Multi-kernel Correlation Filter for Visual Tracking

Ming TANG and Jiayi FENG
National Lab of Pattern Recognition, Institute of Automation,
Chinese Academy of Sciences, Beijing 100190, China
{tangm,jyfeng}@nlpr.ia.ac.cn *

Abstract

Correlation filter based trackers are ranked top in terms of performances. Nevertheless, they only employ a single kernel at a time. In this paper, we will derive a multi-kernel correlation filter (MKCF) based tracker which fully takes advantage of the invariance-discriminative power spectrums of various features to further improve the performance. Moreover, it may easily introduce location and representation errors to search several discrete scales for the proper one of the object bounding box, because normally the discrete candidate scales are determined and the corresponding feature pyramid are generated ahead of searching. In this paper, we will propose a novel and efficient scale estimation method based on optimal bisection search and fast evaluation of features. Our scale estimation method is the first one that uses the truly minimal number of layers of feature pyramid and avoids constructing the pyramid before searching for proper scales.

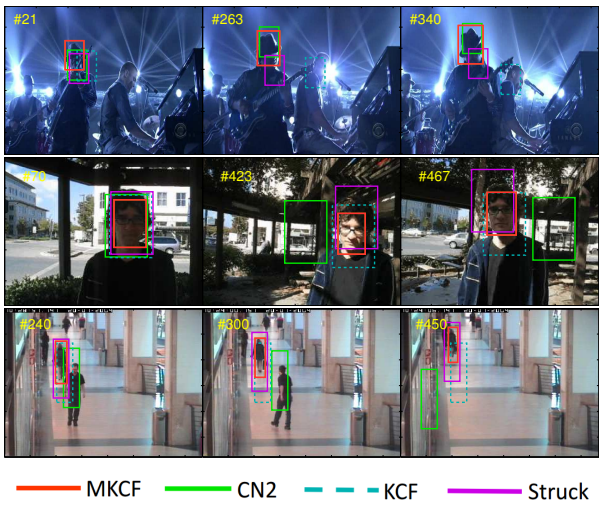

Figure 1. Qualitative comparisons of our Multi-kernel Correlation Filter (MKCF) with state-of-the-art trackers, Struck [13], KCF [15], and CN2 [7] on 3 sequences [36]. The figure is best viewed in color.

1. Introduction

Visual object tracking is one of the most challenging problems in computer vision [32, 18, 20, 26, 23]. To adapt to unpredictable variations of object appearance and background during tracking, one strategy is to select a single strong feature that is robust to any variation. However, this has been known to be difficult [33, 11], especially for the model-free tracking task in which no prior knowledge about the target object is known except for the initial frame. Therefore, designing an efficient scheme to combine several complementary features is a natural alternative.

In such a natural scheme, different features would capture different channels of target information and result in a better performance [35, 37, 21, 7]. Since any feature possesses its own invariance and discriminative power, what really distinguishes one feature from another is its location at the invariance-discriminative power spectrum [33]. And the location may vary for different types of variations. Complementary features should be selected in such a way that they are not at the same location of the spectrum for the same appearance variation. In other words, complementary features must be at different spectrum locations under the same variation, especially in long term tracking because the object appearance and its local background may vary substantially. Concatenation of several features into a single kernel space, however, may confuse their characteristics on the invariance-discriminative power spectrums, and is not a good choice. Consequently, a multiple kernel feature, i.e., a combination of several kernels, one for each feature, is a natural choice.

In recent years, correlation filter based trackers have been proposed [4, 15, 7, 6, 16] and achieved top performance. Bolme *et al.*[4] proposed a correlation filter based tracker, called Minimum Output Sum of Squared Error (MOSSE), with classical signal processing techniques.

*This work was supported by Natural Science Foundation of China. Grant No. 61375035.
The corresponding author of the paper is Ming TANG.

They used a base image patch and several circulant virtual ones to train the filter directly in the Fourier domain. Henriques *et al.* [15, 16] utilized the circulant structure produced by a base sample to propose a kernelized correlation filter (KCF). The KCF used a single kernel and enabled fast learning with fast Fourier transforms instead of costly matrix operation, providing the highest tracking speed [36]. Danelljan *et al.* [7] extended the KCF with low-dimensional adaptive color channels and achieved state-of-the-art performance in a comprehensive evaluation. Nevertheless, the correlation filter based tracker can not yet utilize multiple kernels simultaneously. In this paper, we will extend the correlation filter based tracker further to its multiple kernel version and demonstrate its superior performance. Fig. 1 shows a qualitative comparison to indicate that our approach, MKCF, outperforms other trackers in challenging sequences Shaking, Trellis, and Walking2 [36].

Another challenging problem in visual tracking is the robust estimation of object scale in complex scenes. Currently, there exist two popular strategies to tackle this problem. One is the analytical method [2, 27] which moves and analytically deforms a template to minimize the difference between the template and an image region, and the other exhaustively searches for the proper scale among several discretized scales [1, 6]. In order to handle large variations of scale effectively and efficiently, Danelljan *et al.* [6] applied the idea of kernelized correlation filter to a pyramid representation of candidate bounding boxes to speed up exhaustive searches greatly. Nevertheless, their approach had to construct the pyramid in advance as the search method does, thus was in fact still based on several discrete scales. And in order to estimate object scales accurately, the pyramid had to include 33 layers. In this paper, we will present a novel method, which is based on fast feature pyramids [8] and optimization technique to efficiently determine the optimal scale in the continuous scale space. As far as we know, our scale estimation method is the first one which explores the truly minimal number of layers of feature pyramid and avoids constructing the pyramid before searching for proper scales. Moreover, our optimal scale estimation is generic, and can be incorporated into any tracker which does not contain inherent scale estimation.

In summary, the **main contributions** of our work includes two aspects. 1) A multi-kernel correlation filter (MKCF) based tracker is proposed. In fact, the MKCF can be accepted as a general framework of correlation filter in the sense that it embraces both strengths of multiple channels and multiple kernels. 2) An optimal and efficient algorithm is developed to determine the scale of target object. In this paper, the power law of image scaling [30, 8] is introduced into visual tracking community for the first time in order to quickly determine the object scale.

The remainder of this paper is organized as follows. In Sec.2, we briefly overview the related work. In Sec.3, the general correlation filter framework of multiple kernels and multiple channels is derived. And the optimal and efficient algorithm to search for proper scales is presented. Our tracking algorithm, MKCF, is then described in detail. To understand the MKCF more clearly, Sec.4 provides details of our implementation. Experimental results and comparison with other state-of-the-art approaches are presented in Sec.5. Sec.6 summarizes our work.

2. Related Work

Multiple kernel learning (MKL) aims at simultaneously learning a kernel and the associated predictor in supervised learning settings. Rakotomamonjy *et al.* [28] proposed an efficient algorithm, named SimpleMKL, for solving the MKL problem through reduced gradient descent in a primal formulation. Varma and Ray [33] extended the MKL formulation in [28] by introducing an additional constraint on combinational coefficients and applied it to object classification. Vedaldi *et al.* [34] and Gehler and Nowozin [11] applied MKL based approaches to object detection and classification. Cortes *et al.* [5] studied the problem of learning kernels of the same family with an L_2 regularization for ridge regression (RR) [29]. In this paper, we extend the MKL formulation in [28] to RR, and present a novel multi-kernel RR approach.

In recent years, Bolme *et al.* [4] proposed an extension of traditional correlation filters [19] referred to as Minimum Output Sum of Squared Error (MOSSE) filter. The original MOSSE was expressed in the Fourier domain. In fact, it is easy to observe that the expression of MOSSE in the spatial domain is just the ridge regression [29] with a linear kernel. Therefore, Henriques *et al.* [15] proposed the kernelized correlation filter (KCF) by introducing the kernel trick into ridge regression [29]. The generalizations of MOSSE and KCF to multiple channels have also been proposed [3, 10, 14]. Danelljan *et al.* [7] further extended the KCF to multiple channels, one per adaptive color attribute. Henriques *et al.* [17] utilized the circulant structure of Gram matrix to speed up the training of pose detectors in the Fourier domain. It is noted that all these works [4, 15, 3, 10, 14, 7] can only employ a single kernel. In this paper, we propose a brand new multi-kernel correlation filter which is able to fully take advantage of invariance-discriminative power spectrums of various features.

Ruderman and Bialek [30] explored how the statistics of natural images behave as a function of the scale at which an image ensemble is captured. And their discovery implies that the ratio of the statistics of two image sets under two different scales is approximately the power function of the ratio of the two scales. Dollár [8] extended this power law to a pair of images of the same scene, and pointed out that the feature ratio of two images under different scales is also

approximately the power function of the ratio of the two scales. In this paper, we utilize Dollár's power law to speed up the determination of the object scale during tracking. To the best of our knowledge, it is the first time to make use of the power law in the visual tracking community.

3. Multi-kernel Correlation Filter Based Tracking

3.1. Training Multi-kernel Correlation Filter

The training goal of ridge regression [29] is to find function $f(\mathbf{x})$ which minimizes the squared error over training samples $\mathbf{x}_i$'s and their regression targets y_i's, *i.e.*,

$$\min_f \frac{1}{2}\sum_{i=0}^{l-1}(f(\mathbf{x}_i)-y_i)^2+\lambda||f||_k^2, \tag{1}$$

where l is the number of samples, f lies in a bounded convex subset of a Reproducing Kernel Hilbert Space defined by a positive definite kernel function $k(,)$, and $\lambda \geq 0$ is the regularization parameter. By means of the Representer Theorem [31], the solution f^* to the Tikhonov regularization problem can be expressed as

$$f^*(\mathbf{x})=\sum_{i=0}^{l-1}\alpha_i k(\mathbf{x}_i,\mathbf{x}). \tag{2}$$

Then, $||f||_k^2=\alpha^\top\mathbf{K}\alpha$, where $\alpha=(\alpha_0,\alpha_1,\ldots,\alpha_{l-1})^\top$, and $\mathbf{K}$ is the positive semi-definite kernel matrix with $\kappa_{ij}=k(\mathbf{x}_i,\mathbf{x}_j)$ as its elements. Problem (1) becomes to find α, *i.e.*,

$$\min_{\alpha\in R^l}\frac{1}{2}||\mathbf{y}-\mathbf{K}\alpha||_2^2+\frac{\lambda}{2}\alpha^\top\mathbf{K}\alpha, \tag{3}$$

where $\mathbf{y}=(y_0,y_1,\ldots,y_{l-1})^\top$.

It has been shown that using multiple kernels instead of a single one can improve the algorithms' discrimination [22, 33]. Given the base kernels, k_m where $m=1,2,\ldots,M$, a usual approach is to consider $k(\mathbf{x}_i,\mathbf{x}_j)$ be a convex combination of base kernels, *i.e.*, $k(\mathbf{x}_i,\mathbf{x}_j)=\mathbf{d}^\top\mathbf{k}(\mathbf{x}_i,\mathbf{x}_j)$, where $\mathbf{k}(\mathbf{x}_i,\mathbf{x}_j)=(k_1(\mathbf{x}_i,\mathbf{x}_j),k_2(\mathbf{x}_i,\mathbf{x}_j),\ldots,k_M(\mathbf{x}_i,\mathbf{x}_j))^\top$, $\mathbf{d}=(d_1,d_2,\ldots,d_M)^\top$, $\sum_{m=1}^M d_m=1$, and $d_m\geq 0$. Hence we have

$$\mathbf{K}=\sum_{m=1}^M d_m\mathbf{K}_m, \tag{4}$$

where $\mathbf{K}_m$ is the m^{th} base kernel matrix with $\kappa_{ij}^m=k_m(\mathbf{x}_i,\mathbf{x}_j)$ as its elements. Substituting $\mathbf{K}$ in Eq. (4) for that in (3) and introducing the additional constraint on the sum of d_m's, we obtain the object function $F(\alpha,\mathbf{d})$ of multiple kernel version of ridge regression problem (3) as follows.

$$\min_{\alpha,\mathbf{d}} F(\alpha,\mathbf{d})=\frac{1}{2}\left\|\mathbf{y}-\sum_{m=1}^M d_m^2\mathbf{K}_m\alpha\right\|_2^2+\frac{\lambda}{2}\alpha^\top\sum_{m=1}^M d_m^2\mathbf{K}_m\alpha+\frac{\nu}{2}\left(\sum_{m=1}^M d_m^2-1\right)^2, \tag{5}$$

where $\lambda=10^{-3}$ and $\nu=10^{-2}$ in our current experiments. In order to ensure that all combination coefficients are positive, d_m^2's, instead of d_m's, are used in Eq. (5), *i.e.*, $\mathbf{K}=\sum_{m=1}^M d_m^2\mathbf{K}_m$. It is noted that Eq. (5) is equivalent to a constrained multiple kernel optimization problem with the third item of Eq. (5) as its constraint and ν as its Lagrangian multiplier, and its optimal solution can be expressed as

$$f^*(\mathbf{x})=\sum_{i=0}^{l-1}\alpha_i\mathbf{d}^{2\top}\mathbf{k}(\mathbf{x}_i,\mathbf{x}), \tag{6}$$

where $\mathbf{d}^2\equiv(d_1^2,d_2^2,\ldots,d_M^2)^\top$.

It should be pointed out that Eq. (5) is also the Lagrangian function with ν as its Lagrangian multiplier without a second power on $\left(\sum_{m=1}^M d_m^2-1\right)$. In this case, $\min_{\alpha,\mathbf{d}}F(\alpha,\mathbf{d})$ is a linear programming (LP) problem w.r.t. $\mathbf{d}^2$, given α. Since the optimal solution of LP problem will always be at the vertex of linear feasible region, the optimal $\mathbf{d}^{*2}$ must be a unit vector. This means that the combination of multiple kernels will be discarded and only one kernel left. This case does not meet our goal of exploring multiple kernels simultaneously to improve the tracking performance. Therefore, a second power on $\left(\sum_{m=1}^M d_m^2-1\right)$ is necessary for Eq. (5).

Theorem 1 *Let $\{\mathbf{K}_m\}$ be positive semi-definite. Then, (a) given α, $F(\alpha,\mathbf{d})$ is convex w.r.t. $\mathbf{d}^2$; (b) given $\mathbf{d}^2$, $F(\alpha,\mathbf{d})$ is convex w.r.t. α.*

$F(\alpha,\mathbf{d})$ is a differentiable function, we can find a minimizer simply by taking its gradients w.r.t. α and $\mathbf{d}$ and then solving them. To solve α, we let $\nabla_\alpha F(\alpha,\mathbf{d})=0$, and achieve that

$$\alpha=\left(\sum_{m=1}^M d_m^2\mathbf{K}_m+\lambda\mathbf{I}\right)^{-1}\mathbf{y}, \tag{7}$$

where $\mathbf{I}$ is the $l\times l$ identity matrix. We propose two approaches to find out $\mathbf{d}^2$. One is an iteration approach, and the other is analytic. The iteration approach employs the gradient descent method to achieve the minimum step by step, *i.e.*,

$$\begin{aligned}\mathbf{d}_{t+1}^2 &= \mathbf{d}_t^2+\gamma_{\mathbf{d},t}\nabla_{\mathbf{d}^2}F(\alpha,\mathbf{d}_t)\\ &= \mathbf{d}_t^2+\tfrac{\gamma_{\mathbf{d},t}}{2}(\mathbf{d}^2\mathbf{B}-\mathbf{c}),\end{aligned} \tag{8}$$

where $\mathbf{B}$ is an $M \times M$ matrix with elements $b_{mn} = \alpha^\top \mathbf{K}_{mn}\alpha + 2\nu$, $\mathbf{K}_{mn} = \mathbf{K}_m\mathbf{K}_n + \mathbf{K}_n\mathbf{K}_m$, $\mathbf{c}$ is an M dimensional vector with $c_n = \alpha^\top \mathbf{K}_n(2\mathbf{y} - \lambda\alpha) + 2\nu$ as its elements, $\gamma_{\mathbf{d},t} > 0$ is the optimal step length in the t^{th} iteration [9]. Theorem 1 (a) ensures that such iteration will converge to the minimum. Or, the optimization process (8) will terminate at the boundary of region $\mathbf{d}^2 \geq 0$ whenever any of the component of $\mathbf{d}^2_{t+1}$ is less than or equal to 0.

The analytic approach finds out $\mathbf{d}^2$ through solving the system of equations $\nabla_{\mathbf{d}} F(\alpha, \mathbf{d}) = 0$. Specifically, it is easy to derive from $\nabla_{\mathbf{d}} F(\alpha, \mathbf{d}) = 0$ that

$$d_n \left[\alpha^\top \left(\sum_{m=1}^{M} d_m^2 \mathbf{K}_{mn} \right) \alpha + 2\nu \sum_{m=1}^{M} d_m^2 \right] = d_n c_n, \quad (9)$$

where d_n is an element of $\mathbf{d}$. Therefore,

$$\begin{aligned} d_n = 0, \quad \text{or} \\ \alpha^\top \left(\textstyle\sum_{m=1}^{M} d_m^2 \mathbf{K}_{mn} \right) \alpha + 2\nu \textstyle\sum_{m=1}^{M} d_m^2 = c_n. \end{aligned} \quad (10)$$

Suppose $d_n = 0$ only if $n \in S_0 \subset S_a \equiv \{1, 2, \ldots, M\}$, and $\bar{S}_0 = S_a \setminus S_0$, where the cardinality of S_0 equals N, and $0 \leq N < M$. Then Eq. (10) can be expressed as

$$\sum_{m \in \bar{S}_0} (\alpha^\top \mathbf{K}_{mn}\alpha + 2\nu) d_m^2 = c_n, \quad (11)$$

where $n \in \bar{S}_0$. This is a system of linear equations w.r.t. d_m^2's, and can be briefly expressed as

$$\mathbf{d}_p^2 = \mathbf{c}_p \mathbf{B}_p^{-1}, \quad (12)$$

where $\mathbf{B}_p$ is a $(M-N) \times (M-N)$ matrix with elements $b_{mn} = \alpha^\top \mathbf{K}_{mn}\alpha + 2\nu$, $\mathbf{d}_p^2$ and $\mathbf{c}_p$ are two column vectors with d_m^2's and c_n's as elements, respectively, $m, n \in \bar{S}_0$. If $\mathbf{B}_p$ is not invertible, the generalized inverse $\mathbf{B}_p^+$ will be calculated instead of $\mathbf{B}_p^{-1}$. If any element of $\mathbf{c}_p\mathbf{B}_p^{-1}$ is less than 0, there does not exist any solution $\mathbf{d}_p^2$ for Eq. (12) in such S_0 and $\bar{S}_0$. Otherwise, according to Theorem 1, alternately evaluating Eqs. (7) and (12) will make $F(\alpha, \mathbf{d})$ converge to a point $(\alpha^*, s(\mathbf{d}_p^{*2} \cup \mathbf{d}_0))$, where $\mathbf{d}_p^{*2}$ is evaluated via Eq. (12) and initially $\mathbf{d}_p^2 = (1/M, 1/M, \ldots, 1/M)$, $\mathbf{d}_0$ is the vector with d_n, $n \in S_0$, as its elements, $\mathbf{d}_p^{*2} \cup \mathbf{d}_0$ is the union set of all components of both $\mathbf{d}_p^{*2}$ and $\mathbf{d}_0$, $s(\mathbf{v})$ sorts the elements of $\mathbf{v}$ ascendingly according to their subscripts, and generates a new column vector $\mathbf{d}^{*2}$. For example, suppose $S_0 = \{2\}$, $\bar{S}_0 = \{1, 3\}$, $\mathbf{d}_p^{*2} = \{d_1^2, d_3^2\} = \{0.4, 0.6\}$, and $\mathbf{d}_0 = \{d_2\} = \{0\}$. Then $\mathbf{d}^{*2} = s(\mathbf{v}) = s(\mathbf{d}_p^{*2} \cup \mathbf{d}_0) = \{d_1^2, d_2, d_3^2\} = \{0.4, 0, 0.6\}$. In practice, a satisfactory convergency $(\alpha^*, \mathbf{d}^{*2})$ can be achieved after a couple of iterations of Eqs. (7) and (12).

In fact, N can take any value from 0 to $M-1$, and given N, there are C_M^N different S_0's. To distinguish these S_0's one another, we introduce $S_0^{N,c}$ to represent the S_0 whose cardinality is N and whose elements are the c^{th} combination of C_M^N ones. The C_M^N combinations consist of all possible N elements of S_a. It is clear that $c = 1, 2, \ldots, C_M^N$. Through evaluating the convergent point by means of the above process for every $S_0^{N,c}$, at most $\sum_{N=0}^{M-1} C_M^N$ minimizers of $F(\alpha, \mathbf{d})$ will be found. The minimizer that makes $F(\alpha, \mathbf{d})$ minimal among the minimizers will be accepted as the optimal solution of Eq. (10).

It is interesting to examine the optimization of $F(\alpha, \mathbf{d})$ in the case that only two complementary features are included, *i.e.* $M = 2$, given α. Because $(d_1^2 + d_2^2 - 1)^2$ is one of the three items to optimize in Eq. (5), it is advantage for the optimization that $d_1^2 = 1$ if $d_2^2 = 0$, or vice verse. Therefore, there are three combinations of the values of d_1^2 and d_2^2, *i.e.*, $(1, 0)$, $(0, 1)$, and (v_1, v_2), where (v_1, v_2) is obtained through solving Eq. (12) with $N = 0$. This means that the optimal solution of Eq. (10) will be selected from three candidates: only employing one of two complementary features, and using the linear combination of both features. And the final solution that minimizes $F(\alpha, \mathbf{d})$ will be among the three candidates of $\mathbf{d}^2$, and will be applied to the construction of object appearance model in the current frame.

It is clear that the analytic approach to solving $\mathbf{d}$ is preferred if M is small enough. In our current implementation, we found that the analytic approach is more efficient than the iteration one because $M = 2$. The former spent about half of the time the later spent. Of cause, the iteration approach may be more efficient than the analytic one if M is large. And that how many kernels will be applied depends on the tradeoff between the performance and computational burden.

3.2. Fast Evaluation in Training

All correlation filter based tracking algorithms [4, 15, 17, 7] consider the training samples, $\mathbf{x}_i$, to be generated through cyclically shifting a base sample. Therefore, the optimization of $F(\alpha, \mathbf{d})$ described in the preceding section can be speeded up by means of fast Fourier transform (FFT).

Because kernel matrices $\mathbf{K}_m$s are circulant [15], and the inverses, products, and sums of circulant matrices are still circulant [12], $\mathbf{K}_{mn}$ is circulant. Denote $\mathbf{k}_{mn}$ to be the first row of $\mathbf{K}_{mn}$, and $\mathbf{k}'_m$ the first column of $\mathbf{K}_m$. It is clear that $\mathbf{k}'_m = \mathbf{k}_m^\top$ because $\mathbf{K}_m$ is symmetric. The evaluation of Eq. (7) can be accelerated by

$$\alpha = \mathcal{F}_1^{-1} \left(\frac{\mathcal{F}_1(\mathbf{y})}{\mathcal{F}_1 \left(\sum_{m=1}^{M} d_m^2 \mathbf{k}_m \right) + \lambda} \right), \quad (13)$$

and that of Eq. (12) can be done by

$$\begin{aligned} \mathbf{k}_{mn} \quad = \quad & \mathcal{F}_1^{-1}(\mathcal{F}_1^*(\mathbf{k}'_n) \odot \mathcal{F}_1(\mathbf{k}_m)) + \\ & \mathcal{F}_1^{-1}(\mathcal{F}_1^*(\mathbf{k}'_m) \odot \mathcal{F}_1(\mathbf{k}_n)), \end{aligned} \quad (14a)$$

$$b_{mn} = \alpha^\top \mathcal{F}_1^{-1}(\mathcal{F}_1^*(\mathbf{k}_{mn}) \odot \mathcal{F}_1(\alpha)) + 2\nu, \quad (14b)$$

$$c_m = (\lambda\alpha - \mathbf{y})^\top \mathcal{F}_1^{-1}(\mathcal{F}_1^*(\mathbf{k}_m) \odot \mathcal{F}_1(\alpha)) - 1. \quad (14c)$$

It is clear that the linear combination of circulant matrices is still circulant. Therefore, $\sum_{m=1}^{M} d_m^2 \mathbf{K}_m$ is circulant, and its first row is $\sum_{m=1}^{M} d_m^2 \mathbf{k}_m$. The evaluation of $F(\alpha, \mathbf{d})$ can then be accelerated as follows.

$$F_2(\alpha, \mathbf{d}) = \frac{1}{2}\left\|\mathbf{y} - \mathcal{F}_1^{-1}\left(\mathcal{F}_1^*\left(\sum_{m=1}^{M} d_m^2 \mathbf{k}_m\right) \odot \mathcal{F}_1(\alpha)\right)\right\|_2^2 + \frac{\lambda}{2}\alpha^\top \mathcal{F}_1^{-1}\left(\mathcal{F}_1^*\left(\sum_{m=1}^{M} d_m^2 \mathbf{k}_m\right) \odot \mathcal{F}_1(\alpha)\right). \quad (15)$$

It is noted that only the first two items of $F(\alpha, \mathbf{d})$ are necessary for the evaluation of $F(\alpha, \mathbf{d})$, as its third item is equivalent to a constraint.

3.3. Fast Detection

3.3.1 Determine Central Location

According to Eq. (6), the multiple kernel correlation filter evaluates the responses of all test samples $\mathbf{z}_j = \mathbf{P}^j \mathbf{z}$, $j = 0, 1, \ldots, l-1$, in the current frame $\mathcal{I}$ as

$$y^j(\mathbf{z}) = \sum_{m=1}^{M} d_m^2 \sum_{i=0}^{l-1} \alpha_i k_m(\mathbf{z}_j, \overline{\mathbf{x}}_i), \quad (16)$$

where $\mathbf{z}$ is the base test sample, $\mathbf{P}$ is the permutation matrix [15, 16]. $\overline{\mathbf{x}}_i = \mathbf{P}^i \overline{\mathbf{x}}$, $i = 0, 1, \ldots, l-1$, $\overline{\mathbf{x}}$ is the weighted average of located samples in several earlier frames (*i.e.*, $\overline{\mathbf{x}}^{\text{new}}$ in Sec. 3.3.4). Because $k_m(,)$, $m = 1, 2, \ldots, M$, is permutation-matrix-invariant, the response map, $\mathbf{y}(\mathbf{z})$, of all virtual samples generated by $\mathbf{z}$ can be evaluated as

$$\mathbf{y}(\mathbf{z}) \equiv (y^0(\mathbf{z}), \ldots, y^{l-1}(\mathbf{z}))^\top = \sum_{m=1}^{M} d_m^2 \mathbf{C}(\overline{\mathbf{k}}_m)\alpha, \quad (17)$$

where $\overline{\mathbf{k}}_m = (\overline{k}_{m,0}, \ldots, \overline{k}_{m,l-1})$, $\overline{k}_{m,i} = k_m(\mathbf{z}, \mathbf{P}^i \overline{\mathbf{x}})$, and $\mathbf{C}(\overline{\mathbf{k}}_m)$ is the circulant matrix with $\overline{\mathbf{k}}_m$ as its first row. Therefore, the response map can be accelerated as follows.

$$\mathbf{y}(\mathbf{z}) = \sum_{m=1}^{M} d_m^2 \mathcal{F}_1^{-1}\left(\mathcal{F}_1^*(\overline{\mathbf{k}}_m) \odot \mathcal{F}_1(\alpha)\right). \quad (18)$$

The element of $\mathbf{y}(\mathbf{z})$ which takes the maximal value is accepted as the optimal object location in $\mathcal{I}$.

3.3.2 Identify Optimal Scale

Suppose that $\iota(\mathcal{I}) \equiv \iota_{\max}(\mathbf{y}(\mathbf{z}))$ is the center location of the test patch whose response is maximal among all test patches

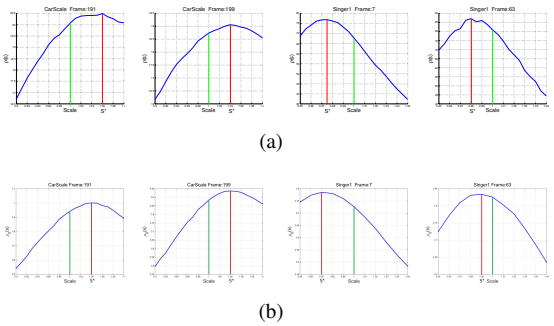

Figure 2. (a) $\rho(s)$ usually has a dominant mode, which is assumed to be the optimal object scale s^* in current frame. 0.618 method [24] is employed to search for s^*. The object scale s of the last frame is supposed to be 1. (b) $\rho_0(s)$ usually also has a dominant mode. Its curve around s^*, however, is sometime a little bit too flat in comparison with $\rho(s)$'s. See Sec. 3.3.2 for details. The figure is best viewed with high resolution display.

in $\mathcal{I}$, $D(\iota, s)$ is an $\mathcal{I}$'s patch, whose center is ι, with scale s, $s_{\mathbf{x}}$ is the scale of patch $\mathbf{x}$, and $R(D, s)$ denotes the image patch D re-sampled by s. Let

$$\rho(s) \equiv \text{PSR}\left(\mathbf{y}\left(R(D(\iota(\mathcal{I}), s), s_{\overline{\mathbf{x}}})\right)\right), \quad (19)$$

where PSR is the peak to sidelobe ratio [4] in the response map $\mathbf{y}(\cdot)$. Intuitively, the evaluation of $\rho(s)$ consists of four steps. The first is to extract a patch $D(\iota(\mathcal{I}), s)$, and the second to re-sample $D(\iota(\mathcal{I}), s)$ by scale $s_{\overline{\mathbf{x}}}$. The re-sampled $D(\iota(\mathcal{I}), s)$ will be used as base sample then. The third is to generate the response map $\mathbf{y}(\cdot)$ by using the base sample and Eq. (17), and the last to evaluate the PSR of $\mathbf{y}(\cdot)$. It is noted that $s_{\overline{\mathbf{x}}}$ is the template scale in the last frame. We experimentally observed that $\rho(s)$ usually possesses a dominant mode. Fig. 2 (a) shows several $\rho(s)$'s generated in our experiments on sequences *CarScale* and *Singer1*. Therefore, it is reasonable to assume that the correct scale, s^*, of the object will maximize PSR, *i.e.*, $s^* = \arg\max_s \rho(s)$. Then the optimization technique is employed to seek out s^* efficiently in our tracker.

In fact, the optimal solution, s^*, of $\rho(s)$ can easily be sought out through exact line search because $\rho(s)$ is a univariate function. In our current implementation, we employed the golden section method, also called 0.618 method [24], to optimize $\rho(s)$. This is because the 0.618 method only evaluates $\rho(s)$ once per iteration. Specifically, suppose that the aspect ratio of the target object is constant in the whole sequence, the object scale is 1 and the length and width of the located bounding box are l_o and w_o, respectively, in the last frame, and the initial interval of s for searching is set to be $\delta_s = [0.9, 1.1]$. Then, $||\delta_s|| = 0.2$, where $||*||$ is the length of interval $*$, the maximum of length and width of the maximal candidate bounding boxes are $1.1 \cdot \max(l_o, w_o)$, the minimum of those are $0.9 \cdot \max(l_o, w_o)$, and their difference is $0.2 \cdot \max(l_o, w_o) =$

$||\delta_s|| \max(l_o, w_o)$. As iteration proceeds, $||\delta_s||$ gets smaller and smaller until $||\delta_s|| \max(l_o, w_o) < 1$ pixel. When the iteration stops, s^* is obtained and returned.

In each iteration, the length of searching interval is narrowed to 0.618 times the one in the last round. In other words, the computational complexity of 0.618 method is $O(\log(\max(l_o, w_o)))$. In comparison, the traditional exhaustive method has to search each layer of pyramid, and requires $O(\max(l_o, w_o))$ to find out the optimal scale.

Another possible choice is to use $f^*()$ of Eq. (6), instead of PSR, to find out the optimal object scale. That is,

$$\rho_0(s) \equiv \mathbf{y}^0 \left(R(D(\iota(\mathcal{I}), s), s_{\overline{\mathbf{x}}})\right). \tag{20}$$

We observed experimentally that, similar to $\rho(s)$, $\rho_0(s)$ also usually possesses a dominant mode. Fig. 2 (b) shows several $\rho_0(s)$'s generated in our experiments in the same sequences and frames as used in Fig. 2 (a). Nevertheless, $\rho_0(s)$ may become a little bit too flat around the optimal scale, therefore, is not as robust as $\rho(s)$. In fact, the performance will decrease at least 1% on the visual tracking benchmark [36] if Eq. (20), rather than Eq. (19), is utilized. Consequently, Eq. (19) is adopted to identify the optimal scale in our tracker.

3.3.3 Fast Feature Scaling

Refer to Eq. (19) again; $R(D(\iota(\mathcal{I}), s), s_{\overline{\mathbf{x}}})$ will re-sample $D(\iota(\mathcal{I}), s)$ by scale $s_{\overline{\mathbf{x}}}$. In order to accelerate the evaluation of $R(D(\iota(\mathcal{I}), s), s_{\overline{\mathbf{x}}})$, rather than rescaling image patch $D(\iota(\mathcal{I}), s)$ and then extracting its feature in every iteration, we applied Dollár's power law [8] to rescale the feature directly. It has been pointed out that the processing time can be reduced several times in this way [8]. Specifically, if X_m is the m^{th} feature of image patch X, then Dollár's power law states that

$$\begin{aligned} &R(D(\iota(\mathcal{I}), s), s_{\overline{\mathbf{x}}})_m \approx \\ &R(D(\iota(\mathcal{I}), s)_m, s_{\overline{\mathbf{x}}}) \cdot \left(\frac{s_{\overline{\mathbf{x}}}}{s}\right)^{-\lambda_m}, \end{aligned} \tag{21}$$

where λ_m is a feature related constant, $R(,)_m$ and $D(,)_m$ are the m^{th} features of patches $R(,)$ and $D(,)$, respectively.

To fully take this advantage to speed up searching for the optimal scale of the object, we extract in practice the features of an image only once. And all the feature scaling operations discussed in Sec. 3.3.2 will be performed on feature channels.

3.3.4 Updating Filter

In our tracker, the object appearance model is $(\alpha, \mathbf{d}^2, \overline{\mathbf{x}})$. We adopt the following formulation to update $\overline{\mathbf{x}}$.

$$\begin{aligned} \overline{\mathbf{x}}_m^{\text{new}} \quad = \quad & (1-\eta_m) R(\overline{\mathbf{x}}_m, s^*) \cdot \left(\frac{s^*}{s_{\overline{\mathbf{x}}}}\right)^{-\lambda_m} \\ & + \eta_m D(\iota(\mathcal{I}), s^*)_m, \end{aligned} \tag{22}$$

where η_m is the learning rate, s^* is the optimal object scale in the current frame (Sec. 3.3.2), $m = 1, 2 \ldots, M$. It is clear that $\overline{\mathbf{x}}_m^{\text{new}}$ is the weighted sum of historical templates and the m^{th} channel feature of object bounding box in the current frame, where the historical template has to be fast scaled to the current scale s^* by means of Dollár's power law [8]. $\overline{\mathbf{x}}^{\text{new}} \equiv (\overline{\mathbf{x}}_1^{\text{new}}, \ldots, \overline{\mathbf{x}}_M^{\text{new}})$. α^{new} and $(\mathbf{d}^{\text{new}})^2$ are evaluated by using $\overline{\mathbf{x}}^{\text{new}}$ and Eqs. (12), (13), (14), and (15). It is noted that $k_m(\mathbf{x}, \mathbf{x}')$ will be evaluation by only using $\mathbf{x}_m$ and $\mathbf{x}'_m$.

The whole process of our tracker is summarized in Alg.1.

4. Implementation Details

Each of color and HOG features uses a kernel, *i.e.*, $M = 2$. The color scheme proposed in [7] is adopted in our tracker. To enhance the robustness against object deformation and speed up tracking, the HOG feature has only six gradient orientations and the cell size is 4×4. Gaussian kernel is used for both features with $\sigma_{\text{color}} = 0.4$ and $\sigma_{\text{HOG}} = 0.5$, which ensures that all $\mathbf{K}_m$s are positive definite [25]. According to Dollár *et al.* [8], the resizing coefficients λ_m in Eq.(21) are set as follows: $\lambda_{\text{color}} = 0$ and $\lambda_{\text{HOG}} = 0.078$.

In order to reduce high-frequency noise in the frequency domain resulted from the large discontinuity between opposite edges of a cyclic-extended image patch, all feature patches (*e.g.* $\mathbf{x}_m$ and $D(\iota_{\max}(\mathbf{y}(\mathbf{z})), s)_m$) are banded with a sine window for a sine window can reduce values near the borders to zero, and eliminate discontinuities.

In spite of the high efficiency of our continuous scale estimation, introducing scale processing inevitably increases the time complexity of our tracker. It is generally observed that, in most cases, the variation of target scale is much slower than that of its location. Therefore, it is superfluous to execute scale estimation as frequently as location. In our current experiments, the ratio of the number of scale estimations to that of locations is 0.5.

5. Experimental Results

Our tracker MKCF was implemented in MATLAB. The experiments were performed on a PC with Intel Core i5 3.20GHz CPU and 16GB RAM. We compared our MKCF to other 4 state-of-the-art trackers, Struck [13], DSST [6], CN2 [7], and KCF [16] with the visual tracking benchmark [36] which includes 50 image sequences. All parameter values of MKCF were kept consistent across all experimental comparisons. It is noted that DSST is the winner of VOT2014 challenge.[1] The mean fps of MKCF over the 50 sequences is about 15.

The performance of our tracker were quantitatively evaluated with popular criteria used in [1, 15, 36, 7, 6, 16], *i.e.*,

[1] www.votchallenge.net/vot2014/results.html

Algorithm 1 Multi-kernel Correlation Filter Based Tracker

- **Input**: Frame t, $t = 0, 1, 2, \ldots$, initial object patch $\mathbf{x}_0$ of $l_1 \times l_2 \times c'$, where c' is the number of feature channels, $l_1 \times l_2$ Gaussian-shaped regression target $\mathbf{y}$.
- **Output**: optimal locations $l_1^*, l_2^* \ldots$ in subsequent frames.
- **Initialization**: $t = 0$, $\overline{\mathbf{x}} = \mathbf{x}_t$.
- **Training** (Secs. 3.1 and 3.2):

1. Generate virtual training set based on $\overline{\mathbf{x}}$, $v_F = \infty$.
2. for $N = 0$ to $M - 1$

   for $c = 1$ to C_M^N

   Calculate $(\alpha, \mathbf{d}^2)$ by using $\overline{\mathbf{x}}$ and Eqs. (12), (13), (14), and (15).

   If $v_F > F(\alpha, \mathbf{d}^2)$

   $v_F = F(\alpha, \mathbf{d}^2)$, $(\alpha_o, \mathbf{d}_o^2) = (\alpha, \mathbf{d}^2)$.

- **Location** (Sec. 3.3):

3. Determine the object location by using $(\alpha_o, \mathbf{d}_o^2)$ and Eq. (18) in frame $t + 1$.
4. Determine the object scale by using Eqs. (19) and (21).
5. Update $\overline{\mathbf{x}}$ by using Eqs. (22).
6. $t = t + 1$, go to 1.

center error, distance precision, precision plot, overlap ratio, overlap precision, and success plot. Center error is calculated as the average Euclidean distance between the centers of located objects and their ground truths in a sequence. Distance precision is the percentage of frames where the objects are located within the center errors in 0 to t_c pixels, where $t_c = 20$, and the precision plot is simply a curve of the distance precisions with t_c changing from 0 to 50 pixels. Overlap ratio is defined as the average ratio of intersection and union of the estimated bounding box and ground truth in a sequence, overlap precision as the percentage of frames where the overlap ratio exceeds t_o in a sequence, where $t_o = 0.5$. And the success plot is simply a curve of overlap precisions with t_o changing from 0 to 1.

5.1. Fast Feature Scaling vs. Traditional One

In our approach, we extract the feature from a patch only once in a frame, and then scaling the patch's feature is approximately implemented by directly scaling the extracted feature channels. Such a way will significantly reduce the computational cost in extracting features [8]. While saving the processing time, it is also desirable not to lose much in tracking accuracy. Fig. 3 includes the average precision and success plots of the two versions, MKCF and MKCFN, of our tracker over the 28 sequences annotated with scale variation [36]. MKCF is just described in Alg. 1, while

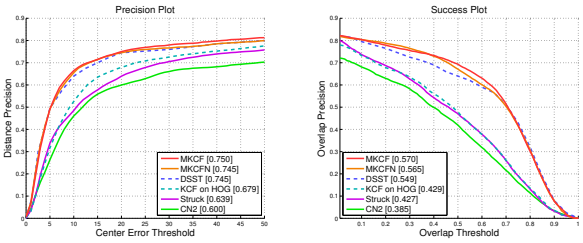

Figure 3. The average success plot of two versions, MKCF and MKCFN, of our tracker, DSST [6], KCF [16], Struck [13], and CN2 [7] over 28 sequences annotated with scale variation [36]. See Secs. 5.1 and 5.2 for details. The area under curve (AUC) of three trackers are reported in the legend.

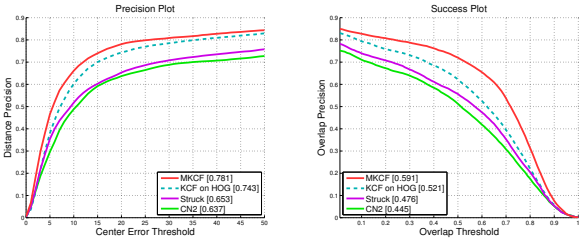

Figure 4. The average precision and success plots of our MKCF, Struck [13], KCF [16], and CN2 [7] over 50 sequences [36]. The mean distance precision scores and AUCs of each tracker are also reported in the legend. The figure is best viewed in color.

MKCFN is a variant of MKCF of which the fast Dollár's law [8] is replaced by a traditional one. The traditional scheme scales the patch's feature by first scaling original patches and then calculating their features in the scaled original channels. Except for the scaling schemes pointed out above, MKCF and MKCFN are exactly the same in terms of implementation and parameter setting. Surprisingly, Fig. 3 indicates that using the fast Dollár's law scheme generates higher distance and overlap precisions than using the traditional one, and the AUC of MKCF is also larger than that of MKCFN. In contrary to the conclusion in object detection domain that approximately scaling features with Dollár's law will slightly reduce the detection accuracy [8], approximate feature extraction scheme indeed improves the robustness and accuracy of tracking. The experimental comparison reveals the *contrary affects* of the approximate scheme on object detection and visual tracking.

5.2. Comparison to State-of-the-art Trackers

Fig. 4 shows the average precision and success plots of our tracker MKCF, Struck [13], KCF [16], and CN2 [7] over the 50 sequences. The mean distance precisions and AUCs are also included in it. It is seen that our tracker outperforms other three ones.

We also compared our MKCF to Struck, KCF (on HOG), and CN2 over the 50 sequences with respect to the 11 anno-

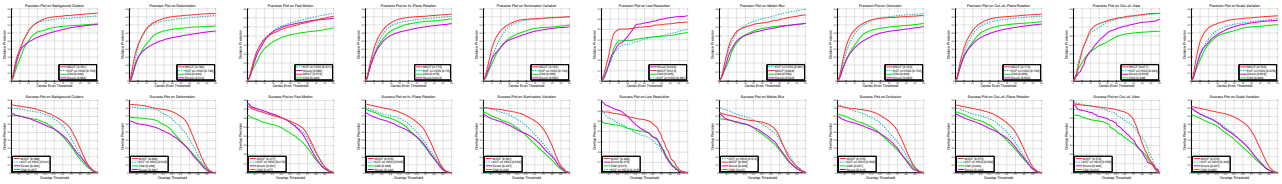

Figure 5. The average precision plots of our MKCF and Struck [13], KCF [16], and CN2 [7] on 11 attributes [36]. The mean distance precision scores and AUCs of each tracker are also reported in legends. The figure is best viewed with high resolution display.

	Boy	Car4	CarScale	Couple	Crossing	David	Dog1	Doll	Dudek	Fleetface	Freeman1	Freeman3	Freeman4	Girl
MKCF	97.2	**100**	**89.7**	45	**100**	99.6	**100**	**100**	**98.8**	63.6	**61**	**38.7**	31.1	83
DSST [6]	**100**	**100**	84.5	10.7	**100**	**100**	**100**	99.6	98.1	66.5	36.8	31.3	**41.7**	24.2
KCF [16]	99.2	36.4	44.4	24.3	95	62.2	65.1	55.2	97.6	**66.9**	16.3	29.1	18.4	74.2
CN2 [7]	95.3	27.6	44.8	10.7	96.7	58.8	65.3	72.8	96.1	58.7	15	33	17.3	46.4
Struck [13]	99.7	39.8	43.3	**54.3**	94.2	23.6	65.3	68.8	98.0	66.6	21.8	20.0	15.9	**98.0**

	Ironman	Lemming	Liquor	Matrix	mRolling	Shaking	Singer1	Skating1	Skiing	Soccer	Trellis	Walking	Walking2	Woman
MKCF	13.3	27.2	**99.4**	**19**	7.3	96.2	**100**	**63.5**	4.9	13.5	96.7	**99.8**	99.8	**93.8**
DSST [6]	13.3	26.9	40.9	18	6.7	**100**	**100**	54.8	4.9	**52.8**	**96.8**	**99.8**	**100**	93.3
KCF [16]	**15.1**	44.2	99	13	7.9	1.4	27.6	36.3	7.4	39.3	84	51.5	38	93.6
CN2 [7]	13.3	29.1	20.4	1	7.3	67.4	27.6	37.3	**9.9**	48.2	65.9	45.9	38.4	24.5
Struck [13]	4.8	**64.1**	40.6	12.0	**15.9**	16.7	29.9	37.0	3.7	15.6	78.4	56.6	43.4	93.5

Table 1. Overlap precision in percent on the 28 sequences annotated with scale variation. The best scores are shown in bold.

tated attributions [36]. Fig. 5 reports the average precision and success plots, the mean distance precisions, and AUCs. It is seen in Fig. 5 that our MKCF almost consistently outperforms other three trackers, except for the following three cases. 1) MKCF is inferior to KCF (on HOG) over the 12 sequences of motion blur, but superior to all other trackers if the center error threshold $t_c \leq 11$ or the overlap threshold $t_o \geq 0.44$. 2) Over the 17 sequences of fast motion, MKCF is inferior to KCF (on HOG) in the precision plot. 3) MKCF is inferior to Struck over 4 sequences of low resolution at distance precision of $t_c = 20$, but superior to all other trackers if $t_c \leq 17$. It has been accepted through experiments that the representation with HOG-based feature performs poorly over low resolution sequences [6, 16]. By combining HOG and its complementary feature color through multi-kernel technique, our MKCF improves its performance on low resolution sequences by a large margin.

Table 1 lists a per-sequence comparison of our MKCF to DSST [6], KCF, Struck, and CN2 in overlap precision. It is easy to check out that MKCF outperforms all other trackers on 14 sequences, is superior to DSST on 13 out of the 28 sequences, and has a similar accuracy to DSST on 7 sequences. Fig. 3 shows the average distance and success plots of MKCF and DSST over 28 sequences annotated with scale variation. The mean distance precision and AUC are also reported. We do not show the performance curves of other state-of-the-art trackers in Fig. 3, because DSST outperforms them consistently on the 28 sequences [6]. It is clear from Table 1 and Fig. 3 that MKCF outperforms DSST on the benchmark.

It is interesting to notice that the difference of performance between our MKCF and DSST is made by the difference of their curves at around 0.5. This means that MKCF is more robust than DSST in difficult frames where the locations and scales of object are easily bias away from the correct ones a little bit more.

6. Conclusion

A novel tracking algorithm, MKCF, has been presented in this paper. To construct multi-feature appearance models, we proposed an approach to fusing features of different types by means of multiple kernel learning. Instead of building traditional image pyramids in advance, we employed the optimization technique to search for the correct object scale in the continuous scale space efficiently. The whole tracking algorithm is accelerated by FFT. Extensive experiments on the benchmark have shown that our algorithm outperforms the state-of-the-art algorithms.

A. Proof of Theorem 1

(a) $\nabla_{\mathbf{d}^2} F(\alpha, \mathbf{d}) = \frac{1}{2}(\mathbf{d}^2\mathbf{B} - \mathbf{c})$, $\nabla^2_{\mathbf{d}^2} F(\alpha, \mathbf{d}) = \frac{1}{2}\mathbf{B}$, where $\mathbf{B}$ is an $M \times M$ matrix with elements $b_{mn} = \alpha^\top \mathbf{K}_{mn}\alpha + 2\nu$, $\mathbf{K}_{mn} = \mathbf{K}_m\mathbf{K}_n + \mathbf{K}_n\mathbf{K}_m$, $\mathbf{c}$ is an $M \times 1$ vector with $c_n = \alpha^\top \mathbf{K}_n(2\mathbf{y} - \lambda\alpha) + 2\nu$ as its elements. $\because$ All $\mathbf{K}_m$s are circulant (Theorem 1, [15]) and positive definite, $\therefore$ $\mathbf{K}_m = \mathbf{U}\mathbf{\Sigma}^2_m\mathbf{U}^*$ [12], $\therefore$ $\mathbf{K}_m\mathbf{K}_n = \mathbf{U}\mathbf{\Sigma}^2_{mn}\mathbf{U}^*$, where $\mathbf{\Sigma}^2_{mn} = \mathbf{\Sigma}^2_m\mathbf{\Sigma}^2_n$, $\therefore$ $\mathbf{K}_{mn}$s are positive definite, $\therefore$ $b_{mn} > 0$ as $\nu \geq 0$, $\therefore$ $\forall \mathbf{d}^2, \mathbf{d}^2\mathbf{B}\mathbf{d}^{2\top} > 0$ if $\mathbf{d}^2 \neq \mathbf{0}$. Therefore, $F(\alpha, \mathbf{d})$ is convex w.r.t. $\mathbf{d}^2$.

(b) $\nabla^2_\alpha F(\alpha, \mathbf{d}) = \mathbf{K}(\mathbf{K} + \lambda\mathbf{I})$. $\because \forall m$, $\mathbf{K}_m$ is positive definite and circulant, $\therefore$ $\mathbf{K}$ is positive definite and circulant. $\therefore \mathbf{K} = \mathbf{U}\mathbf{\Sigma}^2_K\mathbf{U}^*$ [12], $\mathbf{K} + \lambda\mathbf{I} = \mathbf{U}(\mathbf{\Sigma}^2_K + \lambda\mathbf{I})\mathbf{U}^*$, $\mathbf{K} + \lambda\mathbf{I}$ is positive definite as $\lambda > 0$, $\mathbf{K}(\mathbf{K} + \lambda\mathbf{I}) = \mathbf{U}(\mathbf{\Sigma}^4_K + \lambda\mathbf{\Sigma}^2_K)\mathbf{U}^*$. $\therefore$ $\mathbf{K}(\mathbf{K} + \lambda\mathbf{I})$ is positive definite as $\lambda > 0$. Therefore, $F(\alpha, \mathbf{d})$ is convex w.r.t. α.

References

[1] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *IEEE T-PAMI*, 33(No.8):1619 –1632, 2011. 2, 6

[2] S. Baker and I. Matthews. Lucas-kanade 20 years on: A unifying framework. *IJCV*, Vol.56(No.3):221–255, 2004. 2

[3] V. Boddeti, T. Kanade, and B. Kumar. Correlation filters for object alignment. In *CVPR*, 2013. 2

[4] D. Bolme, R. Beveridge, B. Draper, and Y. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 1, 2, 4, 5

[5] C. Cortes, M. Mohri, and A. Rostamizadeh. l_2 regularization for learning kernels. In *UAI*, 2009. 2

[6] M. Danelljan, G. Hager, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 1, 2, 6, 7, 8

[7] M. Danelljan, F. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, 2014. 1, 2, 4, 6, 7, 8

[8] P. Dollár, R. Appel, S. Belongie, and P. Perona. Fast feature pyramids for object detection. *IEEE T-PAMI*, Vol.36:1532–1545, 2014. 2, 6, 7

[9] R. Fletcher. *Practical Methods of Optimization*. John Wiley & Sons, 2^{nd} edition, 1987. 4

[10] H. Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. In *ICCV*, 2013. 2

[11] P. Gehler and S. Nowozin. On feature combination for multiclass object classification. In *ICCV*, 2009. 1, 2

[12] R. Gray. *Toeplitz and Circulant Matrices: A review*. Now Publishers Inc., 2006. 4, 8

[13] S. Hare, A. Saffari, and P. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011. 1, 6, 7, 8

[14] J. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In *ICCV*, 2013. 2

[15] J. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012. 1, 2, 4, 5, 6, 8

[16] J. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE T-PAMI*, Vol.37(No.3):pp.583–596, 2015. 1, 2, 5, 6, 7, 8

[17] J. Henriques, P. Martins, R. Caseiro, and J. Batista. Fast training of pose detectors in the fourier domain. In *NIPS*, 2014. 2, 4

[18] Z. Hong, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Tracking using multilevel quantizations. In *ECCV*, 2014. 1

[19] B. Kumar, A. Mahalanobis, and R. Juday. *Correlation Pattern Recognition*. Cambridge University Press, 2005. 2

[20] J. Kwon, J. Roh, K.-M. Lee, and L. Van Gool. Robust visual tracking with double bounding box model. In *ECCV*, 2014. 1

[21] X. Lan, A. Ma, and P. Yuen. Multi-cue visual tracking using robust feature-level fusion based on joint sparse representation. In *CVPR*, 2014. 1

[22] G. Lanckriet, T. De Bie, N. Cristianini, M. Jordan, and W. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20:2626–2635, 2004. 3

[23] D. Lee, J.-Y. Sim, and C.-S. Kim. Visual tracking using pertinent patch selection and masking. In *CVPR*, 2014. 1

[24] D. Luenberger. *Introduction to Linear and Nonlinear Programming*. Addison-Wesley Reading, MA, 1973. 5

[25] C. Micchelle. Interpolation of scattered data: Distance matrices and conditionally positive definite functions. *Constructive Appr.*, Vol.2:pp.11–22, 1986. 6

[26] H. Nam, S. Hong, and B. Han. Online graph-based tracking. In *ECCV*, 2014. 1

[27] S. Oron, A. Bar-Hillel, and S. Avidan. Extended lucas-kanade tracking. In *ECCV*, 2014. 2

[28] A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *JMLR*, 9:2491–2521, 2008. 2

[29] R. Rifkin, G. Yeo, and T. Poggio. Regularized least-squares classification. *Nato Science Series Sub Series III: Computer and Systems Sciences*, pp131-154.:2003, 190. 2, 3

[30] D. Ruderman. The statistics of natural images. *Network: Computation in Neural Systems*, 5:517–548, 1994. 2

[31] B. Schölkopf and A. Smola. *Learning with Kernels*. MIT press Cambridge, MA, 2002. 3

[32] M. Tang and X. Peng. Robust tracking with discriminative ranking lists. *IEEE T-IP*, Vol.21(No.7):3273–3281, 2012. 1

[33] M. Varma and D. Ray. Learning the discriminative power-invariance trade-off. In *ICCV*, 2007. 1, 2, 3

[34] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009. 2

[35] Y. Wu, G. Blasch, G. Chen, L. Bai, and H. Ling. Multiple source data fusion via sparse representation for robust visual tracking. In *FUSION*, 2011. 1

[36] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking - a benchmark. In *CVPR*, 2013. 1, 2, 6, 7, 8

[37] F. Yang, H. Lu, and M. Yang. Robust visual tracking via multiple kernel boosting with afnity constraints. *IEEE T-CSVT*, 24:242–254, 2014. 1