular VOT2014 and VOT2015 benchmarks demonstrate that we are able to improve the robustness of all the correlation trackers analysed by between 15.1% and 58.2%, with an average improvement in accuracy of 2.8%.

The rest of this paper is divided as follows. §II briefly describes the current state-of-the-art in single-object tracking, with an emphasis on the methods that are more relevant to our proposal. §III introdues correlation filters and explains how they can be used in a tracking scenario. §IV describes our proposed approach to make correlation trackers sensibly more robust to drift. Finally, §V shows the results of our experiments on the benchmark VOT2014 and VOT2015.

II. Related Work

Online learning and Correlation Filters: Modern approaches to adaptive tracking often use an online version of an object detection algorithm. One approach that achieves strong results [34] and has an elegant formulation is Struck [13], which seeks to minimise the structured output objective for localisation [2]. However, the computation needed limits the number of features and training examples.

Correlation Filters instead minimise a least-squares loss for all circular shifts of the positive examples. Although this might seem a weaker approximation of the true problem, it enables the use of densely-sampled examples and high-dimensional feature images in real-time using the Fourier domain. Initially applied to adaptive tracking in greyscale images by Bolme *et al.* [4], their extension to multiple feature channels [3], [14], [15], [18] and therefore HOG features [6] enabled the technique to achieve state-of-the-art performance in VOT14 [20]. The winner of the challenge, DSST [7], incorporated a multi-scale template for Discriminative Scale-Space Tracking using a 1D Correlation Filter. One deficiency of Correlation Filters is that they are constrained to learn from *all* circular shifts. Several recent works [8], [11], [19] have sought to resolve this issue, and the Spatially Regularised (SRDCF) [8] formulation in particular has demonstrated excellent tracking results. However, this was achieved at the cost of real-time operation.

Robustness to deformation: Correlation Filters are inherently confined to the problem of learning a rigid template. This is a concern when the target experiences shape deformation in the course of a sequence. Perhaps the simplest

method to achieve robustness to deformation is to adopt a representation that is insensitive to shape variation. Image histograms have this property, because they discard the position of every pixel. In fact, histograms can be considered orthogonal to Correlation Filters, since a Correlation Filter is *learnt from* circular shifts, whereas a histogram is *invariant to* circular shifts. However, histograms alone are often insufficient to discriminate the object from the background. While colour histograms were used in many early approaches to object tracking [27], [28], they have only recently been demonstrated to be competitive in modern benchmarks in the Distractor-Aware Tracker (DAT) [29], which uses adaptive thresholding and explicit suppression of regions with similar colours. In general, histograms may be constructed from any discrete-valued feature, including local binary patterns and quantised colours. For a histogram to provide robustness to deformation, the feature must be insensitive to the local changes that arise.

The chief alternative way to achieve robustness to deformation is to learn a deformable model. We believe it is ambitious to learn a deformable model from a single video in which the only supervision is the location in the first frame, and therefore adopt a simple bounding box. Moreover, deformable models have a rich representation of the target object that the most adopted benchmarks [21], [35] do not necessarily reward. Our single-template tracker could be considered a component with which to construct a parts-based model.

Schemes to reduce model drift: Model drift is a result of learning from inaccurate predictions. Several works have aimed to prevent drift by modifying the training strategy rather than improving the predictions. TLD [17] and PROST [30] encode rules for additional supervision based on optical flow and a conservative appearance model. Other approaches avoid or delay making hard decisions. MILTrack [1] uses Multiple-Instance Learning to train with bags of positive examples. Supančič and Ramanan [31] introduce self-paced learning for tracking: they solve for the optimal trajectory keeping the appearance model, then update the model using the most confident frames, and repeat. Grabner *et al.* [12] treat tracking as online semi-supervised boosting, in which a classifier learnt in the first frame provides an anchor for the labels assigned to examples in later frames. Tang *et al.* [32] apply co-training to tracking, learning two independent SVMs that use different features and then obtaining hard negatives from the combined scores.

Combining multiple estimates: Another strategy widely adopted to mitigate inaccurate predictions is to combine the estimates of an *ensemble* of methods, so that the weaknesses of the trackers are reciprocally compensated. In [22], [23], Kwon *et al.* make use of complementary basic trackers, built by combining different observation models and motion models, and then integrate their estimates in a sampling framework. Similarly, [33] combines five independent trackers using a factorial HMM, modelling both the object trajectory and the reliability of each tracker across time. Rather than using trackers of different types, the Multi-Expert Entropy Minimisation (MEEM) tracker [36] maintains a collection of past models and chooses the prediction of one according to an entropy criterion.

Long-term tracking with re-detection: Several recent works have adopted Correlation Filters for the problem of long-term tracking, where the performance of an algorithm will be greatly improved by its ability to re-detect the object. The Long-term Correlation Tracker (LCT) [24] augments a standard Correlation Filter tracker with an additional Correlation Filter for confidence estimation and a random forest for re-detection, both of which are only updated in confident frames. The Multi-Store Tracker (MUSTer) [16] maintains a long-term memory of SIFT keypoints for the object and background, using keypoint matching and MLESAC to locate the object. The confidence of the long-term memory is estimated using the number of inliers, and occlusions can be determined by considering the number of background keypoints that are located inside the rectangle. Since we consider mostly short-term benchmarks, and these long-term trackers are meta-algorithms that are built upon a short-term tracker, there is little value in a comparison. Note that the TLD [17] and self-paced learning [31] algorithms also incorporate some aspects that are well-suited to the long-term tracking problem.

III. Correlation filters

A. Detection

A simple way of expressing *similarity* between a template h (also called filter or model in the following) and a test image f is through the dot product. Since the location of the object in f is unknown, correlation (denoted by $\star$) is used to calculate the dot product for each possible alignment of h with f, *i.e.* $\mathbf{c} = f \star h$. Each element of $\mathbf{c}$ can be considered as a similarity score between one shift of h and f. Besides its simplicity, correlation is highly attractive because of the convolution theorem, which states that correlation becomes element-wise multiplication (Hadamard product, denoted with $\odot$) in the Fourier domain, *i.e.* $\mathbf{c} = f \star h = \mathcal{F}^{-1}(F \odot \overline{H})$, where F and H represent the Fourier Transform (FT) of f and h respectively, the bar represents complex conjugation and $\mathcal{F}^{-1}$ denotes the inverse FT.

Ideally, we want to learn a template (h) that yields a high response for image patches that match the object of interest, and lower responses elsewhere. In a template matching scenario, when the test image contains an instance of the object, the desired correlation output should present a sharp peak in the position of the target; otherwise, no peak should be discernible.

In MACE [25], Mahalanobis *et al.* inherently use *hard labels* in the classification problem, because they label each training image using an impulsive desired response, whose intensity is 1 at the exact position of the target and becomes close to 0 everywhere else. Differently, in ASEF [5] Bolme *et al.* use a circular Gaussian as desired response, thus assigning a soft-label for each possible shift of the template. In this way, in contrast to MACE, a unique "exact filter"

is obtained for every training image. The exact template receives the maximum score, and lower scores are given to shifted templates. Over-fitting is avoided by averaging the filters across all the training images.

B. Tracking-by-detection

MOSSE [4] is an adaptation of ASEF to tracking: instead of averaging thousands of filters learnt on different training images, at each frame a new filter is learnt, and then a *global filter* is updated using a running average approach like (3). The optimisation is formulated as a minimisation of the squared difference between the correlation of the filter with the new frame and the desired response, elements of which can be between 0 and 1. More specifically, at time step t a filter h_t is learnt in the Fourier domain by solving

$$\min_{\overline{H}_t} \left| F_t \odot \overline{H}_t - G \right|^2, \tag{1}$$

where G is the Discrete Fourier Transform (DFT) of the desired output of the correlation, F_t is the DFT of the image patch[1] at t and $\bar{H}_t$ is the complex conjugate of the DFT of h_t. Using the technique explained in [26] to compute stationary points of a real-valued function of a complex variable, the result is

$$\overline{H}_t = \frac{G \odot \overline{F}_t}{F_t \odot \overline{F}_t + \eta}, \tag{2}$$

where η is a regularization parameter (taken strictly greater than zero) added to avoid the division by zero that might occur when the training image frequency contains very little energy. The global filter $\overline{\mathcal{H}}_t$ is then updated with the freshly learnt $\overline{H}_t$ using a running average[2]:

$$\overline{\mathcal{H}}_t := (1-\lambda)\overline{\mathcal{H}}_{t-1} + \lambda \overline{H}_t. \tag{3}$$

Such update rule gradually forgets old object appearances, incorporating new ones at a pace controlled by the learning rate λ.

Finally, when a new frame F_{t+1} appears, the filter response is calculated with

$$\mathcal{F}^{-1}(F_{t+1} \odot \overline{\mathcal{H}}_t) \tag{4}$$

and the new position $\boldsymbol{p}_{t+1}$ of the target is estimated at the location of the response peak,.

IV. Proposed approach

Trackers that update online an internal model of the target are very susceptible to drift. In fact, when the tracker fails to accurately estimate the position of the object, its model implicitly learns an inaccurate representation. Inevitably then, even small inaccuracies can cause drift, due to the positive feedback formed by updating the model using the predicted position in the current frame, and then using this model to estimate the position in the next frame [1].

[1]The image patch can be represented with any feature map. Often, the HOG features of [10] are used.

[2]In practice, for mathematical convenience, the numerator and denominator of (2) are updated separately with the same learning rate.

Model-free tracking algorithms that use correlation filters to learn a linear template with HOG [15] or Color Names [9] features achieve state-of-the-art performance [20], [35] while maintaining real-time speed. However, these approaches can still be observed to drift, particularly when the object of interest experiences rapid and significant shape deformation (e.g. Figure 7), which might result from non-rigid deformation, as well as in-plane rotation and scaling if these are not tracked. This is not surprising, because these features only have compact spatial support and a linear function of these features might not be able to generalize to the range of possible shapes. Our method therefore aims to reduce drift using a simple refinement step that is invariant to shape.

Specifically, we estimate the probability $L(i,j)$ that pixel (i,j) belongs to the target object rather than the background for every pixel in a region around the location estimated by the tracker. We then compute the centroid μ of the likelihood map

$$\mu = \frac{\sum_{ij} L(i,j) \cdot (i,j)}{\sum_{ij} L(i,j)} \ , \tag{5}$$

and translate the centre of the box predicted by the tracker $\mathbf{p}_t$ so that its center occurs at the position of μ. For small translations, this can be achieved by shifting the origin of the Gaussian desired response G in (1). We call this *refined* estimated position $\mathbf{p}_{t,r}$.

Algorithm 1 details the stages involved in the tracking pipeline, while Figure 1 visually represents the steps that comprise our method. The likelihood map allows us to refine the position of the object from that predicted using a linear template (black cross) to the centroid of the object likelihood map (red circle).

This modification (called *object-aware* desired response) is surprisingly effective (*c.f.* §V) in correcting the drifts that naturally occur during tracking. Figure 2 gives an intuition of why this happens. To visually show the efficacy of our method, we run a vanilla correlation tracker based on greyscale features (like MOSSE [4]) until frame 187 of the *ball* sequence from VOT2014, where the object's estimated position becomes significantly imprecise. At that point, we run two trackers in parallel, one with (second row) and one without (first row) our modification. It is possible to observe how the instantaneous filters learnt with our approach are always better centred on the target object. This leads to a better global representation and, in practice, to a quick recover from the drift.

A. Object likelihood

Our method is agnostic to how the object likelihood L_t of (5) is obtained. In this paper, we assume that for every given frame I_t, the colour values of the pixels are i.i.d: we can then use a simple Bayes classifier to compute the per-pixel object likelihood, using as prior knowledge $\mathbf{k}$ the rectangular regions R_O and R_B, respectively representing the object extent $\mathcal{O}$ and its background $\mathcal{B}$, both derived from the estimated position of the target $\mathbf{p}_t$. From R_O and R_B, we compute the colour histograms H_O and H_B to model the

Algorithm 1: Tracking pipeline: iteration between frame $t-1$ and t

Data: Position $\mathbf{p}_{t-1}$, image I_t, global filter $\overline{\mathcal{H}}_{t-1}$, color models $\mathcal{O}_{t-1}$ and $\mathcal{B}_{t-1}$.
Result: Position $\mathbf{p}_t$, global filter $\overline{\mathcal{H}}_t$, color models $\mathcal{O}_t$ and $\mathcal{B}_t$.

while *not end of sequence* **do**

Training
Compute DFT of feature map F_t at $\mathbf{p}_t$ from I_t.
Compute object likelihood map L_t at $\mathbf{p}_t$ from I_t (7).
Compute centroid μ of L_t (5).
Refine position estimate $\mathbf{p}_t$ to $\mathbf{p}_{t,r}$ at μ.
Place the desired response peak at $\mathbf{p}_{t,r}$ and learn new filter $\overline{H}_t$ (2).
Update global filter $\overline{\mathcal{H}}_t$ (3).
Update object and background color models $\mathcal{O}_t$ and $\mathcal{B}_t$.
Testing
Compute DFT of feature map F_t at $\mathbf{p}_{t-1}$ from I_t.
Calculate filter response (4).
Estimation
The new bounding box position $\mathbf{p}_{t+1}$ is estimated at the location of the response peak (4).

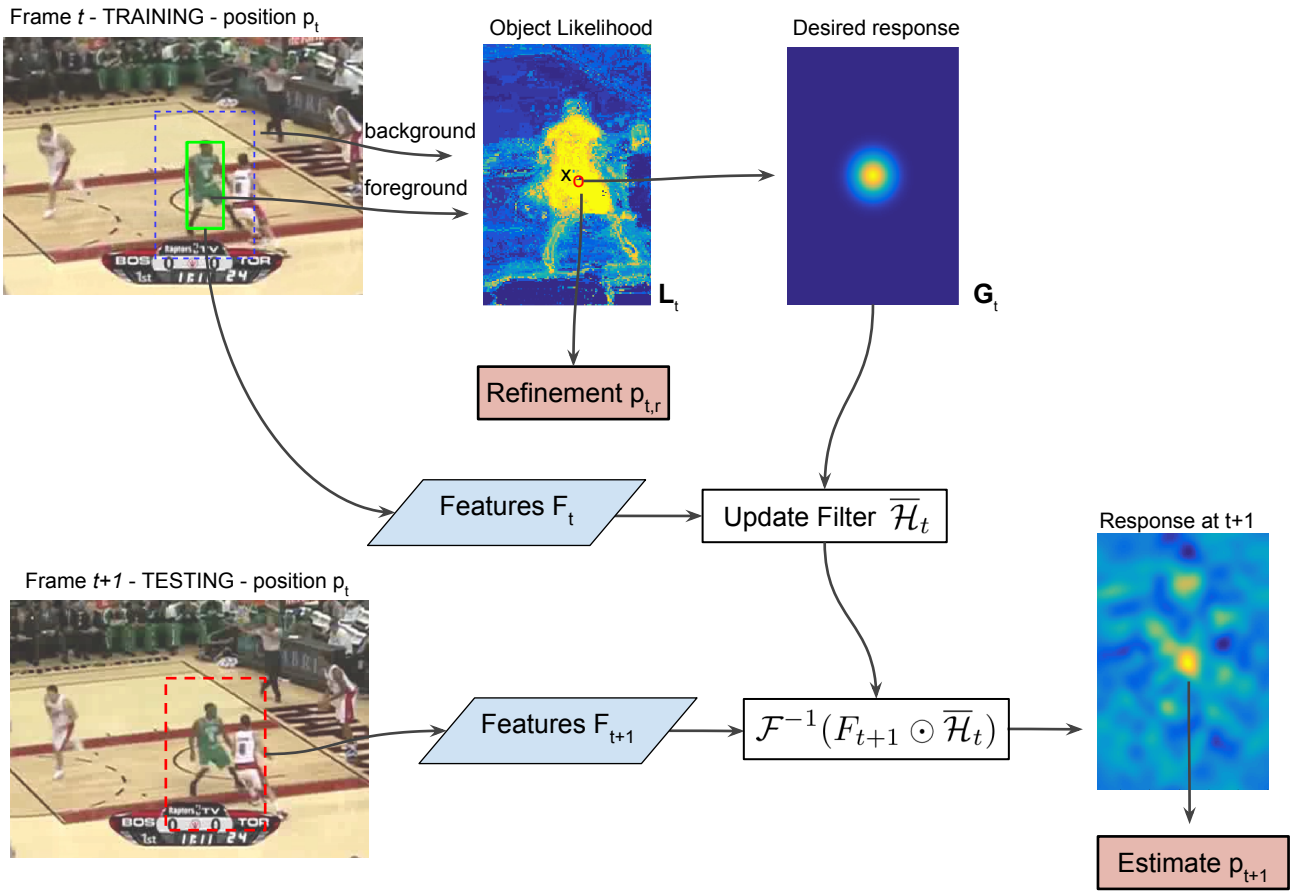

Fig. 1: *At each frame t, we use the current location p_t to extract a training patch from which we compute:* a) *the object likelihood map L_t and* b) *the feature map F_t. The centre of the desired response G_t is then placed on the centroid of L_t and the global filter $\overline{\mathcal{H}}_t$ is updated using (2) and (3). The position of the target in each frame is first estimated at the peak of the correlation between the updated filter and the testing patch (4) and then refined at the centre of mass of the likelihood map.*

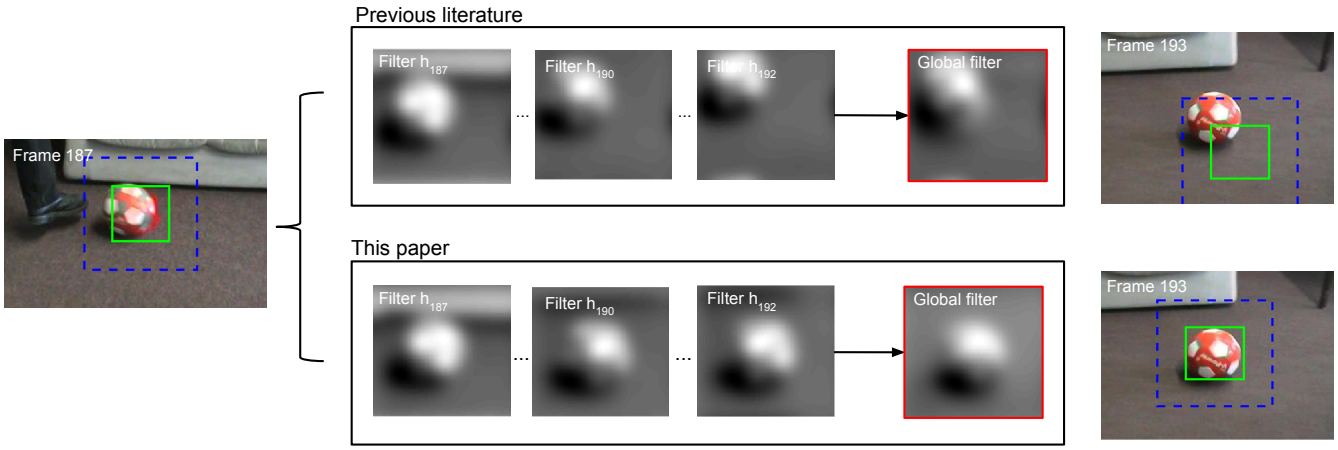

Fig. 2: The object-aware desired response of our method allows to quickly recover the target by learning, at each frame, a filter from non-drifted patches.

foreground object and its background, such that the colour components at location $\mathbf{x}$ are represented by the same bin index $b_{\mathbf{x}}$ in the two histograms.

$$L_t = P(\mathbf{x} \in \mathcal{O}|R_O, R_B) \approx P(\mathbf{x} \in \mathcal{O}|b_{\mathbf{x}}) = \frac{P(b_{\mathbf{x}}|\mathbf{x} \in \mathcal{O})P(\mathbf{x} \in \mathcal{O})}{P(b_{\mathbf{x}}|\mathbf{x} \in \mathcal{O})P(\mathbf{x} \in \mathcal{O}) + P(b_{\mathbf{x}}|\mathbf{x} \in \mathcal{B})P(\mathbf{x} \in \mathcal{B})}. \quad (6)$$

For a given pixel at location $\mathbf{x}$, using the histogram entries $H_O(b_{\mathbf{x}}), H_B(b_{\mathbf{x}})$ and the areas $|R_O|, |R_B|$, we can approximate: $P(b_{\mathbf{x}}|\mathbf{x} \in \mathcal{O}) \approx \frac{H_O(b_{\mathbf{x}})}{|R_O|}$, $P(b_{\mathbf{x}}|\mathbf{x} \in \mathcal{B}) \approx \frac{H_B(b_{\mathbf{x}})}{|R_B|}$, $P(\mathbf{x} \in \mathcal{O}) = \frac{|R_O|}{|R_O|+|R_B|}$ and $P(\mathbf{x} \in \mathcal{B}) = \frac{|R_B|}{|R_O|+|R_B|}$. This allows us to simplify (6) as follows:

$$L_t(\mathbf{x}) \approx P(\mathbf{x} \in \mathcal{O}|b_{\mathbf{x}}) \approx \frac{H_O(b_{\mathbf{x}})}{H_O(b_{\mathbf{x}}) + H_B(b_{\mathbf{x}})}. \quad (7)$$

In order to adapt to the target object during tracking, we update the target object and background colour models at the end of each frame using a running average approach similar to (3).

V. Experiments

A. Setup

In order to demonstrate the benefits of using an *object-aware* desired response, we test our proposed modification on several different setups:

- A vanilla correlation tracker like [4], using greyscale features and a linear kernel (MOSSE+OA).
- KCF, the multi-channel, kernelized version of [15], using fHOG [10] features and a linear kernel (DCF+OA).
- The scale adaptive version of [7], using fHOG [10] features and a linear kernel (DSST+OA).

For the correlation filters, we set the parameters to values that are standard in literature. Namely, we take the variance of the Gaussian desired response proportional to the target size, using a multiplicative factor of $\frac{1}{8}$ for the translation filter and $\frac{1}{4}$ for the scale filter. In the experiments related to the plots of Figures 3 and 4 we used a variable learning rate for the translation filter and a fixed learning rate of 0.025 for the scale. The features F_t from equation (1) are extracted from a rectangle whose sides are obtained by increasing the initial ground truth bounding box sides of a constant value for both dimension, equal to $\frac{w+h}{4}$. The regularization term in (2) is set to $\eta = 0.01$.

For the likelihood map, we use a three-dimensional histogram with 32 bins for each colour channel (R, G, B), and we update the colour models $\mathcal{O}$ and $\mathcal{B}$ from frame to frame with a running average with a fixed learning rate of 0.025 in all the experiments. In case of grayscale sequences, we use a one-dimensional histogram of 32 bins.

B. Benchmark

We perform our experiments on the Visual Object Tracking 2014 [21] and 2015 (VOT14 and VOT15) challenge datasets[3]. The videos of these datasets (25 for VOT14 and 60 for VOT15) were selected by clustering a pool of hundreds of sequences in order to exhibit a large variety of tracking challenges, including camera motion, illumination change, occlusion, size change and motion change. *Accuracy* and *robustness to failures* are used as measures of performance; they are evaluated against a per-frame ground truth with oriented bounding boxes. The per-frame accuracy is expresssed as average intersection over union (IoU) between the bounding box predicted by the tracker A^T and the ground truth A^G, computed as $\frac{A^T \cap A^G}{A^T \cup A^G}$, while the robustness simply counts the number of failures of the tracker per sequence. By default, a failure is detected when the IoU is zero, but it can be configured in the toolkit.

The accuracy and robustness measures of the VOT benchmarks have to be considered carefully because, in case of failure, the tracker is reinitialized by the toolkit using the ground truth. This means that a hypothetical tracker that failed every single frame would have an accuracy of 100%. Conversely, a tracker that always used the full frame as its

[3]We used the vot-toolkit from `github.com/votchallenge/vot-toolkit` at commit `77c0481` of the 30/07/2015.

Tracker	Accuracy	# Failures
DSST+OA (this paper)	0.572	1.94
DAT [29]	0.493	2.26
DSST [7]	0.552	2.40
KCF [15]	0.488	2.29
MOSSE [4]	0.452	3.14

TABLE II: Ranked results for VOT15. First, second and third entries for accuracy (higher is better) and number of failures (lower is better) over 60 sequences are reported.

estimated bounding box would have zero failures. For this reason, it is important to consider these two measures not only separately, in the context of applications that require either high accuracy or high robustness but not both, but also jointly, in order to be able to say which tracker is better in a one-to-one comparison.

C. Results

Firstly, we separately analyse accuracy and robustness. In Figures 3 and 4 we show how performance changes with the learning rate λ of (3). Recall that λ controls how to linearly combine the previously-learnt global filter $\overline{\mathcal{H}}_{t-1}$ with the new one ($\overline{H}_t$). The higher the value, the more importance is given to the latest learnt filters. Observe that the curves of all our modified methods (thick lines labelled *+OA*) have significantly and consistently better performance in terms of robustness than the standard methods (dashed lines). Moreover, it is interesting to observe how increasing the learning rate is only detrimental for the standard methods and not for the *+OAs*. Using an *object-aware* desired response makes the recently-learnt filters much more accurate (as it is illustrated by Figure 2), so that it becomes possible to rely more on them (*i.e.* increasing the learning rate λ) without drifting. In Figures 5 and 6 we show how the gap in performance is very evident also when different values of IoU are used to detect a failure. In particular, the gap in number of failures with higher IoU threshold, while the gap in accuracy shrinks.

In Table I, we report the best detected performance for each method (from Figures 3 and 4) to perform a direct comparison. The robustness of the *+OAs* is always significantly higher than that of the standard methods, with a relative improvement between 15.1% and 58.2%. Also the accuracy is improved, with an average of 2.8%.

In Table II, we report the ranked list obtained using the evaluation toolkit of VOT2015, which considers both measures of performance together [21]. Our DSST+OA (at $\lambda = 0.00175$) performs favourably against all the considered methods, *i.e.* the baseline MOSSE, the state-of-the-art correlation trackers KCF and DSST, and also the very recent DAT [29].

Finally, Figure 7 provides a qualitative comparison at a few critical phases of four sequences from the benchmark VOT14, showing that DCF+OA can cope very well with sudden change of appearance (rows 1-3) and long-range displacements (row 4).

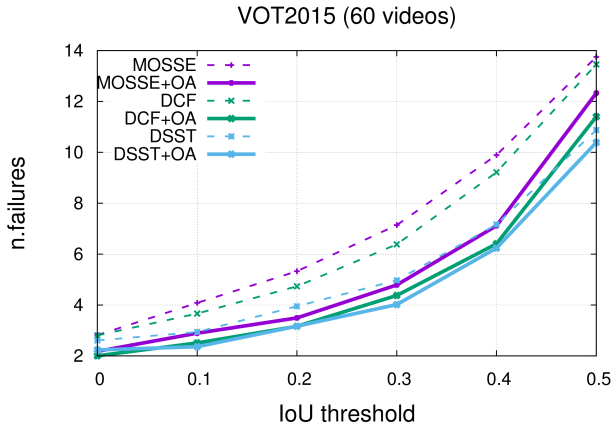

Fig. 5: *Number of failures in relation to the IoU threshold used to detect a failure (the lower the better). Learning rate fixed at* $\lambda = 0.1$.

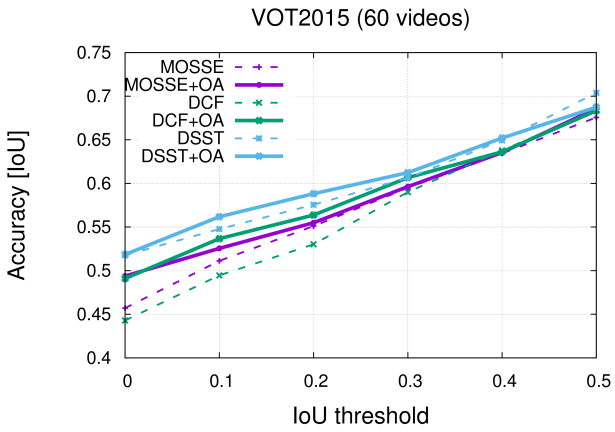

Fig. 6: *Accuracy in relation to the IoU threshold used to detect a failure (the higher the better). Learning rate fixed at* $\lambda = 0.1$.

VI. Conclusion

In this paper, we have shown that it is possible to significantly improve the performance of state-of-the-art correlation trackers by modifying them to optimise for a desired response based on a pixel-wise object likelihood map of the target.

Our paper offers a number of interesting avenues for further work. First, the estimation of the new position of the target in Algorithm 1 uses the filter response only to bootstrap the computation of the object likelihood map. We believe that a probabilistic fusion of the two estimations (the one from the correlation filter and the one from the object likelihood map) would lead to significantly higher performance, because it would implicitly mitigate the weaknesses of HOG (*e.g.* fast changes of appearance) and those of colour features (*e.g.* poor color distinctiveness). Moreover, the likelihood map can be exploited to achieve more accurate scale adaptation, that is not constrained by a fixed aspect ratio like DSST and DSST+OA.

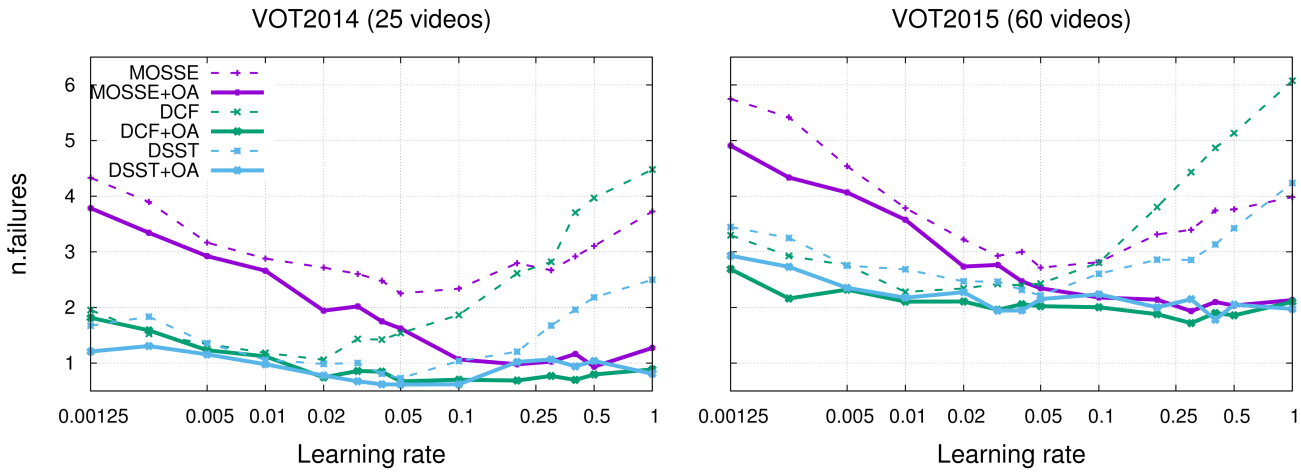

Fig. 3: *Number of failures in relation to the learning rate λ (the lower the better).*

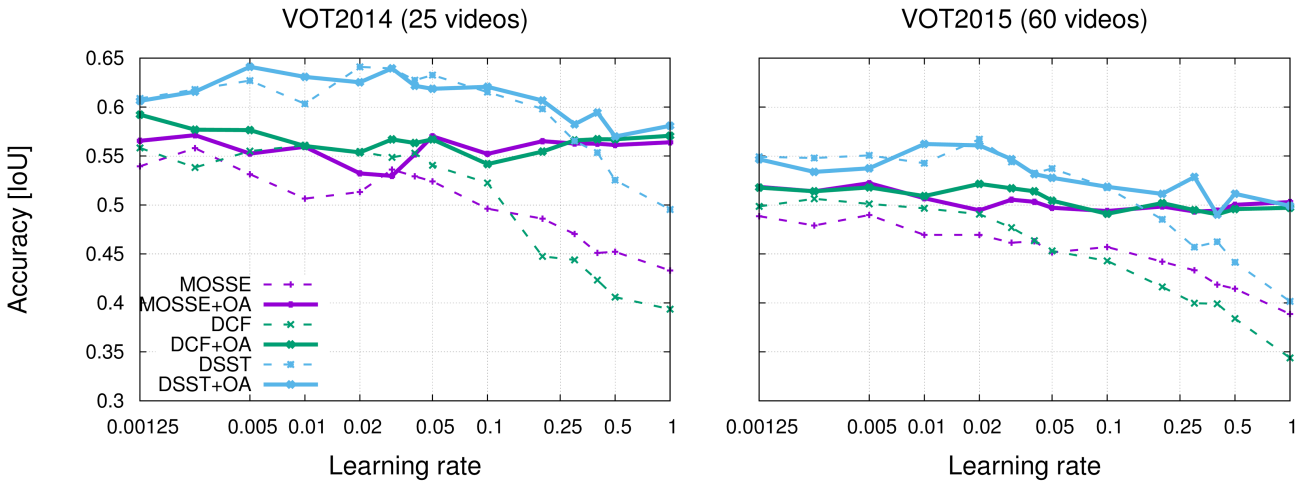

Fig. 4: *Accuracy in relation to the learning rate λ (the higher the better).*

TABLE I: The table reports, separately for robustness to failures and accuracy, the best measures scored by all the considered correlation trackers from Figures 3 and 4.

	MOSSE (VOT14)		**DCF** (VOT14)		**DSST** (VOT14)		**MOSSE** (VOT15)		**DCF** (VOT15)		**DSST** (VOT15)	
	failures	accuracy	failures	accuracy	failures	accuracy	failures	accuracy	failures	accuracy	failures	accuracy
standard	2.25	.558	1.06	.561	0.73	.641	2.72	.490	2.28	.506	2.22	.567
+OA (this paper)	0.94	.571	0.67	.592	0.62	.641	1.93	.522	1.72	.522	1.78	.562
improvement	58.2%	2.3%	36.8%	5.5%	15.1%	0%	29%	6.5%	24.6%	3.2%	19.8%	-0.9%

REFERENCES

[1] B. Babenko, M.-H. Yang, and S. Belongie. Robust Object Tracking with Online Multiple Instance Learning. *TPAMI*, 33(8), 2011.

[2] M. B. Blaschko and C. H. Lampert. Learning to localize objects with structured output regression. In *ECCV*, 2008.

[3] V. N. Boddeti, T. Kanade, and B. V. K. Kumar. Correlation Filters for Object Alignment. In *CVPR*, 2013.

[4] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual Object Tracking using Adaptive Correlation Filters. In *CVPR*, 2010.

[5] D. S. Bolme, B. A. Draper, and J. R. Beveridge. Average of Synthetic Exact Filters. In *CVPR*, 2009.

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[7] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Accurate Scale Estimation for Robust Visual Tracking. In *BMVC*, 2014.

[8] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Learning Spatially Regularized Correlation Filters for Visual Tracking. In *ICCV*, 2015.

[9] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer. Adaptive Color Attributes for Real-Time Visual Tracking. In *CVPR*, 2014.

[10] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object Detection with Discriminatively Trained Part Based Models. *TPAMI*, 32(9), 2010.

[11] J. Fernandez, B. Kumar, et al. Zero-Aliasing Correlation Filters. In *ISPA*, 2013.

[12] H. Grabner, C. Leistner, and H. Bischof. Semi-Supervised On-line Boosting for Robust Tracking. In *ECCV*, 2008.

[13] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured Output Tracking with Kernels. In *ICCV*, 2011.

[14] J. F. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond Hard

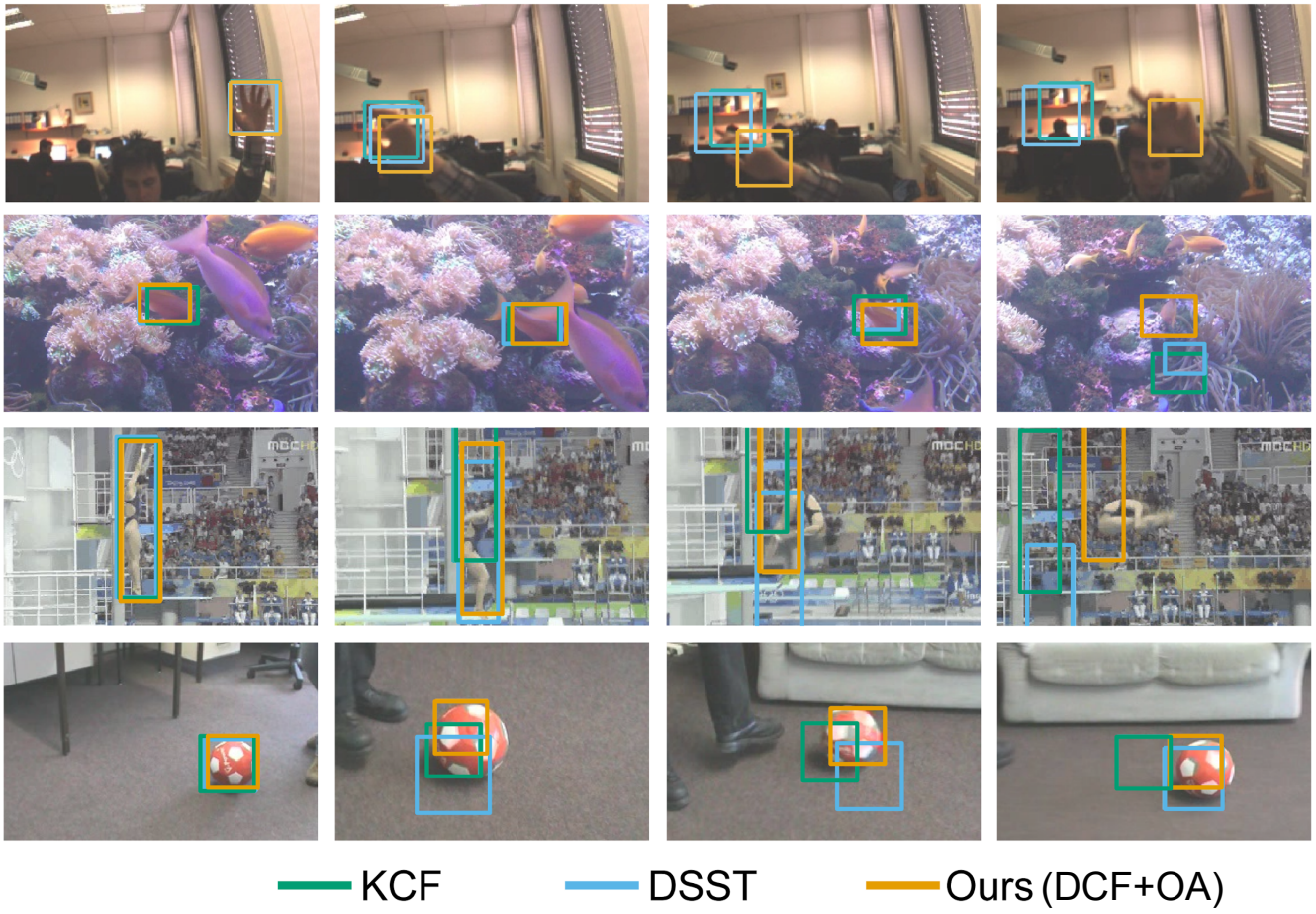

Fig. 7: *Qualitative results on hand1, fish2, diving and ball sequences from VOT2014.*

Negative Mining: Efficient Detector Learning via Block-Circulant Decomposition. In *ICCV*, 2013.

[15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-Speed Tracking with Kernelized Correlation Filters. *TPAMI*, 2015.

[16] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. MUlti-Store Tracker (MUSTer): a Cognitive Psychology Inspired Approach to Object Tracking. In *CVPR*, 2015.

[17] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-Learning-Detection. *TPAMI*, 34(7), 2012.

[18] H. Kiani Galoogahi, T. Sim, and S. Lucey. Multi-Channel Correlation Filters. In *ICCV*, 2013.

[19] H. Kiani Galoogahi, T. Sim, and S. Lucey. Correlation Filters with Limited Boundaries. In *CVPR*, 2015.

[20] M. Kristan et al. The Visual Object Tracking VOT2014 challenge results. In *ECCV*, 2014.

[21] 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. *arXiv preprint arXiv:1503.01313v2*, 2015.

[22] J. Kwon and K. M. Lee. Visual Tracking Decomposition. In *CVPR*, 2010.

[23] J. Kwon and K. M. Lee. Tracking by Sampling Trackers. In *ICCV*, 2011.

[24] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term Correlation Tracking. In *CVPR*, 2015.

[25] A. Mahalanobis, B. Vijaya Kumar, and D. Casasent. Minimum average correlation energy filters. *Applied Optics*, 26(17), 1987.

[26] D. Messerschmitt. Stationary points of a real-valued function of a complex variable. Technical Report UCB/EECS-2006-93, University of California, Berkeley, 2006.

[27] K. Nummiaro, E. Koller-Meier, and L. Van Gool. An adaptive color-based particle filter. *IVC*, 2003.

[28] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-Based Probabilistic Tracking. In *ECCV*, 2002.

[29] H. Possegger, T. Mauthner, and H. Bischof. In Defense of Color-based Model-free Tracking. In *CVPR*, 2015.

[30] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. PROST: Parallel Robust Online Simple Tracking. In *CVPR*, 2010.

[31] J. S. Supančič and D. Ramanan. Self-paced learning for long-term tracking. In *CVPR*, 2013.

[32] F. Tang, S. Brennan, Q. Zhao, and H. Tao. Co-Tracking Using Semi-Supervised Support Vector Machines. In *ICCV*, 2007.

[33] N. Wang and D.-Y. Yeung. Ensemble-Based Tracking: Aggregating Crowdsourced Structured Time Series Data. In *ICML*, 2014.

[34] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013.

[35] Y. Wu, J. Lim, and M.-H. Yang. Online Object Tracking: A Benchmark. In *CVPR*, 2013.

[36] J. Zhang, S. Ma, and S. Sclaroff. MEEM: Robust Tracking via Multiple Experts using Entropy Minimization. In *ECCV*, 2014.