Patch-Based Visual Tracking with Two-Stage Multiple Kernel Learning

Heng Fan[1] and Jinhai Xiang[2(✉)]

[1] College of Engineering, Huazhong Agricultural University, Wuhan 430070, People's Republic of China
hfan@webmail.hzau.edu.cn
[2] College of Informatics, Huazhong Agricultural University, Wuhan 430070, People's Republic of China
jimmy_xiang@mail.hzau.edu.cn

Abstract. In this paper a novel patch-based tracking algorithm is proposed by using two-stage multiple kernel learning. In the first stage, each object patch is represented with multiple features. Unlike simple feature combination, we utilize multiple kernel learning (MKL) method to obtain the optimal combination of multiple features and kernels, which assigns different weight to the features according to their discriminative power. In the second stage, we apply MKL to making full use of multiple patches of the target. This method can automatically distribute different weight to the object patches according to their importance, which improves the discriminative power of object patches as a whole. Within the Bayesian framework, we achieve object tracking by constructing a classifier, and the candidate with the maximum likelihood is chosen to be the target. Experiments demonstrate that the proposed tracking approach performs favorably against several state-of-the-art methods.

Keywords: Visual tracking · Multiple features · Patch-based tracking · Multiple kernel learning

1 Introduction

Visual tracking is one of the most important component of many applications in computer vision, such as surveillance, human-computer interaction, medical imaging and robotics [1]. For robust visual tracking, numerous methods have been presented. Despite reasonably good results from these approaches, some common challenges remain for tracking objects under complex scenes, e.g., when objects undergo significant pose changes or other severe deformations, i.e., object pose variations accompanied with object occlusions or object intersections. To address these problems, a wide range of appearance models for tracking have been proposed by researchers [2]. Roughly speaking, these models can be categorized into two types: discriminative-based model [5,9–13,18,20,21] and generative-based model [3,4,6–8,14,15,19].

Y.-J. Zhang (Ed.): ICIG 2015, Part III, LNCS 9219, pp. 20–33, 2015.
DOI: 10.1007/978-3-319-21969-1_3

Recently multiple kernel learning (MKL) [22,27] has been applied in computer vision, such as object classification [23,24], object detection [25,26]. The MKL method aim to compute an optimal combination of weighted kernels in the supervised learning paradigm. Rather than using one single kernel, the MKL algorithms fuse different features and kernels in an optimal setting, which improves the discriminative power of multiple features.

Motivated by the MKL, we propose a novel patch-based tracking method based on two-stage multiple kernel learning. The patch-based methods utilize the local information of object and can effectively handle partial occlusion and deformation to some extend. However these trackers may cause drift problem because they do not consider the different importance of each patch when occlusion happens. In this work we combine patch-based method with MKL and present a patch-based tracking approach with two-stage MKL. In the first stage, each object patch is represented with multiple features. Unlike simple feature combination, we utilize MKL method to obtain the optimal combination of multiple features and kernels, which assigns different weight to the features according to their discriminative power. In the second stage, we apply MKL to making full use of multiple patches of the target. This method can automatically distribute different weight to the object patches according to their importance, which improves the discriminative power of object patches as a whole. Within the Bayesian framework, we achieve visual tracking by constructing a classifier, and the candidate with the maximum likelihood is selected to be the tracked result. Besides, an effective update method is adopted to help the proposed tracker adapt to the object appearance changes.

The rest of this paper is organized as follows. Section 2 briefly reviews the related works. Section 3 describes the multiple kernel learning method. The proposed two-stage multiple kernel learning is given in Sect. 4. Section 5 describes our tracking method. Experimental results are shown in Sect. 6, and Sect. 7 concludes this paper.

2 Related Work

General tracking approaches can be categorized into either discriminative or generative models [2]. The discriminative methods regard tracking as a classification problem which aims to best separate the object from the ever-changing background. These methods employ both the foreground and background information. Avidan [18] proposes an ensemble tracker which treats tracking as a pixel-based binary classification problem. This method can distinguish target from background, however the pixel-based representation needs more computational resources and thereby limits its performance. In [10], Grabner et al. present an online boosting tracker to update discriminative features and further in [20] a semi-online method is proposed to handle drifting problem. Kalal et al. [13] introduce a P-N learning algorithm to learn effective features from positive and negative samples for object tracking. This tracking method nevertheless is prone to induce drifting problem when object appearance varies. Fan et al. [15] suggest

a weighted P-N learning algorithm and combine it with part-based framework for visual tracking. This method can improve the robustness of tracker in the presence of occlusion. Babenko et al. [9] utilize the multiple instance learning (MIL) method for visual tracking, which can alleviate drift to some extent. Whereas the MIL tracker may detect the positive sample that is less important because it does not consider the sample importance in its learning process. Further in [21], Zhang et al. propose the online weighted multiple instance learning (WMIL) by assigning weight to different samples in the process of training classifier. In [12], Zhang et al. propose a compressive tracker with an appearance model based on features extracted in the compressed domain. This tracker easily induce drift even failure since it is lack of an effective updating strategy in the presence of appearance variations.

On the contrary, the generative models formulate the tracking problem as searching for regions most similar to object. These methods are based on either subspace models or templates. To solve the problem of appearance variations caused by illumination or deformation, the appearance model is updated dynamically. In [3], the incremental visual tracking method suggests an online approach for efficiently learning and updating a low dimensional PCA subspace representation for the object. However, this PCA subspace based representation scheme is sensitive to partial occlusion. Adam et al. [4] present a fragment-based template model for visual tracking. This tracking method estimates the target based on voting map of each part via comparing its histogram with the templates. Nevertheless, static template with equal importance being assigned to each fragment obviously lowers the performance of tracker. Mei et al. [6] apply sparse representation to visual tracking, which can resist occlusion in some degree. However, this method is prone to cause drift because it does not have any update strategy. Jia et al. [8] propose a local structural spare appearance model for object tracking. This method adopts a online update mechanism to help the tracker adapt to appearance changes. Kwon et al. [14] decompose the appearance model into multiple basic observation models to cover a wide range of illumination and deformation.

Recently, MKL method has been widely used in image classification, object detection and recognition. In [23], Yang et al. present a group-sensitive MKL for object categorization. Jawanpuria et al. [24] utilize MKL for non-linear feature selection and apply it to classification. Vedaldi et al. [25] propose a novel three-stage classifier with MKL, which combines linear, quasi-linear, and non-linear kernel SVMs. Zhang et al. [26] proposes an E2LSH based clustering algorithm which combines the advantages of nonlinear multiple kernel combination methods, and use it for object detection.

The most related work to ours is [28], in which a multiple kernel boosting method with affinity constraints is proposed. This method boosts the multiple kernel learning process, thereby facilitating robust visual tracking in complex scenes effectively and efficiently. However, their method does not adopt patch-based representation and hence may be sensitive to partial occlusion. In our work, we segment object into multiple patches and combine it with a two-stage MKL method. Consequently, the proposed tracker is more adaptive to appearance variations.

3 Multiple Kernel Learning

Support vector machine (SVM) has been successfully applied to numerous classification and regression tasks. One of the most important problem in these tasks is to choose an appropriate data representation. In SVM-based approaches, the data representation is implicitly selected by the kernel function $K(x, x_i)$, where $K(\cdot, \cdot)$ is a function associated with a reproducing kernel Hilbert space [28]. Nevertheless, it is difficult for a single SVM classifier to select a good kernel function for the training set in some case. To address this issue, MKL algorithm is proposed. MKL is an extension of kernel learning method. By using different types of kernel to represent different properties of samples (e.g., feature and metric), MKL provides a unified framework for model combination and selection. One of the most popular multiple kernel learning methods is SimpleMKL (SMKL) [29] in which the kernel function is defined as a convex linear combination of kernels

$$K(x, x_i) = \sum_{m=1}^{M} \beta_m K_m(x, x_i), \ \sum_{m=1}^{M} \beta_m = 1, \beta_m \geq 0 \tag{1}$$

where $K(x, x_i)$ denotes the m^{th} kernel and β_m is the corresponding weight. The SMKL is aimed to simultaneously obtain support vectors, support vector coefficients and kernel weights by solving the constrained optimization problem as follows

$$\min_{\beta} J(\beta) \quad s.t. \ \sum_{m=1}^{M} \beta_m = 1, \beta_m \geq 0 \tag{2}$$

where

$$\begin{aligned} J(\beta) &= \min_{\{f\}, b, \xi} \frac{1}{2} \sum_{m} \frac{1}{\beta_m} \|f_m\|^2_{\mathcal{H}_m} + C \sum_{i} \xi_i \\ s.t. \ & y_i \sum_{m} f_m(x_i) + y_i b \geq 1 - \xi_i, \xi_i \geq 0, \forall i \end{aligned} \tag{3}$$

where x_i denotes the i^{th} training sample, y_i is the class label for the i^{th} sample, ξ_i and C represent its slack variable and penalty factor for slack variable respectively, $\mathcal{H}_m$ denotes the reproducing kernel Hilbert space (RKHS), and each function f_m belongs to a different RHSH $\mathcal{H}_m$ associated with a kernel K_m. The Formulation (3) can be solved by reduced gradient method [29], which computes simple differentiation of the dual function of Eq. (3) with respect to β_m

$$\frac{\partial J}{\partial \beta_m} = -\frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K_m(x_i, x_j), \forall m \tag{4}$$

where α_i represents the dual coefficient of x_i. Then the decision function for binary classification is defined as

$$F(x) = \sum_{i} \alpha_i y_i \sum_{m} \beta_m K_m(x, x_i) + b \tag{5}$$

4 Patch-Based Two-Stage MKL

4.1 Object Segmentation

In this paper, we use multiple patches to represent the target, which utilizes the local information of object and can effectively handle partial occlusion and deformation to some extend. Different from [4], we adopt a overlapping slide window segmentation strategy as shown in Fig. 1. After segmentation, we can obtain a patch set $\mathcal{P} = \{p_1, p_2, \cdots, p_P\}$, where p_i is the i^{th} patch and P is the number of patches.

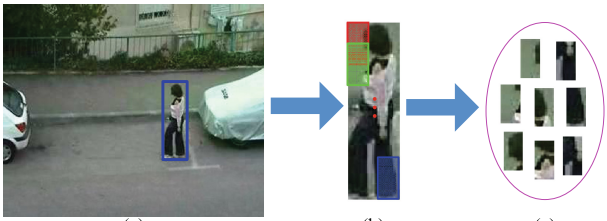

Fig. 1. Illustration of the overlapping slide window segmentation. Image (a) is the object, image (b) shows the segmentation method and image (c) is the set of object patches.

4.2 First-Stage Multiple Kernel Learning

In the first stage, we use multiple features (e.g., HIS histogram, HoG [16] and LBP [17] descriptors) to represent each object patch and apply MKL to the optimal combination for multiple features. For each i^{th} patch, it can be represent with feature set $\{f_{i,1}, f_{i,2}, \cdots, f_{i,D}\}$, where $f_{i,j}$ denotes the $j^{th}(j = 1, 2, \cdots, D)$ feature and D is the number of features. Our goal is to find a strategy to integrate these multiple features to maximize the overall discriminative power. MKL has shown its potential in integrating multiple features in recent research. Therefore, for each i^{th} patch, the output margin of first-stage MKL classifier can be written as the following

$$F_i^{'}(f_i) = \sum_{l=1}^{L} \alpha_l^{'} y_l^{'} \sum_{d=1}^{D} \gamma_{i,d} K_d(f_i, f_{i,l}) + b_i^{'} \tag{6}$$

where $F_i^{'}(\cdot)$ denotes the classification function for the i^{th} patch, $K_d(\cdot, \cdot)$ represent the d^{th} kernel for the d^{th} feature, L is the number of training samples, D stands for the number of features and $\gamma_{i,d}$ weights the discriminative power of the d^{th} feature. Note that in the first stage, the MKL is only used to obtain the weight of each feature. Figure 2 gives a simple illustration about how we make use of MKL to obtain the weight of multiple features for each patch.

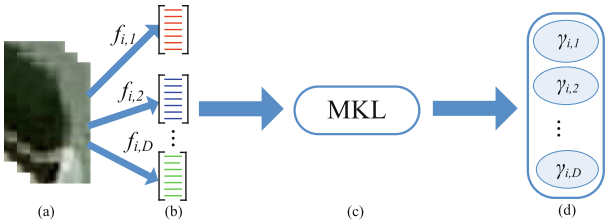

Fig. 2. We firstly collect the training samples for the i^{th} patch in (a), and extract D features for it in (b). The MKL in (c) is then utilized to obtain the weight of multiple features for the patch i as shown in (d).

With the weight of different features, we can obtain the optimal combination of multiple features for the each patch. For the i^{th} patch, we define $\mathcal{F}_i$ as its combined feature

$$\mathcal{F}_i = [\gamma_{i,1} f_{i,1}, \gamma_{i,2} f_{i,2}, \cdots, \gamma_{i,D} f_{i,D}],\ i = 1, 2, \cdots, P \tag{7}$$

4.3 Second-Stage Multiple Kernel Learning

In the second stage, we apply MKL to assigning different weight to the object patches according to their importance. In Sect. 4.1, the target is represented by a patch set $\mathcal{P} = \{p_1, p_2, \cdots, p_P\}$ in which p_i denotes the $i^{th} (i = 1, 2, \cdots, P)$ patch associated with a combination feature $\mathcal{F}_i$. Our goal is aimed to use MKL find an optimal combination for the patches in which the coefficient of each patch stands for the corresponding weight. Therefore, for the target, the output margin of MKL classifier can be written as follows

$$F^*(\mathcal{F}) = \sum_{q=1}^{N} \alpha_q^* y_q^* \sum_{i=1}^{P} \delta_i K_i(\mathcal{F}, \mathcal{F}_q) + b^* \tag{8}$$

where $F^*(\cdot)$ denotes the decision function , $K_i(\cdot, \cdot)$ represents the i^{th} kernel for the i^{th} patch, N is the number of training samples, P stands for the number of patches and δ_i weights the discriminative power of the i^{th} patch. The process of weighing patches can be shown in Fig. 3.

After obtaining the weight of each patch, we can represent the object with a feature vector as follows

$$H = [\delta_1 \mathcal{F}_1, \delta_2 \mathcal{F}_2, \cdots, \delta_P \mathcal{F}_P] \tag{9}$$

where H denotes the feature of the target, δ_i and $\mathcal{F}_i$ are the weight and combined feature for the i^{th} patch.

4.4 Classifier

In this section, a classifier is constructed to discriminative the object from the background. In the initial frame, we randomly sample bounding boxes around

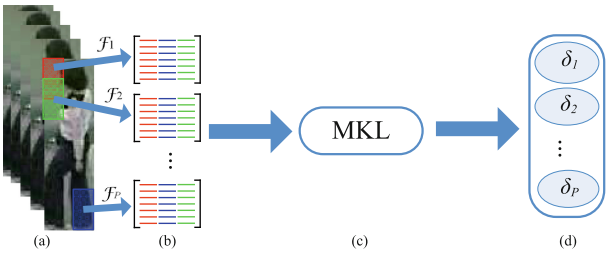

Fig. 3. To start with, we compute the combined features in (b) for all the training patches in (a). Then MKL in (c) is used to obtain the weight of each patch as shown in (d).

the tracked target as positive samples and far away from the target as negative samples. By controlling the distance from the tracked object, the negative samples contain pure background so that they are capable to differentiate from the target to the most extent. We use sets $S^+ = \{s_1^+, s_2^+, \cdots, s_{N^+}^+\}$ and $S^- = \{s_1^-, s_2^-, \cdots, s_{N^-}^-\}$ to denote the positive samples and the negative samples, where N^+ and N^- are the number of positive and negative samples. For each sample, it can be represented by a feature vector with Eq. (9) through two-stage MKL. Therefore, we use sets $\mathcal{H}^+ = \{H_1^+, H_2^+, \cdots, H_{N^+}^+\}$ and $\mathcal{H}^- = \{H_1^-, H_2^-, \cdots, H_{N^-}^-\}$ to represent the features of positive and negative samples. With these features, we can build a LIBSVM classifier G according to [30]. For a new sample s associated with the feature H_s, its classification error can be represented with $G(H_s)$. The smaller the classification error is, the more likely the sample belongs to the object.

5 The Proposed Tracking Method

5.1 Tracking Formulation

Our tracker is implemented via the Bayesian framework. Given the observation set of target $Y^t = \{y_1, y_2, \cdots, y_t\}$ up to the frame t, we can obtain estimation $\widehat{X}_t$ by computing the maximum a posterior via

$$\widehat{X}_t = \max_{X_t^i} p(X_t^i | Y^t) \tag{10}$$

where $\widehat{X}_t$ denotes the i^{th} sample at the state of X_t. The posterior probability $p(X_t^i|Y^t)$ can be obtained by the Bayesian theorem recursively via

$$p(X_t|Y^t) \propto p(y_t|X_t) \int p(X_t|X_{t-1}) p(X_{t-1}|Y^{t-1}) dX_{t-1} \tag{11}$$

where $p(X_t|X_{t-1})$ and $p(X_{t-1}|Y^{t-1})$ represent the dynamic model and observation model respectively.

The dynamic model indicates the temporal correlation of the target state between consecutive frames. We apply affine transformation to model the target motion between two consecutive frames within the particle filter framework. The state transition can be formulated as

$$p(X_t|X_{t-1}) = \mathcal{N}(X_t; X_{t-1}, \Psi) \tag{12}$$

where Ψ is a diagonal covariance matrix whose elements are the variance of affine parameters. The observation model $p(y_t|X_t)$ represents the probability of the observation y_t as state X_t. In this paper, the observation is designed by

$$p(y_t|X_t) \propto 1 - G(X_t) \tag{13}$$

where $G(X_t)$ is the classification error of the t^{th} candidate. Through Bayesian framework, we can determine the candidate sample with the smallest classification error as the tracking result.

5.2 Online Update

Due to the appearance variations of target, updating is essential. In this paper, an effective mechanism is proposed to update the classifier G. To start with, we design a set Φ. In each frame, after locating the target, we randomly sample bounding boxes around the tracked target as positive samples and far away from the target as negative samples. These samples are collected as a group, and added into the set Φ. When the set size v reaches a threshold V, we apply to the set Φ to updating the weight (both the feature weight and patch weight). Then we extract feature for each sample in Φ and train them for the classifier G, and empty Φ in the end. However, when accumulating elements into Φ, the tracking result may contain significant noise and thus is not reliable if the tracking result determined by our tracker has a high classification error which is greater than a threshold E. In this case, we skip this frame to avoid introducing noise into Φ.

So far, we have introduced the overall procedure of the proposed tracking algorithm as shown in Algorithm 1.

6 Experiments

In order to evaluate the performance of our tracking algorithm, we test our method on nine challenging image sequences and compare it with eight state-of-the-art trackers. These algorithms are Frag tracking [4], TLD tracking [13], ℓ_1 tracking [6], IVT tracking [3], MIL tracking [9], CT tracking [12] OAB tracking [10] and SPT tracking [5]. Some representative results are displayed in this section.

The proposed algorithm is implemented in MATLAB and runs at 1.6 frames on a 3.2 GHz Intel E3-1225 v3 Core PC with 8GB memory. We use three features (HSI histogram, HoG and LBP descriptors) and four types of kernels (linear, polynomial, RBF kernel, and sigmoid functions) to represent the target. The

Algorithm 1. Tracking Based on Proposed Method

Initialization:

Given initial state of the target, extract N^+ positive samples and N^- negative samples;
Segment each sample into P patches;
Extract D features for each patch, and apply MKL to obtaining the weights of multiple features;
Compute the combined feature for each patch, and apply MKL to obtaining the weights of patches;
Calculate the feature of each sample and utilize LIBSVM to train them for the classifier G;

Tracking:

for $t = 2$ to the end of the sequence **do**
  Generate N_c candidates $\{s_i\}_{i=1}^{N_c}$;
  for $n_c = 1$ to N_c **do**
    Compute the feature H_{n_c} for the n_c^{th} candidate;
    Calculate the classification error $G(H_{n_c})$;
  end for
  Select the smallest classification error and its index via $e = \min G(H_{n_c})$ and $\mathcal{K} = \min_{n_c} G(H_{n_c})$;
  The $\mathcal{K}^{th}$ candidate target is chosen to be the object;
  if $e \geq E$ **do**
    Skip this frame;
  else
  Extract samples as a group and add this group to set Φ;
  If the size of set Φ is equal V, update the weight and classifier, and then empty set Φ;
  end if
end for

End

parameters of the proposed tracker are fixed in all experiments. The number of particles in Bayesian framework is set to 300 to 500. The training frame N is 4 and the size of the set Φ in this work is set to 5. The parameter classification error threshold E is fixed to 0.4 to 0.6.

6.1 Quantitative Comparison

We evaluate the above mentioned trackers via center location error and overlapping rate [31], and the comparing results are shown in Tables 1 and 2. Figure 4 shows the center location error of the trackers on nine test sequences. Overall, the tracker proposed in this paper outperforms the state-of-the-art algorithms.

Table 1. Center location errors (in pixels). The best result is shown in **red** and the second best in **blue** fonts.

	ℓ_1	Frag	IVT	MIL	TLD	CT	OAB	SPT	Ours
Basketball	131.2	16.2	68.4	94.83	129.3	19.0	86.4	**4.8**	**5.2**
Bicycle	49.0	55.7	7.8	10.5	10.6	51.5	60.9	**5.4**	**4.9**
Bolt	361.7	100.6	374.9	365.4	87.8	348.0	-	**6.8**	**6.6**
Cup	2.9	7.0	**1.8**	40.60	3.1	25.13	4.45	-	**2.8**
Deer	91.7	93.4	222.5	214.0	47.8	235.9	**27.6**	97.0	**9.8**
Face	5.4	**4.8**	62.5	36.22	6.9	57.07	10.5	18.2	**4.5**
Jogging	14.5	9.3	130.0	146.1	**7.2**	124.7	-	-	**10.2**
Lemming	179.5	143.7	182.9	135.3	17.04	82.4	16.2	**7.3**	**5.8**
Woman	134.1	106.2	138.6	116.3	72.6	108.8	-	**12.2**	**8.2**
Average	101.3	56.2	114.7	119.2	38.4	116.9	34.1	**19.9**	**6.4**

Table 2. Overlapping rate. **Red** fonts indicate the best performance while the **blue** fonts indicate the second best.

	ℓ_1	Frag	IVT	MIL	TLD	CT	OAB	SPT	Ours
Basketball	0.03	0.55	0.41	0.21	0.09	0.61	0.16	**0.83**	**0.81**
Bicycle	0.31	0.25	0.33	0.43	0.39	0.29	0.24	**0.55**	**0.72**
Bolt	0.02	0.20	0.01	0.01	0.14	0.01	-	**0.73**	**0.67**
Cup	0.74	0.67	0.71	0.39	0.72	0.53	**0.76**	-	**0.78**
Deer	0.13	0.10	0.03	0.03	0.49	0.04	**0.59**	0.10	**0.68**
Face	0.85	**0.86**	0.36	0.55	0.77	0.38	0.79	0.74	**0.88**
Jogging	0.57	0.65	0.13	0.01	**0.73**	0.13	-	-	**0.68**
Lemming	0.14	0.13	0.12	0.12	0.30	0.33	0.61	**0.65**	**0.81**
Woman	0.06	0.19	0.16	0.13	0.29	0.16	-	**0.60**	**0.62**
Average	0.30	0.41	0.27	0.22	0.44	0.26	0.49	**0.63**	**0.74**

6.2 Qualitative Comparison

Heavy Occlusion: Deformation is a challenge for tracker, because the template features have completely changed when deformation occurs. As shown in Fig. 5, MIL, CT, IVT, OAB, TLD and ℓ_1 do not have good performances in the sequences *Bolt* and *Jogging*. Differently, Frag and SPT have relatively better tracking results in these sequences, because part-based trackers are less sensitive to structure variation than holistic appearance. Whereas, the lack of effective updating strategy still easily cause drifting away even failure. Our tracker have obvious advantage in handling structure deformation even in high frequency, since some local patches of the target remain the same in the presence of the deformation and with the help of effective updating mechanism, our tracking method robustly adapts to the deformation.

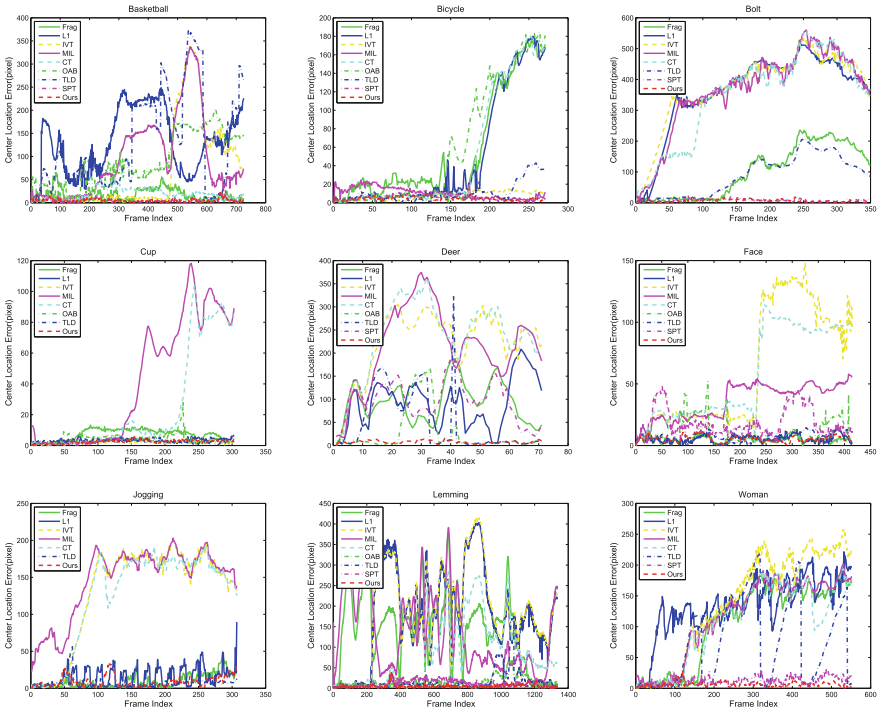

Fig. 4. Quantitative evaluation in terms of center location error (in pixel).

Motion Blur: Fig. 5 demonstrates experimental results on two challenging sequences (*Deer* and *Lemming*). Because the target undergoes fast and abrupt motion, it is more prone to cause blur, which causes drifting problem. It is worth noticing that the suggested approach in this paper performs better than other algorithms. When motion blur happens, our tracker can still effectively represent the target appearance. Besides, our updating mechanism can resist motion blur to some degree. Hence our tracker will not be undermined by the abrupt movement.

Deformation: Deformation is a challenge for tracker, because the template features have completely changed when deformation occurs. As shown in Fig. 5, MIL, CT, IVT, OAB, TLD and ℓ_1 do not have good performances in the sequences *Bolt* and *Jogging*. Differently, Frag and SPT have relatively better tracking results in these sequences, because part-based trackers are less sensitive to structure variation than holistic appearance. Whereas, the lack of effective updating strategy still easily cause drifting away even failure. Our tracker have obvious advantage in handling structure deformation even in high frequency, since some local patches of the target remain the same in the presence of the deformation and with the help of effective updating mechanism, our tracking method robustly adapts to the deformation.

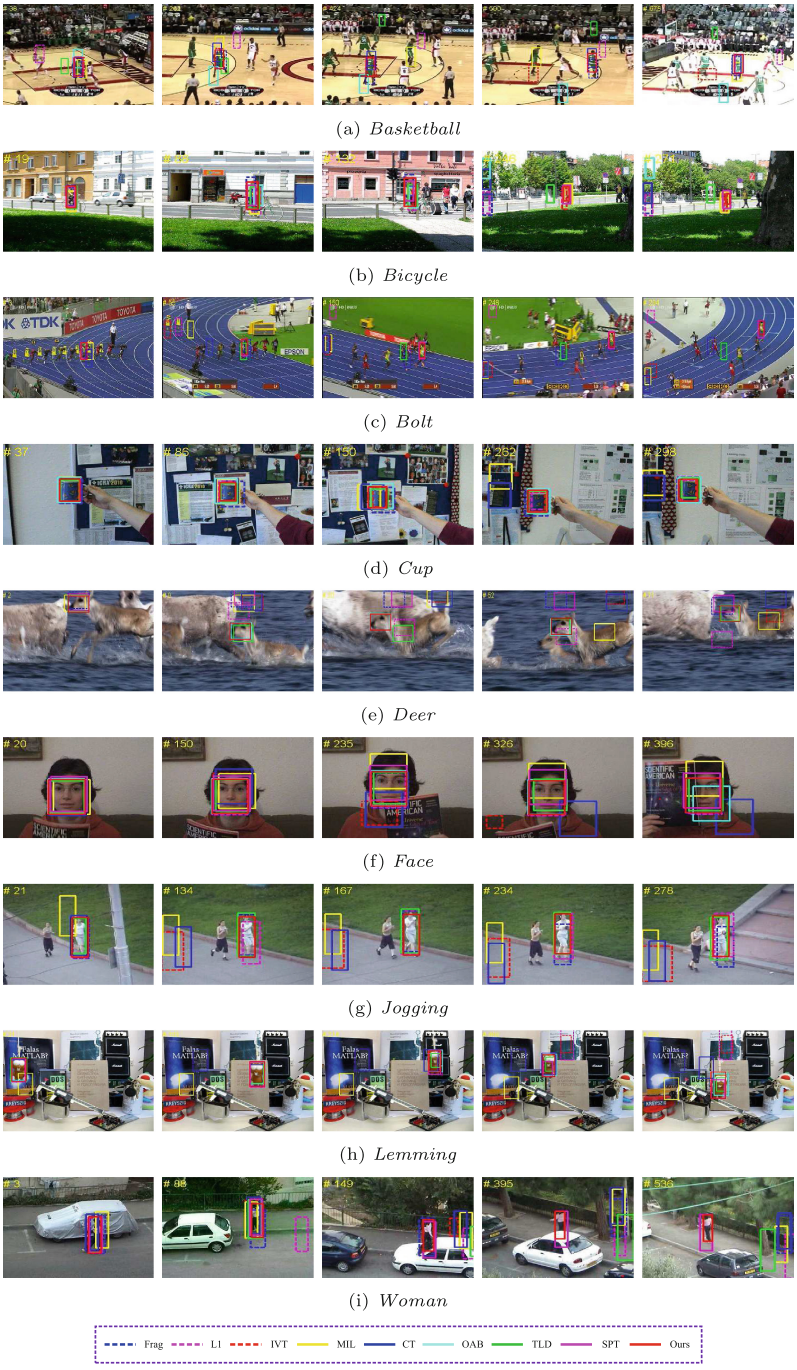

Fig. 5. Screenshots of some sample tracking results.

Background Clutter: The sequences *Cup* and *Basketball* in Fig. 5 are challenging as the background cluttered and the target undergoes the scale variation. Our tracker performs well in this sequence as the target can be differentiated from the cluttered background with the use of our two-stage MKL method. In addition, the updating scheme is also robust to the complex background.

7 Conclusion

In this paper a novel patch-based tracking algorithm is proposed by using two-stage multiple kernel learning. Our method can automatically distribute different weight to the object patches according to their importance, which improves the discriminative power of object patches as a whole. Experiments on challenging image sequences demonstrate that our method performs favorably against several state-of-the-art methods.

Acknowledgement. This work was supported by the Fundamental Research Funds for the Central Universities (Program No. 2014BQ083) and National Training Programs of Innovation and Entrepreneurship for Undergraduates (Program No. 201510504116).

References

1. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: a benchmark. In: 26th IEEE Conference on Computer Vision and Pattern Recognition, pp. 2411–2418. IEEE Press, Portland (2013)
2. Li, X., Hu, W., Shen, C., Zhang, Z., Dick, A.: A survey of appearance models in visual object tracking. ACM Trans. Intell. Syst. Technol. **4**(4), 2411–2418 (2013)
3. Ross, D.A., Lim, J., Lin, R.S., Yang, M.H.: Incremental learning for robust visual tracking. Int. J. Comput. Vision **77**(1–3), 125–141 (2008)
4. Adam, A., Rivlin, E., Shimshoni, I.: Robust fragments-based tracking using the integral histogram. In: 19th IEEE Conference on Computer Vision and Pattern Recognition, pp. 798–805. IEEE Press, New York (2006)
5. Yang, F., Lu, H.C., Yang, M.H.: Robust superpixel tracking. IEEE Trans. Image Process. **23**(4), 1639–1651 (2014)
6. Mei, X., Ling, H.B.: Robust visual tracking and vehicle classification via sparse representation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(11), 2259–2272 (2011)
7. Xiao, Z.Y., Lu, H.C., Wang, D.: L2-RLS based object tracking. IEEE Trans. Circuits Syst. Video Technol. **24**(8), 1301–1308 (2014)
8. Jia, X., Lu, H. C., Yang, M. H.: Visual tracking via adaptive structural local sparse appearance model. In: 25th IEEE Conference on Computer Vision and Pattern Recognition, pp. 1822–1829. IEEE Press, Providence (2012)
9. Babenko, B., Yang, M.H., Belongie, S.: Robust object tracking with online multiple instance learning. IEEE Trans. Pattern Anal. Mach. Intell. **33**(8), 1619–1632 (2011)
10. Grabner, H., Grabner, M., Bischof, H.: Real-time tracking via on-line boosting. In: 17th British Machine Vision Conference, pp. 47–56. Edinburgh (2006)
11. Yang, F., Lu, H.C., Yang, M.H.: Learning structured visual dictionary for object tracking. Image Vis. Comput. **31**(12), 992–999 (2013)

12. Zhang, K.H., Zhang, L., Yang, M.H.: Fast compressive tracking. IEEE Trans. Pattern Anal. Mach. Intell. **36**(10), 2002–2015 (2014)
13. Kalal, Z., Mikolajczyk, K., Matas, J.: Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach. Intell. **34**(7), 1409–1422 (2012)
14. Kwon, J, Lee, K. M.: Visual tracking decomposition. In: 23th IEEE Conference on Computer Vision and Pattern Recognition, pp. 1269–1276. IEEE Press, San Francisco (2010)
15. Fan, H., Xiang, J.H., Xu, J., Liao, H.H.: Part-based visual tracking via online weighted P-N learning. Sci. World J. pp. 13 (2014). Article ID 402158
16. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: 18th IEEE Conference on Computer Vision and Pattern Recognition, pp. 886–893. IEEE Press, San Diego (2005)
17. Ojala, T., Pietikäinen, M., Mäenpää, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary pattern. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 971–987 (2002)
18. Avidan, S.: Ensemble tracking. IEEE Trans. Pattern Anal. Mach. Intell. **29**(2), 261–271 (2007)
19. Oron, S., Bar-Hillel, A., Levi, D., Avidan, S.: Locally orderless tracking. In: 25th IEEE Conference on Computer Vision and Pattern Recognition, pp. 1940–1947. IEEE Press, Providence (2012)
20. Grabner, H., Leistner, C., Bischof, H.: Semi-supervised on-line boosting for robust tracking. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part I. LNCS, vol. 5302, pp. 234–247. Springer, Heidelberg (2008)
21. Zhang, K.H., Song, H.: Real-time visual tracking via online weighted multiple instance learning. Pattern recogn. **46**(1), 397–411 (2013)
22. Cortes, C., Mohri, M., Rostamizadeh, A.: Learning non-linear combinations of kernels. In: 23rd Advances in Neural Information Processing Systems, pp. 396–404. Vancouver (2009)
23. Yang, J., Li, Y., Tian, Y., Duan, L., Gao, W.: Group-sensitive multiple kernel learning for object categorization. In: 12th IEEE International Conference on Computer Vision pp. 436–443. IEEE Press, Kyoto (2009)
24. Jawanpuria, P., Varma, M., Nath, S.: On P-norm path following in multiple kernel learning for non-linear feature selection. In: 31st International Conference on Machine Learning, pp. 118–126. Beijing (2014)
25. Vedaldi, A., Gulshan, V., Varma, M., Zisserman, A.: Multiple kernels for object detection. In: 12th IEEE International Conference on Computer Vision, pp. 606–613. IEEE Press, Kyoto (2009)
26. Zhang, R., Wei, F., Li, B.: E2LSH based multiple kernel approach for object detection. Neurocomputing **124**, 105–110 (2014)
27. Sonnenburg, S., Ratsch, G., Schafer, C., Scholkopf, B.: Large scale multiple kernel learning. J. Mach. Learn. Res. **7**, 1531–1565 (2006)
28. Yang, F., Lu, H.C., Yang, M.H.: Robust visual tracking via multiple kernel boosting with affinity constraints. IEEE Trans. Circuits Syst. Video Technol. **24**(2), 242–254 (2014)
29. Rakotomamonjy, A., Bach, F., Canu, S., Grandvalet, Y.: SimpleMKL. J. Mach. Learn. Res. **9**, 2491–2521 (2008)
30. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**(3), 27:1–27:27 (2011)
31. Everingham, M., Gool, L.V., Williams, C.K., Winn, J., Zisserman, A.: The Pascal visual object classes (VOC) challenge. Int. J. Comput. Vision **88**(2), 303–338 (2010)