Pattern Recognition Letters 90 (2017) 72–79

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Kernelized temporal locality learning for real-time visual tracking

Fanghui Liu[a], Tao Zhou[a], Keren Fu[a,b], Jie Yang[a,*]

[a] Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, Shanghai, 200240, China
[b] School of Electronic Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China

ARTICLE INFO

Article history:
Received 4 June 2016
Available online 22 March 2017

Keywords:
Visual tracking
Kernel method
Temporal smoothness constraint
Appearance model

ABSTRACT

Linear representation-based methods play an important role in the development of the target appearance modeling in visual tracking. However, such linear representation scheme cannot accurately depict the nonlinearly distributed appearance variations of the target, which often leads to unreliable tracking results. To fix this issue, we introduce the kernel method into the locality-constrained linear coding algorithm to comprehensively exploit its nonlinear representation ability. Further, to fully consider the temporal correlation between neighboring frames, we develop a point-to-set distance metric with $L_{2,1}$ norm as the temporal smoothness constraint, which aims to guarantee that the object between the two consecutive frames should be represented by the similar dictionaries temporally. Experimental results on Object Tracking Benchmark show that the proposed tracker achieves promising performance compared with other state-of-the-art methods.

1. Introduction

Visual tracking is an attractive research area in computer vision. It aims to accurately predict the states (i.e., location and target size) of an object of interest in a video sequence given its initial state at the first frame. Although numerous methods [1,2] have been proposed to achieve a robust tracker, it remains a challenging problem because of the intrinsic factors (e.g., pose changes and shape deformation) and extrinsic factors (e.g., partial occlusions and illumination variations).

In recent years, there is an increasing interest in sparse representation-based trackers, e.g., Mei and Ling [3], and Zhang et al. [4]. The rationale of sparse representation based methods is that a possible target candidate[1] is a linear combination of sparse atoms in a dictionary (e.g., a template set **T** in the area of visual tracking). Considering that the running speed of these methods is often limited by solving the ℓ_1 norm optimization problem, the local coordinate coding (LCC) [5] and locality-constrained linear coding (LLC) [6] were proposed to achieve a similar sparsity property as that using a few anchor points [7] or the k-nearest neighbors (NN) selection scheme with high computational efficiency. Further, such two coding methods can also guarantee that similar candidate samples are associated with similar coefficient vectors or share similar dictionaries, which comprehensively exploit the appearance information carried by their local dictionaries. Thus, because of these advantages, these two coding algorithms were incorporated into visual tracking framework [8–10] with a comparable performance.

Despite the demonstrated success, there are two typical drawbacks of these trackers. First, both the methods have the same fundamental assumption that the candidate sample can be well represented by templates through a linear representation scheme. However, the establishment of a target appearance model is more likely a nonlinear problem because of dramatically nonlinearly distributed appearance variations in many practical scenarios. Such linear representation models easily fail to represent the sample distribution problem, and thus can hardly comprehensively understand the target appearance and effectively alleviate error accumulation. Apart from the limitation of the linear representation scheme, the other drawback is that most methods (not limited to the aforementioned trackers) overlook information of neighbouring frames, and only treat the tracking results independently between two consecutive frames. In fact, there is no denying that only a few methods [11,12] are proposed to consider the temporal correlation of coefficient vectors between neighboring frames. However, such correlation constraint on coefficient vectors is not entirely practical for LLC and LCC algorithms. Comparatively, it is more intuitive to consider the temporal smoothness for the dictionary rather than the coefficient vectors in these two coding methods. For example, in the LLC algorithm, the core idea is that if two candidate samples are close to each other, they should be represented by two similar local dictionaries that are constructed by each candidate's k-nearest neighbors from the template set **T**. Considering that the

* Corresponding author.
E-mail addresses: lfhsgre@outlook.com (F. Liu), jieyang@sjtu.edu.cn (J. Yang).

[1] Here "candidate" refers to a sampled image patch, and we aim to develop a tracker to select the optimal candidate sample as the tracking result.

http://dx.doi.org/10.1016/j.patrec.2017.03.019
0167-8655/© 2017 Published by Elsevier B.V.

target usually changes slightly between adjacent frames in the tracking sequences, these two obtained tracking results between two consecutive frames are represented by two similar dictionaries. By doing so, we could fully exploit the temporal and spatial smoothness properties of the local dictionary in the LLC method to enhance the representation ability in appearance modeling.

On the basis of the above discussion, this paper introduces the kernel method into the approximated LLC algorithm with temporal smoothness constraint on the local dictionary termed "Kernelized Temporal Locality Learning" (KTLL) tracker. The main novelties of the proposed method include:

1. A point-to-set distance metric with $L_{2,1}$ norm regularization as the temporal smoothness constraint is proposed to construct the local dictionary, which explicitly exploits the spatial and temporal information of neighboring frames.
2. A kernelized LLC method is derived to exploit its nonlinear representation ability for visual tracking.

We test the KTLL tracker on Object Tracking Benchmark (OTB) demonstrated by its effectiveness.

The rest of the paper is organized as follows. Section 2 reviews the previous works related to the proposed method. Section 3 introduces the details of kernelized temporal dictionary learning model. Section 4 incorporates this model into the visual tracking framework. Experimental results on OTB and parameter analysis are provided in Section 5 followed by conclusion drawn in Section 6.

2. Related work and context

2.1. Nonlinear methods for target appearance model

As aforementioned, many existing methods usually model the appearance of targets by linear representations over a dictionary. However, such linear representation schemes lack the ability to accurately depict the nonlinearly distributed appearance variations of the target. To resolve the nonlinear distribution problem in visual tracking, various nonlinear learning based methods were proposed to enhance the target representation ability. Ma et al. [13] proposed to learn multiple linear and nonlinear subspaces to better model the nonlinear relationship of the target appearances. Gong et al. [14] regarded visual tracking as a ranking problem, where PageRank algorithm [15] was incorporated into the visual tracking framework to discover the underlying geometrical structure of the target in a given image database. Further, Wu et al. [16] proposed the landmark-based label propagation for visual tracking, which is nonparametric and makes no specific assumption about sample distribution.

The kernel method [17] is a powerful and widely used approach to effectively capture the nonlinear similarity of samples. Henriques et al. [18] incorporated the kernel trick [19] into ridge regression to enhance the discriminative ability to separate the target from its surrounding background. Tang and Feng [20] combined multiple kernel learning with the correlation filter-based tracker, thus incorporating the strengths of both multiple channels and multiple kernels in a single method.

2.2. Approximated LLC

Approximated LLC method [6] considers local sparsity by selecting the k nearest neighbors of a candidate sample $\mathbf{y}$ from the template set $\mathbf{T}$ to construct the local dictionary $\mathbf{B}$. The candidate $\mathbf{y}$ is sparsely represented by a linear combination of several base vectors in the local dictionary $\mathbf{B}$ with a coefficient vector $\mathbf{c}$, and it is given by:

$$\min_{\mathbf{c}} \|\mathbf{y}-\mathbf{B}\mathbf{c}\|_2^2 \quad s.t. \quad \mathbf{1}^\top\mathbf{c}=1\,, \tag{1}$$

where $\mathbf{1}$ denotes all-one vector and the constraint $\mathbf{1}^\top\mathbf{c}=1$ ensures shift-invariance. In our method, the coefficients $\mathbf{c}$ is used to search the most similar candidate region to the given target with minimal reconstruction error.

3. Kernelized temporal locality learning model

3.1. Temporal smoothness constraint for dictionary

We define the following notations for easy understanding. At $(t-1)$th frame, the obtained tracking result is defined as y_*^{t-1}, and its corresponding local dictionary is expressed as $\mathbf{B}_*^{t-1}$. Likewise, at tth frame, the corresponding local dictionary $\mathbf{B}_i^t$ is constructed by the k-nearest neighbors of the ith candidate sample $\mathbf{y}_i^t$. Such k-nearest neighbors searching scheme of $\mathbf{y}_i^t$ can be expressed as follows:

$$\begin{cases} \mathrm{idx}_1 = \operatorname*{argmin}_{j\in\mathcal{S}_1} \|\mathbf{y}_i^t-\mathbf{T}_j\|_2^2 \quad \mathcal{S}_1=\{1,2,\ldots,(p+n)\}; \\ \mathrm{idx}_2 = \operatorname*{argmin}_{j\in\mathcal{S}_2} \|\mathbf{y}_i^t-\mathbf{T}_j\|_2^2 \quad \mathcal{S}_2=\mathcal{S}_1\backslash\mathrm{idx}_1; \\ \;\;\cdots\cdots \\ \mathrm{idx}_k = \operatorname*{argmin}_{j\in\mathcal{S}_k} \|\mathbf{y}_i^t-\mathbf{T}_j\|_2^2 \quad \mathcal{S}_k=\mathcal{S}_{k-1}\backslash\mathrm{idx}_{k-1}. \end{cases}, \tag{2}$$

where the template set $\mathbf{T}=[\mathbf{T}^{pos},\mathbf{T}^{neg}]\in\mathbb{R}^{M\times(p+n)}$ is composed of the positive template set $\mathbf{T}^{pos}$ (i.e., the target) and the negative template set $\mathbf{T}^{neg}$ (i.e., the background), and M is the dimension of a vectorized template. The numbers of positive and negative templates are p and n, respectively. By using such k-NN searching scheme, the corresponding local dictionary of $\mathbf{y}_i^t$ is given by $\mathbf{B}_i^t=[\mathbf{T}_{\mathrm{idx}_1},\mathbf{T}_{\mathrm{idx}_2},\cdots,\mathbf{T}_{\mathrm{idx}_k}]$.

Nevertheless, the above searching scheme does not consider the temporal correlation between $\mathbf{B}_i^t$ and $\mathbf{B}_*^{t-1}$, but treat them independently between two consecutive frames. In what follows, we introduce the temporal smoothness constraint for the local dictionary construction and a proper distance metric measuring the similarity between $\mathbf{B}_i^t$ and $\mathbf{B}_*^{t-1}$.

Among the three recent metric learning methods, namely point-to-point distance, point-to-set distance and set-to-set distance [21], the set-to-set distance is the most convenient and straightforward metric to meet the requirement of our method, which consists of measuring the similarities between two matrices (e.g., $\mathbf{B}_i^t$ and $\mathbf{B}_*^{t-1}$). However, the specificity of our model is that the similarity score measured between such two matrices measured by a distance metric should be independent of their column orders. That is, two local dictionaries constructed by the same templates are assigned to the same similarity score by a proper distance metric irrespective of their column orders. In fact, there are some faithful distance metrics [21,22] to achieve what we want it to be, that is, the similarity score for such two matrices is independent of their column orders. Unfortunately, they are either time-consuming or depend on the pre-known data distribution, which are not suitable for real-time visual tracking. According to this, to exclude the effect of column orders in our proposed method, we relax the set-to-set metric, and consider the point-to-set distance (PSD) metric. The PSD metric between the rth template $\mathbf{T}_r$ and the local dictionary $\mathbf{B}_*^{t-1}$ is defined as follows:

$$d_{PSD}(\mathbf{T}_r,\mathbf{B}_*^{t-1}) \triangleq \|\underbrace{[\mathbf{T}_r,\mathbf{T}_r,\ldots,\mathbf{T}_r]}_{\mathcal{T}}-\mathbf{B}_*^{t-1}\|_{2,1}\,, \tag{3}$$

where $\mathcal{T}=[\mathbf{T}_r,\mathbf{T}_r,\ldots,\mathbf{T}_r]\in\mathbb{R}^{M\times k}$ and the $L_{2,1}$ norm of a matrix $\mathbf{A}\in\mathbb{R}^{M\times k}$ is defined as the sum of ℓ_2 norm of each column in $\mathbf{A}$. Compared to the conventional $\|\cdot\|_F$ norm (the average distance) and the $\|\cdot\|_1$ norm (the minimal distance), the $L_{2,1}$ norm imposes column-wise ℓ_1 norm which not only retains the Euclidean property of data vectors but also alleviates the influence of outliers

[23,24]. In Eq. (3), a smaller d_{PSD} indicates that the template $\mathbf{T}_r$ is closer to $\mathbf{B}_*^{t-1}$, which allows for the measurement of the similarity (or smoothness) between $\mathbf{T}_r$ and $\mathbf{B}_*^{t-1}$.

To incorporate this temporal smoothness constraint in the k-nearest neighbors searching scheme for the local dictionary $\mathbf{B}_i^t$, Eq. (3) is substituted into Eq. (2), we have

$$\begin{cases} \text{idy}_1 = \underset{r\in\mathcal{S}_1}{\text{argmin}} \underbrace{\|(\mathbf{Y}_i^t - \mathbf{T})_{\cdot r}\|_2^2}_{\mathbf{e}(r)} + \beta \underbrace{\|[\mathbf{T}_r, \mathbf{T}_r, \ldots, \mathbf{T}_r] - \mathbf{B}_*^{t-1}\|_{2,1}}_{\mathbf{d}(r)}; \\ \text{idy}_2 = \underset{r\in\mathcal{S}_2}{\text{argmin}} \|(\mathbf{Y}_i^t - \mathbf{T})_{\cdot r}\|_2^2 + \beta\|[\mathbf{T}_r, \mathbf{T}_r, \ldots, \mathbf{T}_r] - \mathbf{B}_*^{t-1}\|_{2,1}; \\ \cdots\cdots \\ \text{idy}_k = \underset{r\in\mathcal{S}_k}{\text{argmin}} \|(\mathbf{Y}_i^t - \mathbf{T})_{\cdot r}\|_2^2 + \beta\|[\mathbf{T}_r, \mathbf{T}_r, \ldots, \mathbf{T}_r] - \mathbf{B}_*^{t-1}\|_{2,1}. \end{cases}, \tag{4}$$

where $\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_k$ are defined in Eq. (2), and β is a tradeoff parameter to be tuned between dictionary similarity and dictionary temporal smoothness. We define the matrix $\mathbf{Y}_i^t = [\mathbf{y}_i^t, \mathbf{y}_i^t, \ldots, \mathbf{y}_i^t] \in \mathbb{R}^{M\times(p+n)}$ whose dimension is equal to the number of templates. In Eq. (4), the residual error $\mathbf{e}(r)$ reflects the similarity between the i-th candidate $\mathbf{y}_i^t$ and the r-th template $\mathbf{T}_r$, and $\mathbf{d}(r)$ denotes the temporal smoothness level between $\mathbf{T}_r$ and the local dictionary $\mathbf{B}_*^{t-1}$. A smaller distance $\mathbf{e}(r) + \beta\mathbf{d}(r)$ indicates that the r-th template $\mathbf{T}_r$ is more acceptable as a near(est) neighbor of $\mathbf{y}_i^t$. Finally, the local dictionary $\mathbf{B}_i^t = [\mathbf{T}_{\text{idy}_1}, \mathbf{T}_{\text{idy}_2}, \cdots, \mathbf{T}_{\text{idy}_k}]$ of the candidate $\mathbf{y}_i^t$ is constructed from the template set $\mathbf{T}$ with the first k smaller distances in Eq. (4). Note that Eqs. (2) and (4) can be efficiently computed using matrix operations, thereby making such k-nearest neighbors selection scheme very efficient.

3.2. Kernelized LLC method

3.2.1. Kernel trick

By considering that there exists a nonlinear mapping function ϕ, the kernel method [19] maps the data $\mathbf{x}$ into $\phi(\mathbf{x})$ in a potentially much higher dimensional feature space $\mathcal{F}$. Specifically, we do not need to calculate $\phi(\mathbf{x})$ explicitly: for a certain feature space $\mathcal{F}$ and the corresponding mapping ϕ, there is a highly effective "kernel trick" for computing the inner products between the two features $\phi(\mathbf{x})$ and $\phi(\mathbf{x}')$ in feature space using the kernel function denoted as $\phi(\mathbf{x})^\top\phi(\mathbf{x}')$. To able to map arbitrary data into an infinite dimensional space for nonlinear representation, we use the radial basis function (RBF) the kernel function, namely:

$$\mathcal{K}(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|_2^2}{2\sigma^2}\right), \tag{5}$$

where σ is the tuning parameter.

3.2.2. Kernel trick in the LLC method

To introduce the kernel trick in our KTLL tracker, we use the mapping function ϕ to map a candidate sample $\mathbf{y}$[2] and the corresponding local dictionary $\mathbf{B}$ into a high-dimensional space: $\mathbf{y} \to \phi(\mathbf{y})$ and $[\mathbf{B}_1, \mathbf{B}_2, \cdots, \mathbf{B}_r, \cdots\mathbf{B}_k] \to [\phi(\mathbf{B}_1), \phi(\mathbf{B}_2), \cdots, \phi(\mathbf{B}_r), \cdots\phi(\mathbf{B}_k)]$, where $\mathbf{B}_r$ is the rth atom in the local dictionary $\mathbf{B}$. Supposing that the mapped candidate $\phi(\mathbf{y})$ can be linearly represented by the mapped dictionary $\phi(\mathbf{B})$ with a sparse vector $\mathbf{c}$ in a high-dimensional space, the corresponding objective function in Eq. (1) is formulated as follows:

$$\min_{\mathbf{c}} \frac{1}{2}\|\phi(\mathbf{y}) - \phi(\mathbf{B})\mathbf{c}\|_2^2 + \frac{\lambda}{2}\|\mathbf{c}\|_2^2 \quad s.t. \quad \mathbf{1}^\top\mathbf{c} = 1, \tag{6}$$

[2] Note that the candidate index i is omitted for simplicity.

where λ is the regularization parameter. By omitting the irrelevant term with respect to $\mathbf{c}$, the objective function in Eq. (6) is reformulated as follows:

$$\min_{\mathbf{c}} \frac{1}{2}\mathbf{c}^\top\phi(\mathbf{B})^\top\phi(\mathbf{B})\mathbf{c} - \mathbf{c}^\top\phi(\mathbf{B})^\top\phi(\mathbf{y}) + \frac{\lambda}{2}\|\mathbf{c}\|_2^2 \quad s.t. \quad \mathbf{1}^\top\mathbf{c} = 1, \tag{7}$$

where $\phi(\mathbf{B}) = [\phi(\mathbf{B}_1), \phi(\mathbf{B}_2), \ldots, \phi(\mathbf{B}_k)]$. By introducing a kernel function $\mathcal{K}(\mathbf{B}_i, \mathbf{B}_j) = \langle\phi(\mathbf{B}_i), \phi(\mathbf{B}_j)\rangle$, we rewrite the objective function in Eq. (7) as:

$$\min_{\mathbf{c}} \frac{1}{2}\mathbf{c}^\top K_{\mathbf{BB}}\mathbf{c} - \mathbf{c}^\top K_{\mathbf{By}} + \frac{\lambda}{2}\|\mathbf{c}\|_2^2 \quad s.t. \quad \mathbf{1}^\top\mathbf{c} = 1, \tag{8}$$

where $K_{\mathbf{BB}}$ is a $k \times k$ kernel matrix with $[K_{\mathbf{BB}}]_{ij} = \mathcal{K}(\mathbf{B}_i, \mathbf{B}_j)$, and $K_{\mathbf{By}}$ is a $k \times 1$ vector with $[K_{\mathbf{By}}]_i = \mathcal{K}(\mathbf{B}_i, \mathbf{y})$. $K_{\mathbf{BB}}$ and $K_{\mathbf{By}}$ can be easily represented respectively by:

$$K_{\mathbf{BB}} = \begin{pmatrix} \mathcal{K}(\mathbf{B}_1, \mathbf{B}_1) & \mathcal{K}(\mathbf{B}_1, \mathbf{B}_2) & \cdots & \mathcal{K}(\mathbf{B}_1, \mathbf{B}_k) \\ \mathcal{K}(\mathbf{B}_2, \mathbf{B}_1) & \mathcal{K}(\mathbf{B}_2, \mathbf{B}_2) & \cdots & \mathcal{K}(\mathbf{B}_2, \mathbf{B}_k) \\ \vdots & \vdots & \ddots & \vdots \\ \mathcal{K}(\mathbf{B}_k, \mathbf{B}_1) & \mathcal{K}(\mathbf{B}_k, \mathbf{B}_2) & \cdots & \mathcal{K}(\mathbf{B}_k, \mathbf{B}_k) \end{pmatrix}, \tag{9}$$

and

$$K_{\mathbf{By}} = [\mathcal{K}(\mathbf{B}_1, \mathbf{y}), \mathcal{K}(\mathbf{B}_2, \mathbf{y}), \cdots, \mathcal{K}(\mathbf{B}_k, \mathbf{y})]^\top. \tag{10}$$

Let β be a Lagrange multiplier for the shift-invariant constraint $\mathbf{1}^\top\mathbf{c} = 1$ on the coefficient vector $\mathbf{c}$, then the Lagrange function of Eq. (8) can be defined as:

$$\mathcal{L} = \frac{1}{2}\mathbf{c}^\top K_{\mathbf{BB}}\mathbf{c} - \mathbf{c}^\top K_{\mathbf{By}}\mathbf{1}^\top\mathbf{c} + \frac{\lambda}{2}\mathbf{c}^\top\mathbf{c} + \beta(1 - \mathbf{1}^\top\mathbf{c}). \tag{11}$$

The partial derivative of $\mathcal{L}$ with respect to $\mathbf{c}$ is given by:

$$\frac{\partial\mathcal{L}}{\partial\mathbf{c}} = K_{\mathbf{BB}}\mathbf{c} - 2K_{\mathbf{By}}\mathbf{1}^\top\mathbf{c} + \lambda\mathbf{c} - \beta\mathbf{1}. \tag{12}$$

By using the Karush–Kuhn–Tucker condition, we obtain the following analytic solution:

$$\mathbf{c} = \beta\Big[K_{\mathbf{BB}} - 2K_{\mathbf{By}}\mathbf{1}^\top + \lambda\mathbf{I}\Big]^{-1}\mathbf{1}. \tag{13}$$

where $\mathbf{I}$ is the $k \times k$ identity matrix. The Lagrange multiplier β can be also omitted by the shift-invariant constraint. The kernel matrix $K_{\mathbf{TT}}$ can be pre-computed. As a result, $K_{\mathbf{BB}}$ is easily selected with the corresponding elements in $K_{\mathbf{TT}}$. Thus, we need to compute $K_{\mathbf{By}}$ per frame during the visual tracking process, which makes our algorithm very efficient.

4. Proposed KTLL tracker

4.1. Particle filtering

In visual tracking, a particle filter [3,9] is often used for state estimation. The implicit rationale behind using a particle filter is to approximately estimate the posterior distribution $p(\mathbf{x}_t|\mathbf{Y}_{1:t})$ by a finite set of randomly sampled particles. Given some observed image patches at the t-th frame $\mathbf{Y}_{1:t} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{t-1}\}$, the state of the target $\mathbf{x}_t$ can be estimated recursively as follows:

$$p(\mathbf{x}_t|\mathbf{Y}_{1:t}) \propto p(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{Y}_{1:t-1})\,d\mathbf{x}, \tag{14}$$

where the motion model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ denotes the state transition between two consecutive frames, and $p(\mathbf{y}_t|\mathbf{x}_t)$ denotes the observation model that estimates the likelihood of observing $\mathbf{y}_t$ at state $\mathbf{x}_t$. By defining the target state $\mathbf{x}_t = [l_x, l_y, \theta, s, \alpha, \phi]^\top$, where l_x, l_y, θ, s, α, ϕ denote the translations in the direction of x and y, rotation angle, scale, aspect ratio, and skew respectively, we apply the affine transformation with the above six parameters to model the target motion $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. The optimal state at the tth frame is obtained by maximizing the approximate posterior probability:

$$\mathbf{x}_t^* = \underset{\mathbf{x}_t^i}{\operatorname{argmax}}\, p(\mathbf{y}_t^i|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}), i = 1, 2, \cdots, N, \tag{15}$$

where $\mathbf{x}_t^i$ is the ith sample of the state $\mathbf{x}_t$, and $\mathbf{y}_t^i$ is a candidate sample predicated by $\mathbf{x}_t^i$.

Here, we briefly introduce how to obtain templates at the beginning of a video sequence. In addition to positive templates, we also collect several negative templates (i.e., background) to enhance the discriminative ability of the tracker. Generally, the tracking result in the first frame is manually chosen as a rectangle box. Assuming that $\mathcal{I}(x, y)$ is the center of the rectangle box, the initial positive templates $\mathbf{T}^{pos} = [\mathbf{T}_1, \mathbf{T}_2, \ldots, \mathbf{T}_p]$ are sampled from an inner circular area that satisfies $\|\mathcal{I}_i - \mathcal{I}(x, y)\| < R_r$, where $\mathcal{I}_i$ is the center of the ith sampled image patch ($i = 1, 2, \ldots, p$), and R_r is the radius of the inner circular area. Similarly, the negative templates $\mathbf{T}^{neg} = [\mathbf{T}_{p+1}, \mathbf{T}_{p+2}, \cdots, \mathbf{T}_{p+n}]$ are sampled from the annular region $R_r < \|\mathcal{I}_j - \mathcal{I}(x, y)\| < R_s$, where $\mathcal{I}_j$ is the center of the jth sampled image patch ($j = p + 1, \cdots, p + n$), and R_s is the outer radius of the annular region. These positive and negative templates form the template set $\mathbf{T} \in \mathbb{R}^{M\times(p+n)} = [\mathbf{T}^{pos}, \mathbf{T}^{neg}]$. Some N observed vectorization image patches at the tth frame $\mathbf{Y}_{1:N} = \{\mathbf{y}_1, \mathbf{y}_2, \cdots, \mathbf{y}_N\} \in \mathbb{R}^{M\times N}$ are sampled according to random sampling scheme with the above six affine parameters.

4.2. Observation model

The observation model measures the similarity between a candidate sample and the template set by the reconstruction error. The proposed KTLL model provides the solution to the coefficient vector $\mathbf{c}$ to calculate the reconstruction error. Considering that several negative templates might be selected to construct the local dictionary $\mathbf{B}$, the dictionary and its resulting coefficient vector $\mathbf{c}$ should be divided into two parts, namely $\mathbf{c}_i = [\mathbf{c}^+; \mathbf{c}^-]$ which corresponds to the selected positive local dictionary $\mathbf{B}^+$ and the selected negative local dictionary $\mathbf{B}^-$, respectively. The reconstruction error of the candidate $\mathbf{y}$ with $\mathbf{B}^+$ is defined as follows:

$$\begin{aligned}\varepsilon^+ &= \|\phi(\mathbf{y}) - \phi(\mathbf{B}^+)\mathbf{c}^+\|_2^2 \\ &= (\mathbf{c}^+)^\intercal(K_{\mathbf{B}^+\mathbf{B}^+})\mathbf{c}^+ - 2(\mathbf{c}^+)^\intercal K_{\mathbf{B}^+\mathbf{y}} + \mathcal{K}(\mathbf{y}, \mathbf{y})\end{aligned} \tag{16}$$

Likewise, $\varepsilon^- = \|\phi(\mathbf{y}) - \phi(\mathbf{B}^-)\mathbf{c}^-\|_2^2$ is the reconstruction error of the candidate $\mathbf{y}$ with the selected negative local dictionary $\mathbf{B}^-$. The observation likelihood $p(\mathbf{y}_t|\mathbf{x}_t)$ can be measured by the reconstruction error of the observed image patch $\mathbf{y}$ at the tth frame, using the following expression:

$$p(\mathbf{y}_t|\mathbf{x}_t) \propto (\varepsilon^- - \varepsilon^+)\frac{\exp(-\alpha\varepsilon^+)}{\exp(-\alpha\varepsilon^-) + C}, \tag{17}$$

where the parameter α is a normalization factor, fixed to 2 in our experiments, and C is a small number to avoid dividing by zero.

4.3. Online model updating scheme

Since the tracking result is obtained from a new frame, the appearance model of the target needs to be updated. In our method, the model updating scheme contains two parts: one is for updating the templates, and the other is for the incremental updating of the kernel matrix $K_{\mathbf{TT}}$. For the first part, we maintain a constant number of target templates by a substitution scheme to update the templates since the computational complexity of the kernel matrix is proportional to the number of templates. The negative templates are regularly updated for every five frames. For the positive template set, a positive template (denoted as $\mathbf{T}_i$) with the maximum deviation from the mean value of the whole positive template set is first selected. Then, it is substituted by the current tracking result $\mathbf{y}_t^*$ to maintain the number of positive templates unchanged. For the second part, the corresponding ith column (and row) elements in the kernel matrix $K_{\mathbf{TT}}$ are recalculated by:

$$\mathcal{K}(\mathbf{T}_i, \mathbf{T}_j) \Rightarrow \mathcal{K}(\mathbf{y}_t^*, \mathbf{T}_j), \quad \mathcal{K}(\mathbf{T}_j, \mathbf{T}_i) \Rightarrow \mathcal{K}(\mathbf{T}_j, \mathbf{y}_t^*) \quad \forall j. \tag{18}$$

The algorithm for the KTLL tracker is summarized in Algorithm 1.

Algorithm 1: Algorithm for the proposed KTLL Tracker

```
1  Obtain templates T from 1st to 5th frames, and then calculate
   the kernel matrix $K_{\mathbf{TT}}$.
2  for $t = 6$ to the end of the sequence do
3      $N$ particles $\mathbf{Y}_{1:N}$ are sampled;
4      for i=1:N do
5          Construct the local dictionary $\mathbf{B}_i$ by Eq. (4);
6          Obtain the coefficient vector by Eq. (13);
7          Compute the corresponding likelihood by Eq. (16) and
           Eq. (17);
8      end
9      Choose $\mathbf{x}_t^*$ with the highest confidence value;
10     Update: for each 5 frames do
11         Choose the best tracking result $\mathbf{y}^*$ and then substitute
           a positive template;
12         Incrementally recalculate the kernel matrix $K_{\mathbf{TT}}$ by Eq.
           (18);
13     end
14 end
```

5. Experiments

We test the proposed KTLL tracker on OTB [25] with 29 trackers and 51 video sequences. Besides, several state-of-the-art methods including Linearization to Nonlinear Learning Tracker (LNLT) [9], Structural Sparse Tracker (SST) [4], and Interacting Multiview Tracker (IMT) [26] are also compared with our tracker.

5.1. Experimental setup

The proposed method was implemented in MATLAB on a PC with Intel i5-6500 CPU (3.20 GHz) and 8 GB memory. The following parameters were used for our tests: each observation (i.e. the patch of image) was normalized to 32×32 pixels; the numbers of the initial positive and negative templates were $p = 14$ and $n = 36$, respectively; the spread parameter $\sigma = 1$ was used in RBF Gaussian kernel function; the k-nearest neighbors were fixed to 8; and the regularization parameters λ and β were set to 1 and 0.1, respectively.

5.2. Evaluation metrics

Two typical evaluation criteria were used in OTB. One is the mean Center Location Error (CLE), which is the pixel distance between the centroid of the tracking result and the ground truth. Smaller CLE indicates that the tracker could predict the better location ability. The other is the Pascal VOC Overlap Ratio (VOR) [27], which measures the overlapping degree between the tracked bounding box and the ground truth box, defined as $e = \frac{area(R_T \cap R_G)}{area(R_T \cup R_G)}$, where R_T and R_G are the area of the tracked and ground truth box, respectively. A higher overlap rate implies that the tracker achieves a promising accuracy performance.

On the basis of these two evaluation metrics, the precision plot and the success plot [25] show the percentage of the threshold and

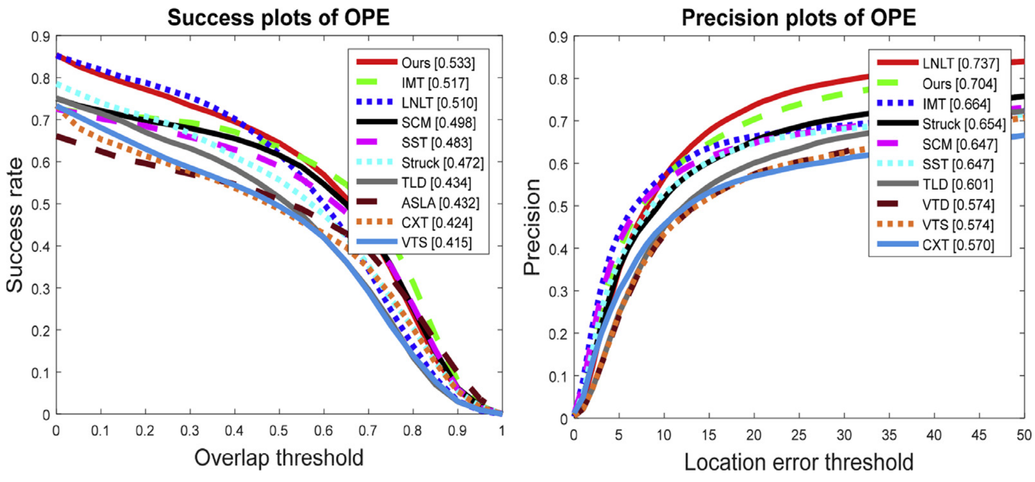

Fig. 1. Plots of OPE. The performance score for each tracker is shown in the embedded legend. For each figure, the top 10 trackers are presented for clarity.

the successfully tracked frames, respectively, to rank these trackers[3]. Generally, the success plot is relatively more impeccable than the precision plot.

5.3. Results on OTB

5.3.1. Overall performance

We show the overall performance of our tracker and compare it with other state-of-the-art methods (ranked within top 10) shown in Fig. 1.

The top 5 trackers on the success plot include LNLT [9], IMT [26], SCM [28], SST [4] and our method. Our tracker ranks the first on the success plot and the second on precision plot. It has a better ability to track the location and a superior overlap rate than existing state-of-the-art methods owing to its nonlinear representation ability and temporal smoothness constraint.

5.3.2. Attribute-based performance analysis

Each sequence in OTB is annotated with 11 attributes that indicates what kinds of challenging factors occur within it. These 11 attributes are *Occlusion, Illumination Variation, Scale Variation, Deformation, Motion Blur, Fast Motion, In-Plane Rotation, Out-of-Plane Rotation, Out-of-View, Background Clutter* and *Low Resolution*. The success plots in Fig. 2 illustrate that the proposed KTLL tracker ranks the first or the second best on four main attributes. Specifically, when compared with the second-best method, our method achieves an improvement of 6.0% and 9.7% for *Occlusion* and *Deformation* attributes, respectively. This demonstrates the interest of our kernel-based tracker by exploiting the nonlinear relationship in appearance modeling instead of that by a linear representation scheme.

5.4. Qualitative analysis

Fig. 3 shows the corresponding tracking results of our method and other top three trackers, namely IMT, SCM and SST on several representative challenging sequences.

Occlusion is one of the most important attributes causing to tracking drifts. In *Suv, Tiger2, Carscale, Woman* and *Jogging.1* sequences, these targets suffer various kinds of occlusions (e.g., the telegraph pole and the branches; partial occlusions and full occlusions). In *Tiger2* sequence, at #255 frame, the SCM and LNLT methods fail to locate the target and the IMT shows tracking drifts to some extent, in comparison with our method. In *Woman* sequence, the woman is partially occluded by the car. During this process, LNLT and IMT methods could not capture the target. Only our method could accurately locate the target during severe occlusions. Results from these sequences show that our method can effectively handle occlusions owing to its robust nonlinear representation ability.

Fast motion and **Motion blur:** There are two reasons leading to motion blur, one is fast motion as shown in *Jumping, Bolt*, and *Lemming* sequences, and the other is drastic camera shake as shown in *Couple* sequence. It is difficult to accurately predict the location of the target when abrupt motion occurs, and the appearance changes because of motion blur. In *Jumping* video, IMT and our tracker show promising results compared with SCM and LNLT trackers. In *Couple* sequence, most of these methods fail to locate the target because of dramatic camera shake. The proposed method shows a promising tracking performance on these sequences attributed to its temporal correlation.

Out-of-plane rotation is an easily overlooked but widely existing attribute that leads to tracking failures as shown in *Couple, Bolt, Lemming, Carscale, Woman*, and *Jogging.1* sequences. It often incurs appearance variations and makes the current appearance model different from the initial one. For example, at #1300 frame in *Lemming* sequence, when the target exhibits pose variations, LNLT and SCM cannot capture these variations accurately. The proposed method and IMT still performs well despite that the appearance of the object changes extensively.

Other attributes: In addition to the abovementioned attribute, there are also others that are not explained in this article. *Tiger2* and *Lemming* sequences contain **illumination variation** attribute and the challenging issues in *Carscale* are **occlusion** and **scale variation**. Most baseline trackers show difficulty in capturing appearance changes when the current appearance differs much from its initial one. The proposed tracker achieves an encouraging performance in terms of accuracy and robustness.

From the abovementioned various sequences, our test results on these videos demonstrate that the proposed tracker is effective and robust to challenging factors because of its nonlinear representation ability and temporal smoothing constraint.

5.5. Self-verification

In this section, we first compare our method with other methods based on LLC or LCC and then analyze the importance of temporal smoothness constraint. Finally, we provide the corresponding tracking results to quantitatively analyze the influence of different parameters in our methods.

[3] The different colors and line styles of the trackers present their rankings on the success plot and the precision plot.

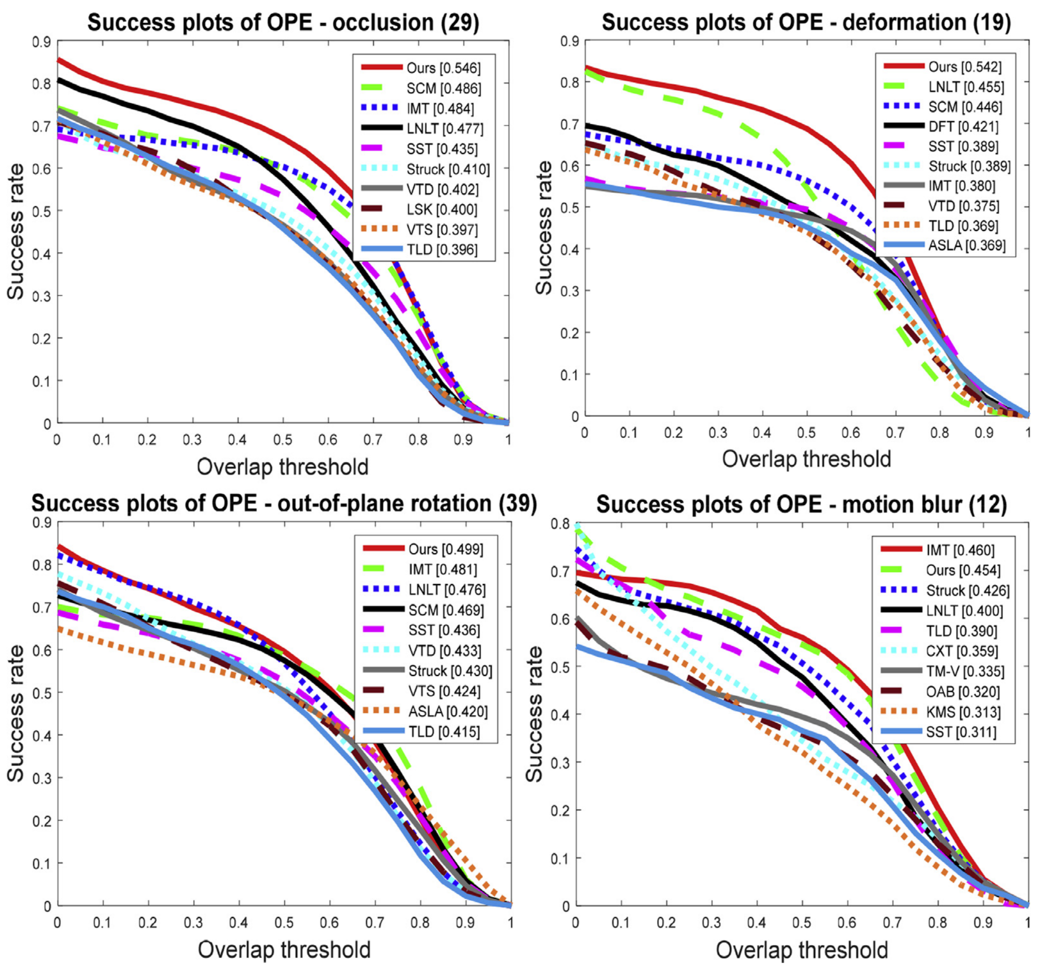

Fig. 2. Success plots of OPE on four main attributes.

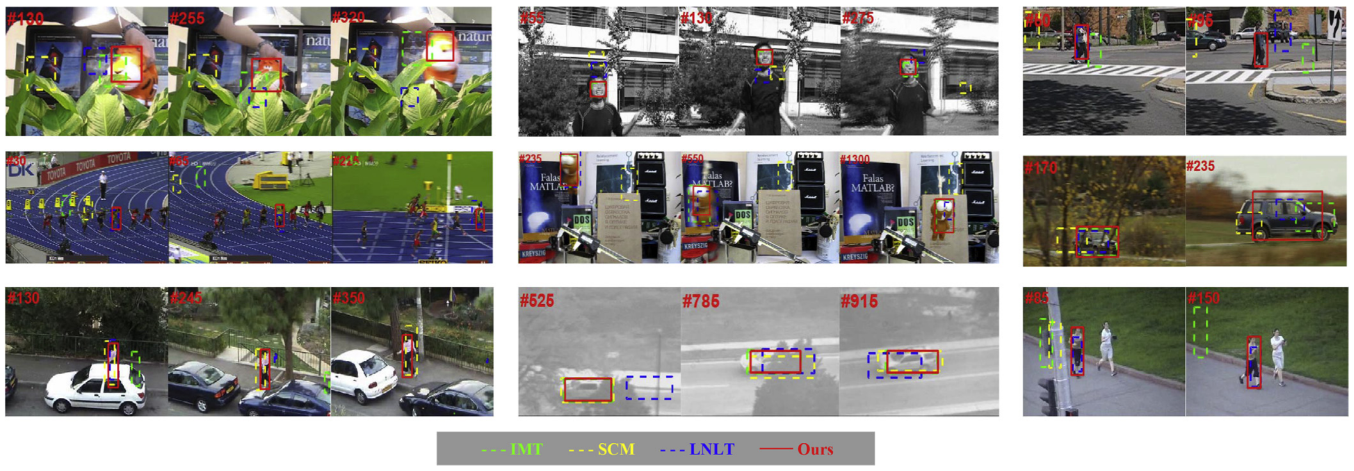

Fig. 3. Representative frames of some sampled tracking results. The figures from left to right, from top to bottom depict *Tiger2*, *Jumping*, *Couple*, *Bolt*, *Lemming*, *Carscale*, *Woman*, *Suv* and *Jogging.1* sequences.

5.5.1. *Comparisons with other local coding-based methods*

We compare the proposed method with three methods based on LLC or LCC algorithms, namely the LSK [8], SRT [29], NMCT [10] and LNLT [9] methods as shown in Fig. 4. Compared with these methods, our tracker ranks the first on the success plot and the second on the precision plot, narrowly followed by the LNLT method.

Comparison of the LNLT method with different iteration algorithms to obtain coefficient vectors, classification label and discriminative dictionary shows that the proposed KTLL method can directly obtain the coefficient vectors and the local dictionary for the optimal candidate, which makes it much faster than the LNLT method.

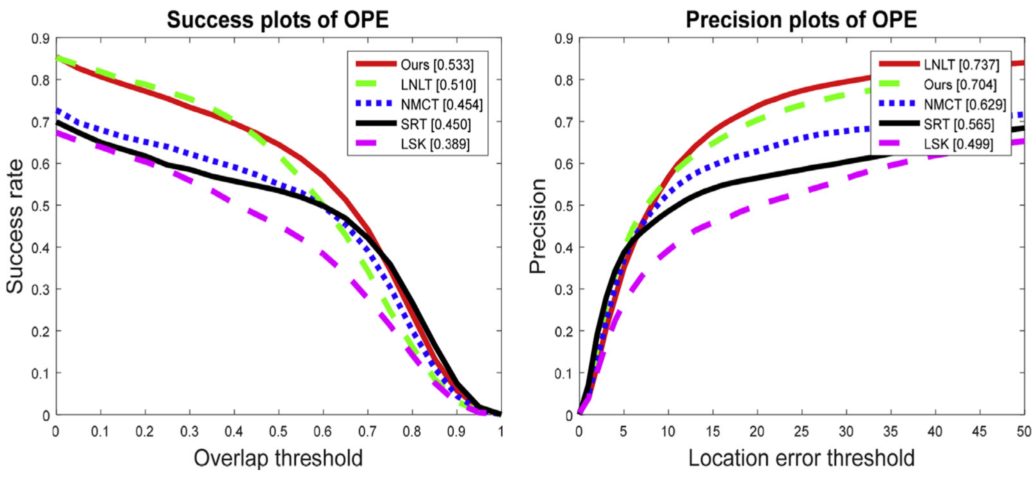

Fig. 4. Plots of OPE in comparisons with other local coding-based methods.

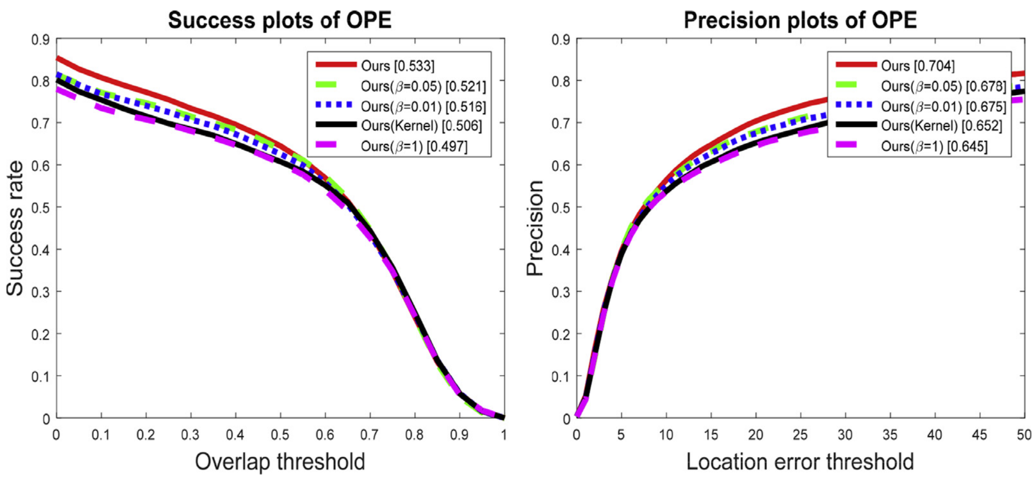

Fig. 5. Plots of OPE in comparisons with different parameter values in temporal smoothness constraint.

Table 1
Average Center Location Errors (CLE) and Average VOC Overlap Ratio (VOR) on OTB with 51 tasks.

Method	Ours	Linear	Polynomial	$\sigma = 0.5$	$\sigma = 5$	$\lambda = 5$	$\lambda = 0.1$	$k = 5$	$k = 10$	$p:n = 10:40$	$p:n = 17:33$	$p:n = 25:25$
Average CLE(pixels)	33.3	58.1	59.4	38.8	46.9	56.9	55.1	43.5	44.4	40.2	37.6	55.5
Average VOR(%)	53.8%	47.6%	48.8%	52.2%	51.1%	50.1%	49.3%	49.7%	50.7%	51.7%	52.9%	49.8%

5.5.2. Influence of temporal smoothness constraint

We also analyzed the improvements of the tracker with regard to the temporal smoothness constraint as shown in Fig. 5.

Without the temporal smoothness constraint (named as “Ours(kernel)”), the proposed method shows degradations of about 2.7% on the success plot. This shows the superiority of the tracker with temporal smoothness constraint over the one without this constraint. When the temporal smoothness increases from 0 to 1, the performances on the success rate and the precision reach the maximum value with $\beta = 0.1$. If β is small, there is slight influence on the final tracking result. A larger value of β (e.g., $\beta = 1$) results in huge degradations from 53.3% to 49.7% on the success plot. It is necessary to seek for a tradeoff between the accuracy of k-nearest neighbors selection and the temporal smoothness constraint.

5.5.3. Influence of different parameter values

In Table 1 are given the experimental results for different kernel functions (linear, polynomial, and RBF Gaussian kernel), different parameter values (e.g., λ and k), and different ratios of positive templates and negative templates on the average CLE and VOR.

Compared with the linear and polynomial kernels, the proposed method with RBF Gaussian kernel shows an obvious improvement by a gain of 6.2% and 5.0% on the average CLE and VOR, respectively, which indicates the considerable importance of such nonlinear mapping method. However, determining the spread parameter σ in RBF Gaussian kernel function is diffcult because of the lack of prior knowledge. Therefore, this parameter is empirically fixed to 1. Table 1 shows that the selection of σ values (e.g., 0.5 and 5) has a slight influence on the tracking performance.

A small λ plays a little role in determining the final tracking result, whereas a large one will lead to over-fitting. When λ is set to 0.1 and 5 with respect to the fixed setting $\lambda = 1$, the average VOR decreases from 53.8% to respectively 49.3% and 50.1%.

The number of k-nearest neighbors determines the representation ability of a local dictionary. A small k is prone to noise, while error accumulation will occur if k is too large. We verify three different nearest neighbors (i.e., $k = 5$, 8, and 10), and the experimental results in Table 1 show that $k = 8$ in the proposed method is appropriate for a better compromise between accuracy and robustness.

Finally, we provide the corresponding results for different values of the ratios of positive and negative templates. Generally, the number of positive templates is smaller than that of negative templates. We compare the results of the proposed method for three different ratios (i.e., 1: 4, nearly 1: 2, and 1: 1) as shown in the last three columns of Table 1. The corresponding results on the average CLE and VOR illustrate that such different ratios have slight influences on the final tracking performance, at most 2% degeneration.

5.6. Running time and computational complexity

The frame per speed (fps) of LSK (as baseline tracker) and our method is about 16. The main computation cost of our method is spent on k-nearest neighbors searching operation and the kernel matrix $K_{\mathbf{By}}$ computation. The computational complexity of k-nearest neighbors searching scheme is $\mathcal{O}(k^2)$. The running time of the kernel matrix computation depends on matrix-vector multiplication $\mathcal{O}(Mk)$. Thus the complexity of the total operations is $\mathcal{O}(k^2 + Mk)$.

6. Conclusion

This paper proposed a kernelized LLC method named KTLL tracker for visual tracking, which effectively exploits its nonlinear representation ability for appearance modeling of the target. The incorporated temporal smoothness constraint enable us to effectively exploit the appearance correlation information between neighboring frames. Thus, the coefficient vector obtained by these two schemes in the KTLL model can accurately characterize the target appearance, which leads to accurate and robust tracking results. The experimental results on the OTB dataset show the rationality of the two schemes with 6.2% and 2.7% improvement on the average overlap rate and success plot, respectively, and demonstrate the effectiveness and robustness of the proposed KTLL tracker when compared with other state-of-the-art methods.

Acknowledgements

The authors would like to thank Prof. Zhu Yuemin for his proofreading and the anonymous reviewers for their insightful comments. This research is partly supported by NSFC, China (No. 61572315, 6151101179), 863 Plan, China (No. 2015AA042308) and the China Postdoctoral Science Foundation (No. 2016M601597).

References

[1] N. Wang, J. Shi, D.-Y. Yeung, J. Jia, Understanding and diagnosing visual tracking systems, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 3101–3109.
[2] K. Zhang, Q. Liu, Y. Wu, M.-H. Yang, Robust visual tracking via convolutional networks without training, IEEE Trans. Image Process. 25 (4) (2016) 1779–1792.
[3] X. Mei, H. Ling, Robust visual tracking and vehicle classification via sparse representation, IEEE Trans. Pattern Anal. Mach. Intell. 33 (11) (2011) 2259–2272.
[4] T. Zhang, S. Liu, C. Xu, S. Yan, B. Ghanem, N. Ahuja, M.-H. Yang, Structural sparse tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 150–158.
[5] K. Yu, T. Zhang, Y. Gong, Nonlinear learning using local coordinate coding, in: Proceedings of Advances in Neural Information Processing Systems, 2009, pp. 2223–2231.
[6] J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, Y. Gong, Locality-constrained linear coding for image classification, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010, pp. 3360–3367.
[7] Y. Chen, J. Zhang, D. Cai, W. Liu, X. He, Nonnegative local coordinate factorization for image representation, IEEE Trans. Image Process. 22 (3) (2013) 969–979.
[8] B. Liu, J. Huang, C. Kulikowski, L. Yang, Robust visual tracking using local sparse appearance model and k-selection, IEEE Trans. Pattern. Anal. Mach. Intell. 35 (2013) 2968–2981.
[9] B. Ma, H. Hu, J. Shen, Y. Zhang, F. Porikli, Linearization to nonlinear learning for visual tracking, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4400–4407.
[10] F. Liu, T. Zhou, J. Yang, I.Y. Gu, Visual tracking via nonnegative regularization multiple locality coding, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2015, pp. 72–80.
[11] T. Liu, G. Wang, L. Wang, K.L. Chan, Visual tracking via temporally smooth sparse coding, IEEE Signal Process. Lett. 22 (9) (2015) 1452–1456.
[12] T. Zhang, S. Liu, N. Ahuja, M.-H. Yang, B. Ghanem, Robust visual tracking via consistent low-rank sparse learning, Int. J. Comput. Vis. 111 (2) (2015) 171–190.
[13] L. Ma, X. Zhang, W. Hu, J. Xing, J. Lu, J. Zhou, Local subspace collaborative tracking, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 4301–4309.
[14] C. Gong, K. Fu, A. Loza, Q. Wu, J. Liu, J. Yang, Pagerank tracker: from ranking to tracking, IEEE Trans. Cybern. 44 (6) (2014) 882–893.
[15] T.H. Haveliwala, Topic-sensitive pagerank: a context-sensitive ranking algorithm for web search, IEEE Trans. Knowl. Data Eng. 15 (4) (2003) 784–796.
[16] Y. Wu, M. Pei, M. Yang, J. Yuan, Y. Jia, Robust discriminative tracking via landmark-based label propagation, IEEE Trans. Image Process. 24 (5) (2015) 1510–1523.
[17] D. Wang, H. Lu, M.-H. Yang, Kernel collaborative face recognition, Pattern Recognit. 48 (10) (2015) 3025–3037.
[18] J.F. Henriques, R. Caseiro, P. Martins, J. Batista, High-speed tracking with kernelized correlation filters, IEEE Trans. Pattern Anal. Mach. Intell. 37 (3) (2015) 583–596.
[19] B. Schölkopf, A. Smola, Learning with kernels: support vector machines, regularization, optimization, and beyond, J. Am. Stat. Assoc. 16 (3) (2011). 781–781
[20] M. Tang, J. Feng, Multi-kernel correlation filter for visual tracking, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 3038–3046.
[21] P. Zhu, L. Zhang, W. Zuo, D. Zhang, From point to set: Extend the learning of distance metrics, in: Proceedings of the IEEE International Conference on Computer Vision, 2013, pp. 2664–2671.
[22] O. Pele, M. Werman, Fast and robust earth mover's distances, in: Proceedings of the IEEE International Conference on Computer Vision, IEEE, 2009, pp. 460–467.
[23] H. Zhang, Z.-J. Zha, Y. Yang, S. Yan, T.-S. Chua, Robust (semi) nonnegative graph embedding, IEEE Trans. Image Process. 23 (7) (2014) 2996–3012.
[24] Y. Yang, Z. Huang, Y. Yang, J. Liu, H.T. Shen, J. Luo, Local image tagging via graph regularized joint group sparsity, Pattern Recognit. 46 (5) (2013) 1358–1368.
[25] Y. Wu, J. Lim, M.H. Yang, Online object tracking: A benchmark, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2013, pp. 2411–2418.
[26] J.H. Yoon, M.-H. Yang, K.-J. Yoon, Interacting multiview tracker, IEEE Trans. Pattern Anal. Mach. Intell. 38 (5) (2016) 903–917.
[27] M. Everingham, L. Van Gool, C.K. Williams, J. Winn, A. Zisserman, The pascal visual object classes (voc) challenge, Int. J. Comput. Vis. 88 (2) (2010) 303–338.
[28] W. Zhong, H. Lu, M.H. Yang, Robust object tracking via sparse collaborative appearance model, IEEE Trans. Image Process. 23 (5) (2014) 2356–2368.
[29] G. Wang, X. Qin, F. Zhong, Y. Liu, H. Li, Q. Peng, M.-H. Yang, Visual tracking via sparse and local linear coding, IEEE Trans. Image Process. 24 (11) (2015) 3796–3809.