Robust Visual Tracking via Speedup Multiple Kernel Ridge Regression

Cheng Qian,[a,b,c] Toby P. Breckon[b], Hui Li[a]
[a]Changzhou Institute of Technology, College of Computer and Information Engineering, Tongjiang South Road, No 299, Changzhou City, China, 213002
[b]Durham University, School of Engineering and Computing Sciences, UK, DH1 3LE
[c] Key Laboratory of Image and Video Understanding for Social Safety (Nanjing University of Science and Technology), Nanjing, China, 210094

Abstract. Most of the tracking methods try to build up feature spaces to represent the appearance of the target. However, limited by the complex structure of the distribution of features, the feature spaces constructed in a linear manner cannot characterize the nonlinear structure well. We propose an appearance model based on kernel ridge regression for visual tracking. Dense sampling is fulfilled around the target image patches to collect the training samples. In order to obtain a kernel space in favor of describing the target appearance, multiple kernel learning is introduced into the selection of kernels. Under the framework, instead of a single kernel, a linear combination of kernels is learned from the training samples to create a kernel space. Resorting to the circulant property of kernel matrix, a fast interpolate iterative algorithm is developed to seek coefficients that are assigned to these kernels so as to give an optimal combination. After the regression function is learned, all candidate image patches gathered are taken as the input of the function, and the candidate with the maximal response is regarded as the object image patch. Extensive experimental results demonstrate that the proposed method outperforms over other state-of-the-art tracking methods.

Keywords: visual tracking, kernel ridge regression, multiple kernel learning, fast interpolate iterative algorithm.

Address all correspondence to: Changzhou Institute of Technology, College of Computer and Information Engineering, Tongjiang South Road No. 299, Changzhou City, China, 213002; E-mail: qc_hz@163.com

1 Introduction

As one of the fundamental tasks in computer vision field, visual tracking obtains comprehensive applications such as automatic surveillance, robot navigation and human computer interaction. Although much work has been dedicated to it in recent years, the accurate tracking of a generic

target in complex environments remains a challenging problem due to the factors including illumination changes, camera motion, occlusion and background clutter.

A regular approach to tackling the problem is to establish an appearance model based on the observations of the target in the previous frames.[1] The appearance model assimilates the features characterizing the target and the background, and then the likelihood of an image patch as an object image region is estimated. A distance metric over the features often directly serves as the measurement of the likelihood. Hence, it is obvious that features extracted from the image patches play the important role in the determination of the object image patches. At present, the appearance models can grossly fall into two categories.[2,3] One is the generative appearance model, while the other is called the discriminative appearance model. The generative appearance model typically employs the features from the target occurring in the latest frames to predict the most likely appearance of the target in the future. For this type of models, the accuracy of predictions depends on the cohesion of features from the target. As for the discriminative appearance models, they focus on the distinction between the features from the target and from the background. The maximal margin between features is pursued by these models. It will have a great impact on the accuracy of the subsequent classification over the image patches. No matter which type of the appearance model the tracking methods use, the distribution of the features has the significant influence on the tracking results. Thus, this requires that an appropriate feature space be constructed available for the successive distance metric, further enhancing the representative ability of the appearance model.

Revolving around the construction of the feature space, a large number of tracking methods take advantage of all kinds of classic features to depict the target and the background. Considering the motion smoothness, it often holds that the features extracted from the target are

coherent in a short time. However, in company with tracking, the appearance of the target also evolves. As for the two object image patches sampled at the different frames over a significant time interval, more often than not, there exists a significant difference in the features corresponding to them. Consequently, this leads to a phenomenon that, in a short time interval, the cohesion of the features can be observed. Once the time span is stretched, it is evident that the distribution of the features gradually exhibits its own disorder.[4] Currently, numerous appearance models alleviate the problem through online learning. They replace the previous training samples with the incoming samples, and try to use latest samples to construct a local linear feature space. In fact, the distribution of features is often multi-modal in the feature space, but most of the appearance models deal with the features in a linear manner, which leads to that the distance metric in the feature space becomes invalid. This is one of the main reasons that the tracking failure happens.

It is difficult to depict the multi-modal structure of the feature distribution with a linear vector space. Compared with an explicit linear vector space, the nonlinear representation for the structure, such as manifold, kernel space and so on, achieves higher generality to the description of this likes of distribution. Li et al. characterize the target appearance with the covariance matrix descriptors, and learn a Riemannian manifold based on them in an incremental fashion.[5,6] Khan et al. assume that the image vectors representing the target lie on a Grassmann manifold.[7] However, the manifold learning is sensitive to outliers, which tends to introduce the errors into the description of the structure. Another approach to representing the multi-modal structure is to introduce the kernel space into the representation of the feature distribution. Recently, Henriques et al. accomplish tracking with the kernelized correlation filters.[8] The usage of the dense sampling strategy ensures sufficient training samples. Linear regression is implemented in the

kernel space of the features. This tracking method shows competitive performances in both the efficiency and the tracking accuracy. Danelljan et al. adjust the update theme for the kernelized correlation filters and incorporate a subspace into the appearance model, which gets promising results.[9] For both of the kernelized correlation filters-based tracking methods, the selection of the kernels has great influence on the performance. Faced with different scenarios, these methods need to empirically construct different kernels.

To address the problem of choosing kernels, we substitute multiple kernels for single kernel under the kernel ridge regression framework, and assign these kernels with optimal coefficients. In this way, the distribution structure is expected to be captured in a proper kernel space. The contribution of our work can be summarized in three aspects: (1) We give a proof that a linear combination of circulant Gram matrices still satisfies circulant property. This lemma lays the foundation for the subsequent speedup multiple kernel ridge regression. (2) A group of coefficients, which measure the contributions of all the kernels, are learned under the multiple kernel learning framework. A fast interpolate iterative algorithm is created for accelerating the learning. (3) A tracking method based on multiple kernel ridge regression is proposed in this paper. It strikes a balance between the efficiency of learning and the accuracy of tracking.

The remainder of the paper is organized as followings. A brief review of the related works is given in section 2. The detailed description of our method is presented in section 3. The experimental results are offered and discussed in section 4. Finally, the paper is concluded in section 5.

2 Related Work

There is numerous literature reporting the tracking results based on different features. Most of these methods are confronted with the problem how to refine the features. A promising strategy

is to find a feature space, regardless of either low-dimensional subspace or high-dimensional kernel space, where all features are embedded into. It is expected that the original feature distribution can be transformed into a distribution that is in favor of discrimination or high-quality matching.

Among the tracking methods based on the embedding space, the subspace leaning-based tracking methods seek a compact representation for the raw pixel values. This type of the appearance models try to accommodate as many observations of the target appearance as possible. Black et al. propose an eigenspace representation for the target appearance to address the appearance variation problem resulting from the viewpoint changes.[10] Ross et al. devise a low-dimensional subspace over the image patches resorting to incremental principal component analysis.[11] The reconstruction error of each image patch is taken as the distance metric. Hu et al. construct a tensor subspace as the description of the target appearance, where the mean and the eigenbasis of tensors can be updated online.[12] Nevertheless, the update theme for the holistic template makes these methods less effective in coping with occlusions.

Recently, the applications of sparse representation in visual tracking have gradually attracted significant attention. For the tracking methods, all image patches can be considered as a sparse linear combination of a set of templates. Consequently, these templates constitute an over-complete dictionary, and the corresponding feature space is thus spanned by them. Mei et al. utilize the object image patches and trivial templates to construct a dictionary.[13,14] The reconstruction error of a candidate in the target template subspace is taken as the distance metric. Based on L1 tracker, Bao et al. make use of the accelerated proximal gradient approach to reduce the computational load for L1 minimization.[15] Instead of the object image patches as the basis vectors of the dictionary, Wang et al. integrate the PCA-based representation into the

construction of the dictionary,[16] which facilitates the online update of the dictionary. Jia et al. develop a structural local sparse appearance model that uses the alignment-pooling technique over the local patches to search the target.[17] Zhong et al. combine a sparsity-based classifier over holistic image patches and sparsity-based generative model over local image patches,[18] and this method is able to achieve the tracking in the case of occlusion. For the sparse representation-based tracking methods, the sparsity constraint enforced on the coefficients is helpful for the selection of the basis vectors of the dictionary. In this manner, the subspace spanned by the basis vectors selected effectively represents the variations that both occur in the target and the background. It is obvious that the new feature space is only relevant to the construction of the dictionary.

Considering the multi-modal structure of the distribution, the kernel methods bring the nonlinearity of the structure into the decision of the object image patches via mapping the features into the high-dimensional kernel space. This makes it feasible to handle the kernelized features in a linear fashion. Avidan uses support vector machine (SVM) under the optical flow framework to distinguish the object image patch from the background.[19] Subsequently, Hare et al. introduce a structured output SVM into the design of the appearance model.[20] Through setting up a joint kernel map for the features and the position translations, the possible position of the target in the new frame is predicted under the large-margin framework. Gao et al. exploit the graph structure of the features in the Hilbert space to devise a discriminative tracker.[21] Wang et al. extend sparse representation to kernel space, and evaluate the similarity between a candidate and the template with residual error.[22] Yang et al. build up a group of SVMs over different feature, and combine them under the boosting framework so as to get a strong discriminative tracker.[23] For these methods, the forms of kernels, including the kernel types and the parameters for the

kernels, rely on whether the kernel space established is suitable for representing the structure of the distribution.

As an effective solution to the selection of kernels, the multiple kernel learning algorithms pursue an optimal combination of kernels for classification tasks.[24,25] According to it, we attempt to exploit multiple kernel learning to server the purpose of building up a kernel space. The key problem concerning the assignment of coefficients for the base kernels is addressed through exerting the norm constraints on the coefficients.[26-28] With L1 norm constraint, the sparse coefficients can be obtained, but the correlation among the base kernels is discarded, which degrades the generalization performance. In contrast to L1 norm constraint, L2 norm constraint takes the contribution of every kernel into account,[29] and it is proven to resist against noises occurring in the data. Motivated by the characteristic of L2 norm constraint, we exploit it to seek a combination of kernel functions that creates a feature space most suitable for the identification of the object image patch.

3 Visual Tracking based on Multiple Kernel Ridge Regression

3.1 Preliminaries

Given a vector $\mathbf{x} \in \mathbb{R}^{n\times 1}$ ($\mathbf{x} = [x_1, x_2 \cdots, x_n]^T$), the corresponding circulant matrix $\mathbf{X}$ generated from $\mathbf{x}$ is expressed as following.

$$\mathbf{X} = \mathrm{C}(\mathbf{x}) = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_n \\ x_n & x_1 & x_2 & \cdots & x_{n-1} \\ x_{n-1} & x_n & x_1 & \cdots & x_{n-2} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_2 & x_3 & x_4 & \cdots & x_1 \end{bmatrix} \tag{1}$$

$C(\cdot)$ denotes a cyclic shift operator. It can be seen that each row of $\mathbf{X}$ is obtained through shifting the vector $\mathbf{x}$. Furthermore, the shift of the vector $\mathbf{x}$ can be illustrated with the product between $\mathbf{x}$ and a permutation matrix $\mathbf{P}$.

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{2}$$

Subsequently the circulant matrix $\mathbf{X}$ can be denoted with respect to $\mathbf{x}$ and $\mathbf{P}$.

$$\mathbf{X} = [\mathbf{x}, \mathbf{P}\mathbf{x}, \mathbf{P}^2\mathbf{x}, \cdots, \mathbf{P}^{n-1}\mathbf{x}]^T \tag{3}$$

It is noted that each entry $\mathbf{X}_{i,j}$ of the circulant matrix can be derived in the form of $\mathbf{X}_{i,j} = (\mathbf{P}^{(i-1)}\mathbf{x})_j^T = \mathbf{X}_{1,\mathrm{mod}(n+j-i+1,n)}$ (Here, $\mathrm{mod}(\cdot,\cdot)$ denotes the modular operator). As a result, there exits an attractive characteristic for the circulant matrix. Using Discrete Fourier Transform (DFT), the circulant matrix can be diagonalized as follows.

$$\mathbf{X} = (\mathbf{F}^*)^T \cdot \mathrm{diag}(\hat{\mathbf{x}}) \cdot \mathbf{F} \tag{4}$$

Where $\mathbf{F}$ denotes the DFT coefficient matrix that is a constant matrix, and $\mathbf{F}^*$ is the complex-conjugate of $\mathbf{F}$. $\mathrm{diag}(\hat{\mathbf{x}})$ is a diagonal matrix with the vector $\hat{\mathbf{x}}$ aligned along the diagonal line. $\hat{\mathbf{x}}$ is the DFT result corresponding to the vector $\mathbf{x}$. The outstanding speed achieved by the kernelized correlation filters-based tracking method[8] is largely attributed to this transformation in Eq. (4).

3.2 Kernel ridge regression with single kernel

For visual tracking, in general, there exists a unique object image region in each frame. Other than the object image region, the rest of the image patches that are sampled at arbitrary locations are all taken as the background image patches. In fact, in the proximity of the object image region, the image patches are immersed with less background pixels than the image patches far

from the target. Babenko et al. state that the hard labels for these image patches tend to augment the tracking drift due to the errors of target locations.[30] Hence it is reasonable to assign an image patch with a real value label instead of a binary label. Kernel ridge regression is capable of meeting the requirement of the assignment of real values labels.

When the features are mapped into a kernel space, considering the linear structure of the feature distribution in this space, a regression function describing the structure can be expressed as.

$$y = \boldsymbol{\beta}\varphi(\mathbf{x}) \tag{5}$$

Where $\varphi(\cdot)$ defines a function that accomplishes the mapping of a feature vector $\mathbf{x}$. For kernel learning-based methods, in practice, the feature mapping is implicitly defined by the kernel function as an inner product $k(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$. y is a real value, and can be also deemed as a soft label. $\boldsymbol{\beta}$ is a weight vector that needs to be learned from the training samples. For the kernel ridge regression, $\boldsymbol{\beta}$ can be derived via minimizing an objective function.

$$\boldsymbol{\beta} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{N} \|y_i - \boldsymbol{\beta}\varphi(\mathbf{x}_i)\|^2 + \lambda \|\boldsymbol{\beta}\|^2 \tag{6}$$

Where $\lambda > 0$ is a tradeoff parameter. With N training samples and their corresponding soft labels $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)\}$, the solution to the above objective function is get as follows.

$$\boldsymbol{\beta} = \boldsymbol{\phi}(\mathbf{x})^T (\boldsymbol{\phi}(\mathbf{x})\boldsymbol{\phi}(\mathbf{x})^T + \lambda \mathbf{I})^{-1} \mathbf{y} \tag{7}$$

Where $\boldsymbol{\phi}(\mathbf{x}) = [\varphi(\mathbf{x}_1), \varphi(\mathbf{x}_2), \cdots, \varphi(\mathbf{x}_N)]^T$. $\mathbf{y}$ is a real value vector, $\mathbf{y} = [y_1, y_2, \cdots, y_N]^T$. Let $\boldsymbol{\alpha} = (\boldsymbol{\phi}(\mathbf{x})\boldsymbol{\phi}(\mathbf{x})^T + \lambda \mathbf{I})^{-1}\mathbf{y}$, and then the solution can be formulated as follows.

$$\boldsymbol{\beta} = \boldsymbol{\phi}(\mathbf{x})^T \boldsymbol{\alpha} = \sum_{i=1}^{N} \alpha_i \varphi(\mathbf{x}_i) \tag{8}$$

From Eq. (8), it is noted that, with the fixed mapping function $\varphi(\cdot)$, $\boldsymbol{\beta}$ only associates to $\boldsymbol{\alpha}$. Once that $\boldsymbol{\alpha}$ is derived, $\boldsymbol{\beta}$ can be determined directly. For the minimization problem in Eq. (6), a dual form with respect to $\boldsymbol{\alpha}$ is explored to give the solution.

$$\boldsymbol{\alpha}^* = \arg\max_{\boldsymbol{\alpha}}(-\lambda\,\boldsymbol{\alpha}^T\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{K}\boldsymbol{\alpha} + 2\boldsymbol{\alpha}^T\mathbf{y}) \tag{9}$$

In the case of single kernel, $\mathbf{K}$ is a Gram matrix ($\mathbf{K} = \boldsymbol{\phi}(\mathbf{x})\boldsymbol{\phi}(\mathbf{x})^T$). The solution to this optimization problem in Eq. (9) is expressed as.

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda\mathbf{I})^{-1}\mathbf{y} \tag{10}$$

It can be seen that the calculation of $\boldsymbol{\alpha}$ is involved in the computation of the inverse matrix, and it inhibits the application of kernel ridge regression in real-time visual tracking. However, under the assumption that the training samples $[\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N]$ are generated by the cyclic shifts of a base vector $\mathbf{x}$, namely $\mathbf{x}_i = \mathbf{P}^{(i-1)}\mathbf{x}$, when the kernel function $k(\cdot,\cdot)$ satisfies the property $k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{P}^p\mathbf{x}_i, \mathbf{P}^p\mathbf{x}_j)$ for all entries of the Gram matrix $\mathbf{K}$, $\mathbf{K}$ is actually a circulant matrix. In this case, Henriques et al. give an excellent approach to accelerating its calculation remarkably.[8] Resorting to Eq. (4), the DFT vector $\hat{\boldsymbol{\alpha}}$ corresponding to $\boldsymbol{\alpha}$ can be acquired in the Fourier form.

$$\hat{\boldsymbol{\alpha}} = \hat{\mathbf{y}}/(\hat{\mathbf{k}} + \lambda\mathbf{1}) \tag{11}$$

Where $\hat{\mathbf{y}}$ denotes DFT vector of $\mathbf{y}$, and $\hat{\mathbf{k}}$ is the DFT vector of the first row from the Gram matrix $\mathbf{K}$. $\mathbf{1}$ denotes a vector $[1,1,\cdots,1]^T$. $\hat{\boldsymbol{\alpha}}$ is the DFT vector that can yield $\boldsymbol{\alpha}$ through inverse DFT.

3.3 Fast kernel ridge regression based on multiple kernel learning

Since that kernel ridge regression is upset by the issue in the selection of a most suitable kernel, we bring multiple kernel learning into the solution to the problem. Under the multiple kernel

learning framework, the kernel space is created by a linear combination of kernels. Then the Gram matrix $\mathbf{K}$ is made up of a set of Gram matrices corresponding to different kernel functions.

$$\mathbf{K} = \sum_{i=1}^{m} \theta_i \mathbf{K}_i \tag{12}$$

Where θ_i is the i-th coefficient ($i = 1,2,\cdots,m$). $\mathbf{K}_i$ is i-th Gram matrix represented by a specific type of kernel function. As a result, the regression function in Eq. (8) can be expressed in terms of multiple kernels as follows.

$$y = \sum_{i}^{N} \alpha_i \sum_{l=1}^{m} \theta_l \, \varphi_l(\mathbf{x}_i) \cdot \varphi_l(\mathbf{x}) \tag{13}$$

Substitute Eq. (12) for $\mathbf{K}$ in Eq. (9), and the objective function can be rewritten as follows.

$$\arg\min_{\boldsymbol{\Theta}} \arg\max_{\boldsymbol{\alpha}} -\lambda\, \boldsymbol{\alpha}^T \boldsymbol{\alpha} - \sum_{i=1}^{m} \theta_i \boldsymbol{\alpha}^T \mathbf{K}_i\, \boldsymbol{\alpha} + 2\boldsymbol{\alpha}^T \mathbf{y} \tag{14}$$

In that the coefficient vector $\boldsymbol{\Theta} = [\theta_1, \theta_2, \cdots, \theta_m]^T$ is not given in advance, the optimization problem in Eq. (14) not only depends on $\boldsymbol{\alpha}$ but also is related to $\boldsymbol{\Theta}$. With the L2-norm constraint on $\boldsymbol{\Theta}$,[29] the coefficient vector $\boldsymbol{\Theta}$ can be restricted to a part of the sphere centered around a positive mean.

$$\{\boldsymbol{\Theta} | \boldsymbol{\Theta} \geq 0, \|\boldsymbol{\Theta} - \boldsymbol{\Theta}_0\|^2 \leq \gamma^2\} \tag{15}$$

Where $\boldsymbol{\Theta}_0$ is an initial vector of $\boldsymbol{\Theta}$, and $\gamma > 0$ defines the radius of the sphere. The optimization problem turns out to a convex optimization problem with respect to $\boldsymbol{\alpha}$ and $\boldsymbol{\Theta}$. Since the close-form solution for Eq. (14) does not exist, an interpolate iterative algorithm is developed to compute $\boldsymbol{\alpha}$ and $\boldsymbol{\Theta}$. This algorithm is outlined in Algorithm 1.

Algorithm 1: Interpolate iterative algorithm

1: **Input**: A group of Gram matrices $\{\mathbf{K}_1, \mathbf{K}_2, \cdots, \mathbf{K}_m\}$, and the coefficient vector $\boldsymbol{\Theta}$ with each coefficient $\theta_i = 1/m$. Given $\mathbf{y}$, the initial value for $\boldsymbol{\alpha}$ is $(\sum_{i=1}^{m} \theta_i \mathbf{K}_i + \lambda \mathbf{I})^{-1} \mathbf{y}$.

2: **do**

3: $\boldsymbol{\alpha}' = \boldsymbol{\alpha}$.

4: $\mathbf{v} = [\boldsymbol{\alpha}^T \mathbf{K}_1 \boldsymbol{\alpha}, \boldsymbol{\alpha}^T \mathbf{K}_2 \boldsymbol{\alpha}, \cdots, \boldsymbol{\alpha}^T \mathbf{K}_m \boldsymbol{\alpha}]^T$.

5: $\boldsymbol{\Theta} = \boldsymbol{\Theta}_0 + \gamma \cdot \mathbf{v}/\|\mathbf{v}\|$.

6: $\boldsymbol{\alpha} = \eta\boldsymbol{\alpha}' + (1-\eta)(\sum_{i=1}^{m}\theta_i\mathbf{K}_i + \lambda\mathbf{I})^{-1}\mathbf{y}$. ($\eta \in (0,1)$ is an interpolated parameter)

7: **while** $\|\boldsymbol{\alpha} - \boldsymbol{\alpha}'\| > \varepsilon$ ($\varepsilon > 0$ is a threshold)

8: **Output**: $\boldsymbol{\alpha}$ and the coefficient vector $\boldsymbol{\Theta}$.

In order to facilitate the real-time application of this algorithm in multiple kernel learning, we also seek an efficient way to get the solutions to $\boldsymbol{\alpha}$ and $\boldsymbol{\Theta}$. Assuming that the selection of the kernels only ranges over the Gram matrices with circulant structures, a fast solution can be derived. At first, aiming at multiple kernel learning, a proof is given that the linear combination of Gram matrices still satisfies the circulant property.

Theorem 1. Given a base vectors $\mathbf{x} \in \mathbb{R}^{M\times 1}$, a group of Gram matrices $\{\mathbf{K}_1, \mathbf{K}_2, \cdots, \mathbf{K}_Q\}$ are generated from a set of kernel functions $\{k_1(\cdot,\cdot), k_2(\cdot,\cdot), \cdots, k_Q(\cdot,\cdot)\}$. An entry of l-th Gram matrix is $\mathbf{K}_{i,j}^{l} = k_l(\mathbf{P}^{i-1}\mathbf{x}, \mathbf{P}^{j-1}\mathbf{x})$ ($l = 1,2,\cdots,Q$). If the set of kernel functions all satisfy $k_l(\mathbf{x},\mathbf{x}') = k_l(\mathbf{P}^p\mathbf{x}, \mathbf{P}^p\mathbf{x}')$ (Here, p is an integer), any linear combination of the Gram matrices $\sum_{q=1}^{Q}\theta_q\mathbf{K}_q$ is still a circulant matrix (θ_q denotes q-th coefficient).

Proof: In that each entry of the Gram matrix satisfies $\mathbf{K}_{i,j}^{l} = k_l(\mathbf{P}^{i-1}\mathbf{x}, \mathbf{P}^{j-1}\mathbf{x}) = k_l\left(\mathbf{x}, \mathbf{P}^{\mathrm{mod}(M+(j-i),M)}\mathbf{x}'\right) = \mathbf{K}_{1,\mathrm{mod}(M+j-i+1,M)}^{l}$, $\mathbf{K}_{i,j}^{l}$ is a circulant matrix. Let $\mathbf{K} = \sum_{q=1}^{Q}\theta_q\mathbf{K}_q$. An entry of $\mathbf{K}$ can be expressed with the entries from the Gram matrices, $\mathbf{K}_{i,j} = \sum_{q=1}^{Q}\theta_q\mathbf{K}_{i,j}^{q} = \sum_{q=1}^{Q}\theta_q k_q(\mathbf{P}^{i-1}\mathbf{x}, \mathbf{P}^{j-1}\mathbf{x})$.

With the property for the kernel function, $k(\mathbf{x},\mathbf{x}') = k(\mathbf{P}^p\mathbf{x}, \mathbf{P}^p\mathbf{x}')$, then the entry of $\mathbf{K}$ is equivalent to $\mathbf{K}_{i,j} = \sum_{q=1}^{Q}\theta_q k_q(\mathbf{P}^{(-i+1)}\mathbf{P}^{(i-1)}\mathbf{x}, \mathbf{P}^{(-i+1)}\mathbf{P}^{(j-1)}\mathbf{x})$. Considering the circulant property of $\mathbf{P}$, $\mathbf{K}_{i,j}$ also satisfies $\mathbf{K}_{i,j} = \sum_{q=1}^{Q}\theta_q k_q(\mathbf{x}, \mathbf{P}^{\mathrm{mod}(M+(j-i),M)}\mathbf{x}) = \mathbf{K}_{1,\mathrm{mod}(M+j-i+1,M)}$. It can be seen that $\mathbf{K}_{i,j}$ is only relevant to $\mathrm{mod}(M + j - i,\ M)$. This meets the criteria of the circulant matrix that, if all entries of the matrix is only related to $\mathrm{mod}(M + j - i,\ M)$, then the matrix is a circulant matrix.

After the conclusion of theorem 1 is reached, we are able to improve the efficiency of interpolate iterative algorithm by means of the diagonalization trick for the circulant matrix in Eq. (4). Suppose that all of the Gram matrices are circulant matrices, in Algorithm 1, the calculation of $\mathbf{v}$ can be rewritten as.

$$\mathbf{v} = \left[\boldsymbol{\alpha}^T(\mathbf{F}^*)^T \cdot \mathrm{diag}(\hat{\mathbf{k}}_1) \cdot \mathbf{F}\boldsymbol{\alpha}, \boldsymbol{\alpha}^T(\mathbf{F}^*)^T \cdot \mathrm{diag}(\hat{\mathbf{k}}_2) \cdot \mathbf{F}\boldsymbol{\alpha}, \cdots, \boldsymbol{\alpha}^T(\mathbf{F}^*)^T \cdot \mathrm{diag}(\hat{\mathbf{k}}_m) \cdot \mathbf{F}\boldsymbol{\alpha}\right]^T \quad (16)$$

Where $\hat{\mathbf{k}}_i$ is the DFT of the first row of the Gram matrix $\mathbf{K}_i$. As the DFT result of $\boldsymbol{\alpha}$, $\hat{\boldsymbol{\alpha}} = \mathbf{F}\boldsymbol{\alpha}$. Additionally, the product between a vector and a diagonal matrix is an element-wise product, then the calculation of $\mathbf{v}$ can be simplified as following.

$$\mathbf{v} = \left[\hat{\boldsymbol{\alpha}} \odot \hat{\mathbf{k}}_1 \cdot \hat{\boldsymbol{\alpha}}^T, \hat{\boldsymbol{\alpha}} \odot \hat{\mathbf{k}}_2 \cdot \hat{\boldsymbol{\alpha}}^T, \cdots, \hat{\boldsymbol{\alpha}} \odot \hat{\mathbf{k}}_1 \cdot \hat{\boldsymbol{\alpha}}^T\right]^T \quad (17)$$

For the time-consuming step in Algorithm 1, we update $\boldsymbol{\alpha}$ in the Fourier domain.

$$\mathbf{F}\boldsymbol{\alpha} = \eta\mathbf{F}\boldsymbol{\alpha} + (1-\eta)\mathbf{F}(\textstyle\sum_{i=1}^{m} \theta_i(\mathbf{F}^*)^T \cdot \mathrm{diag}(\hat{\mathbf{k}}_i) \cdot \mathbf{F} + \lambda\mathbf{I})^{-1}\mathbf{y} \quad (18)$$

Subsequently, the Fourier form of $\boldsymbol{\alpha}$ is obtained.

$$\hat{\boldsymbol{\alpha}} = \eta\hat{\boldsymbol{\alpha}} + (1-\eta)\,\hat{\mathbf{y}}/(\textstyle\sum_{i=1}^{m} \theta_i\,\hat{\mathbf{k}}_i + \lambda \cdot \mathbf{1}) \quad (19)$$

Based on the DFT result, $\boldsymbol{\alpha}$ can be derived through inverse DFT. For the task of visual tracking, it is straightforward to fulfill the kernel ridge regression with $\hat{\boldsymbol{\alpha}}$. Therefore, here, the DFT result is preserved for the successive calculation of confidence of a candidate. The fast interpolate iterative algorithm is summarized in Algorithm 2.

Algorithm 2: Fast interpolate iterative algorithm

1: **Input**: A group of Gram matrices $\{\mathbf{K}_1, \mathbf{K}_2, \cdots, \mathbf{K}_m\}$, and the coefficient vector $\boldsymbol{\Theta}$ with each coefficient $\theta_i = 1/m$. Given $\mathbf{y}$, the Fourier form of $\boldsymbol{\alpha}$ is $\hat{\boldsymbol{\alpha}} = \hat{\mathbf{y}}/(\sum_{i=1}^{m} \theta_i\,\hat{\mathbf{k}}_i + \lambda \cdot \mathbf{1})$.

2: **do**

3: $\hat{\boldsymbol{\alpha}}' = \hat{\boldsymbol{\alpha}}$.

3: $\mathbf{v} = \left[\hat{\boldsymbol{\alpha}} \odot \hat{\mathbf{k}}_1 \cdot \hat{\boldsymbol{\alpha}}^T, \hat{\boldsymbol{\alpha}} \odot \hat{\mathbf{k}}_2 \cdot \hat{\boldsymbol{\alpha}}^T, \cdots, \hat{\boldsymbol{\alpha}} \odot \hat{\mathbf{k}}_1 \cdot \hat{\boldsymbol{\alpha}}^T\right]^T$.

4: $\boldsymbol{\Theta} = \boldsymbol{\Theta}_0 + \gamma \cdot \mathbf{v}/\|\mathbf{v}\|$.

5: $\hat{\boldsymbol{\alpha}} = \eta\hat{\boldsymbol{\alpha}}' + (1-\eta)\,\hat{\mathbf{y}}/(\sum_{i=1}^{m} \theta_i\,\hat{\mathbf{k}}_i + \lambda \cdot \mathbf{1})$.

6: **while** $\|\hat{\boldsymbol{\alpha}} - \hat{\boldsymbol{\alpha}}'\| > \varepsilon$

7: **Output**: $\hat{\boldsymbol{\alpha}}$ and $\boldsymbol{\Theta}$.

3.4 Detection of the target via kernel ridge regression

As for the establishment of the regression function, it is learned from a group of training samples. This requires that there be s features gathered as the training samples. In addition, the tracking task is regarded as the detection of the target based on the features. It can be achieved by evaluating the confidence of an image patch as the object image patch. This also requires that a large number of image patches be sampled as the candidates. In our tracking method, the strategy of dense sampling is adopted to collect the image patches.

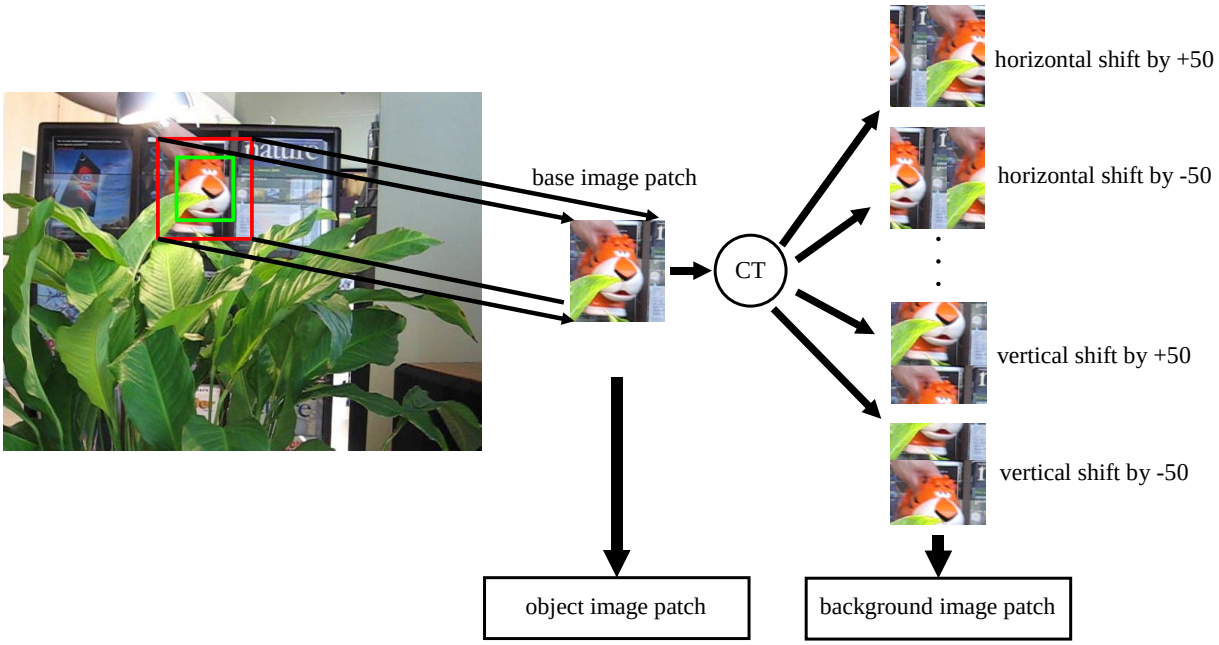


Fig. 1 The collection of the training samples by the cyclic shifts of the base image

The features are extracted from the image patches, and they act as the training samples that are characteristic of the image patches. The object image patches and background image patches are all generated by the cyclic shift of a base image patch. Resorting to the translation, both the object image patch and background image patches are collected. The collection is illustrated in Fig.1. "CT" in Fig.1 is the abbreviation of "cyclic shift". $\mathbf{x}(u, v)$ is used to denote a feature extracted from an image patch (u and v denote the horizontal shift and vertical shift of the

current image patch from the center of the base image patch respectively). When both u and v are equal to 0, $\mathbf{x}(0,0)$ represents the feature that is from the base image patch, and it is also the feature extracted from the target. When either u or v is non-zero, the feature $\mathbf{x}(u,v)$ characterizes the mixture of the object region and the background region.

When learning the kernel regression function, an amount of training samples and soft labels are in demand. The training samples originate from the HOG features that are extracted from the image patches.[31] As for the soft labels, a Gaussian function $f(u,v)$ with respect to the shifts u and v serves as the label assignment function that generates the soft labels for the training samples. In fact, u and v reflects how far away the target is from the center of the image patch. The label assignment function $f(u,v)$ follows the intuition that the further the image patch is away from the center, the less likely the image patch is to be the object image patch.

The candidates are also sampled via the cyclic shifts of a base image patch that is centered at the location of the target in the previous frame. As a result, the responses of the decision function to all the candidate image patches should be checked.

$$\mathbf{y} = \mathbf{K}^T \boldsymbol{\alpha} \tag{20}$$

Considering that $\mathbf{K} = \sum_{j=1}^{m} \theta_j \mathbf{K}_j$ is a linear combination of the circulant matrices with respect to a feature $\mathbf{z}(u,v)$ with an unknown soft label, it can be computed in Fourier domain.

$$\hat{\mathbf{y}} = (\sum_{l=l}^{m} \theta_l \hat{\mathbf{k}}_l)^* \odot \hat{\boldsymbol{\alpha}} \tag{21}$$

The inverse DFT of $\hat{\mathbf{y}}$ gives the responses of all candidates. The response reflects the confidence of a candidate as the object image patch.

$$\mathrm{conf}(\mathbf{z}_i(u,v)) = y_i \tag{22}$$

Finally, the candidate with the maximal response is taken as the object image patch. As Eq. (23) shows, the translation can be induced from the shifts of the object image patch.

$$(u, v) = \arg\max_{u,v}(\,\mathrm{conf}(\mathbf{z}(u, v))) \tag{23}$$

In the method, every target image patch is denoted with a rectangular bounding box. With the horizontal shift u and the vertical shift v, the location of the target bounding box at the current frame can be estimated based on the center of the bounding box at the previous frame. Suppose that the central coordinate of the target bounding box at the previous frame is (LX_{t-1},LY_{t-1}) (LX_{t-1} and LY_{t-1} denote the horizontal coordinate and vertical coordinate of the target bounding box at the $(t-1)$-*th* frame respective), with the translation, the central coordinate of the target bounding box at the current frame is ($LX_t = LX_{t-1} + u,\ LY_t = LY_{t-1} + v$).

After that the location of the object image patch is identified, the training samples are collected again in order to update the detector online. The parameters involved in the regression function in Eq. (21) are updated based on the HOG descriptors extracted from the cyclic versions of the image patch. The updating for the regression function is fulfilled as Algorithm 2 illustrates. It lays the foundation for the determination of the object image patch in the next frame.

4 Experiment and Discussion

In order to evaluate the performance of our tracking method, the experiments are conducted on the publicly available video sequences covering the scenarios such as cluttered background, fast motion, illumination variation, partial occlusion and viewpoint change. Our method is evaluated compared with other state-of-the-art tracking methods including CNT,[9] KCF,[8] LOT,[32] MEEM,[33] PMT,[34] SCM,[18] STC,[35] Struck[20] and TLD[36] over 30 video clips.

4.1 Experiment setup

Instead of the intensities of the raw pixels, we choose the histogram of oriented gradients (HOG) descriptors as the features representing the image patches. In the view of the tradeoff between the

tracking accuracy and the efficiency,[37] we adopt the same setting of the cell size of 4×4 pixels as KCF.[8] Since that the HOG descriptor closely associates to the spatial layout of the pixels, when the cyclic shift happens to an image patch, the cyclic shift also occurs in the HOG descriptor. Hence the theorem of speedup multiple kernel ridge regression still holds for HOG. HOG feature is adequate to the low-level feature in our method. Our goal lies in the validation of the application of multiple kernel learning in visual tracking. Without loss of generality, eleven kernels are combined to create a kernel space. Among them, there is one linear kernel, five polynomial kernels and five Gaussian kernels, each of which satisfies the circulant property. For the polynomial kernel $k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + a)^b$, the parameter a is set to 0.2, 0.7, 1.2 , 1.7, 2.2 for the five kernels respectively, and b is set to 7, 8, 9, 10, 11. For the Gaussian kernel $k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\|\mathbf{x}_i - \mathbf{x}_j\|^2/\sigma^2)$, the parameter σ is set to 0.1, 0.4, 0.7, 1, 1.3 for the five Gaussian kernels. The search region is often set to be three times larger than the size of the object image patch. The tradeoff parameter λ is fixed to 0.0001 for all sequences. The distribution of the soft label y satisfies the two-dimensional Gaussian distribution with the coordinate of the center location as the mean as well as the variance of the Gaussian distribution is set to 0.1. All the tracking methods take the bounding box labelled manually in the first frame as the initial input.

Table 1 All groups of the sequences covering various scenarios[a]

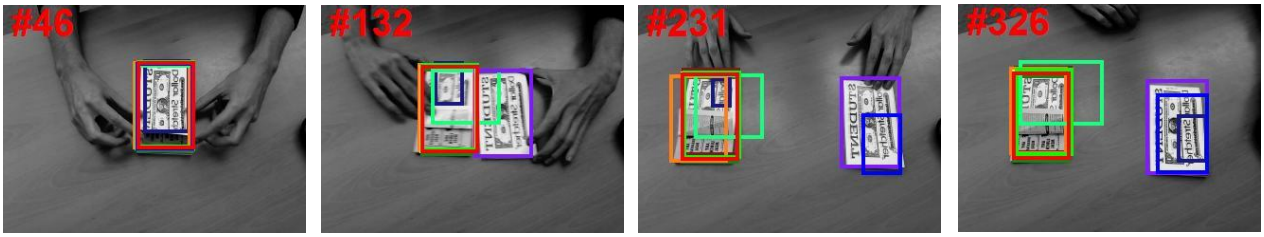
Dominant Factor	Video Sequences
Background Clutter (BC)	*Car11*, *Dollar*, *Stone*, *Tiger1*
Fast Motion (FM)	*Ball*, *Deer*, *Juice*, *Jumping*
Illumination Variation (IV)	*Car4*, *Davidface*, *Shaking*, *Skating1*, *Sylvester*
Viewpoint Change (VC)	*Couple*, *Cup on table*, *Dog1*, *Girl*, *Person*
Non-rigid Deformation (NRD)	*Basketball*, *Bolt*, *Gym*, *Mountain-bike*
Partial Occlusion (PO)	*Davidoutdoor*, *Faceocc1*, *Person partial occluded*, *Woman*
Heavy Occlusion (HO)	*Coke*, *Faceocc2*, *Soccer*, *Suv*

[a] https://sites.google.com/site/trackerbenchmark/benchmarks/v10

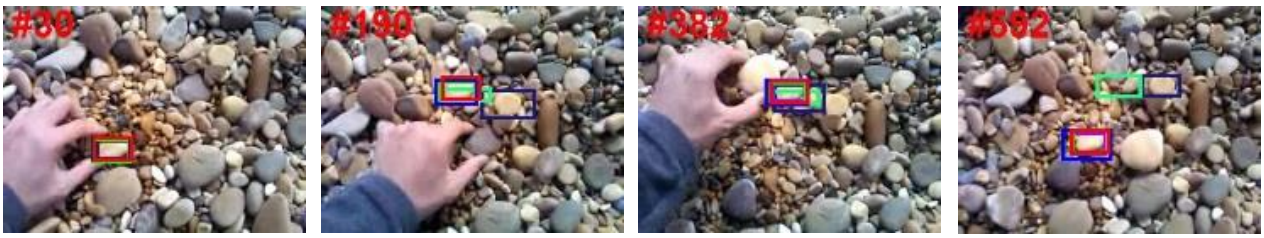
As for the dataset, considering the factors that affect the tracking performance, the video sequences can be categorized into several groups. All groups of the sequences are summarized in Table 1. For a sequence, the task of tracking a target is usually confronted with several challenges simultaneously. To facilitate the evaluations of the tracking performances when facing different challenges, all factors are independently analyzed according to the scenarios. Therefore, only one dominant factor is taken into consideration for a sequence when the experiments are carried out. These video sequences cover most of the challenges that a tracker may come cross.

4.2 Qualitative comparison

(a) Car11



(b) Dollar



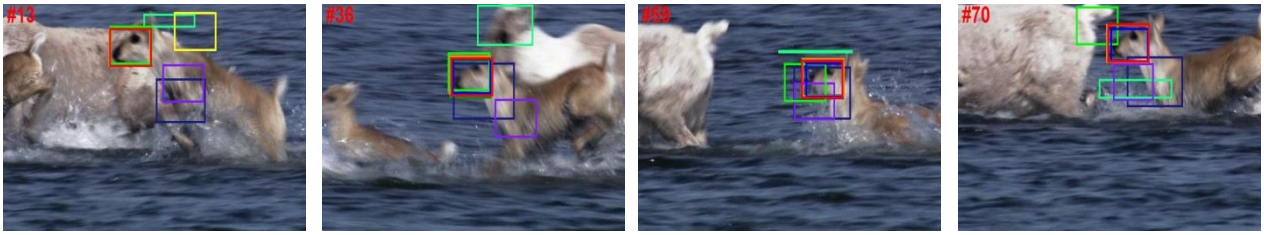
(c) Stone

(d) Tiger1

Fig. 2 Screen shots of tracking results under the situations of cluttered background.

Background Clutter: In the cluttered background, the objects with the same appearances to the specific target, more often than not, arise in the company of the target. Some trackers are prone to be distracted from these objects. In Fig. 2(a), the complex background impairs the descriptive power of the color information characterizing the car. In Fig. 2(b), after a pile of dollars are divided into two piles, both of them look like each other. In Fig. 2(c), there are several stones with the similar shapes and color, which are stacked up in the proximity of the stone tracked. According to the tracking results, it seems that LOT and PMT struggle to resist against the distractors occurring in the cluttered background. For these two methods, the image regions of interest are diverted to the image patches with similar appearances in the four sequences shown in Fig. 2. In addition, it can be seen that TLD and SCM are also readily interrupted by distractors. When the object image patch is immersed with the background pixels, the matching of these background pixels also inhibits object tracking to some extent. In Fig. 2(d), while most of the tracking methods lose the toy due to the influence of the background, our method can still follow the target stably.

(a) Ball



(b) Deer

(c) Juice

(d) Jumping

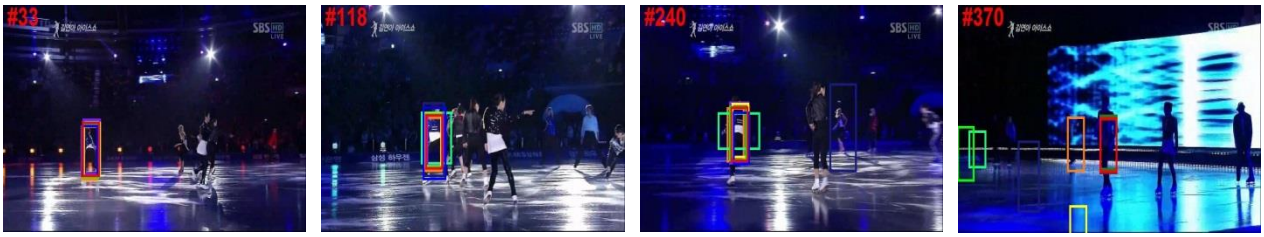
Fig. 3 Screen shots of tracking results under the situations of fast motion.

Fast Motion: For the tracking methods, fast motion means that the search region needs to be expanded for the identification of the target. In general, this expansion will result in the heavy computational burden, which makes it impractical for the tracking methods with the complicate appearance model to accomplish tracking. In all of the sequences shown in Fig. 3, our method,

KCF, MEEM and TLD are able to acquire the robust tracking results. It is noted that these methods all learn the detectors from a set of training samples, and rely on the detector to track the target. Once the detector is trained, the rapid detection will enhance the efficiency of tracking. The expansion of the search region also benefits from this enhancement of the efficiency.

(a) Davidface

(b) Shaking

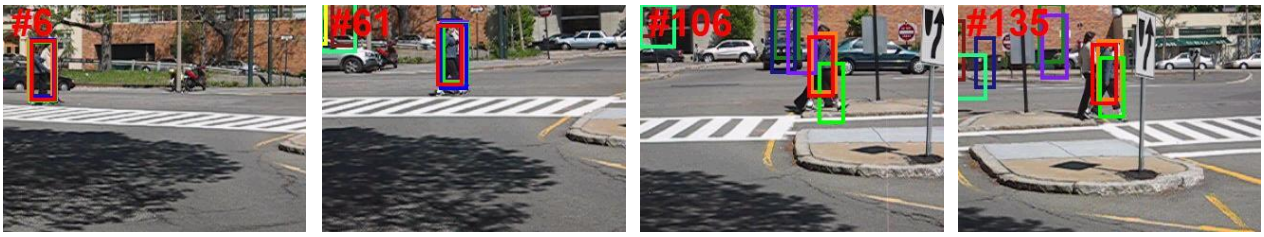


(c) Skating1

(d) Sylvester

Fig. 4 Screen shots of tracking results under the situations of illumination variation.

Illumination Variation: The image intensity of the target tends to vary with the change of the illumination condition. Hence, the raw color information is not adequate to the role of a robust feature. In Fig. 4(a), the face varies from bright to dark. It can be seen that, in this sequence, PMT, CNT, LOT, SCM and Struck all produce drifts. In Fig. 4(b), the head of the man is immersed into the glow while shaking. After the glow fades out, our method, MEEM, SCM and STC recover the tracking. In Fig. 4(c), the illumination condition of the environment causes the significant changes in the intensity contrast for both the environment and the target. There are two methods including our method and KCF that can still keep tracking of the target. In Fig. 4(d), the toy undergoes different lighting. Most of the methods perform well in this type of illumination variation case. The results demonstrate the stable tracking performance of our method. This is largely attributed to that the HOG is adopted as the feature that can overcome the effect brought by the illumination variation.



(a) Couple

(b) Cup on table

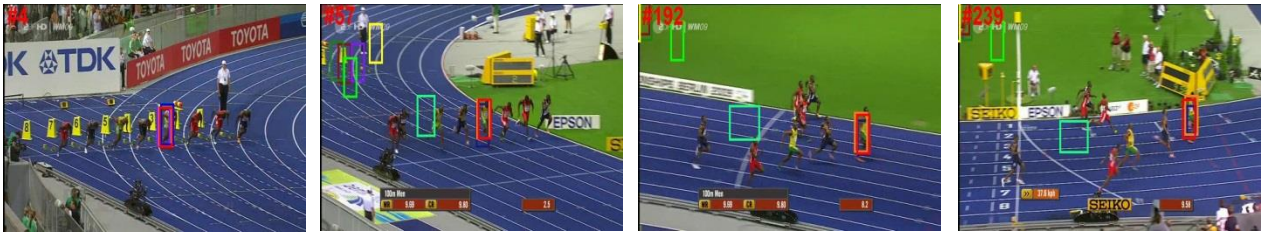
(c) Girl

(d) Person

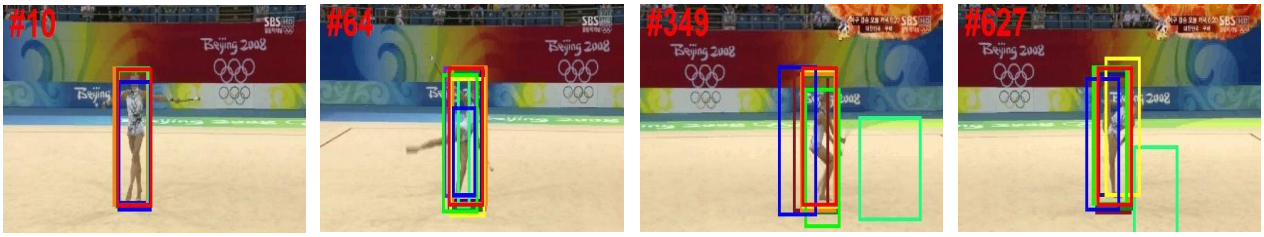
Fig. 5 Screen shots of tracking results under the situations of viewpoint changes

Viewpoint Change: When the viewpoint changes, the appearance of the target usually does not keep still, and the consistence of the appearances among the consecutive frames will be violated. For this type of the tracking task, the stability of tracking largely depends on the flexibility of the appearance model. In Fig. 5(a) and (b), when the cameras move, different side views of the targets are exhibited. Benefiting from the efficient updating for the detectors, our method and MEEM both perform better than others. In Fig. 5(c) and (d), it can be seen that the pose change produces entirely different views for the same objects. Due to that all these methods are equipped with the ability to update the appearance model online, in the case of smooth variation, they are still capable of catching the target even though there exist drift errors.

(a) Basketball



(b) Bolt



(c) Gym

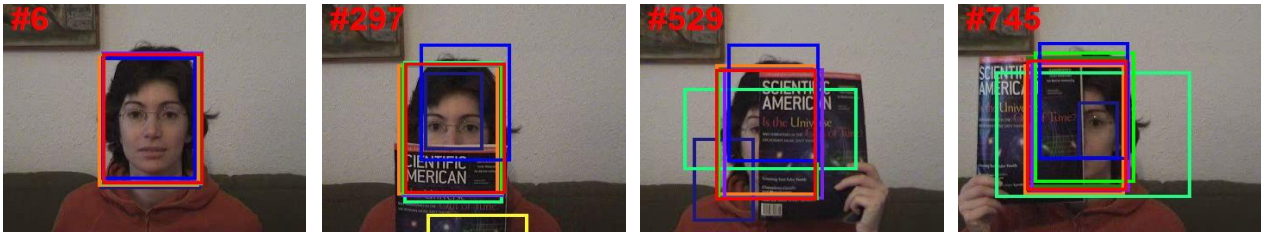
(d) Mountain-bike

Fig. 6 Screen shots of tracking results under the situations of non-rigid deformation.

Non-rigid Deformation: When the target is deformable, the spatial layout of the pixels inside the object image patch is not fixed. Except for the statistical color information, the spatial information like shape is hard to be utilized by this type of tracking tasks. In Fig. 6, it can be seen that our method, MEEM, KCF and LOT cope with these tasks well. As for our method, it is

obvious that the usage of the dense sampling strategy and the HOG feature mitigates the negative effect on tracking exerted by the disorder of the pixels positions, and yields stable tracking results.

(a) Davidoutdoor



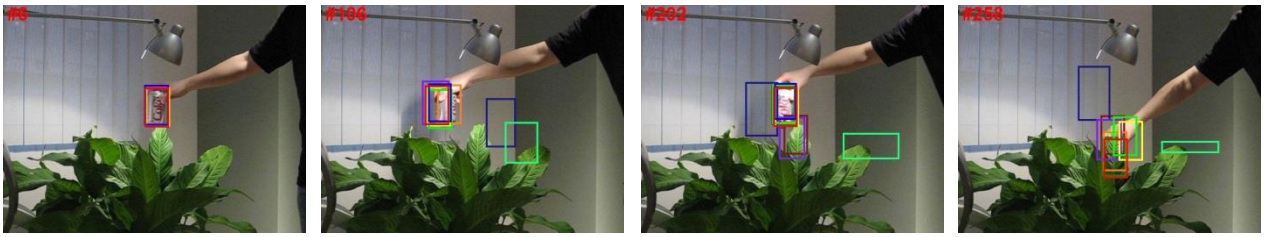
(b) Faceocc1

(c) Person partial occluded

(d) Woman

Fig. 7 Screen shots of tracking results under the situations of partial occlusion

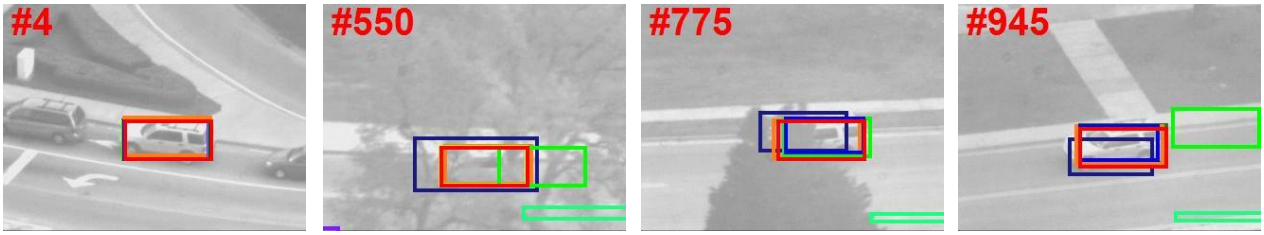
Partial Occlusion: Partial occlusion is a classic problem for tracking. It produces an incomplete appearance of the target. While some parts of the target are visible, the rest of the parts are covered by other objects. Aimed at this problem, the part based appearance models provide reliable parts for tracking. In Fig. 7(a), the man is occluded by a tree while walking. TLD is get trapped into the rear of the car. In Fig. 7(b), when some of the face of the woman is covered by a book, only parts of the face offer the local cue for tracking. PMT gives a poorer performance than other methods. In Fig. 7(c), the camera motion leads to the partial occlusion happening in the man. Most of the methods can locate the man even under the condition of occlusion. In Fig. 7(d), when the woman bypasses the car, there remains the half part of the woman that is visible. It is observed that our method, STC, MEEM, KCF, SCM still work well. PMT can only provide an object image patch containing much background, while other methods even lose the target. In our method, due to that the HOG features are extracted from the cells representing image blocks, in fact, the detection of the target is also implemented on the local parts of the target.



(a) Coke

(b) Faceocc2

(c) Soccer



(d) Suv

Fig. 8 Screen shots of tracking results under the situations of heavy occlusion.

Heavy Occlusion: In the case of heavy occlusion, the large part of the target and even the entire appearance is invisible. After the target recurs, it requires that the tracker be able to recover tracking. In Fig. 8(a), the can hides behind the leaves. Once that the can appears, the trackers such as MEEM, SCM, STC, KCF and our method still find it. In Fig. 8(b), only a small part of the face is exposed to the camera, which makes the tracking difficult. CNT and LOT drift away from the face. In Fig. 8(c), in the combination of full occlusion and viewpoint change, the face of the player is hard to be distinguished from the background. The kernel space and dense sampling endow our method and CNT with the ability to detect the player. In Fig. 8(d), the car is overlapped with the tree while moving along the road. Benefiting from the detection mechanism, our method, MEEM, TLD and KCF achieve satisfactory results for the sequence.

4.3 Quantitative Evaluation

Despite of either the target image patch identified by our method or the ground truth, they are both located with bounding boxes in all frames. To investigate the tracking performance quantitatively, first of all, the center location error is employed to measure the tracking accuracy. The Euclidean distance is calculated between the center of the bounding boxe given by our method and the ground truth at each frame. The average errors of all tracking methods over the sequences are reported in Table 2.

Table 2 Center location errors of all tracking methods over the video clips

Algo.	CNT	KCF	LOT	MEEM	PMT	SCM	STC	Struck	TLD	Ours
BC	4.49	**2.87**	39.16	**3.65**	16.89	12.09	9.82	13.40	19.64	**2.66**
FM	66.31	**2.80**	13.52	5.40	43.35	6.47	600.43	5.95	**4.06**	**2.22**
IV	8.62	10.66	46.23	**7.07**	13.18	16.56	9.52	12.84	**7.64**	**8.16**
VC	8.88	**4.29**	6.41	**4.53**	22.18	5.11	34.93	5.08	7.59	**4.96**
NRD	47.21	47.53	**19.74**	**5.27**	74.06	104.08	65.79	78.70	44.38	**8.39**
PO	46.70	7.33	37.08	**6.80**	10.26	73.18	59.34	**5.85**	19.77	**6.48**
HO	34.56	16.17	22.46	**15.77**	44.91	80.18	54.23	24.04	**6.88**	**5.79**
Aver	26.20	**13.67**	25.61	**7.00**	29.92	39.50	83.91	19.38	14.41	**5.85**

Among the methods listed, it is noted that our method performs best in the scenes including background clutter, fast motion and heavy occlusion. As for the target undergoing the non-rigid deformation or partial occlusion, the performances of our method are inferior to MEEM and Struck resepectively. Under the condition of illumination variation and viewpoint change, MEEM, KCF and TLD achieve better results than other methods.

It is worth noting that, in terms of center location error, KCF obtains the most accurate result under the situation of viewpoint change. It is mainly attributed to two aspects: 1) According to the validation experiments on the performances of multiple kernel learning,[37] the learned combination of kernels cannot consistently perform better than the best single kernel in an arbitrary classification task. Once that the cyclic version of a base image patch cannot provide enough training samples to identify strong kernels, it is possible for our method to be inferior to

the method based on the single best performing kernel. 2) The fine-tuning parameters. The tracking performances of the kernelized correlation filters-based methods not only depend on the selection of a kernel function but also are affected by other factors such as the size of the base image patch, the tradeoff parameter λ and so on. In terms of the center location error, the setting of small size is in favor of the decrease in the error. While KCF sets the size to be 1.5 times larger than the size of the target image patch, our method sets it to be 3 times in order to generalize the tracker to as many scenes as possible. In some cases, compared with our method, this setting of small size enables KCF to incur less center location error. Nevertheless, in terms of all criteria, overall, our method outperforms KCF.

Table 3 Overlap ratios of all tracking methods over the video clips

Algo.	CNT	KCF	LOT	MEEM	PMT	SCM	STC	Struck	TLD	Ours
BC	0.754	**0.763**	0.478	**0.761**	0.461	0.728	0.723	0.716	0.668	**0.757**
FM	0.496	**0.696**	**0.737**	0.601	0.513	**0.716**	0.471	0.594	0.645	0.693
IV	0.544	0.536	0.475	**0.597**	0.550	**0.600**	0.547	0.543	**0.616**	0.524
VC	0.623	0.651	**0.692**	0.643	0.487	**0.739**	0.499	0.660	0.663	**0.668**
NRD	0.621	**0.664**	0.630	**0.661**	0.277	0.602	0.567	0.567	0.585	**0.675**
PO	0.771	**0.776**	0.541	0.729	0.582	**0.765**	0.764	0.759	0.599	**0.777**
HO	0.673	**0.727**	0.547	**0.744**	0.406	0.680	0.724	0.684	0.689	**0.752**
Aver	0.645	**0.672**	0.586	0.670	0.474	**0.687**	0.601	0.642	0.638	**0.678**

Along with the center location error, the average overlap ratio over the valid sequences is also provided for the evaluation of the tracking performance. Given the tracked bounding box ROI_T and the ground truth bounding box ROI_G, the corresponding overlap ratio can be derived, and it is defined as:

$$\text{overlap ratio} = \frac{|\mathrm{ROI}_T \cap \mathrm{ROI}_G|}{|\mathrm{ROI}_T \cup \mathrm{ROI}_G|} \tag{24}$$

Where ROI_T and ROI_G represent the areas of the bounding boxes of the tracking result and the ground truth bounding box respectively. The symbols $\cap$ and $\cup$ denote the intersection and union of two bounding boxes respectively. Besides it, in the definition of overlap ratio, $|\cdot|$ denotes the number of pixels that the corresponding area contains.

In Table 3, it can be seen that SCM possesses the highest overlap ratio. It is ascribed to that SCM establishes a motion model under the particle filter framework, and takes the rotation and scale into consideration. This is helpful for the acquirement of the large intersection part between the tracked box and the ground truth. Our method ranks the second place. Finally, the success rate and the precision plot are introduced into the overall evaluation. Combining all kinds of criteria, our method totally performs better than SCM.

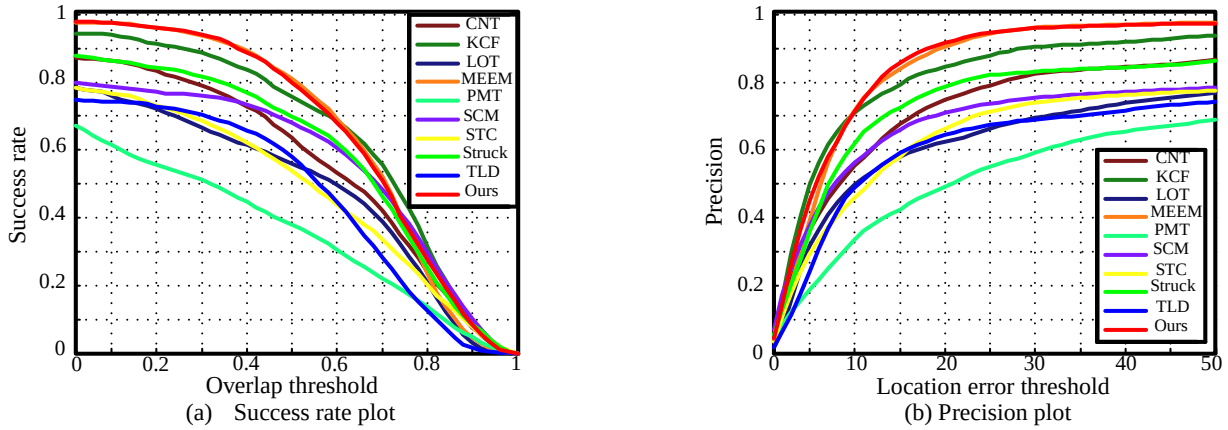
As the more effective criteria for the evaluation of the overall tracking performance over 30 video sequences, the success rate plot and the precision plot with respect to the thresholds are provided in Fig. 9.

For the tracking result at a frame, when the corresponding overlap ratio is larger than a given overlap threshold, it is considered as a success. The number of successful frames is counted, and then the ratio of the number of successful frames to the total number of frames in the video clip is defined as success rate. When the overlap threshold varies from 0 to 1, the success rate also changes. The changes can be reflected with a two-dimensional plot in Fig. 9(a).

The precision plot is largely based on the center location error. The frame, at which the center location error is less than a given distance threshold, is deemed as the successful frames. Subsequently, the ratio of the number of the successful frames to the total number of frames is defined as the precision. When the distance threshold varies from 0 pixel to 50 pixels, the precision also changes. This variation is shown in Fig. 9(b).

As Fig. 9(a) shows, in terms of success rate, our method is close to MEEM and outperforms other methods. However, as the threshold decreases, the success rate of KCF gradually surpasses our method and MEEM. In addition, in view of the precision shown in Fig. 9(b), when the threshold varies from 0 to 10 pixels, KCF achieves the most accurate results compared with other

methods. Once the threshold reaches up to 10 and continues to increase, the tracking errors of our method and MEEM are dramatically less than other methods. Both the success rate and the precision demonstrate that our method is competent to handle general challenges that most trackers are often confronted with.



(a) Success rate plot (b) Precision plot

Fig. 9 The success rate plot and the precision plot for all the tracking methods.

4.4 Comparison of speeds

To investigate the efficiency of the proposed method, the average runtimes of the methods are compared. All of the trackers proposed by these methods are implemented on a PC with a 2.53GHz Intel Core i3 CPU and 4 GB memory. The speed is measured in the number of frames tackled by a tracker in one second (fps). An overall efficiency comparison is exhibited in Table 4. In addition, the proposed tracking methods based on Algorithm 1 and Algorithm 2 are investigated. Among the trackers, "Ours1" denotes the method based on Algorithm 1, and "Ours2" stands for the method based on Algorithm 2.

Table 4 The speeds of all methods running over 30 video clips (measured in fps)

Algo.	CNT	KCF	LOT	MEEM	PMT	SCM	STC	Struck	TLD	Ours1	Ours2
Speed(fps)	34.85	43.96	0.21	6.65	0.20	0.31	104.32	0.07	9.18	1.3×10^{-3}	6.76

From Table 4, it can be seen that our method ranks in the middle of these methods. STC achieves the highest speed. It largely benefits from the simplified appearance model and the

location model. Besides the simple models, it uses FFT to accelerate the implementation of the algorithm. KCF and CNT both exploit the correlation filters to devise trackers. However, due to that the tracker developed by CNT constructs the low-dimensional color feature to take the place of the HOG feature, it acquires a more efficient tracking performance than KCF. Compared with KCF, our method not only needs to seek an optimal relation among a set of kernels but also needs to speed up the implementation of the tracking. Hence, in terms of speed, it is reasonable that our method is slower than KCF. Thanks to the real-time detection mechanism, TLD also runs faster than our method. Nevertheless, while there does not exist a wide gap in the speed, our method can obtain a better tracking accuracy than TLD. Other than these methods mentioned, our method is more efficient than the remainder of the methods. Moreover, our method is able to maintain a better tracking performance.

According to the resultant comparison, it is evident that the tracking method based on Algorithm 2 runs faster than Algorithm 1. Since that Algorithm 1 is involved in the product and inversion of the matrices, these operations consume much time. In terms of the computational complexity, while the cost of Algorithm 1 is at least $O(n^3)$, the cost of Algorithm 2 based on the element-wise products and DFT of the vectors is $O(n\log n)$. In contrast to Algorithm 1, Algorithm 2 is only involved in the element-wise operation, which avoids the time-consuming operations. Hence, our method based on fast interpolate iterative algorithm is more efficient than the traditional interpolate iterative algorithm that is illustrated in Algorithm 1.

4.5 Tracking performances versus the selection of kernels

In the proposed method, the tracking accuracy concerns with the selection of the kernel functions. The type of kernel function plays the important role in the speedup algorithm. Through the proof of Theorem 1, it can be seen that whether the Theorem 1 holds relies on the circulant property of

single kernel function. Only under the assumption that each kernel function satisfies circulant property, we can reach the conclusion that the linear combination of the Gram matrices is still a circulant matrix. The kernels such as Gaussian kernel, linear kernel and polynomial kernel all satisfy this requirement.[38] To realize the acceleration presented in Algorithm 2, the selection of the kernel function must be constrained to this likes of kernel functions.

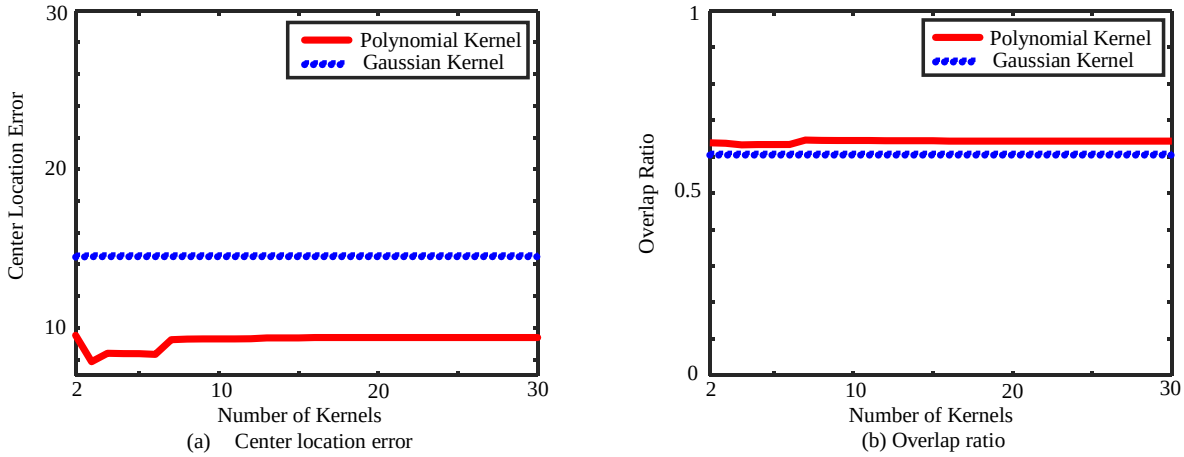


Fig. 10 Tracking performances versus the number of kernels.

Aside from the type of the kernel function, the influence of the number of kernels on the tracking performance is also investigated. Due to that the cyclic version of a base image patch only produces a finite number of features as the training samples, it is necessary that the sufficient kernels be maintained for the improvement on the discriminative power of the regression function.[37] We examine the tracking performance of our method with the increasing number of different kernels. Since that the linear kernel does not involve parameters, the linear combination of kernels incorporates at most one linear kernel. Hence, the linear kernel is excluded from the investigation. Rather than the linear kernel, we focus on the numbers of the polynomial kernels and the Gaussian kernels. Two linear combinations of kernels are constructed for the examination, one of which consists of the polynomial kernels while the other is made up

of the Gaussian kernels. In addition, the parameters of each kernel are different from each other. The examination is implemented on 30 video clips listed in Table1.

In Fig. 10, the tracking performances including center location error and overlap ratio are summarized as the number of kernels increases. As Fig. 10(a) shows, when the number of polynomial kernels is small, the center location error diminishes. But, once that the number of the polynomial kernels exceeds 6, the error increases and then stays at the level. In Fig. 10(b), it is noted that, for the polynomial kernels, the overlap ratio drops a litter and then keeps still as the number increases. However, for the Gaussian kernels, regardless of the center location error and the overlap ratio, they are still consistent even though more and more Gaussian kernels are combined.

According to the observations, when the number of polynomial kernels is small, the tracking performance can be improved through adding the polynomial kernels. Nevertheless, more polynomial kernels cannot bring more enhancements in the performance. As for the Gaussian kernels, it is obvious that the number of kernels does not have influence on the tracking performance.

Table 5 Tracking performances versus types of kernels

(*OR*: Overlap Ratio, *CLE*: Center Location Error, L: Linear kernel, P: Polynomial kernel, G: Gaussian kernel)

Kernel	Linear(L)	Polynomial(P)	Gaussian(G)	L + P	L + G	P + G	L + P + G
OR	0.631	0.637	0.633	0.639	0.641	0.637	0.641
CLE	15.41	14.49	14.92	14.46	14.22	14.46	14.17

In order to investigate the influence of the kernel type on the tracking performance, three kernels are selected from the set of the eleven kernels above, which consist of a linear kernel, a polynomial kernel (the parameters $a = 1.2,\ b = 9$) and a Gaussian kernel (the parameter $\sigma = 1$). The tracking performances, including the center location error and the overlap ratio, over

all possible combinations of the three kernels are checked. The corresponding results are listed in Table 5.

In terms of a single kernel, the tracker based on the polynomial kernel outperforms other two kernels. The Gaussian kernel is inferior to the polynomial kernel. The linear kernel is weakest among them. As for the combination of two kernels, the tracker based on the combination of the polynomial kernel and the Gaussian kernel performs best. In addition, it is noted that a tracker based on any combination of a pair of different kernels obtains more accurate tracking results than an arbitrary single kernel. When the three kernels are combined, compared with a single kernel or a pair of kernels, the performance is enhanced. According to the comparisons, the combination of different kernels is able to improve on the tracking performance to some extent.

4.6 Nonlinear combination of kernel functions

The appearance model based on the regression function is relevant to the linear structure of the feature distribution. However, it is usually difficult to depict the feature distribution with a linear structure. Thanks to the kernel trick, all features can be mapped into a kernel space implicitly, where the distribution of the features mapped can be approximated with a linear structure. The key problem turns out to be how to establish an appropriate kernel space that meets the requirement.

Instead of a single kernel, under the framework of multiple kernel learning, we seek a linear combination of kernel functions to construct the kernel space. Every kernel selected from a group of kernels gives its own measurement of the similarity between a pair of features. While some of the kernels are in favor of the discrimination between features, the other may cripple the discrimination. It is expected that the former kernels are selected while the latter kernels are suppressed. Through assigning different coefficients to the kernels, an optimal combination of

the kernels can be determined. Eventually, a kernel space with a proper similarity measurement is given.

Aiming at the inherent nonlinear relation of a group of kernels, the nonlinear combination of kernels has been proposed.[39] But the nonlinear combination of kernels has to face the non-convex optimization problem that will significantly reduce down the computational efficiency. It contradicts to the attempt to improve on the efficiency of visual tracking.

Conclusion

In this paper, we propose a method for visual tracking based on kernel ridge regression. A kernel regression function is learned from the image patches with the soft labels representing the likelihoods of the target. Multiple kernel learning is introduced into the selection of kernels. A linear combination of different kernels is employed to create a kernel space for the features extracted from the image patches. With interpolate iterative algorithm, the coefficients are determined for the combination of kernels. In order to enhance the efficiency in learning coefficients, the circulant property of the Gram matrix is exploited to develop a fast version of interpolate iterative algorithm. It is integrated into the framework of kernel ridge regression, which further accelerates the computation of the confidence of an image patch as the object image patch. Both qualitative and quantitative evaluations over the challenging sequences demonstrate the competitive performance of our method.

Acknowledgments

This work has been supported by the Project of Natural Science Research of Higher Education Institutions of Jiangsu Province (Grant No. 15KJB520003), the Project supported by the Jiangsu Key Laboratory of Image and Video Understanding for Social Safety (Nanjing University of

Science and Technology) (Grant No. SHAQKFKT201505) and the Natural Science Foundation of Changzhou Institute of Technology (No. YN1204). The contribution of Miss Xinxin Peng is appreciated.

References

1. S. Salti, A. Cavallaro, and L. D. Stefano, "Adaptive appearance modeling for video tracking survey and evaluation," *IEEE Trans. Pattern Anal. Mach. Intell.,* **21**(10), 4334-4348 (2012).
2. X. Li, W. Hu, C. Sehn, Z. Zhang, A. Dick, and A. V. D. Hengel, "A survey of appearance models in visual object tracking," *ACM TIST*, **4**(4), 58-100 (2013).
3. H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: a review," *Neurocomputing*, **74**(18), 3823-3831 (2011).
4. N. Jiang, W. Liu, "Data-driven spatially-adaptive metric adjustment for visual tracking," *IEEE Trans. on Image Processing*, **23**(4), 1556-1568 (2014).
5. X. Li, W. Hu, Z. Zhang, X. Zhang, M. Zhu, J. Cheng, "Visual tracking via incremental log-euclidean riemannian subspace learning," in *Proc. CVPR*, pp. 1-8, IEEE, Anchorage (2008).
6. W. Hu, X. Li, W. Luo, X. Zhang, S. Maybank, and Z. Zhang, "Single and multiple object tracking using log-euclidean riemannian subspace and block-division appearance model," *IEEE Trans. Pattern Anal. Mach. Intell.*, **34**(12), 2420-2440 (2012).
7. Z. H. Khan, I. Y. H. Gu, "Nonlinear dynamic model for visual object tracking on grassmann manifolds," *IEEE T. Cybernetics*, **43**(6), 2005-2019 (2013).
8. J. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. ECCV*, (4), pp. 702-715, Springer, Firenze (2012).
9. M. Danelljan, F. S. Khan, M. Felsberg, and J. V. D. Weijer, "Adaptive color attributes for real-time visual tracking," in *Proc. CVPR*, pp. 1090-1097, IEEE, Columbus (2014).
10. M. J. Black, A. D. Jepson, "Eigentracking: robust matching and tracking of articulated objects using a view-based representation," *International Journal of Computer Vision*, **26**(1), 63-84 (1998).

11. D. A. Ross, J. Lim, R. Lin, and M. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, **77**(1), 125-141 (2008).
12. M. Hu, X. Li, X. Zhang, X. Shi, S. Maybank, and Z. Zhang, "Incremental tensor subspace learning and its applications to foreground segmentation and tracking," *International Journal of Computer Vision*, **91**(3), 303-327 (2011).
13. X. Mei, H. Ling, "Robust visual tracking using l1 minimization," in *Proc. Int Conf. on Computer Vision (ICCV)*, pp. 1436-1443, IEEE, Kyoto (2009).
14. X. Mei, H. Ling, "Robust visual tracking and vehicle classification via sparse representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, **33**(11), 2259-2272 (2011).
15. C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l1 tracker using accelerated proximal gradient approach," in *Proc. CVPR*, pp. 1830-1837, IEEE, Providence (2012).
16. D. Wang, H. Lu, and M. Yang, "Online object tracking with sparse prototypes," *IEEE Transactions on Image Processing*, **22**(1), 314-325 (2013).
17. X. Jia, H. Lu, and M. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *CVPR*, pp. 1822-1829, IEEE, Providence (2012).
18. W. Zhong, H. Lu, and M. Yang, "Robust object tracking via sparse collaborative appearance model," *IEEE Transactions on Image Processing*, **23**(5), 2356-2368 (2014).
19. S. Avidan, "Support vector tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, **26**(8), 1064-1072 (2004).
20. S. Hare, A. Saffari, and P. H. S. Torr, "Struck: structured output tracking with kernels," in *Proc. Int Conf. on Computer Vision (ICCV)*, pp. 263-270, IEEE, Barcelona (2011).
21. J. Gao, J. Xing, W. Hu, and X. Zhang, "Graph embedding based semi-supervised discriminative tracker," in *Proc. Int Conf. on Computer Vision Workshops*, pp.145-152, IEEE, Sydney (2013).
22. L. Wang, H. Yan, K. Lv, and C. Pan, "Visual tracking via kernel sparse representation with multikernel fusion," *IEEE Trans. Circuits Syst. Video Techn.*, **24**(7), 1132-1141 (2014).

23. F. Yang, H. Lu, M. Yang, "Robust visual tracking via multiple kernel boosting with affinity constraints," *IEEE Trans. Circuits Syst. Video Techn.*, **24**(2), 242-254 (2014).

24. G. R. G. Lanckriet, N. Cristianini, P. Bartlett, L. E. Ghaoui, and M. I. Jordan, "Learning the kernel matrix with semidefinite programming," *Journal of Machine learning Research*, **5**, 27-72 (2004).

25. S. Sonnenburg, G. Ratsch, C. Schafer, and B. Scholkopf, "Large scale multiple kernel learning," *Journal of Machine learning Research*, **7**, 1531-1565 (2006).

26. M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K. R. Muller, and A. Zien, "Efficient and accurate l_p-norm multiple kernel learning," in *Proc. NIPS*, pp. 997-1005, Springer, Vancouver (2009).

27. H. Yang, Z. Xu, J. Ye, I. King, and M. R. Lyu, "Efficient sparse generalized multiple kernel learning," *IEEE Transactions on Neural Networks*, **22**(3), 433-446 (2011).

28. M. Kloft, U. Brefeld, P. Laskov, and S. Sonnenburg, "Non-sparse multiple kernel learning," in *Proc. NIPS Workshop*, 1-4, Springer, Vancouver (2008).

29. C. Cortes, M. Mohri, and A. Rostamizadeh, "L_2 regularization for learning kernels," in *Proc. Conf. in Uncertainty Artificial Intelligence*, pp. 109-116, AUAI, Montreal (2009).

30. B. Babenko, M. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, **33**(8), 1619-1632 (2011).

31. N. Dalal, B. Triggs, "Histograms of oriented gradients for human detection", in *CVPR*, pp. 886-893, IEEE, Providence (2005).

32. S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking", in *Proc. CVPR*, pp. 1940-1947, IEEE, Providence (2012).

33. J. Zhang, S. Ma, and S. Sclaroff, "MEEM: robust tracking via multiple experts using entropy minimization," in *Proc. ECCV*, pp. 188-203, Springer, Zurich (2014).

34. Z. Zhang, K. H. Wong, "Pyramid-based visual tracking using sparsity represented mean transform," in *Proc. CVPR*, pp. 1226-1233, IEEE, Columbus (2014).

35. K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M. Yang, "Fast visual tracking via dense spatio-temporal context learning", in *Proc. ECCV*, pp. 127-141, Springer, Zurich (2014).

36. Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, **34**(7), 1409-1422 (2012).
37. S. S. Bucak, R. Jin, A. K. Jain, "Multiple kernel learning for visual object recognition: a review", *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(7), 1354-1369 (2014).
38. J. F. Henriques, R. Caseiro, , P. Martins, J. Batista, "High-speed tracking with kernelized correlation filters", *IEEE Trans. Pattern Anal. Mach. Intell.*, **37**(3), 583-596(2015).
39. C. Cortes, M. Mohri, A. Rostamizadeh, "Learning non-linear combination of kernels", in *Proc. NIPS*, pp. 396-404, Springer, Vancouver (2009).

Cheng Qian is a lecturer at Changzhou Institute of Technology. He received his BS and MS degrees in automation from Hangzhou Dianzi University in 2004 and 2007, respectively, and his Ph.D degree in computer science from Zhejiang University in 2011. His current research interests include machine learning, pattern recognition, and computer vision.

Toby Breckon is a senior lecturer in the innovative computing group at Durham University. He received his Ph.D degree from University of Edinburgh in 2006. His current research interests include computer vision, image processing and robotic sensing.

Hui Li is a lecturer at Changzhou Institute of Technology. Her research interest includes data mining and software engineering.

Caption List

Fig. 1 The collection of the training samples by the cyclic shifts of the base image.

Fig. 2 Screen shots of tracking results under the situations of cluttered background: (a) Car11, (b) Dollar, (c) Stone and (d) Tiger1.

Fig. 3 Screen shots of tracking results under the situations of fast motion: (a) Ball, (b) Deer, (c)Juice and (d) Jumping.

Fig. 4 Screen shots of tracking results under the situations of illumination variation: (a) Davidface, (b) Shaking, (c) Skating1 and (d) Sylvester.

Fig. 5 Screen shots of tracking results under the situations of viewpoint changes: (a) Couple, (b) Cup on table, (c) Girl and (d) Person.

Fig. 6 Screen shots of tracking results under the situations of non-rigid deformation: (a) Basketball, (b) Bolt, (c) Gym and (d) Mountain-bike.

Fig. 7 Screen shots of tracking results under the situations of partial occlusion: (a) Davidoutdoor, (b) Faceocc1, (c) Person partial occluded and (d) Woman.

Fig. 8 Screen shots of tracking results under the situations of heavy occlusion: (a) Coke, (b) Faceocc2, (c) Soccer and (d) Suv.

Fig. 9 The success plot and the precision plot for all the tracking methods. (a) Success rate plot and (b) Precision plot.

Fig. 10 Tracking performances versus the number of kernels.

Table 1 All groups of the sequences covering various scenarios.

Table 2 Center location errors of all tracking methods over the video clips.

Table 3 Overlap ratios of all tracking methods over the video clips.

Table 4 The speeds of all methods running over 30 video clips (measured in fps).