Multi-Template Scale-Adaptive Kernelized Correlation Filters

Adel Bibi and Bernard Ghanem
King Abdullah University of Science and Technology (KAUST), Saudi Arabia
adel.bibi@kaust.edu.sa, bernard.ghanem@kaust.edu.sa

Abstract

This paper identifies the major drawbacks of a very computationally efficient and state-of- the-art-tracker known as the Kernelized Correlation Filter (KCF) tracker. These drawbacks include an assumed fixed scale of the target in every frame, as well as, a heuristic update strategy of the filter taps to incorporate historical tracking information (i.e. simple linear combination of taps from the previous frame). In our approach, we update the scale of the tracker by maximizing over the posterior distribution of a grid of scales. As for the filter update, we prove and show that it is possible to use all previous training examples to update the filter taps very efficiently using fixed-point optimization. We validate the efficacy of our approach on two tracking datasets, VOT2014 and VOT2015.

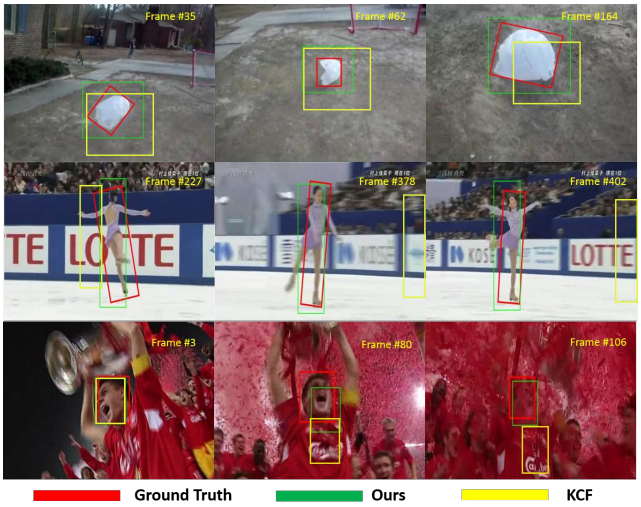

Figure 1: Comparsion between our method and KCF on 3 sequences from the VOT2015 dataset.

1. Introduction

Visual object tracking is a classical and very popular problem in computer vision with a plethora of applications such as vehicle navigation, human computer interface, human motion analysis, surveillance, and many more. The problem involves estimating the location of an initialized visual target in each frame of a video. Despite numerous object tracking methods that have been proposed in recent year [25, 21, 24, 22], most of these trackers suffer a degradation in performance mainly because of several challenges that include illumination changes, partial or full occlusion, motion blur, complex motion, out of plane rotation, and camera motion.

In this paper, we build upon a correlation based tracker popularly known as the Kernelized Correlation Filter (KCF) tracker [13, 12]. KCF has achieved impressive results on the visual tracking benchmark [24] and ranked third in the VOT2014 challenge [14] achieving real-time performance. KCF, similar to other correlation filter based trackers, tries to find the best filter taps that maximize the response when correlated with a target template that looks similar in appearance to the training data. KCF solves the problem of tracking by solving a simple rigid regression problem over training data in the dual form, which allows the use of both multi-dimensional features and nonlinear kernels (e.g. Gaussian).

The combination of multi-dimensional features and kernels when learning the filter taps pushed the tracker to be among the best performers. This is made possible because of the over-sampling strategy used in KCF. Instead of taking random samples of the target to train over, an over-sampling method is used to consider all possible translations of the target in a given window. This over-sampling was considered previously to be a drawback because of the large number of redundant samples that are required. However, when these samples are collected and organized properly, they form a circulant matrix that has very desirable properties, the most interesting of which is that its diagonalization can be efficiently computed using the DFT matrix. Using this over-sampling (in both training and detection), this DFT diagonalization property is used in formulating and solving the dual rigid regression problem entirely in the frequency domain. In fact, the only necessary operations are elements-wise addition, element-wise multiplication, and the FFT.

Due to the attractive properties highlighted above, KCF can accurately predict the target's location in the video frame in real-time. However, two performance-impeding drawbacks with KCF are its use of a fixed target scale in detection [15, 9] and a heuristic filter update rule that makes use of only one template at a time. We solve the issue of scaling through a voting scheme, which selects a scale that maximizes the posterior probability when the prior is Gaussian centered around the scale selected in the previous frame.

Our second contribution lies in how the filter taps are updated from frame-to-frame. In the landmark KCF paper [13, 12], it is argued that the filter taps can be computed in either the primal or the dual form each with its own limitations. Solving the problem in the primal form allows multiple base template in training, but only with one-dimensional features (i.e. gray scale images). On the other hand, solving it in the dual form allows the use of multi-dimensional features (e.g. HoG) and non-linear kernels, but only with a single training template. It was stated in [13] that using both multiple templates and multi-dimensional features is not possible. As can be seen in Table 1, using multiple templates in training with a linear kernel in the primal formulation gives very interesting results even outperforming KCF with non-linear kernels. In fact, using the primal formulation with multiple templates ranks first among all the considered trackers that use the same features. This observation motivates us to develop a kernelized correlation filter scheme that incorporates multiple multi-dimensional templates in training within the dual formulation, while reusing properties of the resulting circulant matrix structure. In this paper, we show that this can be done through an iterative scheme where each iteration solves for multiple filters each corresponding to a training example with a set of constraints that encourage these filters to be the same. By incorporating multiple templates (e.g. tracking results from previous frames), our approach allows for a more systematic update scheme for the filter taps, as compared to the heuristic update rule used in KCF, KCF based trackers, and many other correlation filter based trackers [13, 17, 16, 7]. This iterative update scheme does not tradeoff performance and accuracy for computational efficiency, owing to the underlying circulant matrix structures that arise. By integrating both updates, we demonstrate that our tracker is superior to the original KCF formulation and other state-of-the-art trackers.

2. Related Work

Trackers in general can be divided into two main categories: generative and discriminative trackers. Generative trackers adopt an appearance model to describe the target observations. The main aim of the tracker is to search for the target that is the most similar in appearance to the generative model. Therefore, the major contribution of this type of tracker is in developing complex generative representative models that can reliably describe the object even when it undergoes different appearance changes. Examples on generative models include mean shift tracker [8], incremental tracker (IVT) [20], fragment-based tracker (Frag) [2], L1-min tracker [19], multi-task tracker (MTT) [27], low-rank sparse tracker [26], and structural sparse tracking [28] to name a few.

	Acc. Rank	Rob. Rank	Rank
KCF_Gauss_HoG	3.23	2.83	3.03
KCF_Lin_HoG	3.35	2.79	3.07
KCF_Lin_Gray_Multi	**3.41**	**3.20**	**3.30**
KCF_Gauss_Gray	3.79	3.00	3.39
KCF_Lin_Gray	3.99	4.04	4.02

Table 1: Multi-templates vs single template updates

On the other hand, discriminative trackers formulate visual object tracking as a binary classification problem that searches for the target location that is the most distinctive from the background. Examples of discriminative trackers include multiple instance learning tracking (MIL) [5], ensemble tracking [4], support vector tracking [3], and all the correlation filter based trackers. Correlation filters have been used for a long time in classification problems, where the objective is to learn filter taps that minimize the energy response of the filter [18] or minimize the variance of the response over a set of given training examples [23]. The filter taps are usually computed in the time domain in a simple least squares formulation. Later, Bolme *et al.* proposed a method for learning the taps in the frequency domain [7], thus, achieving impressively high frame rates since all the necessary computations degenerate to elementary additions and multiplications. The correlation filter was extended even further to handle multi-dimensional features (beyond gray scale), when its kernelized version (KCF) was proposed in [13, 12]. KCF solves for the filter taps very efficiently by utilizing kernel functions and the circulant structure of the underlying kernel matrix. Since then, using correlation filters for tracking has attracted more attention in the vision community. For example, some trackers use multiple KCF trackers to represent different parts of the object and track them jointly [16]. In [17], they learn an online random fern classifier over a larger region to identify failures and re-detect the object in case of long term occlusion or when the target exits out of the field-of-view. Despite KCF's popularity and versatility, some drawbacks of the method remain. We investigate and address two of these drawbacks in this paper, namely multiple template training and scale adaptation.

We organize the rest of the paper as follows. Section 3 is a brief description of the KCF tracker, while Section 4 describes our method. Section 5 provides empirical evidence

for our method as compared to other trackers, while Section 6 concludes the paper.

3. Review of KCF Tracker

KCF tracker has gained attention recently for achieving very impressive results on the visual tracking benchmark [24], as well as, a high rank in the 2014 visual object tracking (VOT) competition [14]. Moreover, KCF has very attractive computational properties, since it can easily reach real-time frame rates. Much like other detection based trackers, KCF can be trained using a set of training templates. At the first frame, only one template is available. Some methods generate more templates by sampling patches around the first patch to collect more training data. On the other hand, KCF exhaustively samples the whole region around the target by taking advantage of cyclic shifts, which simulate translations of the target object.

Assuming for simplicity all examples are 1D and $\mathbf{x}$ represents the base template at the first frame, then $\mathbf{P}\mathbf{x} = [x_n, x_1, ..., x_{n-1}]^T$ represents one circular shift of that base patch, where $\mathbf{P}$ is a permutation matrix. Then, the set representing all possible circular shifts is given by $\{\mathbf{P}^i\mathbf{x}|i = 0, ..., n-1\}$. Obviously in case of 2D signals, there will be two possible shifts one in either direction. The matrix containing all possible circular shifts is a circulant matrix $\mathbf{X}$, which is also known as the data matrix.

The goal of KCF training is to learn the filter taps $\mathbf{w}$ that minimize the error between the filtered circular shifts of the training template $\mathbf{x}$ and the regression targets $\mathbf{y}$, which take on Gaussian values the highest of which is at the middle of the patch (i.e. when no circular shift exists).

$$\min_{\mathbf{w}} \sum_{i}^{n} (f(\mathbf{w}; \mathbf{P}^i\mathbf{x}) - y_i)^2 + \lambda ||\mathbf{w}||_2^2 \tag{1}$$

The representation function is given as $f(\mathbf{w}) = \mathbf{w}^T\phi(\mathbf{x})$, where $\phi(\mathbf{x})$ denotes a mapping of the template $\mathbf{x}$ into another space that can possibly be of a higher dimension. Therefore, Eq (1) can be re-written as,

$$\min_{\mathbf{w}} ||\Phi\mathbf{w} - \mathbf{y}||_2^2 + \lambda ||\mathbf{w}||_2^2 \tag{2}$$

where Φ contains the mapping of all the circular shifts of template $\mathbf{x}$. The most efficient solution to Eq (2) highly depends on the nature of the mapping function $\phi(.)$.

ϕ is linear. In this case, $\phi(\mathbf{x}) = \mathbf{x}$, $\Phi = \mathbf{X}$, and the solution is known to be: $\mathbf{w} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$. In fact, Eq (2) can be solved directly and efficiently in terms of the filter taps $\mathbf{w}$ (primal formulation) by exploiting the circulant matrix structure of $\mathbf{X}$ (i.e. it can be diagonalized using the DFT matrix), as follows:

$$\hat{\mathbf{w}}^* = \frac{\hat{\mathbf{x}}^* \odot \hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda} \tag{3}$$

where $\hat{\mathbf{z}}$ and $\hat{\mathbf{z}}^*$ denote the FFT of signal $\mathbf{z}$ and its complex conjugate, respectively. Interestingly, when *multiple* templates are available in training, it can easily be shown that the optimal filter taps can be computed in a similar way [6]:

$$\hat{\mathbf{w}}^* = \frac{\sum_{j=1}^{m} \hat{\mathbf{x}}_j^* \odot \hat{\mathbf{y}}}{\sum_{j=1}^{m} \hat{\mathbf{x}}_j^* \odot \hat{\mathbf{x}}_j + \lambda} \tag{4}$$

ϕ is non-linear. In this case, the single template $\mathbf{x}$ can be chosen to have multiple features (e.g. HoG), but the solution to Eq (2) cannot be computed efficiently in the primal form. However, using a kernel decomposition of the filter taps to transform the problem into its *dual* form, which is also a ridge regression problem. The dual variable solution α for this kernelized version, $\mathbf{w} = \Phi^T\alpha$, in the dual form can be shown to be: $\alpha = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y}$, where $\mathbf{K} = \Phi_{\mathbf{x}}\Phi_{\mathbf{x}}^T$ which is known as the kernel matrix. This kernel matrix evaluates the kernel (e.g. Gaussian) on all pairs of the circular shifts of the template $\mathbf{x}$, thus, these shifts can contain multiple features, as opposed to the case when ϕ is linear. As shown in [13], if the kernel used is permutation invariant, matrix $\mathbf{K}$ is also circulant and thus we can exploit its DFT diagonalization property to compute the dual solution α (in the frequency domain) efficiently, as follows:

$$\hat{\alpha}^* = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{xx} + \lambda} \tag{5}$$

where $\hat{\mathbf{k}}^{xx}$ is defined to be the FFT of the kernel correlation, and $\hat{\mathbf{w}}$, $\hat{\mathbf{y}}$ denotes the FFT of the filter taps and the target patches respectively.

Target Detection. After the optimal taps are learned for the template $\mathbf{x}$, some variants of KCF trackers update these taps as a convex combination of these taps and those from a previous frame [13, 7, 15]. In this way, historical information can be propagated, despite the fact that it is done heuristically and non-adaptively. Detecting the target in the next frame can be simply done by applying the filter $\mathbf{w}$ to some search regions in the frame. In other words, we apply $f(\mathbf{z}; \mathbf{w}) = \mathbf{w}^T\phi(\mathbf{z})$ where $\mathbf{z}$ is the sample to be evaluated over, and $\Phi_{\tilde{\mathbf{x}}}$ is the latest model that has been updated. If the dual form of the solution is computed, then the detection formula is: $f(\mathbf{z}; \alpha) = \alpha^T\Phi_{\tilde{\mathbf{x}}}\phi(\mathbf{z})$. To measure the response over all circular shifts (i.e. translations) of the signal $\mathbf{z}$, the mapping of these circular shifts is computed in the same way it is done in the training, namely using

a circulant matrix Φ_z. In this case, all the filter responses (in the frequency domain) can be computed efficiently: $\hat{\mathbf{f}}(\mathbf{z};\alpha) = (\hat{k}^{\tilde{\mathbf{x}}\mathbf{z}} \odot \hat{\alpha})$, where $\hat{k}^{\tilde{\mathbf{x}}\mathbf{z}}$ is the FFT of $\Phi_{\tilde{\mathbf{x}}}\Phi_z^T$. It is worthwhile to note that it was stated in [13] that solving the ridge regression problem in Eq (2) for multi-dimensional feature vectors with non-linear kernels, as well as, multiple templates is not possible. We will address this issue next.

4. Multiple Template Scale Adaptive KCF

The following subsections will provide a detailed derivation of our proposed tracker.

4.1. Multiple Templates

Although the primal formulation of the rigid regression problem allows for training with multiple templates (i.e. more than one circulant matrix $\mathbf{X}$) [6], it does not directly allow the use of multi-dimensional features and non-linear kernels. Experimental results in Table 1 demonstrate a significant performance improvement when using multiple templates in training as compared to using only one, while utilizing the same features. In the seminal KCF work [13], the authors argue that multiple templates are feasible only for linear kernels in the primal formulation. On the other hand, the dual form allows for non-linear, multidimensional features, but using only one template. They argue that using both is not possible. In this paper, we show how filter taps can be computed using multiple templates with multi-dimensional features and non-linear kernels in the dual formulation.

We formulate the problem as follows. Ideally, at frame $n+1$, there are n samples, on which the filter $\mathbf{w}$ should be trained. In this case, the data matrix containing the templates and all their shifted versions is no longer circulant, but rather blockwise circulant. In fact, the augmented data matrix $\mathbf{X} = [\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_n]^T$, where each $\mathbf{X}_i$ is a circulant matrix generated from the i^{th} template (i.e. the detected patch at frame i). Therefore, the multiple template kernelized correlation problem can be formulated as:

$$\min_{\mathbf{w}} \left\| \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_n \end{pmatrix} \mathbf{w} - \begin{pmatrix} \mathbf{y} \\ \mathbf{y} \\ \vdots \\ \mathbf{y} \end{pmatrix} \right\|_2^2 + \lambda ||\mathbf{w}||_2^2 \qquad (6)$$

Note that Φ_i is the mapping matrix evaluated for template $\mathbf{x}_i$ and its circulant shifts. Since $\mathbf{X}_i$ is circulant, Φ_i is also circulant. In what follows, we will provide details on how this optimization problem can be solved iteratively using the inherent properties of block circulant matrices. Without loss of generality, our derivation will be highlighted for 1-D signals with two training examples (i.e. $n = 2$), but the results can be easily extended to 2-D signals and to multiple training examples. The resulting problem can then be written as:

$$\min_{\mathbf{w}} \left\| \begin{pmatrix} \Phi_1 \\ \Phi_2 \end{pmatrix} \mathbf{w} - \begin{pmatrix} \mathbf{y} \\ \mathbf{y} \end{pmatrix} \right\|_2^2 + \lambda ||\mathbf{w}||_2^2 \qquad (7)$$

By adding auxiliary variables, Eq (7) can be re-written by considering two independent filters under the constraint that these filters are the same, as follows:

$$\min_{\mathbf{w}_1, \mathbf{w}_2} \left\| \begin{pmatrix} \Phi_1\mathbf{w}_1 - \mathbf{y} \\ \Phi_2\mathbf{w}_2 - \mathbf{y} \end{pmatrix} \right\|_2^2 + \lambda \left\| \begin{pmatrix} \mathbf{w}_1 \\ \mathbf{w}_2 \end{pmatrix} \right\|_2^2 \qquad (8)$$
$$\text{subject to:} \quad \mathbf{w}_1 = \mathbf{w}_2.$$

This problem can be solved by replacing the hard constraint with a soft one in the objective: $\frac{\mu}{2}||\mathbf{w}_1 - \mathbf{w}_2||_2^2$. This additional regularizer encourages that both filters be the same. The value of μ is increased in each iteration. In each iteration, we solve for both $\mathbf{w}_1$ and $\mathbf{w}_2$ via alternating fixed-point optimization. The method starts by initializing a solution for $\mathbf{w}_2$ and uses that to update $\mathbf{w}_1$. Then, we use the updated $\mathbf{w}_1$ to solve for $\mathbf{w}_2$, so on and so forth, until a stopping criterion has been met. Since the original problem is convex, this strategy is guaranteed to converge to a global minimum. We initialize the filters with the solution to the single-template KCF. Therefore, in the j^{th} iteration, we solve the following two coupled problems:

$$\min_{\mathbf{w}_1} \; ||\Phi_1\mathbf{w}_1 - \mathbf{y}||_2^2 + \lambda||\mathbf{w}_1||_2^2 + \mu||\mathbf{w}_1 - \mathbf{w}_2^j||_2^2 \qquad (9)$$
$$\min_{\mathbf{w}_2} ||\Phi_2\mathbf{w}_2 - \mathbf{y}||_2^2 + \lambda||\mathbf{w}_2||_2^2 + \mu||\mathbf{w}_2 - \mathbf{w}_1^{j+1}||_2^2$$

Solving Eq (9) Note that the problems above have the exact same form, so we will only derive the solution to one of them. To solve Eq (9), we formulate its dual in Eq (10) to allow for non-linear kernel functions and multidimensional features.

$$\min_{\mathbf{w}_1} \sum_i ({\mathbf{w}_1}^T\phi(\mathbf{x_i}) - y_i)^2 + \lambda||\mathbf{w}_1||_2^2 + \mu||\mathbf{w}_1 - \mathbf{b}||_2^2 \qquad (10)$$

Here, we take $\mathbf{b} = \mathbf{w}_2^j$ for readability. Following the same strategy as in [13], we set the gradient of Eq (10) to zero and identify the left hand side to the right. By doing so, we obtain: $\mathbf{w}_1 = \Phi_1^T\mathbf{a}_1 + k\mathbf{b}$, where $a_1^i = -\frac{1}{\lambda+\mu}({\mathbf{w}_1}^T\phi(\mathbf{x_i}) - y_i)$ and $k = \frac{\mu}{\lambda+\mu}$. By substituting the dual formulation of $\mathbf{w}_1$ into Eq (10) and setting its gradient to zero, we obtain the following linear system:

$$(\Phi_1\Phi_1^T + (\lambda+\mu)\mathbf{I})\mathbf{a}_1 = \mathbf{y} - (k\mathbf{I} + (\lambda k + \mu(k-1))(\Phi_1\Phi_1^T)^{-1})\tilde{\mathbf{b}} \qquad (11)$$

where $\tilde{\mathbf{b}} = \Phi_1 \mathbf{b} = \Phi_1 \Phi_2^T \mathbf{a}_2$. Now Eq (11) looks very similar to the solution derived using a single training sample in the original KCF formulation but with an extra additive term. A similar approach can be used to solve this linear system efficiently. Using the FFT diagonalization of both $\Phi_1 \Phi_1^T$ and $\Phi_1 \Phi_2^T$ (both are circulant matrices), we can invert the left hand side and compute the right hand side efficiently, as follows:

$$\tilde{\mathbf{b}} = \mathbf{F} diag(\hat{k}^{x_2 x_1}) \mathbf{F}^H \mathbf{a}_2 = \mathbf{F}(\hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \tag{12}$$

where $\hat{\mathbf{k}}^{x_2 x_1}$ is the FFT of the correlation kernel vector between the two signals $\mathbf{x}_1$ and $\mathbf{x}_2$. Similarly,

$$(\Phi_1 \Phi_1^T)^{-1} \tilde{\mathbf{b}} = \mathbf{F}((\hat{k}^{x_1 x_1})^{-1} \odot \hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \tag{13}$$

This result extends easily to 2-D signals and to multiple features, as in [13]. Therefore, the final solution would be very similar to the solution given in [13] with a modified target vector. In this way, we avoid using heuristics to combine the effect of the current template with historical information as done in previous work [13, 7, 15].

$$\hat{\mathbf{a}}_1 = \frac{\hat{\psi}}{\hat{\mathbf{k}}^{x_1 x_1} + (\lambda + \mu)} \tag{14}$$

where $\hat{\psi} = \mathbf{F}(-(k\mathbf{I} + (\lambda k + \mu(k-1))(\Phi_1 \Phi_1^T)^{-1}) \tilde{\mathbf{b}} + \mathbf{y})$.

As stated earlier, we update $\mathbf{w}_1$ and $\mathbf{w}_2$ (equivalently $\hat{\mathbf{a}}_1$ and $\hat{\mathbf{a}}_2$) by increasing the tradeoff coefficient μ in each iteration. Upon convergence, we obtain the solution to the original problem in Eq (7). The stopping criterion we use in our experiments depends on the variation in the overall objective cost in previous iterations. Once that variation is small enough, the optimization process is terminated. Here, we note that evaluating the cost function efficiently is non-trivial; however, we can use diagonalization ideas similar to those seen earlier to compute it very quickly.

4.2. Scale Integration

One of the main drawbacks of KCF is that it does not address the target scale issue. We use a simple yet effective method (similar in spirit to [15, 9]) to counteract this issue. Specifically, we apply max-pooling over multiple scales in the detection phase of the tracker. The main difference between our approach and previous work is that we maximize over the posterior probability instead of the likelihood, as illustrated in the following equation.

$$\max_i \; P(s_i|\mathbf{y}) = P(\mathbf{y}|s_i) P(s_i) \tag{15}$$

where s_i represents the i^{th} scale and $P(\mathbf{y}|s_i)$ is the likelihood that is defined by the maximum detection response at the i^{th} scale: $\hat{\mathbf{f}}(\mathbf{z}) = (\hat{k}^{\hat{\mathbf{x}}\mathbf{z}} \odot \hat{\alpha})$. The prior term $P(s_i)$ is assumed to follow a Gaussian distribution, which is centered around the previous scale and has a standard deviation σ which is set experimentally. This will allow for smooth transition between scales, since the target's scale is assumed to not change much between consecutive frames. This strategy produces more stable detections than previous methods [15, 9] that only use the likelihood.

5. Experimental Results

We conduct two experiments to evaluate the efficiency and accuracy of our proposed tracker. First, we compare our tracker against state-of-art trackers that participated in the VOT2014 challenge [14], which comprises 25 challenging sequences. Secondly, we evaluate our tracker using the VOT2015 toolkit [1] on a set of 60 challenging videos. We show how each of the proposed modifications to the original KCF tracker (i.e. the use of multiple templates in training and scale adaptation in detection) leads to improvement in overall performance.

5.1. Features and Parameters

In the implementation, we used HoG features with a Gaussian kernel function. The σ used in the Gaussian kernel is set to 0.5, as in [13]. The HoG cell size is 4x4 and the number of orientation bins is 9. The extracted feature vector is multiplied by a Hanning window, as described in [7]. The regularization parameter over the energy term is set experimentally to $\lambda = 10^{-4}$. Also, we set $\mu = 10^{-5}$ as an initial value and it is doubled every iteration. The search grid of scales is set to be $[0.76, 0.80...1.20, 1.24]$ of the original size of the target. The stopping criterion used in the optimization of Section 4.1 is based on the standard deviation of the overall cost in the past 5 iterations. The stopping threshold for this standard deviation is set to $\eta = 10^{-5}$.

5.2. Experimental Setup

In all our experiments, we run our proposed approach in MATLAB on an Intel(R) Xeon(R) 2.67GHz CPU with 32GB RAM.

Datasets: We conduct the experiments on 2 different datasets, namely VOT2014[14] and VOT2015[1]. VOT2014 has 25 annotated videos in total, while VOT2015 comprises 60 videos. Both datasets pose challenging problems to object tracking, including partial occlusion, illumination change, motion blur, and background clutter. Both datasets have annotations to account for non-standard rectangles that can be rotated or scaled.

Evaluation Methodology: We use the VOT2015 toolkit to evaluate the results for both VOT2014 and VOT2015. The toolkit reports three measures of evaluation: accuracy,

robustness, and speed, along with the overall tracking score. Our proposed tracker is deterministic, so it is evaluated by the toolkit three consecutive times and the average score over the three turns is recorded. As for accuracy, we use the same criterion used in the Pascal VOC, namely the overlap ration (VOR) [10], which is defined as:

$$\text{VOR} = \frac{area(ROI_{T_i} \cap ROI_{G_i})}{area(ROI_{T_i} \cup ROI_{G_i})} \tag{16}$$

where T and G denote the target and ground truth bounding boxes respectively. As for robustness, it measures the number of times the tracker loses the object. Therefore, a robustness score of 0 means the tracker does not lose the object at all, during the video throughout the 3 runs. A failure is defined when the overlap measure is below some threshold for a certain number of consecutive frames. The toolkit automatically re-initializes the tracker when a failure happens allowing the tracker to resume from the re-initialized frame. Obviously, a smaller robustness score is desired. The toolkit also reports the speed of the tracker in processing each frame.

5.3. Results

VOT2014 Experiments: We compare our method with some of the available results of the state-of-art trackers on the VOT2014 dataset. The trackers we used for comparisons were NCC (baseline), Struck [11], MIL[5], IVT[20], KCF [13], and KCF_Scale trackers. In the VOT2014 challenge, a scaled version of KCF was introduced which we denote as KCF_Scale. Since the parameter setup for this tracker is not available and to ensure a fair comparison with our method, we run KCF_Scale with the same methodology used in VOT2014 but with the same parameters we use in our tracker. It is clear from Table 2 that our method (incorporating KCF with multiple templates and scale adaptation) substantially surpasses all the other trackers in both accuracy and robustness.

	Acc. Rank	Rob. Rank	Rank	FPS
Ours	2.68	2.96	2.82	25.1
KCF_Scale	3.00	3.48	3.24	47.42
KCF	3.46	3.13	3.29	66.5
Struck	3.68	3.85	3.75	19.77
MIL	5.02	3.88	4.45	1.94
IVT	5.00	4.65	4.81	27.51
NCC	5.21	6.08	5.65	27.9

Table 2: Comparison on the VOT2014 dataset.

VOT2015 Experiments: Since the tracking results of other trackers has not been released on VOT2015 just yet, we use this dataset to demonstrate how each of our proposed components (multiple templates and scale adaptation) contributes to the overall improvement of our approach over the original KCF tracker. To do this, we compare the KCF tracker with: **(KCF+MT)** which is our multiple template version of the KCF (refer to Section 4.1) and **(KCF+MT+Sc)** which is the latter tracker but with scale adaptation (refer to Section 4.2). In Table 3, we compare the performance and runtime of these three trackers. Clearly, adding any of the two components to the original KCF tracker improves its performance. By adding both, the improvement in performance is even more significant. It may be noted that the improvement by adding the multiple template update may not be as significant as for adding the scale for this dataset. This is primarily due to the fact that this dataset in particular includes examples of a target object undergoing substantial variations in scale. The effect of adding the MT component is more evident in the VOT2014 dataset, where less scale variations are encountered. For a detailed per-video comparison, we show the accuracy of our method and KCF on all 60 VOT2015 videos in Figure 2. It is clear from the plot that our tracker outperforms KCF in the majority of videos.

	Acc. Rank	Rob. Rank	Rank	FPS
KCF+MT+Sc	1.80	1.95	1.88	24.3
KCF+MT	2.04	2.04	2.04	31.54
KCF	2.16	2.01	2.08	44.76

Table 3: Comparisons on the VOT2015 dataset.

6. Conclusion

We have identified the main drawbacks of the KCF tracker that cause failure and we propose two components to address these drawbacks. First, we show that it is possible to incorporate multiple multi-dimensional templates in computing the optimal filter taps. By reformulating the kernel correlation problem and by using fixed-point optimization, we demonstrate that using multiple templates improves the performance of the original KCF tracker that uses only one template and a heuristic update scheme. Second, we address the problem of fixed scale tracking. We use a similar grid search approach as in previous methods, but we week the scale with maximum posterior instead of likelihood. This subtle difference makes the tracker more robust to gradual scale changes. Our experimental results on VOT2014 and VOT2015 show that our tracker substantially outperforms many state-of-the-art trackers.

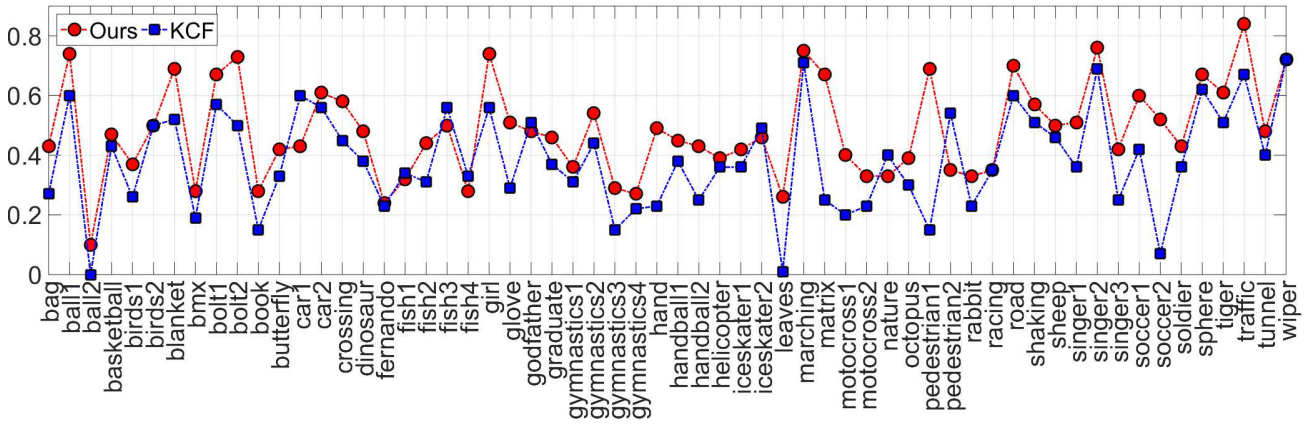

Figure 2: Accuracy results on VOT2015 dataset for 60 videos, comparing our proposed method and KCF.

7. Appendix

7.1. Derivation of Eqs (12) and (13)

This section shows a detailed derivation of Eq (12) as follows:

$$\tilde{\mathbf{b}} = (\Phi_1 \Phi_2^T)\mathbf{a}_2 = \mathbf{F} diag(\hat{k}^{x_2 x_1})\mathbf{F}^H \mathbf{a}_2 \quad (17)$$

$$= \mathbf{F} diag(\hat{k}^{x_2 x_1})\hat{\mathbf{a}}_2^* \quad (18)$$

$$= \mathbf{F}(\hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \quad (19)$$

and for Eq. 13:

$$(\Phi_1 \Phi_1^T)^{-1}\tilde{\mathbf{b}} = (\mathbf{F} diag(\hat{k}^{x_1 x_1})\mathbf{F}^H)^{-1}\hat{\mathbf{b}} \quad (20)$$

$$= \mathbf{F} diag^{-1}(\hat{k}^{x_1 x_1})\mathbf{F}^H \hat{\mathbf{b}} \quad (21)$$

$$= \mathbf{F} diag^{-1}(\hat{k}^{x_1 x_1})(\hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \quad (22)$$

$$= \mathbf{F}((\hat{k}^{x_1 x_1})^{-1} \odot \hat{k}^{x_2 x_1} \odot \hat{\mathbf{a}}_2^*) \quad (23)$$

7.2. Efficient Computation of Objective Function

Here, we show a more detailed derivation for how to efficiently compute the regression cost that will be used in the stopping criterion for the alternating fixed-point optimization of Eq. (10)

$$||\Phi_2 \mathbf{w}_2 - \mathbf{y}||_2^2 = (\Phi_2 \mathbf{w}_2 - \mathbf{y})^H (\Phi \mathbf{w}_2 - \mathbf{y}) \quad (24)$$

$$= \mathbf{w}_2^H \Phi_2^H \Phi_2 \mathbf{w}_2 - \mathbf{w}_2^H \Phi_2^H \mathbf{y} - \mathbf{y}^H \Phi_2 \mathbf{w}_2 + \mathbf{y}^H \mathbf{y} \quad (25)$$

$$= \mathbf{a}_2^H \Phi_2 \Phi_2^H \Phi_2 \Phi_2^H \mathbf{a}_2 - \mathbf{a}_2^H \Phi_2 \Phi_2^H \mathbf{y} - \mathbf{y}^H \Phi \Phi^H \mathbf{a}_2 + \mathbf{y}^H \mathbf{y} \quad (26)$$

$$= \mathbf{a}_2^H \mathbf{F}(\hat{\mathbf{k}}^{22} \odot \hat{\mathbf{k}}^{22} \odot \hat{\mathbf{a}}_2^*) - \mathbf{a}_2^H \mathbf{F}(\hat{\mathbf{k}}^{22} \odot \hat{\mathbf{y}}_2^*) - \mathbf{y}^H \mathbf{F}(\hat{\mathbf{k}}^{22} \odot \hat{\mathbf{a}}^*) + \mathbf{y}^H \mathbf{y} \quad (27)$$

References

[1] The vot 2015 evaluation kit. http://www.votchallenge.net.

[2] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 798–805. IEEE, 2006.

[3] S. Avidan. Support vector tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 26(8):1064–1072, 2004.

[4] S. Avidan. Ensemble tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 29(2):261–271, 2007.

[5] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 983–990. IEEE, 2009.

[6] V. N. Boddeti, T. Kanade, and B. Kumar. Correlation filters for object alignment. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2291–2298. IEEE, 2013.

[7] D. S. Bolme, J. R. Beveridge, B. Draper, Y. M. Lui, et al. Visual object tracking using adaptive correlation filters. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 2544–2550. IEEE, 2010.

[8] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(5):564–577, 2003.

[9] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.

[10] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[11] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 263–270. IEEE, 2011.

[12] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *Computer Vision–ECCV 2012*, pages 702–715. Springer, 2012.

[13] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 37(3):583–596, 2015.

[14] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, L. Čehovin, G. Nebehay, T. Vojíř, G. Fernandez, A. Lukežič, A. Dimitriev, et al. The visual object tracking vot2014 challenge results. In *Computer Vision-ECCV 2014 Workshops*, pages 191–217. Springer, 2014.

[15] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *Computer Vision-ECCV 2014 Workshops*, pages 254–265. Springer, 2014.

[16] T. Liu, G. Wang, and Q. Yang. Real-time part-based visual tracking via adaptive correlation filters. *Intelligence*, page 2345390, 2015.

[17] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5388–5396, 2015.

[18] A. Mahalanobis, B. Vijaya Kumar, and D. Casasent. Minimum average correlation energy filters. *Applied Optics*, 26(17):3633–3640, 1987.

[19] X. Mei and H. Ling. Robust visual tracking using &# x2113; 1 minimization. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1436–1443. IEEE, 2009.

[20] T. Poggio and G. Cauwenberghs. Incremental and decremental support vector machine learning. *Advances in neural information processing systems*, 13:409, 2001.

[21] S. Salti, A. Cavallaro, and L. D. Stefano. Adaptive appearance modeling for video tracking: Survey and evaluation. *Image Processing, IEEE Transactions on*, 21(10):4334–4348, 2012.

[22] A. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, July 2014.

[23] B. Vijaya Kumar. Minimum-variance synthetic discriminant functions. *JOSA A*, 3(10):1579–1584, 1986.

[24] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *Computer vision and pattern recognition (CVPR), 2013 IEEE Conference on*, pages 2411–2418. IEEE, 2013.

[25] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[26] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Low-rank sparse learning for robust visual tracking. In *Computer Vision–ECCV 2012*, pages 470–484. Springer, 2012.

[27] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja. Robust visual tracking via multi-task sparse learning. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2042–2049. IEEE, 2012.

[28] T. Zhang, S. L. C. Xu, S. Yan, B. Ghanem, N. Ahuja, and M.-H. Yang. Structural sparse tracking.