Discriminative Tracking Using Tensor Pooling

Bo Ma, Lianghua Huang, Jianbing Shen, *Senior Member, IEEE*, and Ling Shao, *Senior Member, IEEE*

***Abstract*—How to effectively organize local descriptors to build a global representation has a critical impact on the performance of vision tasks. Recently, local sparse representation has been successfully applied to visual tracking, owing to its discriminative nature and robustness against local noise and partial occlusions. Local sparse codes computed with a template actually form a three-order tensor according to their original layout, although most existing pooling operators convert the codes to a vector by concatenating or computing statistics on them. We argue that, compared to pooling vectors, the tensor form could deliver more intrinsic structural information for the target appearance, and can also avoid high dimensionality learning problems suffered in concatenation-based pooling methods. Therefore, in this paper, we propose to represent target templates and candidates directly with sparse coding tensors, and build the appearance model by incrementally learning on these tensors. We propose a discriminative framework to further improve robustness of our method against drifting and environmental noise. Experiments on a recent comprehensive benchmark indicate that our method performs better than state-of-the-art trackers.**

***Index Terms*—Discriminative, sparse representation, subspace, tensor pooling, tracking.**

I. Introduction

VISUAL object tracking is one of the fundamental areas in computer vision. It plays a critical role in numerous applications, including surveillance, video analysis, human-computer interaction, robotics, and intelligent transportation. Despite decades of studies [1]–[5], [55], tracking is still a challenging task due to the difficulty of handling lots of severe appearance variations.

Appearance and motion models are two essential components for building a robust tracker. Recent years have witnessed significant advances in both adaptive appearance modeling [6]–[12], [50], [51], [53], [54] and robust motion modeling [13]–[16] that improve the tracking performance. In this paper, we focus on the appearance model, which is closely related to the performance of a tracker against various challenging appearance variations. Ross *et al.* [10] presented an incremental visual tracker (IVT)-based on online principal component analysis (PCA) to account for appearance variations. Wang *et al.* [17] extended IVT with a Gaussian–Laplacian noise assumption, which is more effective in dealing with outliers. However, with flattened intensity vector representing a target, IVT results in a high-dimensional data learning problem. Moreover, converting two-order intensity images to one-order vectors could lose the spatial information. Li *et al.* [18] extended IVT to a higher order case, which tackles the tracking task in an online tensor subspace learning framework. In this way, the method is able to take spatial redundancies into consideration, and involves relatively low-computational and memory costs. However, the holistic template used in the above methods is still susceptible to local noise and transformation. In [6], [19], and [48], local integral histograms-based representations are proposed for the purpose of improving the robustness of trackers against partial occlusions and object transformation. Ali *et al.* [20] proposed to handle local noise by using salient feature points. The basic idea is that when objects are partially occluded, the unoccluded feature points are still available for distinguishing the target. Wang *et al.* [21] proposed a discriminative appearance model based on superpixels. The target and its surrounding are segmented into several superpixels, which are used for training a discriminative model online to distinguish foreground superpixels from background ones. Therefore, the method is effective in handling heavy occlusions and nonrigid transformations.

Recently, sparse representation has been attracting much attention in visual object tracking [8], [9], [22]–[24], [52], [56]. It provides an elegant model for representing candidates with very few but most related target templates to minimize the adverse impacts of background noise [8], [9], [23], [53]. In addition, it is also an adaptive model for representing the target appearance with local sparse codes to exploit the discriminative nature of sparse representation [24]–[28]. These two different kinds of sparse coding-based tracking methods could be categorized as target searching based on sparse representation (TSSR) and appearance modeling based on sparse coding (AMSC), respectively [22]. Experimental comparison [22] indicates that AMSC methods significantly outperform TSSR ones. Since most AMSC methods model the target appearance with local sparse codes, we call them local sparse representation (LSR)-based methods as a unified expression.

It is noteworthy that local sparse codes computed with a template constitute a three-order tensor, as shown in Fig. 1. For example, a template with $n_r \times n_c$ patches (n_r and n_c

Manuscript received May 15, 2015; revised July 28, 2015; accepted September 7, 2015. Date of publication September 28, 2015; date of current version October 13, 2016. This work was supported in part by the National Natural Science Foundation of China under Grant 61472036 and Grant 61272359, in part by the National Basic Research Program of China (973 Program) under Grant 2013CB328805, in part by the Fok Ying-Tong Education Foundation for Young Teachers, and in part by the Specialized Fund for Joint Building Program of Beijing Municipal Education Commission. This paper was recommended by Associate Editor J. Su. *(Corresponding author: Jianbing Shen.)*

B. Ma, L. Huang, and J. Shen are with the Beijing Laboratory of Intelligent Information Technology, School of Computer Science, Beijing Institute of Technology, Beijing 100081, China (e-mail: bma000@bit.edu.cn; shenjianbing@bit.edu.cn).

L. Shao is with the Department of Computer Science and Digital Technologies, Northumbria University, Newcastle upon Tyne NE1 8ST, U.K. (e-mail: ling.shao@northumbria.ac.uk).

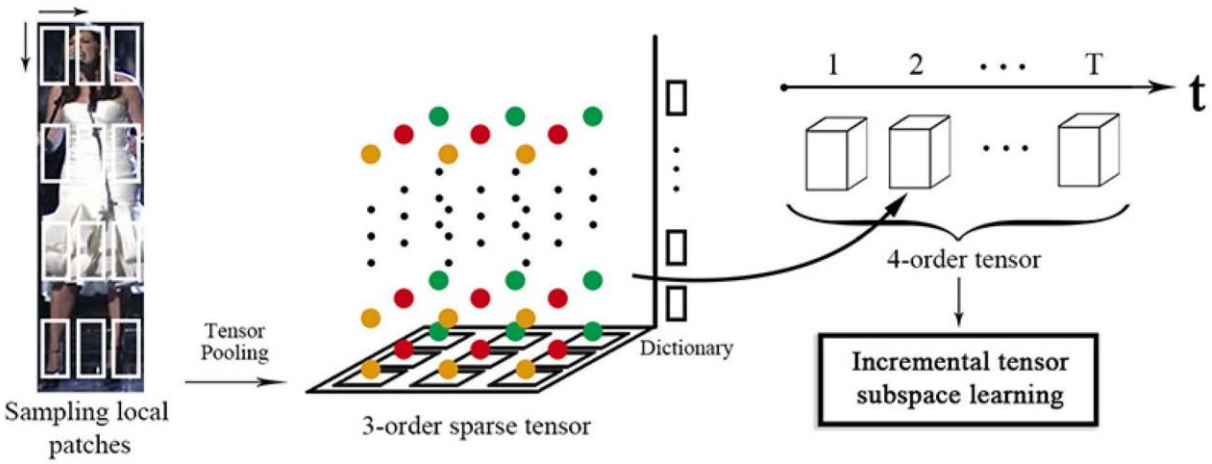


Fig. 1. Overview of our approach, which includes tensor pooling and online subspace learning.

are numbers of patches per row and per column in a target template, respectively) coded on a dictionary with L items would build a three-order tensor $J \in \mathbf{R}^{L\times n_r\times n_c}$. Almost all the LSR-based tracking methods convert the tensor to a vector by using a pooling operator. Max pooling [28] and average pooling [26], [27], [49] operators compute statistics of local sparse codes to yield the pooled representation. In this way, the dimensionality is significantly reduced, while they completely ignore the spatial arrangement of local patches, causing much loss of discriminability. On the other hand, concatenation pooling [9], [24] directly concatenates the local codes to a long vector, so as to preserve spatial orders of local codes. However, the concatenated feature results in a high dimensionality learning problem, which is difficult to handle.

Compared to pooling vectors, the original tensor form could deliver more intrinsic structural information. Taking coding tensor $J \in \mathbf{R}^{L\times n_r\times n_c}$ for an example, the first order vectors of J represent different local sparse codes, and the second and third orders of J can be interpreted as the row-wise and column-wise distributions of dictionary items on the target template, respectively. Furthermore, training with the tensor form could avoid high dimensionality learning problems. For instance, performing dimension reduction on a three-order tensor of size $50 \times 50 \times 50$ to size $10 \times 10 \times 10$ in a vector form needs to learn a $125\,000 \times 1000$ basis matrix. In a tensor form, however, the dimension reduction task only requires three 50×10 basis matrices, whose size is 1.2×10^{-5} of that of the vector form. In this paper, we propose a novel approach for constructing local descriptors named tensor pooling, which models a template as a three-order tensor feature. Our tensor pooling-based visual tracking algorithm represents the target or candidates with tensor-pooled sparse features, and considers visual tracking task as an online tensor learning problem.

The advantages of tensorial representation have attracted significant interest from vision researchers. Several works of tensor-based extensions of fundamental methods have been proposed, such as tensor decomposition [18], [29]–[31], multilinear PCA [32], linear discriminative analysis [33], support tensor machines [34], and canonical analysis correlation of tensors [35]. In visual tracking, the tensorial representation of the target has also been introduced recently [18], [36]–[38]. Since the subspace learning [10], [17], [18] methods could capture compact and informative appearance of an object and can be easily extended to online learning algorithms, which are suitable for tracking tasks, in this paper, we propose to address the tracking problem within an online tensor subspace learning framework.

The overview of our approach is shown in Fig. 1. Target appearance variations are captured by incremental subspace learning of pooling tensors. When predicting the target, the likelihood of a candidate is evaluated with reconstruction error norms of its pooling tensor to the learned subspace. To further improve the robustness of our approach against background clutter and drifting, the basic tracker is incorporated with a discriminative framework and a robust updating scheme. Our source code will be publicly available online.[1]

The rest of this paper is organized as follows. We first introduce the foundation of tensor and its decomposition algorithm in Section II, and then present details of our tracking algorithm in Section III. Experiment results and analysis are shown in Section IV. We conclude this paper in Section V.

II. Preliminaries

In this section, we give a brief introduction to the basic concept about tensor and its decomposition algorithms.

A. Tensor Fundamentals

A tensor can be regarded as a higher order generalization of a vector or a matrix, which is one-order or two-order tensor, respectively. We denote an N-order tensor as $\mathcal{A} \in \mathbf{R}^{I_1\times I_2\times\cdots\times I_N}$, and each of its elements as $a_{i_1\cdots i_n\cdots i_N}$, where $1 \le i_n \le I_n$. In tensor terminology, the n-mode matrix unfolding of tensor $\mathcal{A} \in \mathbf{R}^{I_1\times I_2\times\cdots\times I_N}$, which is denoted as $A_{(n)} \in \mathbf{R}^{I_n\times(\prod_{i/=n} I_i)}$, is obtained by varying index i_n while keeping the other indices fixed. The process of matrix unfolding of a three-order tensor is illustrated in Fig. 2 for better understanding.

[1] http://github.com/shenjianbing/tensortracking

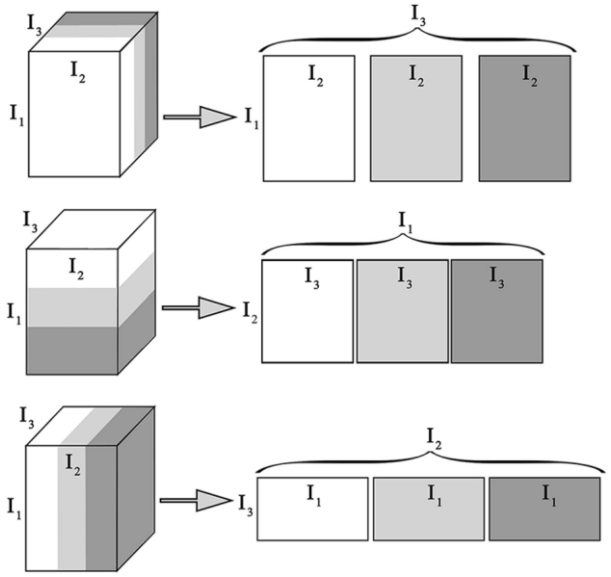


Fig. 2. Illustration of unfolding a third-order tensor in terms of different modes.

The inverse process of mode-n unfolding is mode-n folding, which can restore the original tensor $\mathcal{A}$ from the unfolded matrix $A_{(n)}$.

The mode-n product between a tensor $\mathcal{A}$ and a matrix $U \in \mathbf{R}^{J \times I_n}$ is of size $I_1 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N$ and defined element-wise as

$$(\mathcal{A} \times_n U)_{i_1 \cdots i_{n-1} j i_{n+1} \cdots i_N} = \sum_{i_n} a_{i_1 i_2 \cdots i_N} u_{j i_n}. \quad (1)$$

It can be better characterized in a metricized form

$$Y = \mathcal{A} \times_n U \Leftrightarrow \mathbf{Y}_{(n)} = \mathbf{U}\mathbf{A}_{(n)}. \quad (2)$$

The scalar product between two tensors $\mathcal{A}, \mathcal{B}$ is defined as

$$\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} a_{i_1 \cdots i_N} b_{i_1 \cdots i_N}. \quad (3)$$

The Frobenius norm of $\mathcal{A}$ is defined as $\|\mathcal{A}\| = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$. The mode-$n$ rank R_n of $\mathcal{A}$ is defined as $R_n = \text{rank}(A_n)$. For more notations or theories about tensor, we refer the reader to [30].

B. Low-Rank Approximation

An N-order tensor $\mathcal{A} \in \mathbf{R}^{I_1 \times I_2 \times \cdots \times I_N}$ can be approximated by a low-rank tensor $\mathcal{S} \in \mathbf{R}^{R_1 \times R_2 \times \cdots \times R_N}$ obtained by

$$\min_{\mathcal{S}, U^{(1)}, \ldots, U^{(N)}} \| \mathcal{A} - \mathcal{S} \times_1 U^{(1)} \times_2 \cdots \times_N U^{(N)} \| \quad (4)$$

where $U^{(n)} \in \mathbf{R}^{I_n \times R_n}$, $n = 1 \cdots N$ corresponds to the basis matrix of the n-mode unfolding matrix of $\mathcal{A}$ with $R_n \leq I_n$. $\mathcal{S}$ is the so-called core tensor [30]. Equation (4) can be readily solved by the Tucker decomposition (TD) technique [30]. Note that the eigen value matrix $D^{(n)}$ and row basis $V^{(n)}$ can also be obtained in the procedure of TD, which are useful for the incremental version of tensor subspace learning described in Section II-D.

Algorithm 1 Procedure of Incremental Updating of Eigenbasis and Mean

Input: *CVD* results $U_p D_p V_p^T$ of existing data X_p, new coming data X_q, column mean m_p of X_p and column mean m_q of X_q.

Output: Column mean m_r of $X_r = [X_p | X_q]$ and *CVD* results $U_r D_r V_r^T$ of X_r.

1: $n^r = f * n$;
2: Compute $m_r = \frac{n^r}{n^r+m} m_p + \frac{m}{n^r+m} m_q$, and $\hat{E} = (X_q - m_r \mathbf{1}_{1\times m} | \sqrt{\frac{n^r m}{n^r+m}}(m_p - m_q))$;
3: Compute R-SVD with $U(fD)V^T$ and $\hat{E}$ to obtain $U_r D_r V_r^T$.

C. Incremental Updating of Eigenbasis and Mean

We first introduce the incremental low-rank approximation in the vector case, and extend it to the tensor case in the next section. The R-SVD algorithm [39] is developed for sequentially computing SVD of a dynamic matrix as new data (columns or rows) arrive. However, this algorithm assumes a zero mean when updating the eigenbasis. Ross *et al.* [10] extended the work of classic R-SVD method. It updates the eigenbasis while taking the shift of the sample mean into account.

Let CVD(H) denote the SVD of a matrix H with the column mean removed. Let $X_p = [x_1, x_2, \ldots, x_n]$ denote the existing data, $X_q = [x_{n+1}, x_{n+2}, \ldots, x_{n+m}]$ denote the new coming data, and let $X_r = [X_p | X_q]$. Given the column means m_p of X_p, m_q of X_q, and the CVD results U_p, D_p, V_p of X_p, the column mean m_r of X_r and the CVD of X_r can be computed efficiently as follows.

1) Compute $m_r = n/(n+m)m_p + m/(n+m)m_q$, and $\hat{E} = (X_q - m_r \mathbf{1}_{1\times m} | \sqrt{(nm/n+m)}(m_p - m_q))$.
2) Compute R-SVD with U_p, D_p, V_p, and $\hat{E}$ to obtain U_r, D_r, V_r.

Levey and Lindenbaum [39] further introduced a forgetting factor f to put more weights on recent observations when updating the eigenbasis, i.e., $A^r = (fA|E) = (U(fD)V^T|E)$, where A and A^r are the original and weighted data matrices, respectively.

The above process could be depicted as

$$[U_r, D_r, V_r, m_r] = \mathbf{ICVD}(U_p, D_p, V_p, m_p, X_q, k, f) \quad (5)$$

where k is the number of preserved dominant bases and f denotes the forgetting factor. The incremental CVD procedure is summarized in Algorithm 1. The analytical proofs of R-SVD and ICVD are given in [10] and [39].

D. Incremental Tensor Subspace Learning

The proposed tracking algorithm actually learns a four-order tensor subspace over time, but for better understanding, here we take the three-order case as an example, which can be easily extended to four-order case. We apply the incremental rank $-(R_1, R_2, R_3)$ tensor subspace analysis (IRTSA) [18] algorithm to update the tensor subspace. Let $\mathcal{A} \in \mathbf{R}^{I_1 \times I_2 \times I_3}$ denote the original tensor, $\mathcal{F} \in \mathbf{R}^{I_1 \times I_2 \times I_3^r}$ denote the new subtensor and $\mathcal{A}^* = (\mathcal{A}|\mathcal{F}) \in \mathbf{R}^{I_1 \times I_2 \times I_3^*}$ denote the merged tensor

Algorithm 2 IRTSA Algorithm

Input: Tensor decomposition results $U^{(k)}D^{(k)}V^{(k)}(1 \le k \le 3)$ of the original tensor A, column mean $M^{(1)}$, $M^{(2)}$ of $A_{(1)}$, $A_{(2)}$ and row mean $M^{(3)}$ of $A_{(3)}$, newly added tensor F, and preserved ranks R_1, R_2, and R_3, forgetting factor f.

Output: Tensor decomposition results $\hat{U}^{(k)}\hat{D}^{(k)}\hat{V}^{(k)}(1 \le k \le 3)$ of $A^* = (A|F)$, column mean $\hat{M}^{(1)}$, $\hat{M}^{(2)}$ of $A^*_{(1)}, A^*_{(2)}$ and row mean $\hat{M}^{(3)}$ of $A^*_{(3)}$

1: $[\hat{U}^{(1)}, \hat{D}^{(1)}, \hat{V}^{(1)}, \hat{M}^{(1)}]$ $= \textbf{ICVD}(U^{(1)}, D^{(1)}, V^{(1)}, M^{(1)}, F_{(1)}, R_1, f)$;

2: $[\hat{U}^{(2)}, \hat{D}^{(2)}, \hat{V}^{(2)}, \hat{M}^{(2)}]$ $= \textbf{ICVD}(U^{(2)}, D^{(2)}, V^{(2)}, M^{(2)}, F_{(2)}, R_2, f)$; $\hat{V}^{(2)} = P^T \cdot \hat{V}^{(2)}$.

3: $[\hat{U}^{(3)}_0, \hat{D}^{(3)}_0, \hat{V}^{(3)}_0, \hat{M}^3_0]$ $= \textbf{ICVD}(U^{(3)}, D^{(3)}, V^{(3)}, M^{(3)}, F^T_{(3)}, R_3, f)$; $\hat{U}^{(3)} = V^{(3)}$, $\hat{V}^{(3)} = \hat{U}^{(3)}_0$, $\hat{D}^{(3)} = (\hat{D}_{(3)})^T$, $\hat{M}^3 = (\hat{M}_0)^T$.

with $I^*_3 = I_3 + I'_3$. Based on TD, the task of IRTSA is to learn the dominant subspace of A^* incrementally, given the TD results of A and subtensor F. With the mergence of the new subtensors, the column spaces of $A^*_{(1)}, A^*_{(2)}$ are extended at the same time when the row space of $A^*_{(3)}$ is extended. The idea of IRTSA is to update $U^{(k)}D^{(k)}V^{(k)^T}(k = 1, 2, 3)$ with new columns or rows of corresponding unfolding matrices using the ICVD algorithm. The procedure of the IRTSA algorithm is listed in Algorithm 2.

III. Proposed Tracking Algorithm

In this section, we present the proposed algorithm in details. Our tracking algorithm is implemented under the particle filter framework [13], which will be introduced first in Section III-A. Its appearance model and likelihood function $p(Y_t|X_t)$ are described in the following sections.

A. Bayesian Framework

Let $X_t = \{x_t, y_t, \theta_t s_t, \beta_t, \varphi_t\}$ denotes the state of a target at time t, where x_t, y_t, $\theta_t s_t$, β_t, φ_t are the x, y translations, the rotation angle, the scale, the aspect ratio and the skew, respectively. Given a set of observations $Y_t = \{Y_1, Y_2, \ldots, Y_t\}$ at the tth frame, the posterior probability is estimated recursively as

$$p(X_t|Y_t) \propto p(Y_t|X_t) \int p(X_t|X_{t-1})p(X_{t-1}|Y_{t-1})dX_{t-1} \quad (6)$$

where $p(X_t|X_{t-1})$ represents the dynamic model between two consecutive states, and $p(Y_t|X_t)$ denotes the observation model that estimates the likelihood of observing Y_t at state X_t. A particle filter [13] is applied for approximating the distribution over the location of the target and producing a set of samples. The optimal state of the target in the current frame given all the observations up to the tth frame is obtained by the maximum a posteriori estimation over these samples by

$$\hat{X}_t = \arg\max_{X^i_t} p(Y^i_t|X^i_t)p(X^i_t|X_{t-1}), \quad i = 1, 2, \ldots, N \quad (7)$$

where N is the number of samples, X^i_t denotes the ith sample of state X_t.

The state transition is formulated by random walk, i.e., $p(X_t|X_{t-1}) = N(X_t; X_{t-1}, W)$, where W is a diagonal covariance matrix whose diagonal elements are $\sigma^2_x, \sigma^2_y, \sigma^2_\theta, \sigma^2_s, \sigma^2_\beta, \sigma^2_\varphi$, respectively. The likelihood function in the observation model $p(Y_t|X_t)$ will be discussed in the following sections.

B. Tensor Pooling

The proposed tracking algorithm applies local sparse codes for object representation. We use overlapped sliding windows on the warped image to obtain $n_r \times n_c$ patches, where n_r and n_c are the numbers of them in each column and row, respectively. Patches are first converted to intensity vectors, and then normalized to make the features more robust against illumination variations. Each of them is represented by a vector $\mathbf{y} \in \mathbf{R}^{G\times 1}$, where G denotes the patch size. The sparse coefficient vector β of each patch is computed by

$$\min_{\beta_i} \|\mathbf{y}_i - \mathbf{D}\beta_i\|^2_2 + \lambda\|\beta_i\|_1 \quad (8)$$

where $\mathbf{D} \in \mathbf{R}^{G\times L}$ is the dictionary learned by k-means++ [40] using patches extracted from the target in the first frame, L is the number of cluster centers.

Unlike most LSR-based methods which either compute statistics of these sparse codes or concatenate them to a long vector, we organize the sparse codes according to their corresponding positions in the target template. Therefore, these $n_r \times n_c$ sparse coefficient vectors with dimension L form a three-order sparse tensor $A \in \mathbf{R}^{L\times n_r\times n_c}$. In this way, which we call tensor pooling, the entire spatial arrangement information of local patches is well preserved.

C. Online Learning and Similarity Evaluation

Each tracking result corresponds to a three-order sparse pooling tensor. Supposing t frames have been tracked, then a four-order tensor $\mathcal{T} \in \mathbf{R}^{L\times n_r\times n_c\times t}$ is constructed with the mergence of these three-order tensors. Let $I_1 = L$, $I_2 = n_r$, $I_3 = n_c$, $I_4 = t$, then $\mathcal{T} \in \mathbf{R}^{I_1\times I_2\times I_3\times I_4}$. We decompose tensor $\mathcal{T}$ incrementally with the IRTSA algorithm described in Section II-D to capture the target appearance variations.

To evaluate a test tensor $\mathcal{J} \in \mathbf{R}^{I_1\times I_2\times I_3\times 1}$ computed with a candidate in a new frame, we define two different reconstruction error norms as

$$\mathrm{RE}_1 = \|(\mathcal{J} - \mathcal{M}) - (\mathcal{J} - \mathcal{M}) \times_{j=1}^{3} U^{(j)} \cdot U^{(j)^T}\|^2$$

$$\mathrm{RE}_2 = \|J_{(i)} - M_{(i)}\| - \|J_{(i)} - M_{(i)} \cdot V^{(4)} \cdot V^{(4)^T}\|^2 \quad (1 \le i \le 3) \quad (9)$$

where $J_{(4)}$ is the mode-4 unfolding matrix of $\mathcal{J}$, $\mathcal{M}$ is the mean tensor of pooling tensors and $M_{(4)}$ is the row mean of the mode-4 unfolding matrix $\mathcal{T}_{(4)}$. The variation cost is computed as $\mathrm{Cost} = \gamma\,\mathrm{RE}_1 + (1-\gamma)\mathrm{RE}_2$, where γ is a weighted parameter.

Algorithm 3 Proposed Tracking Algorithm

Input: Initialized dictionary D and positive subspace parameters; frames: $F_{k+1},\ldots, F_n$; object state s_{k+1}.

Output: Tracking results $\hat{s}_t$ at time t.

for $t = k+1 \rightarrow n$ **do**
  Sample candidate states $p^{(1)},\ldots, p^{(m)}$ with particle filter around state $\hat{s}_{t-1}$;
  for $i = 1 \rightarrow m$ **do**
    $Y^{(i)}$ = ExtractPatches($F_t, p^{(i)}$);
    $\mathcal{J}^{(i)}$ = TensorPooling($Y^{(i)}, D$);
    Evaluate likelihood $l^{(i)}$ with $\mathcal{J}^{(i)}$ and positive and negative subspace parameters using (9) and (10);
  end for
  $[maxval, maxid] = max(l^{(1)},\ldots, l^{(m)})$;
  $\hat{s}_t = p^{(maxid)}$;
  if $maxval > \gamma$ **then**
    Collect $\mathcal{J}^{(maxid)}$ as a positive sample for updating;
    $N_{pos} = N_{pos} + 1$;
    if $N_{pos} == T$ **then**
      Update positive subspace with *IRTSA* algorithm;
      $N_{pos} = 0$;
    end if
  end if
  Sample negative images and obtain their pooling tensors $\mathcal{J}^{(1)} \sim \mathcal{J}^{(s)}$;
  Learn negative subspace with $\mathcal{J}^{(1)} \sim \mathcal{J}^{(s)}$ using *TD* algorithm.
end for

Consequently, the likelihood function is defined as

$$p(Y_t|X_t) \propto \exp(-\text{Cost}). \quad (10)$$

D. Discriminative Framework

Recently, the discriminative framework has been successfully utilized in many tracking-by-detection algorithms. By taking negative samples into account in the appearance modeling process, the tracker is able to distinguish the object from the background and exclude shifted samples to avoid drifting.

In this paper, motivated by the success of discriminative models in visual tracking, we propose a discriminative framework to further improve the performance. The positive samples are collected one per frame using the pooled tensor features of tracking results in the tracked frames, and the positive subspace is learned incrementally using the IRTSA algorithm as described above, while the negative samples are collected only in the last tracked frame by extracting patches several pixels away from the estimated target location. The negative tensor subspace is learned directly via the TD algorithm by pooling tensors of these negative samples.

After positive and negative models are trained, the costs of a candidate tensor on both subspaces are computed to tell how similar the candidate is to be a positive or negative sample. Finally, the likelihood of each candidate tensor is evaluated as

$$p(Y_t|X_t) \propto \exp\left(\text{Cost}^{(+)} - \text{Cost}^{(-)}\right) \quad (11)$$

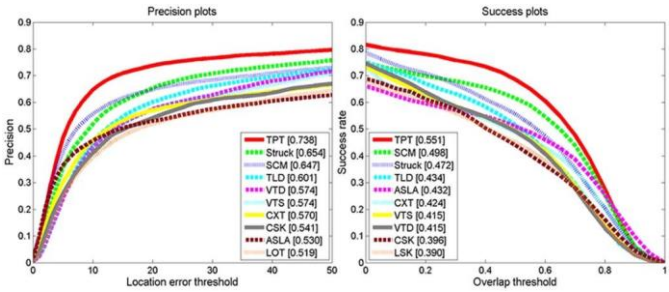


Fig. 3. Overall performance of compared trackers on the 51 sequences. Only the top 11 trackers are displayed for clarity.

where $\text{Cost}^{(+)}$ and $\text{Cost}^{(-)}$ are the costs computed on positive and negative tensor subspaces, respectively.

E. Update Scheme

During tracking, the holistic appearance of the target may change a lot, but its normalized local appearance will roughly remain the same as in the first frame. Therefore, we keep the overcomplete dictionary learned in the first frame fixed throughout the tracking process, which captures most local patterns of the target and can also avoid being deteriorated by tracking failure.

On a fixed dictionary, the target appearance variation is adapted by incrementally updating the subspace of pooled tensors. In the incremental updating procedure, we do not take all the estimated results for updating, which are vulnerable to background clutter, occlusions or tracking failure. Instead, under the discriminative framework, we collect the estimated candidates which are more likely to be positive samples. That is, $\text{Cost}^{(-)} > \text{Cost}^{(+)} + \epsilon$, where ϵ is the threshold parameter. Therefore, the likelihood is updated as

$$p(Y_t|X_t) \propto \exp\left(\text{Cost}^{(+)} - \text{Cost}^{(-)}\right) > \exp(\epsilon). \quad (12)$$

The occluded or missed samples are more similar to negative samples, whose likelihoods are smaller. The above mentioned updating scheme is able to exclude ill samples when updating. Supposing the first few k frames (in our experiments, $k = 5$) have been tracked using a simple tracking approach. The positive subspace is initialized via the TD algorithm with k corresponding pooled tensors. The main tracking procedure for the remaining frames is shown in Algorithm 3.

IV. Experiments

To evaluate the performance of our tracker, we follow the protocol suggested in a recent benchmark [5]. Our tracker is tested on 51 challenging sequences and compared with 29 recent state-of-the-art trackers with their results publicly available on the benchmark [5]. Some of the state- of-the-art trackers are: struck tracker [7], SCM tracker [9], TLD tracker [41], VTD tracker [42], CT tracker [43], ASLA tracker [44], CSK tracker [45] and IVT tracker [10].

A. Implementation Details

The proposed tracker is implemented in MATLAB R2013a on an Intel Core i7-2600 3.4 GHz CPU with 8 GB memory.

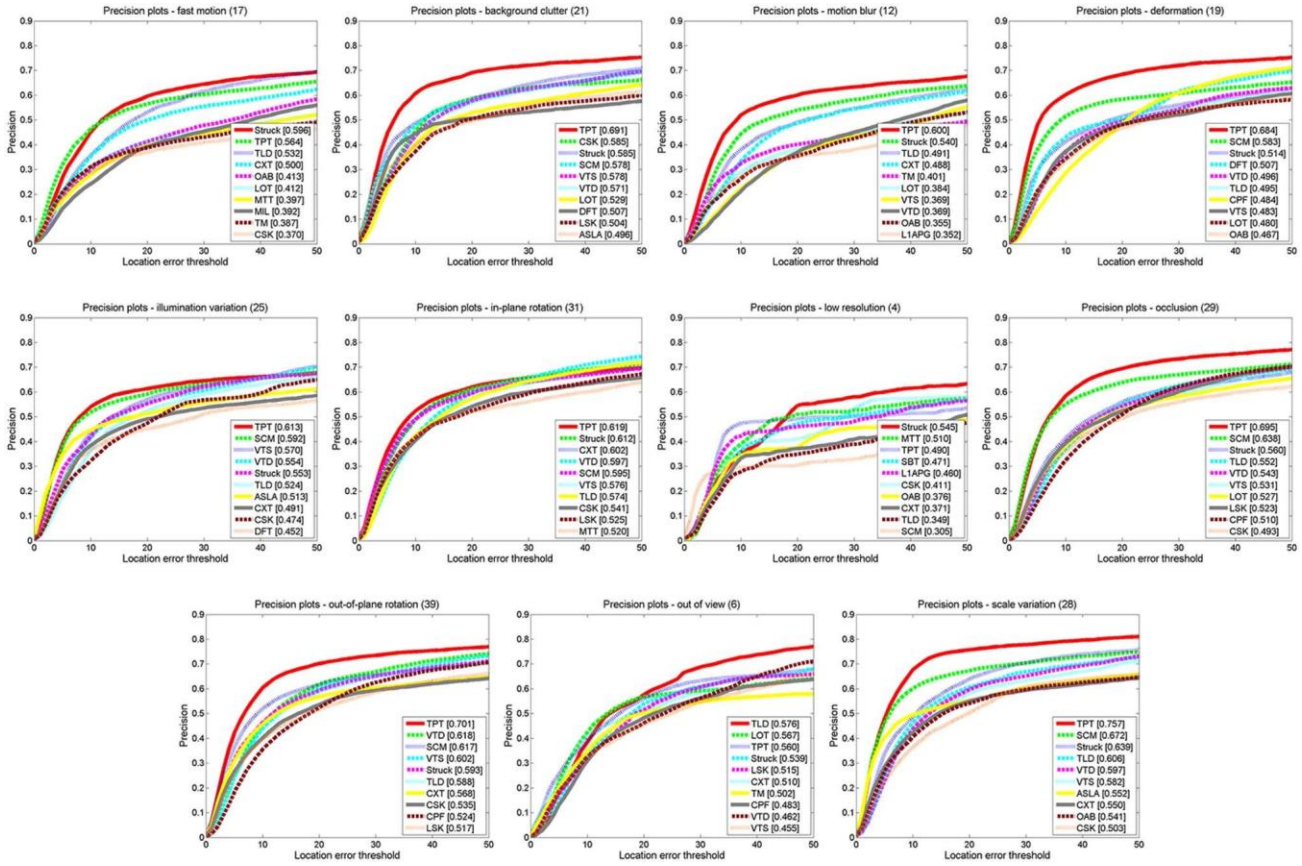


Fig. 4. Precision plots for different challenges namely: background clutter, deformation, fast motion, in-plane rotation, illumination variation, low resolution, motion blur, occlusion, out-of-plane rotation, out-of-view, and scale variation. Our approach performs favorably on 8 out of 11 challenges in terms of precision.

Templates are normalized to 32×32, and the patch size is set to 6×6. Dictionary is learned by k-means++ [40] with $c = 50$ cluster centers. Sparse coding is implemented with the SPAMs package [46]. The number of negative samples is set to 200, and positive ones are obtained one per frame. We set the update frequency to 5 and update threshold τ to 1.0. We use tensor class and TD codes provided publicly by [47]. The preserved ranks of the positive subspace are set as [40 10 10 min$(t, 25)$], where t is the number of tracked frames, and those of the negative subspace are set as [40 10 10 25]. Under the particle framework, the particle number is set as 600. In most videos of the benchmark, only slight shifts of targets exist in consecutive frames, instead of abrupt motions. Therefore, we choose the appropriate values [10 10 0.015 0 0 0] for the affine parameters empirically. The λ in (8) is set to 0.01 and the weighted parameter γ with respect to RE_1 and RE_2 is set to 0.33. All these parameters are fixed on all the tested sequences for fair comparison.

B. Quantitative Evaluation

1) Evaluation Criteria: The precision plots and success plots [5] are applied to evaluate the robustness of trackers. A precision plot indicates the percentage of frames whose estimated location is within the given threshold distance to the ground truth. A success plot demonstrates the ratio of successful frames whose overlap rate is larger than the given threshold. The precision score is given by the score on a selected representative threshold (e.g., 20 pixels). The success score is evaluated by the area under curve (AUC) of each tracker. For clarity, only 11 top trackers are illustrated for each metric.

2) Overall Performance: The overall performances of the top 11 trackers on the 51 sequences are evaluated with the success plots and precision plots, as shown in Fig. 3. For the precision plots, the results at an error threshold of 20 pixels are used for ranking, and for the success plots, we use AUC scores to rank the trackers. The performance score of each tracker is shown in the legend of Fig. 3.

It is observed from Fig. 3 that struck, SCM, and our tracker achieve good tracking performance. In the precision plots, our tracker outperforms struck by 12.08% and outperforms SCM by 13.12%. In the success plots, our tracker performs 9.62% better than SCM and 16.74% better than struck. Overall, our tracker outperforms the state-of-the-art trackers in terms of location accuracy and overlap precision.

The robustness of our tracker could be attributed to the following. First, the discriminative nature of local sparse codes ensures its effectiveness in visual tracking. Compared with pooling vectors, pooling tensors can deliver more discriminative, structural information of the target, which improves the robustness and accuracy of our approach. Second, the incremental tensor subspace learning algorithm could elegantly learn the main patterns of the target appearance and capture its variations timely. Third, the proposed discriminative framework and the robust updating scheme further improve

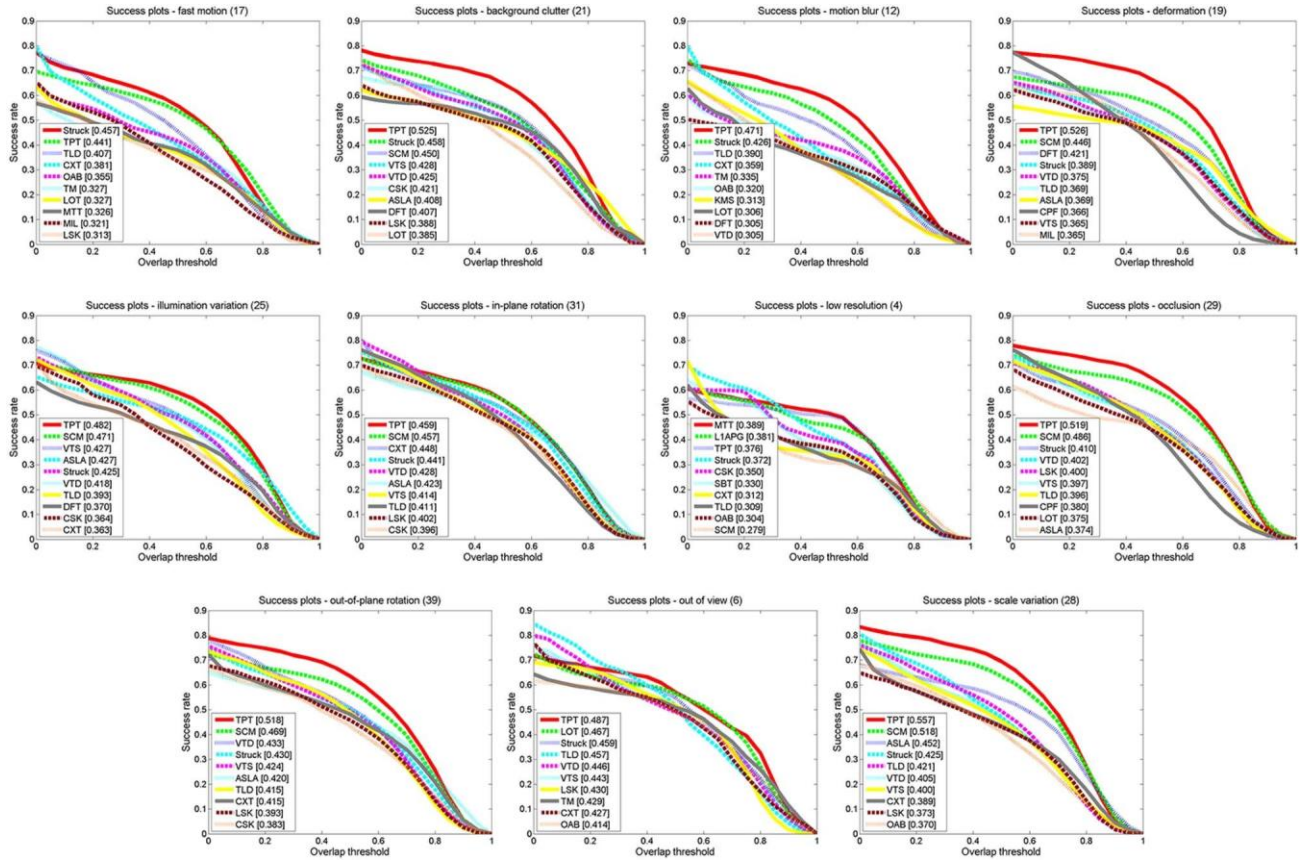


Fig. 5. Success plots for different challenges namely: background clutter, deformation, fast motion, in-plane rotation, illumination variation, low resolution, motion blur, occlusion, out-of-plane rotation, out-of-view, and scale variation. Our approach performs favorably on 9 out of 11 challenges in terms of success.

our approach against challenging factors in tracking such as occlusion, background noise, and drifting.

3) Performance Per Challenge: Several factors can affect the performance of a visual tracker. In the recent benchmark [5], the 51 sequences are annotated with 11 different challenges that may affect tracking performance. Our approach performs favorably on 8 out of 11 challenges in terms of precision plots and 9 out of 11 challenges in terms of success plots. Figs. 4 and 5 illustrate the precision plots and success plots on all the 11 challenges.

On the background clutter subset, our tracker achieves the best performance, which can be attributed to the informative tensor pooling method and the discriminative framework to avoid drifting. On the motion blur subset, both struck and our method outperform others. On the out-of-plane rotation subset, our tracker provides outstanding performance, which may be benefited from the subspace learning framework to capture the main patterns of the target appearance and adapt to the new patterns timely. On the deformation and illumination variation subset, both SCM and our approach perform better than other trackers, which indicates that local representation methods are effective in dealing with transformations, and that normalized intensity features are less vulnerable to illumination changes.

C. Comparison Between Different Pooling Operators

In this section, we qualitatively and quantitatively compare the tracking performance of trackers based on different

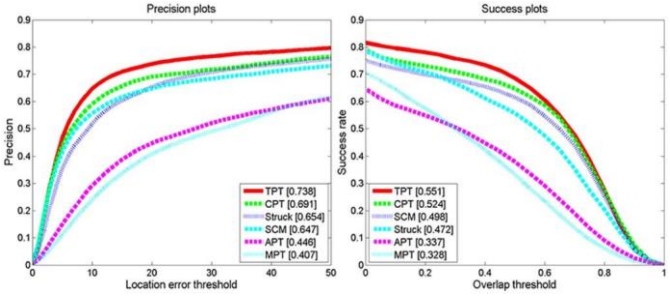


Fig. 6. Overall performance of trackers with different pooling operators on the 51 sequences. The trackers include APT, MPT, CPT, and TPT.

pooling operators. Four trackers, i.e., the tensor pooling-based tracker (TPT), the average pooling-based tracker (APT), the max pooling-based tracker (MPT), and the concatenation pooling-based tracker (CPT), are implemented under the same tracking framework as described in previous sections. The only few differences of these four trackers are the pooling operators and the corresponding subspace learning schemes. For vector pooling-based methods, the incremental PCA is applied to update the target subspaces, and the number of bases is set to $4 = 25$. For the proposed tensor pooling-based method, we use the IRTSA algorithm to learn and update the tracking model, and the preserved ranks are set to $[40, 5, 5, 4]$.

1) Overall Evaluation: The four trackers are tested on 51 benchmark sequences [5] and evaluated with the overall

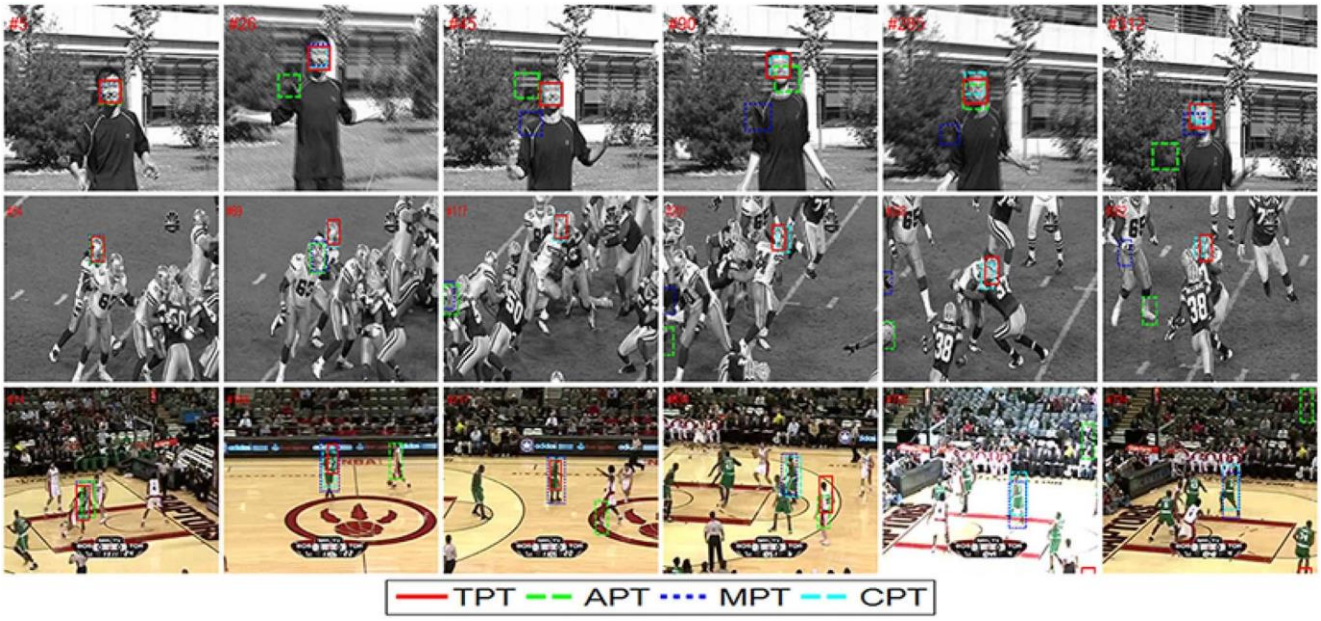


Fig. 7. Visual comparison of four different pooling operators-based trackers on sequences *Jumping*, *Football*, and *Basketball*. In *Jumping*, the target is under severe motion blur. In some frames of *Football*, the background objects are very close to the target. In *Basketball* which is a failure case of our proposed algorithm, the nonrigid transformation is the main challenge.

success plots and precision plots. Two popular trackers, the struck [7] and SCM [9], which are the recent state-of-the-art trackers, are added for comparison. The evaluation results are shown in Fig. 6. As we can see from Fig. 6, TPT and CPT significantly outperforms APT and MPT, and even performs better than struck and SCM, which indicates that the sufficient information contained in the corresponding two kinds of pooling features make them more discriminative and obtain favorable results. TPT obtains much better results than CPT, which can be attributed to the intrinsic structural information exploited in the tensor pooling operator and the corresponding tensor subspace learning scheme. It is also observed from the plots that APT achieves better performance than MPT, which may be because that the APT averages all the local features while MPT only takes a value from one of the local features for each dimension, and the ignorance of most features degrades its performance. Overall, the proposed tensor pooling operator is shown to have superior properties against the other three vector pooling operators.

2) Visual Comparison: To compare the four pooling operators intuitively, we select visual tracking results on some representative sequences, as shown in Fig. 7. In sequence *Jumping*, the target is under severe motion blur caused by the fast motion of the jumping boy. The average pooling and MPTs fail to track the sequence, which drift several pixels away or mistakenly lock to the background. The CPT performs better, but it still shifts away or estimates the scale inaccurately in some frames. Our proposed tensor pooling-based method successfully lock the target till the end and achieves the best performance. In sequence *Football*, the background contains objects that are very similar to the target, which makes the average pooling and MPTs incorrectly locks to the background. The CPT drifts away when occlusion occurs. The TPT produces promising results. The visual comparison results on the two sequences indicate that, in complex situations such as background clutter, motion blur, and similar objects interference, the tensor pooling-based method is more capable of exploiting discriminative information to identify the target.

Sequence *Basketball* is a failure case of our method but a successful example of max pooling and concatenation pooling-based methods. The player rotates its body and changes his poses frequently, and is sometimes occluded by other players. The max pooling-based method performs well on this sequence, though sometimes shakes around the target center. This is because that the ignorance of spatial information of local features makes the method invariant to nonrigid transformations. The concatenation pooling-based method, although not so accurate, successfully tracks the target. Our method, however, fails in this sequence. It is mainly because the structural information of local features contained in pooled tensors limits its flexibility in handling severe target deformations.

D. Qualitative Evaluation

In this section, we present a qualitative evaluation of the tracking results. Twelve representative sequences with different challenges are selected from the 51 sequences. The four dominant challenges of these sequences are occlusion, illumination variation, object deformation, and out-of-plane rotation. Figs. 8–10 show some screenshots of the tracking results of our tracker and some competitive state-of-the art trackers.

1) Occlusion: Occlusion is one of the most general and critical challenges in visual tracking. Fig. 8 illustrates tracking results on three representative sequences (i.e., *Walking2*, *Jogging1*, and *Woman*) where objects are severely or long-term occluded. In the *Walking2* sequence, the walking woman is occluded by a man over a long term (e.g., #187–#235 and #369–#378), and the similarity between

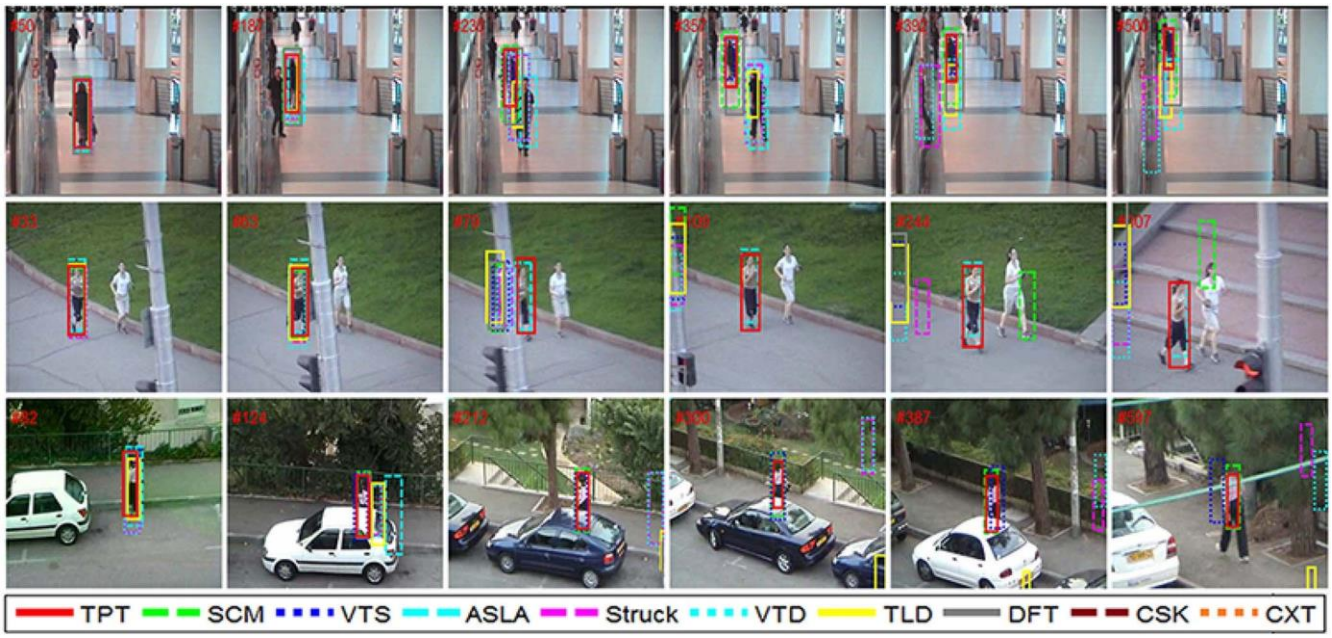


Fig. 8. From top to bottom are representative results of trackers on sequences *Walking2*, *Jogging1*, and *Woman*, where objects are heavily occluded.

Fig. 9. From top to bottom are representative results on sequences *Car4*, *Fish*, and *Trellis*, where objects suffer from significant illumination variations.

the target and the background makes the tracking more challenging. The ASLA, struck, VTD, and TLD methods miss the walking man when occlusion occurs (e.g., #235–#357). SCM and DFT fails to track the scale variation of the woman. Only VTS and Our method accurately track the target till the end. In the *Jogging1* sequence, there is a short-term complete occlusion for the person (e.g., #73) as well as scale variation. Most trackers lock to the obstacle during occlusion. Only TLD and our tracker are able to reacquire the object and track the person to the end of the sequence. The robustness of TLD against complete occlusion is because of its reinitialization scheme, while the robust updating scheme in our method also ensures that our model will not be contaminated by missed objects. In the *Woman* sequence, the person undergoes frequent long-term occlusions by cars. The TLD, VTD, VTS, CSK, and ASLA methods drift away from the target when the person gets occluded. Only struck, SCM and our tracker successfully lock the target for the whole sequence. The robustness of our method against occlusion could be attributed to the robust updating scheme to prevent the model from being contaminated by obstacles.

2) Illumination Variations: Fig. 9 shows tracking results on three challenging sequences (i.e., *Car4*, *Fish*, and *Trellis*), where objects undergo significant illumination changes.

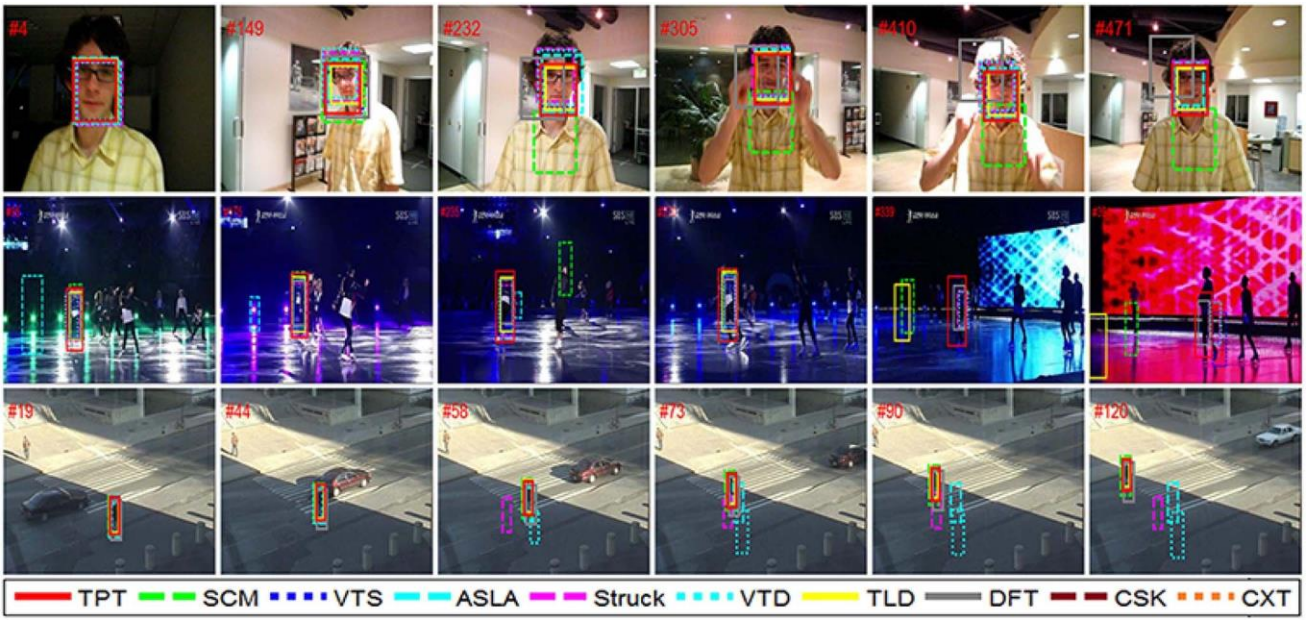


Fig. 10. From top to bottom are representative results on sequences *David*, *Skating1*, and *Crossing*. Object deformation is the main challenge of these sequences.

In sequence *Car4*, the target undergoes drastic illumination variation when the car runs under the tree shade (e.g., #5 and #430) and passes through a bridge (e.g., #200). Along with the illumination changes are scale variations. It is difficult to handle both of these two challenges. TLD, VTD, and VTS drift away when the car goes under the arch of the bridge, and struck and CSK fail to track the scale variations. ASLA, SCM, and our tracker achieve the best accuracies in this sequence. In sequence *Fish*, the light condition of object changes and the camera moves fast. CSK drifts away and VTD and VTS methods perform unstable and shake around the target position. Struck, ASLA, and our method obtain more accurate tracking results. In sequence *Trellis*, the person walks from a dark place to a sunny outdoor environment, where the target undergoes drastic illumination variations. The out-of-plane rotation and background clutter cast more difficulties in tracking. Most trackers drift away when the illumination condition changes, and some of their tracking results shake away frequently. Struck and our method lock the target more stably than others, and our tracker obtains the best accuracy. The robustness of our tracker against illumination variations can be attributed to the normalized local intensity features, since they capture local contrast information that is not susceptible to global light condition changes. The discriminative nature of sparse codes further improves the robustness of our method.

3) Object Deformation: As shown in Fig. 10, sequences *David*, *Skating1* and *Crossing* are selected to evaluate the robustness of trackers against nonrigid object deformation. In the *David* sequence, the person changes the orientation of his face over time, and the varying illumination also makes the tracking harder. SCM and DFT fail to lock the target. Struck, VTS and our tracker achieve the best performance. In the *Skating1* sequence, the dancer continuously changes her pose under drastic illumination variations and a complex background. Struck, TLD, SCM, VTD, ASLA, and VTS gradually drift away when the target's pose and light condition change. CSK and our method successfully track the sequence and achieve the most stable performance. In the *Crossing* sequence, the walking person moves from a shadow area to a bright one. Nonrigid deformation and drastic illumination variation are the main challenges when performing tracking on this sequence. TLD, VTD, and VTS lose the target when the target passes through the dark area. ASLA, struck, SCM, and our method successfully track the person till the end and our tracker obtains the highest accuracy. The reason that our tracker performs well on these three sequences can be explained as follows. First, the flexibility of the local appearance model makes our method less susceptible to object transformation. Second, the incremental subspace learning algorithm is able to capture main patterns of most target poses and adapts to the new ones timely. Third, the discriminative framework further improves the tracking performance of our tracker.

V. Conclusion

In this paper, we have proposed to represent the target and candidates by pooling sparse tensors, and formulate tracking as an online tensor subspace learning problem in a discriminative framework. The pooled tensors could deliver more informative and structured information, which potentially enhances the discriminative power of the appearance model and improves the tracking performance. The proposed robust updating scheme also proves to be effective in avoiding introducing tracking failures into model updating. Experiments on a recent comprehensive benchmark with 29 state-of-the-art trackers demonstrate the effectiveness and robustness of our tracker. More comparison results between different pooling operators demonstrate the superior performance of the tensor

pooling approach. As a general method for building global representation with local descriptors, we believe the tensor pooling scheme can be extended in a wide range of vision tasks such as object detection and pose estimation, where local appearance model should be further exploited.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, 2006, Art. ID 13.

[2] Z. H. Khan and I. Y.-H. Gu, "Nonlinear dynamic model for visual object tracking on Grassmann manifolds with partial occlusion handling," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2005–2019, Dec. 2013.

[3] T. Bai, Y.-F. Li, and X. Zhou, "Learning local appearances with sparse representation for robust and fast visual tracking," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 663–675, Apr. 2015.

[4] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song, "Recent advances and trends in visual tracking: A review," *Neurocomputing*, vol. 74, no. 18, pp. 3823–3831, Nov. 2011.

[5] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE CVPR*, Portland, OR, USA, 2013, pp. 2411–2418.

[6] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Proc. IEEE CVPR*, New York, NY, USA, 2006, pp. 798–805.

[7] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. IEEE ICCV*, Barcelona, Spain, 2011, pp. 263–270.

[8] X. Mei and H. Ling, "Robust visual tracking using l1 minimization," in *Proc. IEEE ICCV*, Kyoto, Japan, 2009, pp. 1436–1443.

[9] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Proc. IEEE CVPR*, Providence, RI, USA, 2012, pp. 1838–1845.

[10] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, May 2008.

[11] J. Fan, X. Shen, and Y. Wu, "Scribble tracker: A matting-based approach for robust tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 8, pp. 1633–1644, Aug. 2012.

[12] J. Ning, L. Zhang, D. Zhang, and C. Wu, "Robust mean-shift tracking with corrected background-weighted histogram," *IET Comput. Vis.*, vol. 6, no. 1, pp. 62–69, Jan. 2012.

[13] M. Isard and A. Blake, "Contour tracking by stochastic propagation of conditional density," in *Proc. ECCV*, Cambridge, U.K., 1996, pp. 343–356.

[14] X. Zhou, Y. Lu, J. Lu, and J. Zhou, "Abrupt motion tracking via intensively adaptive Markov-chain Monte Carlo sampling," *IEEE Trans. Image Process.*, vol. 21, no. 2, pp. 789–801, Feb. 2012.

[15] M. K. Lim, C. S. Chan, D. Monekosso, and P. Remagnino, "Refined particle swarm intelligence method for abrupt motion tracking," *Inf. Sci.*, vol. 283, pp. 267–287, Nov. 2014.

[16] C. Yoon, M. Cheon, and M. Park, "Object tracking from image sequences using adaptive models in fuzzy particle filter," *Inf. Sci.*, vol. 253, pp. 74–99, Dec. 2013.

[17] D. Wang, H. Lu, and M.-H. Yang, "Online object tracking with sparse prototypes," *IEEE Trans. Image Process.*, vol. 22, no. 1, pp. 314–325, Jan. 2013.

[18] X. Li, W. Hu, Z. Zhang, X. Zhang, and G. Luo, "Robust visual tracking based on incremental tensor subspace learning," in *Proc. IEEE Int. Conf. Comput. Vis.*, Rio de Janeiro, Brazil, 2007, pp. 1–8.

[19] S. He, Q. Yang, R. W. H. Lau, J. Wang, and M.-H. Yang, "Visual tracking via locality sensitive histograms," in *Proc. IEEE CVPR*, Portland, OR, USA, 2013, pp. 2427–2434.

[20] M. M. N. Ali, M. Abdullah-Al-Wadud, and S.-L. Lee, "Multiple object tracking with partial occlusion handling using salient feature points," *Inf. Sci.*, vol. 278, pp. 448–465, Sep. 2014.

[21] S. Wang, H. Lu, F. Yang, and M.-H. Yang, "Superpixel tracking," in *Proc. IEEE ICCV*, Barcelona, Spain, 2011, pp. 1323–1330.

[22] S. Zhang, H. Yao, X. Sun, and X. Lu, "Sparse coding based visual tracking: Review and experimental comparison," *Pattern Recognit.*, vol. 46, no. 7, pp. 1772–1788, Jul. 2013.

[23] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Minimum error bounded efficient l1 tracker with occlusion detection," in *Proc. IEEE CVPR*, Colorado Springs, CO, USA, 2011, pp. 1257–1264.

[24] Q. Wang, F. Chen, W. Xu, and M.-H. Yang, "Online discriminative object tracking with local sparse representation," in *Proc. IEEE WACV*, Breckenridge, CO, USA, 2012, pp. 425–432.

[25] T. Bai, Y. F. Li, and Y. Tang, "Structured sparse representation appearance model for robust visual tracking," in *Proc. IEEE ICRA*, Shanghai, China, 2011, pp. 4399–4404.

[26] S. Zhang, H. Yao, and S. Liu, "Robust visual tracking using feature-based visual attention," in *Proc. IEEE ICASSP*, Dallas, TX, USA, 2010, pp. 1150–1153.

[27] B. Liu, J. Huang, L. Yang, and C. Kulikowsk, "Robust tracking using local sparse appearance model and K-selection," in *Proc. IEEE CVPR*, Providence, RI, USA, 2011, pp. 1313–1320.

[28] Q. Wang, F. Chen, J. Yang, W. Xu, and M.-H. Yang, "Transferring visual prior for online object tracking," *IEEE Trans. Image Process.*, vol. 21, no. 7, pp. 3296–3305, Jul. 2012.

[29] L. R. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, Sep. 1966.

[30] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *Soc. Ind. Appl. Math. Rev.*, vol. 51, no. 3, pp. 455–500, 2009.

[31] M. A. O. Vasilescu and D. Terzopoulos, "Multilinear analysis of image ensembles: TensorFaces," in *Proc. ECCV*, Copenhagen, Denmark, 2002, pp. 447–460.

[32] H. Lu, K. N. Plataniotis, and A. N. Venetsanopoulos, "Multilinear principal component analysis of tensor objects for recognition," in *Proc. Int. Conf. Pattern Recognit.*, Hong Kong, 2006, pp. 776–779.

[33] S. Yan *et al.*, "Multilinear discriminant analysis for face recognition," *IEEE Trans. Image Process.*, vol. 16, no. 1, pp. 212–220, Jan. 2007.

[34] I. Kotsia, W. Guo, and I. Patras, "Higher rank support tensor machines for visual recognition," *Pattern Recognit.*, vol. 45, no. 12, pp. 4192–4203, Dec. 2012.

[35] T.-K. Kim and R. Cipolla, "Canonical correlation analysis of video volume tensors for action categorization and detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 8, pp. 1415–1428, Aug. 2009.

[36] J. Wen, X. Li, X. Gao, and D. Tao, "Incremental learning of weighted tensor subspace for visual tracking," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, San Antonio, TX, USA, 2009, pp. 3688–3693.

[37] Y. Wu, J. Cheng, J. Wang, and H. Lu, "Real-time visual tracking via incremental covariance tensor learning," in *Proc. IEEE ICCV*, Kyoto, Japan, 2009, pp. 1631–1638.

[38] J. Gao, J. Xing, W. Hu, and S. Maybank, "Discriminant tracking using tensor representation with semi-supervised improvement," in *Proc. IEEE ICCV*, Sydney, NSW, Australia, 2013, pp. 1569–1576.

[39] A. Levey and M. Lindenbaum, "Sequential Karhunen–Loeve basis extraction and its application to images," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1371–1374, Aug. 2000.

[40] D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proc. 18th Annu. ACM-SIAM Symp. Discrete Algorithms*, New Orleans, LA, USA, 2007, pp. 1027–1035.

[41] Z. Kalal, J. Matas, and K. Mikolajczyk, "PN learning: Bootstrapping binary classifiers by structural constraints," in *Proc. IEEE CVPR*, San Francisco, CA, USA, 2010, pp. 49–56.

[42] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. IEEE CVPR*, San Francisco, CA, USA, 2010, pp. 1269–1276.

[43] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. ECCV*, Florence, Italy, 2012, pp. 864–877.

[44] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Proc. IEEE CVPR*, Providence, RI, USA, 2012, pp. 1822–1829.

[45] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. ECCV*, Florence, Italy, 2012, pp. 702–715.

[46] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proc. ICML*, Montreal, QC, Canada, 2009, pp. 689–696.

[47] B. W. Bader *et al*. (2012). *MATLAB tensor toolbox version 2.5*. [Online]. Available: http://www.sandia.gov/~tgkolda/TensorToolbox/

[48] E. Erdem, S. Dubuisson, and I. Bloch, "Fragments based tracking with adaptive cue integration," *Comput. Vis. Image Understand.*, vol. 116, no. 7, pp. 827–841, Jul. 2012.

[49] B. Liu, J. Huang, C. Kulikowski, and Y. Lin, "Robust visual tracking using local sparse appearance model and K-selection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 12, pp. 2968–2981, Dec. 2013.

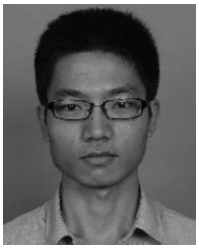
[50] H. Song, "Robust visual tracking via online informative feature selection," *Electron. Lett.*, vol. 50, no. 25, pp. 1931–1933, Dec. 2014.

[51] D. Wang, H. Lu, and C. Bo, "Visual tracking via weighted local cosine similarity," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1838–1850, Sep. 2015.

[52] D. Wang, H. Lu, Z. Xiao, and M.-H. Yang, "Inverse sparse tracker with a locally weighted distance metric," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2646–2657, Sep. 2015.
[53] F. Yang, Z. Jiang, and L. S. Davis, "Online discriminative dictionary learning for visual tracking," in *Proc. WACV*, Steamboat Springs, CO, USA, 2014, pp. 854–861.
[54] K. Zhang, L. Zhang, and M.-H. Yang, "Fast compressive tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 2002–2015, Oct. 2014.
[55] B. Ma *et al.*, "Visual tracking using strong classifier and structural local sparse descriptors," *IEEE Trans. Multimedia*, vol. 17, no. 10, pp. 1818–1828, 2015.
[56] T. Zhang *et al.*, "Structural sparse tracking," in *Proc. IEEE CVPR*, Boston, MA, USA, Jun. 2015, pp. 150–158.

Bo Ma received the Ph.D. degree in computer science from the Harbin Institute of Technology, Harbin, China, in 2003.

From 2004 to 2006, he was with the Department of Computer Science, City University of Hong Kong, Hong Kong, involving on research projects in computer vision and pattern recognition. In 2006, he joined the Department of Computer Science, Beijing Institute of Technology, Beijing, China, where he is currently an Associate Professor. He has published over 40 journal and conference papers such as the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE ICCV, the IEEE SIGNAL PROCESSING LETTERS, and *Pattern Recognition*. His current research interests include statistical pattern recognition, object tracking, and information fusion.

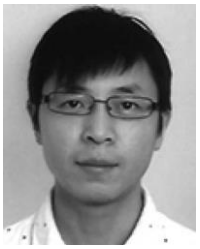


Lianghua Huang is currently pursuing the M.S. degree with the School of Computer Science, Beijing Institute of Technology, Beijing, China.

His current research interests include visual tracking algorithms.

Jianbing Shen (M'11–SM'12) is a Full Professor with the School of Computer Science, Beijing Institute of Technology, Beijing, China. He has published over 50 journal and conference papers such as the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, the IEEE TRANSACTIONS ON CYBERNETICS, the IEEE TRANSACTIONS ON MULTIMEDIA, the IEEE CVPR, and the IEEE ICCV. His current research interests include computer vision and pattern recognition.

Dr. Shen was a recipient of several flagship honors, including the Fok Ying Tung Education Foundation from Ministry of Education, the Program for Beijing Excellent Youth Talents from Beijing Municipal Education Commission, and the Program for New Century Excellent Talents in University from Ministry of Education. He is on the editorial boards of *Neurocomputing*.



Ling Shao (M'09–SM'10) is a Full Professor and the Head of the Computer Vision and Artificial Intelligence Group with the Department of Computer Science and Digital Technologies, Northumbria University, Newcastle upon Tyne, U.K. He is an Advanced Visiting Fellow with the Department of Electronic and Electrical Engineering, University of Sheffield, Sheffield, U.K. His current research interests include computer vision, image processing, pattern recognition, and machine learning.

Dr. Shao is an Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING, the IEEE TRANSACTIONS ON CYBERNETICS, and other journals. He is a fellow of the British Computer Society and IET, and a Life Member of ACM.