Online-Learning Structural Appearance Model for Robust Visual Tracking

Min Yang, Mingtao Pei, Yuwei Wu, Bo Ma, and Yunde Jia

Beijing Laboratory of Intelligent Information Technology,
School of Computer Science, Beijing Institute of Technology,
Beijing 100081, P.R. China
{yangminbit,peimt,wuyuwei,bma000,jiayunde}@bit.edu.cn

Abstract. The main challenge of robust visual tracking comes from the difficulty in designing an adaptive appearance model to account for appearance variations. Existing tracking algorithms often build an representation for the tracked object, and perform self-updating of the object representation with examples from recently tracking results. Slight inaccuracies in the tracker can degrade the appearance models. In this paper, we propose a robust tracking method with an online-learning structural appearance model based on local sparse coding and online metric learning. Our appearance model employs structural feature pooling over the local sparse codes of an object region to obtain a robust object representation. Tracking is then formulated as seeking for the most similar candidate within a Bayesian inference framework where the distance metric used for similarity measurement is learned in an online manner to match the varying object appearances. Both qualitative and quantitative evaluations on various challenging image sequences demonstrate that the proposed algorithm outperforms the state-of-the-art methods.

Keywords: Visual tracking, appearance modeling, sparse coding, online metric learning.

1 Introduction

Appearance modeling is a critical prerequisite for successful visual tracking. An appearance model generally consists of two modules: object representation which captures the visual characteristics of an object and appearance matching scheme that measures the similarity between observed samples and the model. Due to appearance variations caused by background clutters, object deformation, illumination changes and occlusions *etc*, designing a robust appearance model is a challenging task.

Recently, sparse representation based appearance modeling has received considerable attention in the visual tracking community [12,10,15,4,19]. The pioneer work introduced by Mei and Ling [12] models the object appearance as a sparse linear combination of object and trivial templates via ℓ_1 minimization. Wang *et al.* [15] employed subspace learning method to construct and update the object

C. Sun et al. (Eds.): IScIDE 2013, LNCS 8261, pp. 30–39, 2013.
© Springer-Verlag Berlin Heidelberg 2013

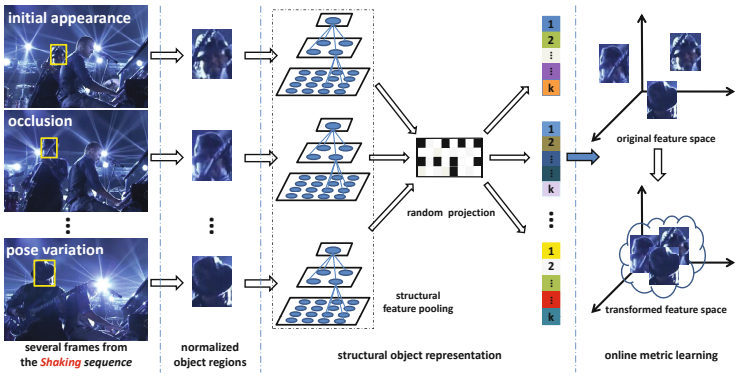

Fig. 1. Motivation of our work. The tracked object is represented by using a structural representation strategy, and the varying object appearances caused by occlusions and pose variations *etc.* can be successfully matched via online metric learning.

templates used for sparse representation. However, only the holistic information of the object is used in these methods, which makes it difficult to handle drastic view or pose variations. Zhong *et al.* [19] presented a sparsity-based collaborative appearance model which exploits the advantages of both holistic and local information. Jia *et al.* [4] introduced an alignment-pooling method across the sparse codes of local patches to improve the accuracy of location estimation. Motivated by the success of sparse representation for object tracking, one aspect we focus on is to design an effective object representation containing local and spatial information using sparse coding.

Most of appearance models based on sparse representation measure the similarity between the candidates and the model by reconstruction errors [12,15,19]. However, when the object undergoes significant appearance variations, the true candidate of the object might have large reconstruction errors, leading to ambiguity of the tracker. In fact, the magnitude of the reconstruction errors depend largely on the dictionary which should be updated in an online manner to account for the varying appearances. Nevertheless, straightforward updating of the dictionary with newly obtained results is prone to potential drift because of the accumulated errors. To keep the flexibility, our another emphasis is placed on seeking for a suitable feature space via online metric learning to match the varying appearances rather than updating the dictionary to ensure the correspondence between minimal reconstruction error and true object location.

Metric learning has been introduced to object tracking by several methods [5,17,9]. Jiang *et al.* [5] integrated neighborhood component analysis (NCA) into kernel-based tracking framework to improve the tracking performance. Wang *et al.* [17] formulated appearance modeling and motion estimation into a unified framework based on metric learning. However, methods proposed in [5,17] only use simple representation strategies, *e.g.*, color histogram, to represent the object appearance, thus are sensitive to significant appearance variations. Furthermore, these methods learn the distance metric in an off-line manner, which often leads

to expensive computation. Li *et al.* [9] presented a non-sparse linear representation with the learned Mahalanobis distance metric for visual tracking. This method only utilizes the holistic information and ignores the trivial template, resulting in failure of tackling occlusions.

Considering the two objectives mentioned above, we develop an online-learning structural appearance model for robust visual tracking. More specifically, we sample several image patches inside an object region using overlapped sliding windows, and employ a structural feature pooling process to concatenate the sparse codes of these patches into a structural representation of the object region. This structural representation captures local and spatial information of the image patches, followed by a very sparse random projection to generate a low-dimensional compact representation. An online metric learning algorithm is then advocated to get a discriminative and adaptive metric for appearance matching. The learned metric makes the different appearances of the object close to each other, and separates the object from the background simultaneously. The main components of our tracking method are depicted in Fig. 1. Numerous experiments and evaluations on challenging video sequences demonstrate that our method outperforms several state-of-the-art trackers.

2 Structural Object Representation

Given a normalized object region $\boldsymbol{P}$, we first sample N local patches inside the region using overlapped sliding windows. Each patch representing one fixed part of an object is then converted to a d-dimensional vector $\boldsymbol{p}_j \in \mathbb{R}^{d\times 1}$. Therefore, the complete structure of the object can be represented by concatenating all these patches together, *i.e.*, $\boldsymbol{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \cdots, \boldsymbol{p}_N] \in \mathbb{R}^{d\times N}$. Let $\boldsymbol{D}$ be a dictionary (or codebook) with n entries, $\boldsymbol{D} = [\boldsymbol{d}_1, \boldsymbol{d}_2, \cdots, \boldsymbol{d}_n] \in \mathbb{R}^{d\times n}$, then each patch $\boldsymbol{p}_j$ can be converted into a n-dimensional code using sparse coding.

2.1 Locality-Constrained Linear Coding

In visual tracking applications, similarity is more essential than sparsity [10]. The regularization term of ℓ_1 norm in traditional sparse coding scheme is not smooth, which leads to the loss of correlations between codes even though for similar patches. Hence, we employ locality-constrained linear coding (LLC) [16] to preserve the similarity between image patches. Specifically, the LLC code $\boldsymbol{x}_j \in \mathbb{R}^{n\times 1}$ corresponding to $\boldsymbol{p}_j \in \mathbb{R}^{d\times 1}$ is computed by

$$\begin{aligned} &\min_{\boldsymbol{x}_j} \|\boldsymbol{p}_j - \boldsymbol{D}\boldsymbol{x}_j\|_2^2 + \lambda \|\boldsymbol{e}_j \odot \boldsymbol{x}_j\|_2, \\ &s.t.\ \mathbf{1}^\top \boldsymbol{x}_j = 1 \end{aligned} \tag{1}$$

where $\odot$ denotes element-wise multiplication, and $\boldsymbol{e}_j$ is the Euclidean distance vector between $\boldsymbol{p}_j$ and all basis vectors in $\boldsymbol{D}$. Note that the LLC code in Eqn. 1 is not ℓ_0 norm sparse, but is sparse in the sense that the solution only has few significant values. LLC actually selects a set of local basis vectors for $\boldsymbol{p}_j$ to form a local coordinate system. In this work, we use a fast approximation of LLC [16] to efficiently obtain the sparse codes of image patches.

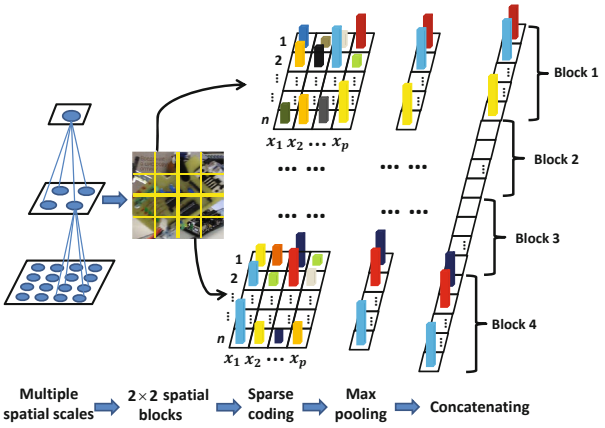

Fig. 2. Illustration of our structural feature pooling process. We show three spatial scales and take 2×2 as an example. In order to illustrate the pooling process clearly, we use 8×8 non-overlapped sliding window to obtain $4 \times 4 = 16$ images patches within the region (the object region is normalized into 32×32). Each of the four spatial blocks (*i.e.*, Block 1, $\cdots$, Block 4) contains $p = 4$ images patches. The higher cylinders in the figure represent the larger values of LLC codes.

2.2 Feature Pooling

Denote the LLC codes of an object as $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N] \in \mathbb{R}^{n \times N}$, the feature pooling function on LLC codes is defined as $\boldsymbol{\alpha} = \xi(\boldsymbol{X})$, where $\boldsymbol{\alpha} \in \mathbb{R}^{n \times 1}$ and $\xi(\cdot)$ is defined on each row of $\boldsymbol{X}$. We use the max pooling process to acquire a middle level representation, which is well established with biophysical evidence in visual cortex and has been shown to be effective for image representation [18]. Therefore, the i-th element of $\boldsymbol{\alpha}$ is given by

$$\boldsymbol{\alpha}_i = \max\left\{|\boldsymbol{x}_{i,1}|, |\boldsymbol{x}_{i,2}|, \cdots, |\boldsymbol{x}_{i,N}|\right\}, \tag{2}$$

where $\boldsymbol{x}_{i,j}$ is the element at i-th row and j-th column of $\boldsymbol{X}$. In this case, $\boldsymbol{\alpha}$ is a global pooled feature, since the pooling function is measured on the whole image patches, discarding the spatial information of the local patches.

In order to capture local and spatial information, we construct a spatial pyramid for the object region and do max pooling on multiple spatial scale. Suppose the object region $\boldsymbol{P}$ is partitioned into $\sigma \times \sigma$ non-overlapped spatial blocks $\{\Delta_b\}_{b=1}^{\sigma^2}$ on the spatial scale σ, the LLC codes $\boldsymbol{X}$ are accordingly divided into $\sigma \times \sigma$ subsets $\{\boldsymbol{X}_{\Delta_b}\}_{b=1}^{\sigma^2}$. The corresponding pooled features are denoted as $\{\boldsymbol{\alpha}_b = \xi(\boldsymbol{X}_{\Delta_b})\}_{b=1}^{\sigma^2}$. Concatenating the pooled features created from each subset of the LLC codes on various spatial scales, we can acquire a structural representation $\boldsymbol{Z}^*$ of the object region,

$$\boldsymbol{Z}^* = \left[\boldsymbol{\alpha}_1^\top, \boldsymbol{\alpha}_2^\top, \cdots, \boldsymbol{\alpha}_\nu^\top\right]^\top, \tag{3}$$

where $\boldsymbol{Z}^* \in \mathbb{R}^m$, $m = n \times \nu$ and ν is the total number of spatial blocks on all spatial scales. Fig. 2 illustrates our structural feature pooling process, in which we show three spatial scales and take 2×2 as an example to demonstrate the feature pooling on a specific spatial scale.

2.3 Dimensionality Reduction

The feature vector $\boldsymbol{Z}^*$ described in Sect. 2.2 is usually high dimensional. We use the random projection to embed the $\boldsymbol{Z}^* \in \mathbb{R}^m$ into a low-dimensional subspace $\boldsymbol{Z} = \boldsymbol{R}\boldsymbol{Z}^*$, where $\boldsymbol{R} \in \mathbb{R}^{m \times k}$ is a random projection matrix and $\boldsymbol{Z} \in \mathbb{R}^k$. A *very sparse random projection* [8] is introduced to help find effective subspace for the original data. The entries of the random projection matrix $\boldsymbol{R}$ are defined as

$$r_{ij} = \sqrt{q} \times \begin{cases} +1 & with\ \ probability\ \ 1/2q \\ 0 & with\ \ probability\ \ 1 - 1/q \\ -1 & with\ \ probability\ \ 1/2q \end{cases}, \tag{4}$$

where $q = \sqrt{m}$. Compared with the traditional dimensionality reduction methods, *e.g.*, PCA, the random projection matrix defined by Eq. 4 is independent with the original data, and hence is suitable for our framework.

3 Online Metric Learning

Given an object template $\boldsymbol{Z}_T \in \mathbb{R}^k$ created from the first frame and a candidate $\boldsymbol{Z}_C \in \mathbb{R}^k$ in the current frame, the Mahalanobis distance between $\boldsymbol{Z}_T$ and $\boldsymbol{Z}_C$ is defined as

$$\mathcal{D}_{\boldsymbol{M}}(\boldsymbol{Z}_T, \boldsymbol{Z}_C) = (\boldsymbol{Z}_T - \boldsymbol{Z}_C)^\top \boldsymbol{M} (\boldsymbol{Z}_T - \boldsymbol{Z}_C), \tag{5}$$

where $\boldsymbol{M} \in \mathbb{R}^{k \times k}$ is required to be a symmetric positive semi-definite matrix. In this work, the matrix $\boldsymbol{M}$ is adaptively obtained by an online metric learning method, and the object template remains fixed during tracking.

We first describe the training examples collection mechanism in our algorithm. Once the object is located, we sample a set of image regions from a small neighborhood around the object location, and label the feature vectors of these regions as positive examples. Similarly, the negative examples are composed of the feature vectors of the image regions far away from the object location.

Following the method in [14], we encode a tuple used for online metric learning as (u, v, l), where (u, v) is a example pair and l is the label which equals $+1$ if u and v are considered similar and -1 otherwise. Given a tuple set, a margin constraint is that the distances between all pairs of dissimilar examples are greater than the distances between all pairs of similar examples at least γ. Alternatively, there exists a threshold δ which is subject to the rule:

$$\begin{cases} \mathcal{D}_{\boldsymbol{M}}(u, v) \leq \delta - \gamma/2 & \forall (u, v, l) :\ l = +1, \\ \mathcal{D}_{\boldsymbol{M}}(u, v) \geq \delta + \gamma/2 & \forall (u, v, l) :\ l = -1. \end{cases} \tag{6}$$

We set γ to be 2, and rewrite the constraint as

$$l \cdot \big(\delta - \mathcal{D}_{\boldsymbol{M}}(u, v)\big) \geq 1. \tag{7}$$

During online learning, at each time step τ, we get a tuple (u_τ, v_τ, l_τ) and calculate the distance $\mathcal{D}_{\boldsymbol{M}_\tau}(u_\tau, v_\tau)$ between the two examples according to the current metric $\boldsymbol{M}_\tau$. After getting the prediction label $\widehat{l}_\tau = sign\big(\mathcal{D}_{\boldsymbol{M}_\tau}(u_\tau, v_\tau) < \delta_\tau\big)$, we can compute a loss if there is a discrepancy between $\widehat{l}_\tau$ and l_τ. The loss function is given by

$$\boldsymbol{\phi}_\tau(\boldsymbol{M}, \delta) \doteq \max\Big\{0, l_\tau\big(\mathcal{D}_{\boldsymbol{M}_\tau}(u_\tau, v_\tau) - \delta_\tau\big) + 1\Big\}. \tag{8}$$

The goal of the online algorithm is to minimize the cumulative loss in Eq. (8). According to [14], this can be done by updating the matrix $\boldsymbol{M}_\tau$ and the threshold δ_τ using two successive projections

$$\begin{aligned} (\boldsymbol{M}_{\widehat{\tau}}, \delta_{\widehat{\tau}}) &= \mathcal{P}_{C_\tau}(\boldsymbol{M}_\tau, \delta_\tau), \\ (\boldsymbol{M}_{\tau+1}, \delta_{\tau+1}) &= \mathcal{P}_{C_a}(\boldsymbol{M}_{\widehat{\tau}}, \delta_{\widehat{\tau}}), \end{aligned} \tag{9}$$

where $\mathcal{P}_C(\boldsymbol{v})$ indicates the orthogonal projection from vector $\boldsymbol{v}$ to a closed convex set C, $C_\tau = \{(\boldsymbol{M}, \delta) : \boldsymbol{\phi}_\tau(\boldsymbol{M}, \delta) = 0\}$ is the set of all $(\boldsymbol{M}_\tau, \delta_\tau)$ pairs which attain zero loss on the example (u_τ, v_τ, l_τ), and $C_a = \{(\boldsymbol{M}, \delta) : \boldsymbol{M} \succeq 0, \delta \geq 1\}$ is the set of all admissible $(\boldsymbol{M}_\tau, \delta_\tau)$ pairs.

4 Proposed Tracking Algorithm

Object tracking can be considered as a Bayesian inference task in a Markov model with hidden state variables. Given the observed image set $\mathcal{O}_{1:t} = \{\boldsymbol{o}_1, \cdots, \boldsymbol{o}_t\}$ up to time t, the optimal state $\boldsymbol{s}_t$ of the tracked object can be estimated by Bayesian theorem

$$p\big(\boldsymbol{s}_t \big| \mathcal{O}_{1:t}\big) \propto p(\boldsymbol{o}_t|\boldsymbol{s}_t) \int p\big(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}\big) p\big(\boldsymbol{s}_{t-1}\big|\mathcal{O}_{1:t-1}\big)\, d\boldsymbol{s}_{t-1}. \tag{10}$$

This inference is governed by the dynamic model $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$ and the observation model $p(\boldsymbol{o}_t|\boldsymbol{s}_t)$. A particle filter [3] is used to approximate the posterior $p\big(\boldsymbol{s}_t\big|\mathcal{O}_{1:t}\big)$ by a finite set of N_s samples $\{\boldsymbol{s}_t^i\}_{i=1}^{N_s}$ with importance weights $\{\omega_t^i\}_{i=1}^{N_s}$. We apply an affine image warp to model the object motion between two consecutive frames. The dynamic model $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1})$ is modeled by Brownian motion, *i.e.*, $p(\boldsymbol{s}_t|\boldsymbol{s}_{t-1}) = \mathcal{N}(\boldsymbol{s}_t; \boldsymbol{s}_{t-1}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is a diagonal covariance matrix.

Given a candidate sample $\boldsymbol{s}_t^i$, the region of interest can be extracted from the observed image $\boldsymbol{o}_t$ by applying an affine transformation using $\boldsymbol{s}_t^i$ as parameters. Then the observation likelihood of the candidate $\boldsymbol{s}_t^i$ is computed by

$$p(\boldsymbol{o}_t|\boldsymbol{s}_t^i) \propto \exp\big(-\mathcal{D}_{\boldsymbol{M}}(\boldsymbol{Z}_T, \boldsymbol{Z}_C^i)\big), \tag{11}$$

where $\boldsymbol{Z}_T$ and $\boldsymbol{Z}_C^i$ are feature vectors of the template and the candidate $\boldsymbol{s}_t^i$, respectively. For the tracking at time t, the candidate with the maximum observation likelihood is chosen as the tracking result.

5 Experimental Results

We run our tracking algorithm on twelve public challenging video sequences, and only gray scale information is used for our experiments. The challenges of these videos include heavy occlusion, illumination changes, pose variations, motion blur, scale variations and complex backgrounds. Our tracker is compared against seven state-of-the-art tracking algorithms denoted as Frag [1], IVT [13], ℓ_1 [12], TLD [6], MIL [2], VTD [7] and SCM [19], respectively. We use the source codes provided by the authors with the same initialization and their default parameters. Since the trackers except Frag involve randomness, we run them 5 times and report the average result for each sequence.

5.1 Implementation Details

We resize the object image to 32×32 pixels and extract overlapped 8×8 patches within the object region with 2 pixels as the step length. Performing $k-$means clustering algorithm on the patches extracted from first frame, the number of dictionary entries n is set to 100. The multi-scale max pooling is performed on three spatial scales 1×1, 2×2 and 3×3 blocks, resulting in a 1400-dimensional feature vector. As discussed in [11], we set the random projection dimensionality $k = 400 \approx 1400/3$. Given the object location at the current frame, 20 positive examples and 15 negative examples are collected for online metric learning. As a trade-off between computational efficiency and effectiveness, the metric matrix $\boldsymbol{M}$ is updated every 10 frames. *The parameters are fixed for all sequences.*

5.2 Quantitative Evaluation

We use the center location error as well as the overlap rate for quantitative evaluations. Center location error is the per-frame distance (in pixels) between the center of the tracking result and that of the ground truth. Overlap rate is defined as $\frac{area(R_T \cap R_G)}{area(R_T \cup R_G)}$, where R_T is the bounding box of tracking result and R_G denotes the ground truth. Table 1 and Table 2 summarize the average center location errors and the average overlap rates, respectively. Note that the TLD tracker does not give tracking result when occlusions occur and the object is need to be re-detected. Thus, we only show the center location errors for the sequences that the TLD can keep track all the time. Overall, the proposed tracker performs favorably against the state-of-the-art algorithms.

To validate the effectiveness of online metric learning, we present the tracking results using a fixed distance metric learned by using the training examples extracted from the first frame (denoted as **Ours w/o ML**) in Table 1 and Table 2. The results show that the online metric learning mechanism provides an effective way to account for appearance variations of the object, and facilities appearance updating and object tracking. More interestingly, **Ours w/o ML** also performs well in some sequences, which demonstrate the effectiveness of the proposed structural object representation.

Table 1. Average center location error (in pixels). **Bold** fonts indicate the best performance while the *italic* fonts indicate the second best ones.

	Frag	IVT	ℓ_1	TLD	MIL	VTD	SCM	Ours w/o ML	Ours
Bird2	25.3	102.1	125.0	—	13.2	54.8	11.9	*11.0*	**6.7**
Bolt	166.6	212.0	166.9	—	**7.0**	95.0	94.4	39.6	*7.3*
Surfer	75.1	125.7	119.3	—	85.1	53.5	35.9	*15.0*	**11.6**
Car6	49.5	53.1	*5.2*	—	89.8	53.4	841.0	27.0	**4.8**
Caviar2	5.7	8.5	54.6	7.2	70.0	6.4	**2.7**	*2.8*	2.9
Woman	*119.2*	246.6	154.8	—	121.2	135.6	121.2	122.7	**5.6**
David	69.6	6.0	57.7	6.4	28.9	27.8	**5.1**	8.3	*5.2*
Shaking	118.5	124.5	56.6	—	19.7	**6.3**	8.6	8.8	*7.7*
Singer2	37.9	88.5	63.0	—	59.4	*18.1*	53.7	177.8	**10.5**
Panda	100.0	95.6	82.6	—	42.9	78.2	*3.7*	**2.9**	3.1
Jumping	8.3	4.4	44.0	*4.1*	39.5	73.0	*4.1*	113.0	**3.6**
Board	85.8	179.3	197.4	150.3	65.5	78.9	*20.4*	27.6	**15.0**

Table 2. Average overlap rate (%). **Bold** fonts indicate the best performance while the *italic* fonts indicate the second best ones.

	Frag	IVT	ℓ_1	TLD	MIL	VTD	SCM	Ours w/o ML	Ours
Bird2	47.5	10.6	8.4	17.3	67.8	15.5	69.5	*71.4*	**79.6**
Bolt	1.3	1.1	15.4	1.1	**71.9**	1.8	13.5	37.0	*71.3*
Surfer	12.8	5.3	5.0	35.4	16.6	20.3	30.1	*56.1*	**64.2**
Car6	54.0	40.3	*78.7*	76.8	14.8	51.5	3.2	63.8	**80.2**
Caviar2	53.6	45.7	33.5	68.1	23.2	61.1	80.2	*81.1*	**81.7**
Woman	16.3	*16.6*	5.4	10.6	15.5	14.9	15.2	16.5	**71.8**
David	29.9	64.8	29.6	55.6	47.7	49.3	42.5	*73.7*	**83.9**
Shaking	21.3	3.0	14.2	12.1	58.7	**73.1**	71.9	70.7	*72.3*
Singer2	50.8	23.3	23.9	9.7	24.0	*67.8*	33.0	8.7	**72.9**
Panda	32.1	9.3	1.4	58.7	45.1	37.2	*63.5*	62.0	**64.4**
Jumping	58.6	70.6	14.4	65.5	20.9	11.5	*72.2*	10.6	**73.5**
Board	58.5	12.6	10.0	16.8	46.6	38.8	**76.2**	67.2	*73.1*

5.3 Qualitative Evaluation

Several screenshots of the visual tracking results on the twelve sequences are illustrated in Fig. 3. We give a qualitative evaluation of the tracking results in four different ways as follows.

Pose Variation. In the *Bird2*, *Bolt* and *Surfer* sequences, the object appearances change drastically due to significant pose variations. We can see that only our method tracks the objects successfully in all these three sequences. Other evaluated algorithms except the MIL and SCM methods fail when the objects start to change their pose (*e.g.*, *Bird2* #50 and *Bolt* #20). In the *surfer* sequence, the MIL and SCM methods gradually drift away when there is severe occlusion and large scale change of the object. Our method adaptively cope with appearance variations via online metric learning, thus provide more accurate and consistent tracking results.

Fig. 3. Sample tracking results of the evaluated algorithms on twelve challenging image sequences. The figures are arranged in the same order as Table. 1.

Occlusion. We test several sequences (*Car6*, *Caviar2* and *Woman*) with severe or long-term partial occlusions. In the *Car6* sequence, only the ℓ_1, TLD and the proposed methods are able to track the object when the long-term occlusion happens (*e.g.*, *Car6* #528). Note that the ℓ_1 tracker involves occlusion resolving scheme, and the TLD method employs a detector to reacquire the object. The *Caviar2* and *Woman* sequences contain scale change, partial occlusion and interference of similar objects. Most of the trackers lock onto a wrong object after occlusion (*e.g.*, *Caviar2* #240 and *Woman* #130). In contrast, our method achieves stable performance in the entire sequence.

Illumination Change. The tracked objects in the *David*, *Shaking* and *Singer2* sequences undergo significant illumination changes and pose variations. The Frag, IVT, L1, TLD and MIL methods can not handle the appearance variations caused by illumination changes together with pose variations (*e.g.*, *David* #178 and *Shaking* #62), whereas the VTD and SCM methods perform better. In the *Singer2* sequence, the contrast between the foreground and the background is very low. Our method success to track the object accurately, but most trackers drift away at the beginning of the sequence (*e.g.*, *Singer2* #64).

Other Challenges. We test three sequences where the objects suffer other challenges including in-plane rotation (*Panda*), motion blur (*Jumping*) and background clutters (*Board*). Overall, the SCM and our method perform well whereas the other trackers fail to track the objects.

6 Conclusion

We have presented a robust appearance model for visual tracking via a structural object representation strategy and online metric learning. Experiments on twelve challenging sequences demonstrate the robustness of our tracker compared with seven state-of-the-art tracking methods.

Acknowledgments. This work was supported in part by the Natural Science Foundation of China (NSFC) under grant No. 61203291 and the 973 Program of China under grant No. 2012CB720000.

References

1. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: CVPR, vol. 1, pp. 798–805 (2006)
2. Babenko, B., Yang, M., Belongie, S.: Robust object tracking with online multiple instance learning. PAMI 33(8), 1619–1632 (2011)
3. Isard, M., Blake, A.: Condensation conditional density propagation for visual tracking. IJCV 29(1), 5–28 (1998)
4. Jia, X., Lu, H., Yang, M.: Visual tracking via adaptive structural local sparse appearance model. In: CVPR, pp. 1822–1829 (2012)
5. Jiang, N., Liu, W., Wu, Y.: Learning adaptive metric for robust visual tracking. TIP 20(8), 2288–2300 (2011)
6. Kalal, Z., Matas, J., Mikolajczyk, K.: Pn learning: Bootstrapping binary classifiers by structural constraints. In: CVPR, pp. 49–56 (2010)
7. Kwon, J., Lee, K.: Visual tracking decomposition. In: CVPR, pp. 1269–1276 (2010)
8. Li, P., Hastie, T., Church, K.: Very sparse random projections. In: SIGKDD, pp. 287–296 (2006)
9. Li, X., Shen, C., Shi, Q., Dick, A., van den Hengel, A.: Non-sparse linear representations for visual tracking with online reservoir metric learning. In: CVPR, pp. 1760–1767 (2012)
10. Liu, B., Huang, J., Yang, L., Kulikowsk, C.: Robust tracking using local sparse appearance model and k-selection. In: CVPR, pp. 1313–1320 (2011)
11. Liu, L., Fieguth, P.: Texture classification from random features. TPAMI 34(3), 574–586 (2012)
12. Mei, X., Ling, H.: Robust visual tracking using $\ell 1$ minimization. In: ICCV (2009)
13. Ross, D., Lim, J., Lin, R., Yang, M.: Incremental learning for robust visual tracking. IJCV 77(1), 125–141 (2008)
14. Shalev-Shwartz, S., Singer, Y., Ng, A.: Online and batch learning of pseudo-metrics. In: ICML (2004)
15. Wang, D., Lu, H., Yang, M.: Online object tracking with sparse prototypes. TIP 22(1), 314–325 (2012)
16. Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., Gong, Y.: Locality-constrained linear coding for image classification. In: CVPR, pp. 3360–3367 (2010)
17. Wang, X., Hua, G., Han, T.X.: Discriminative tracking by metric learning. In: Danilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part III. LNCS, vol. 6313, pp. 200–214. Springer, Heidelberg (2010)
18. Yang, J., Yu, K., Gong, Y., Huang, T.: Linear spatial pyramid matching using sparse coding for image classification. In: CVPR, pp. 1794–1801 (2009)
19. Zhong, W., Lu, H., Yang, M.: Robust object tracking via sparsity-based collaborative model. In: CVPR, pp. 1838–1845 (2012)