Fast Kernelized Correlation Filter without Boundary Effect

Ming TANG[1,3]*, Linyu ZHENG[1,2]*, Bin YU[1,2], and Jinqiao WANG[1,4]
[1] National Lab of Pattern Recognition, Institute of Automation, CAS, Beijing 100190, China
[2] School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China
[3] Shenzhen Infinova Limited, [4] ObjectEye Inc.

Abstract

In recent years, correlation filter based trackers (CF trackers) have attracted much attention from the vision community because of their top performance in both localization accuracy and efficiency. The society of visual tracking, however, still needs to deal with the following difficulty on CF trackers: avoiding or eliminating the boundary effect completely, in the meantime, exploiting non-linear kernels and running efficiently. In this paper, we propose a fast kernelized correlation filter without boundary effect (nBEKCF) to solve this problem. To avoid the boundary effect thoroughly, a set of real *and* dense *patches is sampled through the traditional sliding window and used as the training samples to train nBEKCF to fit a Gaussian response map. Non-linear kernels can be applied naturally in nBEKCF due to its different theoretical foundation from the existing CF trackers'. To achieve the fast training and detection, a set of cyclic bases is introduced to construct the filter. Two algorithms, ACSII and CCIM, are developed to significantly accelerate the calculation of kernel correlation matrices. ACSII and CCIM fully exploit the density of training samples and cyclic structure of bases, and totally run in space domain. The efficiency of CCIM exceeds that of the FFT counterpart remarkably in our task. Extensive experiments on six public datasets, OTB-2013, OTB-2015, NfS, VOT2018, GOT10k, and TrackingNet, show that compared to the CF trackers designed to relax the boundary effect, BACF and SRDCF, our nBEKCF achieves higher localization accuracy without tricks, in the meanwhile, runs at higher FPS.*

1. Introduction

Visual tracking is one of the fundamental problems in computer vision with many applications. Despite significant progress in recent years [22, 19, 18, 20, 25, 21], visual tracking is still a challenge [41] due to some severe interferences (e.g. large appearance changes, occlusions, background clutters and fast motion), very limited training samples, and the requirement of low computational cost. In tracking task, it is crucial to construct a robust appearance model from very limited samples to distinguish a target from distractive background, while maintaining high efficiencies.

Since 2010, correlation filter based trackers (CF trackers) have been achieving a great success [5, 13, 9, 32, 10, 16, 7, 4, 33, 31, 44]. Almost all CF trackers learn their filters with cyclic samples to regress Gaussian response maps, and their training and detection are accelerated with the convolution theorem and fast Fourier transform (FFT). Bolme *et al*. [5] proposed the Minimum Output Sum of Squared Error (MOSSE) for very high speed tracking on gray-scale sequences. They used base image patches and all their cyclical shifts to train the appearance model directly in Fourier domain. Henriques *et al*. [14] reformulated MOSSE as a ridge regression problem in space domain, being able to apply multi-channel features and non-linear kernels naturally to improve the localization accuracy. In their CF tracker, kernelized correlation filter (KCF), the regression function and its optimization problem are expressed as

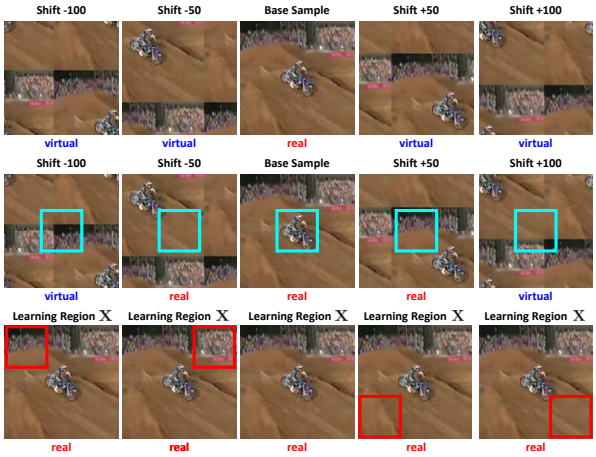
$$f(\mathbf{X}) = \sum_{s=0}^{m-1}\sum_{t=0}^{n-1} \alpha_{s,t}\kappa(\mathbf{X}, \mathbf{X}_{s,t}) \tag{1}$$

and

$$\min_{\boldsymbol{\alpha}} \|\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}\|_2^2 + \lambda \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha}, \tag{2}$$

respectively, where $\mathbf{X}$ and $\mathbf{X}_{s,t}$'s are cyclic sample, and $\mathbf{K}$ is the kernel matrix with $\kappa(\mathbf{X}_{u,v}, \mathbf{X}_{s,t})$ as its elements, $\{\mathbf{X}_{u,v}\} \equiv \{\mathbf{X}_{s,t}\} \equiv \mathcal{X}_{\text{KCF}}$, $\mathcal{X}_{\text{KCF}}$ is the set of cyclic samples. Compared to MOSSE, KCF achieves a much higher accuracy on OTB-2013 [40] when exploiting HOG feature [6] and Gaussian kernel, meanwhile, it is still able to run at a high speed. However, the use of FFT produces the cyclicity of samples which leads to the problem of *boundary effect* [17] in MOSSE, KCF, and many CF trackers [32, 27, 29, 35, 43, 33]. As shown in Fig. 1a, the boundary effect means that almost all training and detection samples are unreal and synthesized by cyclically shifting base samples, and these unreal samples are still supposed to represent those real ones

*Indicates equal contribution. The corresponding author is Ming TANG (tangm@nlpr.ia.ac.cn). This work was supported by the Research and Development Projects in the Key Areas of Guangdong Province (No. 2020B010165001). This work was also supported by NSFC under Grants 61772527, 61976210, 61806200, 61702510 and 61876086.



(a)

$$\mathcal{X} = \{\mathbf{X}^{u,v}\}_{u=0,v=0}^{\tilde{M},\tilde{N}} =$$

$$\mathcal{Z} = \{\mathbf{Z}^{s,t}\}_{s=0,t=0}^{m,n} =$$

(b)

Figure 1: (a) Comparison of sampling methods in KCF [14] (the first row), BACF [16] (the second row), fdKCF* [44] and our nBEKCF (the last row). Training samples of KCF come from all possible cyclic shifts of a base sample (*i.e.*, the central patch), and they are all virtual except for the base one. So do those of SRDCF [9] and ECO [7]. BACF obtains its training samples of target size (cyan boxes) by clipping the middle parts of all training samples of KCF, therefore, some of them are virtual. Different from KCF and BACF, in fdKCF* and our nBEKCF, the training samples of target size (red boxes) are densely sampled from the learning region $\mathbf{X}$ with the traditional sliding window, and they are all real. We call such sampling method as *real and dense sampling*. (b) Illustrations of a training set $\mathcal{X}$ and a set $\mathcal{Z}$ of cyclic bases in nBEKCF. $\mathbf{X}^{u,v}$'s are sampled from $\mathbf{X}$ by using the real and dense sampling method. The elements, $\mathbf{Z}^{s,t}$'s, of $\mathcal{Z}$ are pre-defined and totally *cyclic*. $\mathcal{Z}$ is constructed by *all possible cyclic shifts* of the target patch (*i.e.*, the central red box on the last row of (a)) in $\mathbf{X}$, although theoretically it is not necessary to use target patches to generate $\mathbf{Z}^{s,t}$'s. Note that both the density of $\mathcal{X}$ and the cyclicity of $\mathcal{Z}$ are crucial to the high efficiency of our nBEKCF.

at different translational shifts in training and localization.[1] As for the training of KCF, all training samples, except for a real base sample, *i.e.*, $\mathbf{X}_{0,0}$, are unreal in Problem (2). In practice, the boundary effect dramatically reduces the discriminative power of appearance models and greatly degrades the localization accuracy which MOSSE and KCF could have achieved.

In order to relax the boundary effect to improve localization accuracy of the KCF with linear kernel, Galoogahi *et al*. [16] and Danelljan *et al*. [9] proposed the background-aware correlation filter (BACF) and spatially regularized discriminative correlation filter (SRDCF), respectively. In BACF, a rectangular mask is introduced into the error item of Problem (2) to cover the cyclic samples, and then the alternating direction method of multipliers (ADMM) is employed to solve the optimization problem with equality constraints. Although the boundary effect can be reduced greatly by introducing the mask, it cannot be eliminated completely in BACF, as pointed out in [17] and [39]. In SRDCF, a smooth spatial regularization factor is introduced into the regularizer of Problem (2) to penalize the filter coefficients depending on their spatial locations. The regularization factor acts really similarly to the mask of BACF in practice [39], being not able to eliminate the boundary effect thoroughly. On the other hand, due to the ways they relax the boundary effect, SRDCF and BACF cannot run in high efficiencies when employing the powerful high-dimensional features. Although it has been shown that trackers can improve their accuracies by large margins with non-linear kernels [45], SRDCF and BACF are also unable to exploit the non-linear kernels to improve their localization accuracy because the window shape is unknown in the non-linear kernel space, despite the fact that it is known in an image or filter with linear kernel.

fdKCF* [44] is another try to avoid boundary effects of KCF. It optimizes the regression model of KCF in space domain, eliminates almost all repeated calculations, and then uses GPU to accelerate the remaining ones. But, fdKCF* has to resort to GPU to run in super-real-time even if low dimensional hand-crafted features are employed.

According to the above, it is seen that there still exists the following problem for CF trackers: avoiding or eliminating the boundary effect completely, and in the meantime, exploiting non-linear kernels and running in a high efficiency.

In fact, the efficiencies of MOSSE and KCF totally rely on the boundary effect. Therefore, the efforts of BACF and SRDCF to reduce the boundary effect on the theoretical foundation of KCF is bound to weaken their efficiencies seriously. Moreover, as a side-effect, BACF and SRDCF's efforts to reduce the boundary effect inhibit them from applying non-linear kernels to improve their accuracies. According to this

[1] Readers may refer to the 4th paragraph of Sec.1 in [17] for other details.

analysis, we believe that a totally novel theoretical foundation other than that of KCF is necessary to develop a novel type of correlation filter to address the above problem.

Now that both the efficiency and the boundary effect of KCF is from the cyclicity of $\mathcal{X}_{\text{KCF}}$, if $\{\mathbf{X}_{u,v}\}$ and $\{\mathbf{X}_{s,t}\}$ are treated as two different sets, *i.e.*, treat $\{\mathbf{X}_{u,v}\}$ as a training set and $\{\mathbf{X}_{s,t}\}$ as a base set, the cyclicity of $\{\mathbf{X}_{u,v}\}$ is cancelled, *i.e.*, all $\mathbf{X}_{u,v}$'s are real, and the cyclicity of $\{\mathbf{X}_{x,t}\}$ is kept, then the boundary effect of KCF disappears and the efficient calculation is still able to achieved. According to this idea, in this paper, we propose a novel type of CF tracker, a fast kernelized correlation filter without boundary effect (nBEKCF), which is totally different from KCF in theory, to solve the above problem. As shown in Fig.1a, unlike most existing CF trackers which exploit both real and synthetic patches generated from a base image patch as training samples to train their filters with FFT, our nBEKCF draws a set of training samples by using the real and dense sampling method to fit a Gaussian response map, avoiding the boundary effect thoroughly. To train the filter and locate the target object efficiently, a set of cyclic bases is introduced and exploited without FFT. Specifically, a set of basis functions, $\{\kappa(\cdot, \mathbf{Z}^{s,t})\}$, is generated with a set of pre-defined cyclic bases $\mathcal{Z} = \{\mathbf{Z}^{s,t}\}$, and our filter is formulated with this basis function set, *i.e.*, $f(\cdot; \mathbf{Z}^{s,t}) = \sum_{s,t} \alpha_{s,t} \kappa(\cdot, \mathbf{Z}^{s,t})$. Training set $\mathcal{X}$ consists of real $\mathbf{X}^{u,v}$'s which are *densely sampled* from learning region $\mathbf{X}$ in a pixel-wise way. Fig.1b illustrates training sets $\mathcal{X}$ and $\mathcal{Z}$. Then, filter $f(\cdot; \mathbf{Z}^{s,t})$ regresses training set $\mathcal{X} = \{\mathbf{X}^{u,v}\}$ to a Gaussian response map. Note that the non-linear kernels can be applied naturally in $f(\cdot; \mathbf{Z}^{s,t})$. In order to treat multiple frames and update the filter efficiently, the modeling scheme over multiple frames [11, 33] is adapted to nBEKCF. It is worth noticing that $\mathcal{X} \neq \mathcal{Z}$ and only $\mathcal{Z}$ is cyclic in nBEKCF, but $\mathcal{X} \equiv \mathcal{Z} \equiv \mathcal{X}_{\text{KCF}}$ and $\mathcal{X}_{\text{KCF}}$ is cyclic in KCF.

It is found that the key to improve the efficiency of training and detection is the quick calculation of kernel correlation matrices in nBEKCF. Therefore, we develop two *non-FFT* based algorithms, autocorrelation with squared integral image (ACSII) and cyclic correlation with integral matrix (CCIM), to significantly accelerate the calculation by fully exploiting the density of $\mathcal{X}$ and cyclic structure of $\mathcal{Z}$. In our approach, a kernel correlation matrix is constructed with $\kappa(,)$, $\mathcal{Z}$ and $\mathcal{X}$. By exploiting a great deal of overlap among densely sampled $\mathbf{X}^{u,v}$'s, ACSII calculates the autocorrelation efficiently. By exploiting both the density of $\mathbf{X}^{u,v}$'s and the cyclicity of $\mathbf{Z}^{s,t}$'s, CCIM is remarkably more efficient than that of FFT in calculating the correlation in our task.

Consequently, the regression problem is solved efficiently in space domain, rather than by means of frequency domain, in nBEKCF. It is $\mathcal{X} \neq \mathcal{Z}$ and all elements of $\mathcal{X}$ being real that make nBEKCF free from the boundary effect, and it is $\mathcal{X} \neq \mathcal{Z}$ and the cyclicity of $\mathcal{X}$ that make nBEKCF locate the target object really efficiently.

Our nBEKCF is tested on six public datasets, OTB-2013, OTB-2015, NfS, VOT2018, GOT10k, and TrackingNet. The experimental results show that nBEKCF achieves state-of-the-art accuracy, and compared to the trackers designed to relax the boundary effect, BACF and SRDCF, our nBEKCF with hand-crafted features, HOG and CN, obtains higher localization accuracy without tricks and is able to run at higher FPS (50 on average).

2. Kernelized Correlation Filter without Boundary Effect (nBEKCF)

In this section, we will first introduce our novel nBEKCF with a single frame, then extend it to the historical frames, and finally present how to locate the target object with nBEKCF in the current frame.

2.1. nBEKCF with Single Frame

Let $\mathbf{Z} \in \mathbb{R}^{m \times n \times D}$ be the D-channel feature map of the base patch. In our current implementation, the base patch is the target object patch. The set of cyclic bases, $\mathcal{Z} = \{\mathbf{Z}^{s,t}\}_{s=0,t=0}^{m-1,n-1}$, is generated by

$$\mathbf{Z}^{s,t}(d) = \mathbf{P}_m^s \mathbf{Z}(d) \mathbf{Q}_n^t, \tag{3}$$

where $d = 0, \ldots, D-1$, $\mathbf{Z}(d)$ is the d-th channel of $\mathbf{Z}$, $\mathbf{P}_m$ and $\mathbf{Q}_n$ are the $m \times m$ and $n \times n$ permutation matrices [12], respectively,

$$\mathbf{P}_m = \begin{bmatrix} \mathbf{0}_{m-1}^\top & 1 \\ \mathbf{I}_{m-1} & \mathbf{0}_{m-1} \end{bmatrix}, \quad \mathbf{Q}_n = \begin{bmatrix} \mathbf{0}_{n-1} & \mathbf{I}_{n-1} \\ 1 & \mathbf{0}_{n-1}^\top \end{bmatrix},$$

where $\mathbf{0}_{l-1}$ is the $(l-1) \times 1$ zero vector, $\mathbf{I}_{l-1}$ is the $(l-1) \times (l-1)$ identity matrix, $l \in \{m, n\}$, and $\mathbf{P}_m^\rho$ and $\mathbf{Q}_n^\rho$ are the ρ-th power of $\mathbf{P}_m$ and $\mathbf{Q}_n$, respectively. Note that $\mathbf{Z}^{0,0}(d) = \mathbf{Z}(d)$. Intuitively, $\mathbf{P}_m^\rho \mathbf{Z}(d)$ cyclically shifts $\mathbf{Z}(d)$'s rows down by ρ rows and $\mathbf{Z}(d)\mathbf{Q}_n^\rho$ cyclically shifts $\mathbf{Z}(d)$'s columns right by ρ columns, when $\rho \geqslant 0$.[2] Therefore, $\mathbf{Z}^{0,0} = \mathbf{Z}$, and $\mathcal{Z}$ consists of all possible cyclic shifts of $\mathbf{Z}$ on 2D spatial domain.

Furthermore, let $\mathbf{X} \in \mathbb{R}^{M \times N \times D}$ be the D-channel feature map of learning region. We generate the set of real (*i.e.*, non-cyclic) training samples of target size, $\mathcal{X} = \left\{\mathbf{X}^{u,v} \in \mathbb{R}^{m \times n \times D}\right\}_{u=0,v=0}^{M-m,N-n}$, through real and dense sampling from $\mathbf{X}$, as shown in Fig.1.

Then, let $f : \mathbb{R}^{m \times n \times D} \to \mathbb{R}$, kernel $\kappa : \mathbb{R}^{m \times n \times D} \times \mathbb{R}^{m \times n \times D} \to \mathbb{R}$, and consider $\{\kappa(\cdot, \mathbf{Z}^{s,t})\}_{s=0,t=0}^{m-1,n-1}$ as a set of basis functions. We define the kernelized correlation filter without boundary effect (nBEKCF) as

$$f(\mathbf{X}_r) = \sum_{s=0}^{m-1} \sum_{t=0}^{n-1} \alpha_{s,t} \kappa(\mathbf{X}_r, \mathbf{Z}^{s,t}), \tag{4}$$

[2] If $\rho < 0$, the direction of shift is opposite to that of $\rho > 0$.

where $\mathbf{X}_r \in \mathbb{R}^{m\times n\times D}$, and model the ridge regression problem as

$$\begin{aligned}\min_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha}) &\equiv \sum_{u=0}^{\tilde{M}}\sum_{v=0}^{\tilde{N}} \left(f(\mathbf{X}^{u,v}) - y_{u,v}\right)^2 + \lambda \boldsymbol{\alpha}^\top \boldsymbol{\alpha} \\ &= \|\mathbf{K}\boldsymbol{\alpha} - \mathbf{y}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_2^2,\end{aligned} \tag{5}$$

where $\lambda > 0$ is the regularization parameter, $\tilde{M} = M - m$, $\tilde{N} = N - n$, $\mathbf{y} = [y_{0,0}, y_{0,1}, \ldots, y_{\tilde{M},\tilde{N}-1}, y_{\tilde{M},\tilde{N}}]$ is the vector of gaussian labels, and

$$\mathbf{K} = \begin{bmatrix} \kappa\left(\mathbf{X}^{0,0}, \mathbf{Z}^{0,0}\right) & \cdots & \kappa\left(\mathbf{X}^{0,0}, \mathbf{Z}^{m-1,n-1}\right) \\ \vdots & \ddots & \vdots \\ \kappa\left(\mathbf{X}^{\tilde{M},\tilde{N}}, \mathbf{Z}^{0,0}\right) & \cdots & \kappa\left(\mathbf{X}^{\tilde{M},\tilde{N}}, \mathbf{Z}^{m-1,n-1}\right) \end{bmatrix}$$

is called the *kernel correlation matrix* of $\mathcal{X}$ and $\mathcal{Z}$. Note that the $\boldsymbol{\alpha}$ of Problem 5 is not the variable of dual space, thus different from the $\boldsymbol{\alpha}$ of KCF.

To solve for $\boldsymbol{\alpha}^*$, let $\nabla_{\boldsymbol{\alpha}} F(\boldsymbol{\alpha}) = 0$; it is achieved that $\left(\mathbf{K}^\top\mathbf{K} + \lambda\mathbf{I}\right)\boldsymbol{\alpha}^* = \mathbf{K}^\top\mathbf{y}$. Because $\mathbf{K}^\top\mathbf{K}$ is semi-positive definite, $\mathbf{K}^\top\mathbf{K} + \lambda\mathbf{I}$ is invertible. Consequently, the optimal solution of Problem (5) is

$$\boldsymbol{\alpha}^* = \left(\mathbf{K}^\top\mathbf{K} + \lambda\mathbf{I}\right)^{-1}\mathbf{K}^\top\mathbf{y}. \tag{6}$$

The efficiency of calculating $\boldsymbol{\alpha}^*$ is mainly determined by that of constructing $\mathbf{K}$, and the computational burdens of multiplication and inversion of matrices are almost negligible relative to that of constructing $\mathbf{K}$. It is thanks to the real and dense samples in $\mathcal{X}$ and the totally cyclic bases in $\mathcal{Z}$ that the fast construction of $\mathbf{K}$ can be realized. Sec 3 will elaborate how to exploit the density of $\mathcal{X}$ and the cyclicity of $\mathcal{Z}$ to construct $\mathbf{K}$ efficiently without FFT.

It is clear that if $\mathcal{X}$ is cyclic and $\mathcal{Z} \equiv \mathcal{X}$, $\mathbf{K}$ will be a Gram matrix and the *error item* of Problem (5) will be exactly the same as that of KCF in Problem (2). But, the learned $f(\cdot; \mathbf{Z}^{s,t})$ in Eq.(4) is essentially different from Eq.(1) of KCF, because $\mathcal{Z} \neq \mathcal{X}$, $\mathcal{X}$ is not cyclic and $\mathcal{Z}$ is cyclic in Eq.(4). It is the characteristics of $\mathcal{Z}$ and $\mathcal{X}$ that make possible the efficient optimization of Problem (5) (Sec. 3). Note that KCF is *not* a special case of nBEKCF because their *regularization items* are different. Sec. 4 will present major differences between nBEKCF and the existing CF trackers.

2.2. nBEKCF with Historical Frames

In visual tracking task, the appearance model is often trained with multiple frames of different times to improve its robustness. In this section, we will adapt the update scheme proposed in [11] to our nBEKCF.

Specifically, we model the ridge regression problem of multiple frames as follows.

$$\min_{\boldsymbol{\alpha}_Q} F_Q(\boldsymbol{\alpha}_Q) \equiv \sum_{q=1}^{Q} \beta_q \|\mathbf{K}_q\boldsymbol{\alpha}_Q - \mathbf{y}\|_2^2 + \lambda\|\boldsymbol{\alpha}_Q\|_2^2, \tag{7}$$

where Q is the number of historical frames, $q = 1$ is the initial frame and $q = Q$ is the present one, $\mathbf{K}_q$ is the kernel correlation matrix of $\mathcal{X}_q$ and $\mathcal{Z}_q$ in frame q, $\beta_1 = (1-\gamma)^{Q-1}$, $\beta_q = \gamma(1-\gamma)^{Q-q}$ for all $q \geqslant 2$, $\sum_{q=1}^{Q}\beta_q = 1$, and $\gamma \in [0, 1]$ is the learning rate.

Let $\nabla_{\boldsymbol{\alpha}_Q} F_Q(\boldsymbol{\alpha}_Q) = 0$. It is achieved that

$$\boldsymbol{\alpha}_Q^* = \left[\sum_{q=1}^{Q}\beta_q\mathbf{K}_q^\top\mathbf{K}_q + \lambda\mathbf{I}\right]^{-1}\sum_{q=1}^{Q}\beta_q\mathbf{K}_q^\top\mathbf{y}.$$

While the frames come sequentially, an efficient update scheme can be designed as follows.

$$\begin{gathered}\boldsymbol{\alpha}_Q^* = (\mathbf{A}_Q + \lambda\mathbf{I})^{-1}\mathbf{B}_Q, \\ \mathbf{A}_Q = (1-\gamma)\,\mathbf{A}_{Q-1} + \gamma\mathbf{K}_Q^\top\mathbf{K}_Q, \mathbf{B}_Q = (1-\gamma)\,\mathbf{B}_{Q-1} + \gamma\mathbf{K}_Q^\top\mathbf{y}, \\ \hat{\mathbf{X}}_Q = (1-\gamma)\,\hat{\mathbf{X}}_{Q-1} + \gamma\mathbf{X}_Q, \qquad \hat{\mathbf{Z}}_Q = (1-\gamma)\,\hat{\mathbf{Z}}_{Q-1} + \gamma\mathbf{Z}_Q,\end{gathered}$$

where $\mathbf{X}_Q$ and $\mathbf{Z}_Q$ are the $\mathbf{X}$ and $\mathbf{Z}$ in frame Q, respectively. $\hat{\mathbf{X}}_Q$ is updated for the calculation of $\mathbf{K}_Q$. This scheme allows the model to update without storing the previous ones. Only the current $\{\mathbf{A}_Q, \mathbf{B}_Q, \hat{\mathbf{X}}_Q, \hat{\mathbf{Z}}_Q\}$ needs to be saved.

2.3. Detection of Target Object

Given D-channel feature map $\mathbf{X}'_{Q+1} \in \mathbb{R}^{M\times N\times D}$ of search region in the current frame $Q+1$. Construct $\mathcal{X}'_{Q+1}$ and $\hat{\mathcal{Z}}_Q$ based on $\mathbf{X}'_{Q+1}$ and $\hat{\mathbf{Z}}_Q$, respectively, with the methods of constructing $\mathcal{X}$ and $\mathcal{Z}$ presented in Sec. 2.1. Then, the response map of $\mathcal{X}'_{Q+1}$ can be obtained with $\mathbf{y}' = \mathbf{K}'\boldsymbol{\alpha}_Q^*$, where $\mathbf{K}'$ is the kernel correlation matrix of $\mathcal{X}'_{Q+1}$ and $\hat{\mathcal{Z}}_Q$.

The element of $\mathbf{y}'$ which takes the maximal value is accepted as the optimal location of the target object in frame $Q+1$. The optimal scale of target object is estimated by using DSST [8].

3. Fast Calculation of Correlation Matrix

While solving for the optimal $\boldsymbol{\alpha}^*$ and detecting the target object, kernel correlation matrix, $\mathbf{K}$,[3] has to be constructed first. It is clear that $\mathbf{K}$ can be constructed by using the brute-force approach with computational complexity $O\left(m^2n^2MND\right)$ because $(\tilde{M}+1)(\tilde{N}+1)\times mn$ elements are contained and the calculation of each element involves two samples of mnD dimensions. This complexity, however, is too high for some time-sensitive tasks such as visual tracking, because m^2n^2MND is often too large there. Typically, $M = N = 60$, $m = 15$, $n = 20$, and $D = 31 + 10$ when HOG [6] and CN [36] are adopted with the cell size being 4×4 in CF trackers [32, 7, 33]. Therefore, it is necessary to develop a fast algorithm to construct $\mathbf{K}$. Otherwise, it is almost impossible to apply nBEKCF in such time-sensitive tasks. On the other hand, it is noticed that, while solving

[3] Refer to $\mathbf{K}$, $\mathbf{K}_Q$, and $\mathbf{K}'$ in Sec. 2.

for $\boldsymbol{\alpha}^*$, the computational cost of matrix inversion is usually not a main bottleneck for efficient solution if the inversion is achieved through solving a system of linear equations, because $\mathbf{K}^\top\mathbf{K} \in \mathbb{R}^{mn\times mn}$ and mn is usually not too large, *i.e.*, 300 in the above example.

While constructing $\mathbf{K}$, most common kernels, such as dot-product kernels, polynomial kernels, and Gaussian kernel [14], can be employed to calculate its elements. The calculation of these kernels in calculating $\mathbf{K}$ is involved in the autocorrelation of $\mathcal{X}$,

$$\mathcal{X}\circ\mathcal{X} = \begin{bmatrix} \langle \mathbf{X}^{0,0}, \mathbf{X}^{0,0}\rangle, & \cdots, & \langle \mathbf{X}^{0,\tilde{N}}, \mathbf{X}^{0,\tilde{N}}\rangle \\ \vdots & \ddots & \vdots \\ \langle \mathbf{X}^{\tilde{M},0}, \mathbf{X}^{\tilde{M},0}\rangle & \cdots & \langle \mathbf{X}^{\tilde{M},\tilde{N}}, \mathbf{X}^{\tilde{M},\tilde{N}}\rangle \end{bmatrix},$$

and the cross-correlation (briefly, correlation) of $\mathcal{X}$ and $\mathcal{Z}$,

$$\mathcal{X}\diamond\mathcal{Z} = \begin{bmatrix} \langle \mathbf{X}^{0,0}, \mathbf{Z}^{0,0}\rangle & \cdots & \langle \mathbf{X}^{0,0}, \mathbf{Z}^{m-1,n-1}\rangle \\ \vdots & \ddots & \vdots \\ \langle \mathbf{X}^{\tilde{M},\tilde{N}}, \mathbf{Z}^{0,0}\rangle & \cdots & \langle \mathbf{X}^{\tilde{M},\tilde{N}}, \mathbf{Z}^{m-1,n-1}\rangle \end{bmatrix},$$

where $\langle\cdot,\cdot\rangle$ is the dot product. It is clear that the computational complexities of constructing $\mathcal{X}\circ\mathcal{X}$ and $\mathcal{X}\diamond\mathcal{Z}$ will be $O(mnMND)$ and $O(m^2n^2MND)$, respectively, if the brute-force approach is employed.

A simple idea is to employ FFT to accelerate the construction of $\mathbf{K}$. FFT, however, can only accelerate $\mathcal{X}\diamond\mathcal{Z}$ with the computational complexity $O(mnMND\log MN)$. This is still not satisfactory for time-sensitive tasks. The reason that FFT is not the optimal choice for our task is that it can only take advantage of the density of $\mathcal{X}$ but is *not* able to exploit the cyclicity of $\mathcal{Z}$. Moreover, FFT cannot accelerate $\mathcal{X}\circ\mathcal{X}$.

In this section, by means of the principle of integral image [38], we will present two novel algorithms, autocorrelation with integral image (ACSII) and cyclic correlation with integral matrix (CCIM), to calculate $\mathcal{X}\circ\mathcal{X}$ and $\mathcal{X}\diamond\mathcal{Z}$ efficiently in space domain. The key idea of ACSII and CCIM is to fully exploit the structures of bases and samples, *i.e.*, the bases are cyclic in $\mathcal{Z}$ and samples are densely sampled in $\mathcal{X}$, to eliminate all redundant computations.

3.1. Autocorrelation with Squared Integral Image

Let $\{\mathbf{x}_{p,q}\in\mathbb{R}^D\}_{p=0,q=0}^{M-1.N-1}$ enumerate all 2D spatial locations of $\mathbf{X}$. If the brute-force approach is employed to calculate $\mathcal{X}\circ\mathcal{X}$, *i.e.*, to calculate all its elements via

$$\langle \mathbf{X}^{u,v}, \mathbf{X}^{u,v}\rangle = \sum_{p=u}^{u+m-1}\sum_{q=v}^{v+n-1} \|\mathbf{x}_{p,q}\|_2^2,$$

where $u=[0,\tilde{M}]$, $v=[0,\tilde{N}]$, there will exist large amounts of redundant calculations because there are high overlaps between neighboring samples of $\mathcal{X}$, as shown in Fig.1b. $\|\mathbf{x}_{p,q}\|_2^2$ and $\|\mathbf{x}_{p,q}\|_2^2+\|\mathbf{x}_{p',q'}\|_2^2$ will be performed multiple times for most (p,q)'s and their neighbors (p',q')'s.

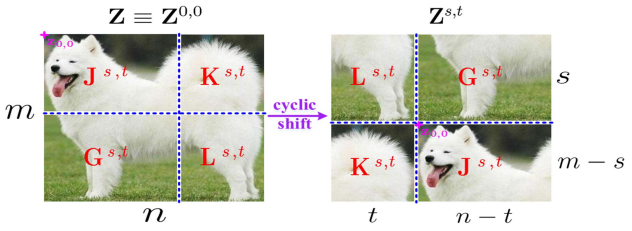


Figure 2: The division of cyclic base $\mathbf{Z}^{s,t}$ into four sub-bases, $\mathbf{L}^{s,t}$, $\mathbf{G}^{s,t}$, $\mathbf{K}^{s,t}$, and $\mathbf{J}^{s,t}$, by the location of $\mathbf{z}_{0,0}$. The relative spatial locations of elements of $\mathbf{D}^{s,t}$ are the same as their relative locations in $\mathbf{Z}$, where $\mathbf{D}^{s,t}\in\{\mathbf{L}^{s,t},\mathbf{G}^{s,t},\mathbf{K}^{s,t},\mathbf{J}^{s,t}\}$.

To eliminate the redundancy, we propose a novel algorithm, ACSII, to fast calculate the autocorrelation $\mathcal{X}\circ\mathcal{X}$ by means of integral image $\mathbf{I}\in\mathbb{R}^{M\times N}$ with $I_{p,q}=\sum_{i=0}^{p}\sum_{j=0}^{q}\|\mathbf{x}_{i,j}\|_2^2$ as its elements. In this way, any $\langle \mathbf{X}^{u,v}, \mathbf{X}^{u,v}\rangle$ is calculated in a constant time, *i.e.*, three additions of scalars, as follows.

$$\langle \mathbf{X}^{u,v}, \mathbf{X}^{u,v}\rangle = I_{u+m-1,v+n-1} - I_{u,v+n-1} - I_{u+m-1,v} + I_{u,v}.$$

The detailed technical steps of ACSII are presented in Alg. 1 of [34] and supplementary material. It can be seen that ACSII is really similar to the integral image. In fact, ACSII is the integral image except that it acts on the squared vector norm.

3.2. Cyclic Correlation with Integral Matrix

Suppose $\mathbf{H}_{((a_1,b_1),(a_2,b_2))}$ is the sub-matrix of matrix $\mathbf{H}$ with (a_1,b_1) and (a_2,b_2) as its top-left and down-right corners. Let $\{\mathbf{z}_{s,t}\in\mathbb{R}^D\}_{s=0,t=0}^{m-1,n-1}$ enumerate all 2D spatial locations of $\mathbf{Z}$. If the brute-force approach is employed to calculate $\mathcal{X}\diamond\mathcal{Z}$, *i.e.*, to calculate $\mathcal{X}\diamond\mathcal{Z}$ through

$$\mathcal{X}\diamond\mathcal{Z} = [\mathrm{vec}(\mathbf{Z}^{0,0}\text{☆}\mathbf{X}),\cdots,\mathrm{vec}(\mathbf{Z}^{m-1,n-1}\text{☆}\mathbf{X})], \quad (9)$$

where ☆ is the correlation operator and $\mathrm{vec}(\mathbf{H})$ indicates the vectorization of $\mathbf{H}$, there will be large amounts of redundant calculations because all bases, $\mathbf{Z}^{s,t}$'s, of $\mathcal{Z}$ are obtained by cyclically shifting $\mathbf{Z}$. In fact, $\langle \mathbf{z}_{s,t}, \mathbf{x}_{p,q}\rangle$ and $\langle \mathbf{z}_{s,t}, \mathbf{x}_{p,q}\rangle + \langle \mathbf{z}_{s+\Delta s,t+\Delta t}, \mathbf{x}_{p+\Delta s,q+\Delta t}\rangle$ will be performed multiple times for most s, t, p, and q.

As shown in Fig. 2, according to the cyclicity of $\mathcal{Z}$, any base $\mathbf{Z}^{s,t}\in\mathcal{Z}$ can always be divided into four sub-bases, $\mathbf{L}^{s,t}$, $\mathbf{G}^{s,t}$, $\mathbf{K}^{s,t}$, and $\mathbf{J}^{s,t}$, according to the location of $\mathbf{z}_{0,0}$, where $\mathbf{z}_{0,0}$ is the top-left element of $\mathbf{Z}$, and the relative spatial locations of elements of each sub-base are the same as their relative locations in $\mathbf{Z}$. Therefore, $\mathbf{Z}^{s,t}\text{☆}\mathbf{X}$ can be decomposed into four items,

$$\begin{aligned}\mathbf{Z}^{s,t}\text{☆}\mathbf{X} = &(\mathbf{L}^{s,t}\text{☆}\mathbf{X})_{((0,0),(M-m,N-n))} \\ &+ (\mathbf{G}^{s,t}\text{☆}\mathbf{X})_{((0,t),(M-m,N-n+t))} \\ &+ (\mathbf{K}^{s,t}\text{☆}\mathbf{X})_{((s,0),(M-m+s,N-n))} \\ &+ (\mathbf{J}^{s,t}\text{☆}\mathbf{X})_{((s,t),(M-m+s,N-n+t))}.\end{aligned}$$

If $\langle \mathbf{z}_{s,t}, \mathbf{x}_{p,q} \rangle$'s for all possible (s,t)'s and (p,q)'s are calculated and stored, then $\mathbf{D}^{s,t}\hat{\mathbf{X}}$, where $\mathbf{D}^{s,t} \in \{\mathbf{L}^{s,t}, \mathbf{G}^{s,t}, \mathbf{K}^{s,t}, \mathbf{J}^{s,t}\}$, can be obtained by first retrieving stored calculations and then summing them. In this way, the multiple calculations on $\langle \mathbf{z}_{s,t}, \mathbf{x}_{p,q} \rangle$'s are eliminated. In our algorithm, $\langle \mathbf{z}_{s,t}, \mathbf{x}_{p,q} \rangle$'s for all possible (s,t)'s and (p,q)'s are called fundamental calculation, and stored as a series of fundamental matrices. In order to obtain $\mathbf{Z}^{0,0}\hat{\mathbf{X}}$ by the summation of all fundamental matrices, cyclic shifting is performed on the fundamental matrices. Each element of $\mathbf{Z}^{0,0}\hat{\mathbf{X}}$ equals to an element of the summation.

Another part of repeated calculations is the summation. It is clear that large amounts of summations calculated in $\mathbf{D}^{s,t}\hat{\mathbf{X}}$ are repeated in $\mathbf{D}^{s',t'}\hat{\mathbf{X}}$, where $s \neq s'$ or $t \neq t'$. For example, if $s' = s+1$ and $t' = t$, most summations of fundamental calculations involved in $\mathbf{J}^{s,t}\hat{\mathbf{X}}$ are repeated by those involved in $\mathbf{J}^{s',t'}\hat{\mathbf{X}}$. Because the fundamental calculations are stored as the fundamental matrices in our algorithm, the repeated summations are the repeated summations of fundamental matrices. To eliminate all these repeated summations, inspired by the integral image, we design the integral matrix $\mathbf{M}$ so as to calculate each $\mathbf{D}^{s,t}\hat{\mathbf{X}}$ in a constant time. Because the fundamental matrices are constructed according to the calculation of $\mathbf{Z}^{0,0}\hat{\mathbf{X}}$, $\mathbf{D}^{s,t}\hat{\mathbf{X}}$'s have to be cyclically shifted to align each other and then are summed to achieve $\mathbf{Z}^{s,t}\hat{\mathbf{X}}$ correctly, if $s \neq 0$ or $t \neq 0$.

Consequently, $\mathbf{Z}^{s,t}\hat{\mathbf{X}}$ is calculated efficiently. Note that $\mathbf{M}$ is a block matrix and the same as the integral image in principle. The difference between them is that $\mathbf{M}$ operates on block matrices, whereas the integral image on scalars.

According to the above analysis, we develop the algorithm, cyclic correlation with integral matrix (CCIM), to calculate $\mathcal{X} \diamond \mathcal{Z}$ efficiently. The formal steps of CCIM is presented in Algorithm 2 of [34] and supplementary material. The appendices also provide an example to illustrate how CCIM works exactly and the proof of its correctness.

3.3. Characteristics of ACSII and CCIM

The computational complexity of ACSII is $O(MND)$ because those of its three steps, constructing squared image, constructing squared integral image, and calculating autocorrelation, are $O(MND)$, $O(MN)$, and $O(MN)$, respectively. Because the computational complexities of the three steps of CCIM, construct fundamental matrices, construct integral matrix, and calculate correlation, are $O(mnMND)$, $O(mnMN)$, and $O(mnMN)$, respectively, its computational complexity is $O(mnMND)$, much lower than that with FFT. In order to experimentally verify the superior efficiency of CCIM over that of FFT, we calculate $\mathcal{X} \diamond \mathcal{Z}$ on a PC with Intel Core i7 CPU under the typical situation stated in the beginning of Sec. 3. It is not surprising to see that CCIM takes only 20 ms, while FFT 530 ms, to achieve $\mathcal{X} \diamond \mathcal{Z}$. In addition, ACSII takes 3 ms to calculate $\mathcal{X} \circ \mathcal{X}$. By fully exploiting the structures of bases and samples, ACSII and CCIM eliminate all redundant computations in calculating $\mathcal{X} \circ \mathcal{X}$ and $\mathcal{X} \diamond \mathcal{Z}$ in space domain. It is seen from the above analysis that the efficiency of CCIM is much higher than that of FFT in both theory and practice, because FFT cannot use the relations among the bases. Table 1 summarizes the characteristics of ACSII, CCIM, and FFT in our task.

4. Related Work

Our novel nBEKCF is essentially different from all existing CF trackers, because it is NOT based on KCF at all. Specifically, there are three key differences between nBEKCF and existing CF trackers. 1) The theoretical foundations are different. 2) The sampling strategies are different (non-cyclic vs. cyclic). 3) The efficient optimization procedures are different (non-FFT vs. FFT-related). The reason to cause these differences is the separation of the sets of real and dense training samples and cyclic bases in nBEKCF. In contrast, the both sets are identical and cyclic in KCF.

We do not model our problem with Tikhonov regularization [33] when deriving nBEKCF. Therefore, Problem (5) does not have to be transferred into the dual space by means of Representer Theorem [30] to solve. In addition, we also do not require the kernelized correlation filter f to lie in a bounded convex subset of a reproducing kernel Hilbert space (RKHS) defined by a positive definite kernel function $\kappa(\cdot,\cdot)$. $S_\kappa \equiv \{\kappa(\cdot, \mathbf{X}^{u,v})\}$ can be considered as a set of basis functions, and it will be orthogonal if f lies in a bounded convex subset of a RKHS defined by $\kappa(\cdot,\cdot)$.

In almost all existing CF trackers, the training sets are constructed through cyclically shifting a single image patch (*i.e.*, base sample) in a pixel-wise way, and the training samples are the cyclically shifted base samples themselves (e.g., in KCF, MKCF [32, 33], and SRDCF) or their clipped parts (e.g., in BACF). Therefore, there are more or less synthetic samples in their training sets. In our nBEKCF, however, the training samples are drawn from an image region in the traditional sliding window way, and there is no cyclic shifting of the image region. Therefore, all training samples are non-synthetic, *i.e.*, real, in nBEKCF, avoiding the boundary effect completely.

Unlike almost all existing CF trackers which resort to FFT for fast optimization, we develop two novel algorithms, ACSII and CCIM, to achieve efficient training and detection. CCIM is even more efficient than its FFT counterpart.

It is clear that nBEKCF is different from fdKCF* because their regression models and optimization schemes are both totally different. While nBEKCF applies ACSII and CCIM to accelerate its optimization, fdKCF* relies on the look-up table to efficiently optimize the regression model of KCF. On the other hand, the major computational costs of fdKCF* and nBEKCF are $O(M^2N^2(D+mn))$ and $O(MNDmn)$, respectively. Because $M(N) > m(n)$, the cost of fdKCF* is

	Object of Calculation	Computational Complexity	Elapsed Time ($M = N = 60$, $m = 15$, $n = 20$, $D = 41$)	Space or Frequency Domains	Exploiting Density of $\mathcal{X}$	Exploiting Cyclicity of $\mathcal{Z}$
ACSII	$\mathcal{X} \circ \mathcal{X}$	$O(MND)$	$3ms$	Space	Yes	-
CCIM	$\mathcal{X} \diamond \mathcal{Z}$	$O(mnMND)$	$20ms$	Space	Yes	Yes
FFT	$\mathcal{X} \diamond \mathcal{Z}$	$O(mnMND \log MN)$	$530ms$	Frequency	Yes	No

Table 1: Characteristics of ACSII, CCIM, and FFT in our task. '-' means the algorithm is not involved in the data.

much larger than that of nBEKCF. In practice, only on GPU can fdKCF* run in super-real-time even if low dimensional hand-crafted features are employed, because one of its major computational costs, $O(M^2N^2mn)$, is large and irrelative to the dimensionality of features. Whereas, nBEKCF can construct its kernel correlation matrices in a high efficiency, running in super-real-time on CPU if low dimensional hand-crafted features are employed. Note that the speed of constructing a kernel correlation matrix of nBEKCF on CPU is really close to that of fkKCF* on GPU, if the pre-trained network features of high dimensionality are exploited.

In comparison to Siamese trackers [2, 24], our nBEKCF, as a CF tracker, can be trained discriminatively, while they cannot, as pointed out by [3]. That is, nBEKCF utilizes both foreground and background to train its filter, while Siamese trackers only use the target patch when training their models. Consequently, in terms of trackers themselves, nBEKCF is more robust than Siamese trackers for various backgrounds.

5. Experiments

We implement two versions of our nBEKCF, nBEKCF-HC and nBEKCF-D by employing hand-crafted features and deep convolutional neural networks (CNNs) features, respectively. nBEKCF-HC is evaluated on three public benchmarks, OTB-2013, OTB-2015, and NfS, and compared against state-of-the-art trackers with hand-crafted features. nBEKCF-D is evaluated on other three public benchmarks, VOT2018, GOT10k, and TrackingNet, and compared to state-of-the-art trackers with CNNs features. We do not test nBEKCF-HC on the latter three ones because the hand-crafted features are too weak on them.

In our experiments, for a fair comparison, the same type of state-of-the-arts trackers are among the list of compared trackers. The same type of state-of-the-arts trackers means the top ones with the similar motivation, similar features, and similar scale adaptation scheme to the nBEKCF's. Therefore, ECO-HC [7] is selected from the trackers with hand-crafted features, and UPDT [4] from the trackers with ImageNet pre-trained features, which, like nBEKCF, both adopt the scale-pyramid scheme to decide proper scales of target. [4]

5.1. Implementation Details

In nBEKCF-HC, as in the top CF tracker with hand-crafted features, ECO-HC, HOG with 31-channels and color-name (CN) with 10-channels are employed, and both their cell sizes are 4×4. Gaussian kernel is applied with standard deviation 6 to construct the kernel correlation matrices. The learning rate γ is 0.008. The size of learning or search region is $M = N = 3\sqrt{mn}$ for the tradeoff between localization accuracy and speed. ACSII and CCIM are implemented in C++, and other parts in Matlab. The experiments are performed on a single Intel I7 CPU.

Regularization parameter $\lambda = 0.01$. Gaussian response $\mathbf{y}$ is identical to that in KCF with variance 0.01.

5.2. Ablation Study

Our nBEKCF is the first CF tracker which is not plagued by the boundary effect at all and is able to exploit non-linear kernels and high-dimensional hand-crafted features, while running in a fast speed with single CPU. Here, we investigate the impacts of avoiding the boundary effect thoroughly and choosing different kernels in nBEKCF. We conduct the ablation experiments with nBEKCF-HC on OTB2015 [41].

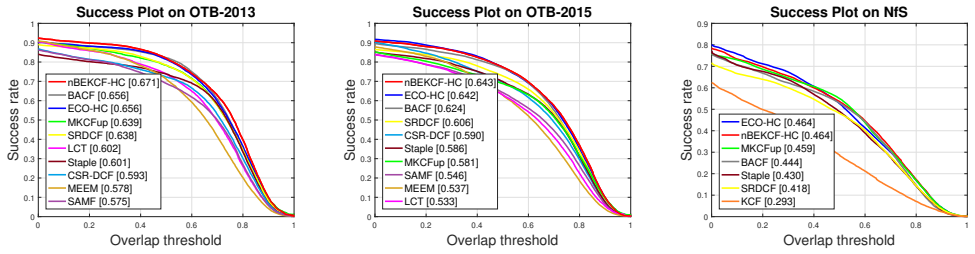
As a baseline tracker in our ablation study, SAMF [26] is the KCF with scale-pyramid scheme, exploits HOG and CN as its features, and suffers from the boundary effect. Because the motivation of BACF and SRDCF is also to address the boundary effect of KCF and they both employ scale-pyramid scheme to determine the scale of target, they are compared with SAMF and nBEKCF in this section. For the fair comparison, we implement SRDCF-HC and BACF-HC by adding color features (CN) into the public source codes of SRDCF and BACH, and the linear kernel is employed in nBKECF-HC because SRDCF and BACF are not able to employ any non-linear kernel.

Table 2 shows the results where AUC [41] is used to evaluate their accuracies. It is seen that the right four trackers outperform the base tracker SAMF-HC with large margins, and nBEKCF-HC-L performs better than SRDCF-HC and BACF-HC. Therefore, alleviating or avoiding the boundary effect will increase the accuracy, and avoiding the boundary effect completely in nBEKCF improves the accuracy more than just alleviating it in SRDCF-HC and BACF-HC.

On the other hand, the accuracy of nBEKCF-HC is higher than that of nBEKCF-HC-L, confirming that employing non-linear kernels will achieve a higher accuracy than employing the linear one, as shown by Zuo *et al*. [45].

Table 2 also shows that nBEKCF-HC-L and nBEKCF-HC not only outperform other trackers, but are able to run at much faster speeds. It is interesting to notice that nBEKCF-HC-L and nBEKCF-HC even run faster than SAMP which,

[4] Due to the space limitation, the implementation details of nBEKCF-D and its comparison to trackers with ImageNet pre-trained features are provided in [34] and supplementary material.



(a) (b) (c)

Figure 3: The mean success plots of our nBEKCF-HC and other state-of-the-art trackers with hand-crafted features on OTB-2013, OTB-2015, and NfS. The mean AUCs are reported in the legend. nBEKCF-HC achieves the best results.

	SAMF	SRDCF-HC	BACF-HC	nBEKCF-HC-L	nBEKCF-HC
AUC	0.572	0.616	0.620	0.632	0.643
mFPS	40	3	30	55	50

Table 2: AUCs and mean FPSs on OTB-2015. "-HC" means the indicated tracker exploits HOG and CN features, and "-L" indicates the tracker applies the linear kernel.

Tracker	nBEKCF-HC	BACF	ECO-HC	LCT	MKCFup
mFPS	50	35	40	30	150
Tracker	CSR-DCF	SRDCF	MEEM	Staple	SAMF
mFPS	15	10	15	70	30

Table 3: The mean FPSs of our nBEKCF-HC and other state-of-the-art trackers with hand-crafted features.

exactly like KCF, is plagued by the boundary effect.

5.3. Evaluation on OTB-2013, OTB-2015, and NfS

OTB-2013/2015 [40, 41]. OTB-2013 and OTB-2015 are the most popular benchmarks with various challenges for the evaluation of trackers, and contain 50 and 100 videos, respectively. On the OTB-2013 and OTB-2015 experiments, we compare our nBEKCF-HC against nine hand-crafted feature based state-of-the-art trackers, MEEM [42], SAMF [26], SRDCF [9], LCT [28], Staple [1], BACF [16], ECO-HC [7], CSR-DCF [27], and MKCFup [33]. All trackers are evaluated by success plot and AUC. Figs. 3a and 3b show that our nBEKCF-HC obtains the mean AUC of 67.1% and 64.3% on OTB-2013 and OTB-2015, respectively, outperforming all other state-of-the-art CF trackers with hand-crafted features.

In addition, table 3 [5] shows the mean FPS of the above trackers on OTB-2015. It can be seen that the running speed of our nBEKCF-HC is faster than all other trackers, except for MKCFup and Staple which take full advantage of the cyclic structure of samples. Although MKCFup and Staple are two fastest trackers, they suffer from the boundary effect, resulting in lower localization accuracies. It is worth noting that ECO-HC applies several tricks, such as sparse update and feature dimension reduction, to speed up. Whereas, nBEKCF-HC does not employ any similar tricks but is still faster than ECO-HC.

[5] We ran nBEKCF-HC, BACF, and ECO-HC on an identical PC for the fair comparison of fps.

NFS [15]. We evaluate nBEKCF-HC on the 240 FPS version of NfS benchmark which contains 100 challenging videos. On the NfS experiment, we compare nBEKCF-HC against five representative CF trackers, KCF, SRDCF, ECO-HC, BACF, and MKCFup. All trackers are evaluated by success plot and AUC. Fig. 3c shows that our nBEKCF-HC obtains the mean AUC of 0.464, outperforming all other CF trackers, except for ECO-HC. It is worth noting that ECO-HC uses sample clustering to improve its accuracy. Whereas, nBEKCF-HC does not apply any similar tricks but still achieves competitive accuracy with ECO-HC. In addition, it is seen that the success rate at overlap threshold 0.5, *i.e.*, the mean overlap precision, of nBEKCF-HC is slightly higher than that of ECO-HC.

It can be concluded from the above experiments that our nBEKCF-HC achieves the best trade-off between accuracy and speed among all hand-crafted feature based trackers.

6. Conclusions

The novel nBEKCF is presented in this paper. Without the boundary effect, being able to exploit non-linear kernels, and running at high fps, nBEKCF possesses all these characteristics that were hard to manage simultaneously before.

Historically, the pioneering work of VanderLugt [37] was the start of applying correlation filters to pattern recognition [23]. MOSSE and KCF were seminal works in applying correlation filters to visual tracking. Nevertheless, they all were plagued by the well-known defect - the boundary effect. After them, the three representative works, CFLB, SRDCF, and BACF, were proposed to address the defect. While the three alleviated the boundary effect, they lost the merits - high speed and the ability of applying non-linear kernels - of KCF. In this paper, our nBEKCF provides a totally different line to cast off the above dilemma thoroughly, not resorting to FFT. We believe that nBEKCF is an alternative in applying correlation filters to pattern recognition and visual tracking, and the performance of most modern CF trackers would be improved by means of nBEKCF. Even in the era of deep learning, nBEKCF may also provide a theoretical basis for the network design of trackers.

References

[1] Luca Bertinetto, Jack Valmadre, Stuart Golodetz, Ondrej Miksik, and Philip HS Torr. Staple: Complementary learners for real-time tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1401–1409, 2016.

[2] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *European conference on computer vision*, pages 850–865. Springer, 2016.

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[4] Goutam Bhat, Joakim Johnander, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Unveiling the power of deep tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 483–498, 2018.

[5] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2544–2550. IEEE, 2010.

[6] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. 2005.

[7] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Eco: Efficient convolution operators for tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6638–6646, 2017.

[8] Martin Danelljan, Gustav Häger, Fahad Shahbaz Khan, and Michael Felsberg. Discriminative scale space tracking. *IEEE transactions on pattern analysis and machine intelligence*, 39(8):1561–1575, 2016.

[9] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 4310–4318, 2015.

[10] Martin Danelljan, Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, pages 472–488. Springer, 2016.

[11] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost Van de Weijer. Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1090–1097, 2014.

[12] Philip J Davis. *Circulant matrices*. American Mathematical Soc., 2013.

[13] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *European conference on computer vision*, pages 702–715. Springer, 2012.

[14] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE transactions on pattern analysis and machine intelligence*, 37(3):583–596, 2014.

[15] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1125–1134, 2017.

[16] Hamed Kiani Galoogahi, Ashton Fagg, and Simon Lucey. Learning background-aware correlation filters for visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1135–1143, 2017.

[17] Hamed Kiani Galoogahi, Terence Sim, and Simon Lucey. Correlation filters with limited boundaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4630–4638, 2015.

[18] Matej Kristan, Aleš Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Čehovin Zajc, Tomas Vojir, Gustav Häger, Alan Lukežič, Abdelrahman Eldesokey, and Gustavo Fernandez. The visual object tracking vot2017 challenge results, 2017.

[19] Matej Kristan, Aleš Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Luka Čehovin Zajc, Tomas Vojir, Gustav Häger, Alan Lukežič, and Gustavo Fernandez. The visual object tracking vot2016 challenge results. Springer, Oct 2016.

[20] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pfugfelder, Luka Čehovin Zajc, Tomas Vojir, Goutam Bhat, Alan Lukezic, Abdelrahman Eldesokey, Gustavo Fernandez, and et al. The sixth visual object tracking vot2018 challenge results, 2018.

[21] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kamarainen, Luka Čehovin Zajc, Ondrej Drbohlav, Alan Lukezic, Amanda Berg, Abdelrahman Eldesokey, Jani Kapyla, and Gustavo Fernandez. The seventh visual object tracking vot2019 challenge results, 2019.

[22] Matej Kristan, Jiri Matas, Aleš Leonardis, Michael Felsberg, Luka Čehovin Zajc, Gustavo Fernandez, Tomas Vojir, Gustav Häger, Georg Nebehay, Roman Pflugfelder, Abhinav Gupta, Adel Bibi, and Alan Lukežič. The visual object tracking vot2015 challenge results. In *Visual Object Tracking Workshop 2015 at ICCV2015*, Dec 2015.

[23] BVK Vijaya Kumar, Abhijit Mahalanobis, and Richard D Juday. *Correlation pattern recognition*. Cambridge University Press, 2005.

[24] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8971–8980, 2018.

[25] Peixia Li, Dong Wang, Lijun Wang, and Huchuan Lu. Deep visual tracking: Review and experimental comparison. *Pattern Recognition*, 76:323–338, 2018.

[26] Yang Li and Jianke Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *European conference on computer vision*, pages 254–265. Springer, 2014.

[27] Alan Lukezic, Tomas Vojir, Luka Cehovin Zajc, Jiri Matas, and Matej Kristan. Discriminative correlation filter with channel and spatial reliability. In *Proceedings of the IEEE*

Conference on Computer Vision and Pattern Recognition, pages 6309–6318, 2017.

[28] Chao Ma, Xiaokang Yang, Chongyang Zhang, and Ming-Hsuan Yang. Long-term correlation tracking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5388–5396, 2015.

[29] Matthias Mueller, Neil Smith, and Bernard Ghanem. Context-aware correlation filter tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1396–1404, 2017.

[30] Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.

[31] Chong Sun, Dong Wang, Huchuan Lu, and Ming-Hsuan Yang. Learning spatial-aware regressions for visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8962–8970, 2018.

[32] Ming Tang and Jiayi Feng. Multi-kernel correlation filter for visual tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 3038–3046, 2015.

[33] Ming Tang, Bin Yu, Fan Zhang, and Jinqiao Wang. High-speed tracking with multi-kernel correlation filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4874–4883, 2018.

[34] Ming Tang, Linyu Zheng, Bin Yu, and Jinqiao Wang. Fast kernelized correlation filter without boundary effect. In *arXiv:1806.06406*, June 2018.

[35] Jack Valmadre, Luca Bertinetto, João Henriques, Andrea Vedaldi, and Philip HS Torr. End-to-end representation learning for correlation filter based tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2805–2813, 2017.

[36] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *IEEE Transactions on Image Processing*, 18(7):1512–1523, 2009.

[37] A VanderLugt. Signal detection by complex spatial filtering. *IEEE transactions on information theory*, 10(2):139–145, 1964.

[38] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[39] Jinqiao Wang, Linyu Zheng, Ming Tang, and Jiayi Feng. A comparison of correlation filter based trackers and struck trackers. *IEEE Transactions on Circuits and Systems for Video Technology*, 2019.

[40] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2411–2418, 2013.

[41] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[42] Jianming Zhang, Shugao Ma, and Stan Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In *European conference on computer vision*, pages 188–203. Springer, 2014.

[43] Tianzhu Zhang, Changsheng Xu, and Ming-Hsuan Yang. Learning multi-task correlation particle filters for visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):365–378, 2018.

[44] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Fast-deepkcf without boundary effect. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

[45] Wangmeng Zuo, Xiaohe Wu, Liang Lin, Lei Zhang, and Ming-Hsuan Yang. Learning support correlation filters for visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, 41(5):1158–1172, 2018.