Computer Vision and Image Understanding 172 (2018) 25–36

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

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

Object tracking via Online Multiple Instance Learning with reliable components

Feng Wu, Shaowu Peng, Jingkai Zhou, Qiong Liu*, Xiaojia Xie

The School of Software Engineering, South China University of Technology, Guangzhou 510006, China

ARTICLE INFO

MSC:
41A05
41A10
65D05
65D17

Keywords:
Object tracking
Online multiple instance learning
Reliable component

ABSTRACT

Visual object tracking is a challenging and essential research problem in the field of computer vision. Recent years, many Online Multiple Instance Learning (MIL) tracking methods have been proposed with promising experimental results. These methods train a discriminative classifier under the boosting framework. The weak classifiers are learned from parts of the object and all classifiers are updated while updating the appearance model. However, due to irregular shape of object or occlusions, some components are not on object and should not be learned. On the contrary, a discriminative weak classifier learned from these components will mislead the tracker to drift away. To overcome this problem, we propose a novel online MIL tracking approach by updating with reliable components (OMRC). It keeps both background and object templates while tracking. By comparing current tracking result with two templates, we can get whether the pixels belong to object. The components which have a higher rate of pixels belong to object than a predefined threshold are reliable components. Moreover, in order to represent images better, we use HOG features and Histogram features instead of the widely used Haar-like features and design a new online weak classifier learning method. Experiments are performed on two challenging datasets including OTB2015 and Temple Color. Experimental results demonstrate the robustness of our OMRC tracker and the effectiveness of each component in the OMRC tracker.

1. Introduction

Visual tracking is an important research topic in computer vision with many practical applications such as human-computer interactive, surveillance and robotics, to name a few. Here, we consider the most general scenario of visual tracking, i.e., single-camera, short-term, model-free, single-object, causality tracking. For other scenarios such as long-term tracking (Kalal et al., 2012; Ma et al., 2015), multi-object tracking (Andriyenko and Schindler, 2011; Chen et al., 2014a) are not included in this study. Despite many object tracking methods have been proposed (Babenko et al., 2011; Ross et al., 2008), it still is a challenging problem to design a robust algorithm due to factors such as partial occlusions, background clutter, illumination and viewpoints changes.

In the tracking system, many tracking-by-detection approaches focus on the adaptive appearance model updating (Grabner and Bischof, 2006; Yang et al., 2014) and it is the key issue to object tracking (Li et al., 2013). In the updating procedure, they collect training samples according to tracking results and the aim is to maximize the separability between positive examples and negative examples. Due to tracking errors, those training samples may be polluted by noise. Recently, multi-instance model has been used to representation training samples, online multi-instance learning (MIL) tracking methods collect samples near tracking results as a positive bag and treat far away samples as a negative bag. This sampling strategy gives a more accurate description and leaves the flexibility to find the decision boundary.

Although many MIL trackers have been proposed in the recent years (Babenko et al., 2011; Chen et al., 2014b; Zhang et al., 2013). There are two aspects that are not fully studied and limit the performance.

Firstly, previous work updates all weak classifiers indiscriminately, which will introduce noise into the final appearance model. Bounding boxes are used to describe the object locations. Most MIL trackers under a boosting framework that training a pool of weak classifiers and select a subset of weak classifiers to construct the final appearance model. Each weak classifier is corresponding to a sub-region which we will call component in the bounding box. At each updating procedure, all the weak classifiers are updated. Note that the tracked object has irregular shape and the cluttered background may be included in the image patch even the tracking location is correct as shown in Fig. 1. Besides partial occlusions also cover some components in the bounding box. For those components, the corresponding classifier should not be updated. On the contrary, if a discriminative weak classifier is learned from these

* Corresponding author.
E-mail address: liuqiong@scut.edu.cn (Q. Liu).

https://doi.org/10.1016/j.cviu.2018.03.008
Received 21 September 2017; Received in revised form 10 March 2018; Accepted 21 March 2018
Available online 11 April 2018
1077-3142/

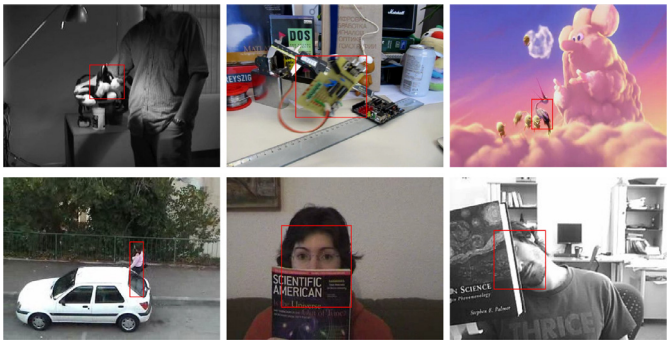


Fig. 1. Several examples that include cluttered background. The top row examples are due to irregular shapes and the bottom row examples are caused by the occlusions.

regions and selected by the final appearance model, it will mislead the tracker to drift away. So it is necessary to filter these components while updating the appearance model.

Secondly, features used in previous work are too weak. The Haar-like features and Gaussian model widely employed in MIL trackers cannot accurately describe the object. High dimensional features are required for more discriminative representation of images (Ning et al., 2016; Wang et al., 2015). HOG features have been used in Gra-MIL (Xie et al., 2012), but the updating of their weak classifier is bounded to the final appearance model updating. In order to learn weak classifiers flexibly, an independent online weak classifier updating method is necessary.

In this paper, we propose a novel online MIL tracking approach by updating the reliable components (OMRC). It consists of the following three parts. Firstly, we use HOG features and Histogram features to represent the images. For each component, we get two feature vectors and learn a linear classifier for each vector in an online schema. Secondly, in order to avoid being polluted by background components in classifier updating, we filter unreliable components by maintaining a background template and an object template respectively while tracking. After tracking the object in one frame, we first update the templates and get the pixels' mask that denotes the pixels belong to the object. Then update the components which have a higher rate of pixels belong to object than a predefined threshold. Thirdly, we use the weighted MIL (Zhang and Song, 2013) weak classifiers selection criterion to get final classifier that could be used to detect object in next frame. Besides, inspired by the part based models (Felzenszwalb et al., 2009; Zhao et al., 2016) where the object models consist of a coarse global template for entire object and higher resolution part templates, we always update and select two weak classifiers that cover the whole object. We conduct experiments on two challenging datasets with several other tracking approaches such as OAB, IVT, Frag, MIL and WMIL. Experimental results have demonstrated the robustness of the proposed OMRC tracking method. Besides, we compare OMRC with its two simplified version and numerical results demonstrate the effectiveness of each component in our method.

The main contributions of this paper are summarized as follows.

- We introduce component filter in the tracking system to filter unreliable components. It maintains a background template and an object template. While updating two templates, a pixels' mask is generated which indicates whether the pixels belong to object. The component classifiers that have a low rate of pixels on object will not be updated and selected. Besides, inspired by the part based models, we always update and select we classifiers that cover the whole rectangle.
- We propose an online learning algorithm to a batch weighted samples. A dual coordinate descent method is used to solve the optimization problem. It is used to learn the component weak classifiers. Then high dimensional features can be adopted in our tracking system. Although we use HOG features and Histogram features in this paper, other features such as color features and deep features can be used as well without any change.
- We conduct experiments on two challenging datasets. Experimental results have demonstrated the robustness of the OMRC tracker and the effectiveness of each component in the method.

The rest of this paper is organized as follows. Section 2 gives brief introductions of related work. In Section 3 we present the details of our algorithms. Experimental results are presented in Section 4. And the last section gives some concluding remarks.

2. Related work

Our work mainly relates to online MIL tracking and component filter. In the following, we review some important related works.

2.1. Online Multiple Instance Learning trackers

Adaptive tracking-by-detection methods update the appearance model after detecting object in one frame. Online boosting (OAB) tracker (Grabner and Bischof, 2006) introduces an online boosting framework for tracking. Each weak classifier corresponds to one feature under Gaussian distribution and OAB updates the parameters by a Kalman filter approach. The training samples in these methods have binary labels. The positive samples are chosen from tracker location or close locations while negative samples are chosen far away from tracker location. Structured SVM trackers (Hare et al., 2011; Ning et al., 2016) define a margin for each location and tracker location according to their overlap rate. Correlation filters (Bolme et al., 2010; Henriques et al., 2015) construct an output response map for each frame according to the distance between the position and tracker location center. However, those self-learning appearance models will accumulate errors and drift away from object when positive samples do not capture the object perfectly. MILTrack (Babenko et al., 2011) introduces the MIL concept into online tracking by modeling the samples close to tracker location as a positive bag and select features by maximizing the bag likelihood. After that, a lot of online boosting based MIL trackers have been proposed.

Some works focus on feature representation. Yan et al. (2015) integrate the sparse representation scheme into the online MIL tracker. They construct a sparse measurement matrix to extract the features. Gra-MIL (Xie et al., 2012) uses HOG features to represent image patch and learn Linear Discriminative analysis weak classifiers. They update weak classifiers in sequence instead of selecting weak classifiers. While Gra-MIL doesn't need to keep a large pool of classifiers, the final classifier contains all weak classifiers even if it is occluded.

The MIL trackers mentioned above treat each sample in positive bags equally without considering the importance of each sample which may cause the imprecise of the tracker. WMIL (Zhang and Song, 2013) gives higher weight to the samples that are closer to tracker location in a positive bag while treating samples equally in a negative bag. Semi-MILBoost (Chen et al., 2014b) treats samples in positive bag as unlabeled while the ones in the negative as negative. They iteratively update the pseudo-labels and importance of all samples under a boosting framework.

Some works focus on feature selection. ODFS (Zhang et al., 2013) selects features to aim at maximizing the difference between average confidence of samples in positive bag and average confidence of samples in negative bag. Fisher-MIL (Xu et al., 2015) proposes a new feature selection criterion via optimizing the trace of the Fisher information matrix of the bag.

Although these methods give promising results in their experiments, high dimensional feature and reliable components filter are helpful to improve the robustness of the tracker. Wang et al. (2015) break a

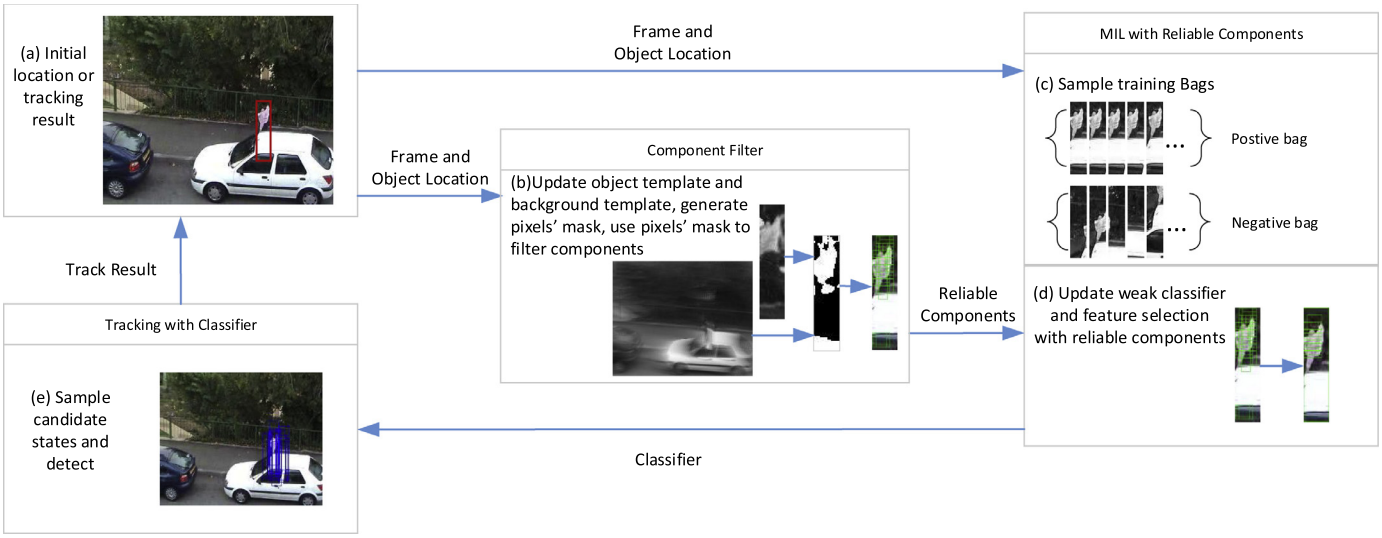


Fig. 2. The basic flow of our system. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

tracker down into five constituent parts and conduct ablative experiments on each component to study how it affects the overall result. They find that the feature extractor plays the most important role in a tracker. Besides, some components should not be learned because of the irregular shape of the object or occlusions. These two aspects will be explored in this paper.

2.2. Component filter

Usually, a bounding box is not full of object because of its irregular shape or partial occlusions. The regions that do not belong to the object will disturb the tracker if they are used to update the appearance model. So, we should find out the regions belong to object and update the appearance model with their features.

There are some generative models that try to distinguish the outlier from the image patch. Wang et al. (2013) and Mei and Ling (2009) handle partial occlusion with trivial templates. The outliers will have larger coefficients with corresponding trivial templates. But the spatial information among outliers is not considered. In another work of Wang and Lu (2014), PCOM models the difference between inliers and reconstruct result as Gaussian, and models the difference between outliers and reconstruct result as a uniform distribution, then a Graph cuts is adopted to separate inlier from outliers. ITWVST (Cruz-Mota et al., 2013) can exploit a priori knowledge about importance region of a tracked object. But it fixes the prior knowledge by predefined templates and this is usually unable to obtain in our situation where only the bounding box location of the first frame is given. These algorithms only focus on the object region without considering the background.

There are some part based discriminative models that try to find reliable patches. Frag (Adam et al., 2006) divides object template into patches. Instead of summing the score of each patch by some threshold, they choose the Qth score by assuming that occlusions will always leave at least a quarter of the target visible. Zhao et al. (2016) learn a weighted part model for object tracking. They adjust the weight of a part model base on the historical distribution to control the sample selections for part models updating, which makes different parts update differently due to partial occlusion or drastic appearance change. RPT (Li et al., 2015) attempts to identify and exploit the reliable patches that can be tracked effectively. It computes the probability of a patch lying on the tracked object according to the motion consistency. Lee et al. (2014) divide input image into patches and construct the foreground and background appearance models. While tracking, they generate a foreground probability map which is then convolved with the pertinence mask to suppress the effects of background clutters. SPT (Yang et al., 2014) constructs a superpixel-based appearance model to separate the target from the background. The above three algorithms rely on the color images. Although our method can use color information by adopting color features, here we also need to work well with grey images.

Different from the above methods that get the occlusions during detection, we get the occlusion information after tracking a frame and use the reliable components to guide model updating. In our online boosting MIL framework, we keep an object template and a background template. After detecting a frame, we have already get the object location. Then we compute the pixels' mask according to the distance to templates. Only the components that have a high rate of pixels on the object are considered as reliable and will be learned. Meantime, the templates are updated to adapt to object and background changes.

3. The proposed approach

In this section, we introduce our online multi-instance learning tracking system. We begin with an overview of the system and motion model. Next, we give a detail discussion about image representation and weak classifiers. We then describe our component filter. After that, we introduce the feature selection method. Finally, we discuss how using component filter deal with irregular shape and occlusions.

3.1. System overview and motion model

The basic flow of our tracking system is illustrated in Fig. 2. Let $l_t(x) \in R^2$ denote the left top coordinate of sample location x at the tth frame. Firstly, we assume that the location of sample x_0 is the object location which is given at the first frame or predicted by the tracker as Fig. 2(a). Secondly, we introduce a component filter process to find out reliable components as Fig. 2(b). We keep a background template and an object template. While updating templates, we obtain a pixels' mask that indicate whether a pixel is on the object. Only the components which have a high rate of pixels belong to object are the reliable components, i.e., the green rectangles in the last image. These are discussed in more detail in Section 3.3. Thirdly, we collect training samples for appearance model updating as Fig. 2(c). The positive bag consists of samples $X^{\alpha} = \{x | \|l_t(x) - l_t(x_0)\| < \alpha\}$ that are densely cropped centering at object location $l_t(x_0)$ within a search radius α. The negative bag consists of samples $X^{(\zeta,\beta)} = \{x | \zeta < \|l_t(x) - l_t(x_0)\| < \beta\}$ that are randomly cropped, where $\alpha < \zeta < \beta$. Fourthly, we only update and select the component classifiers that corresponding to reliable

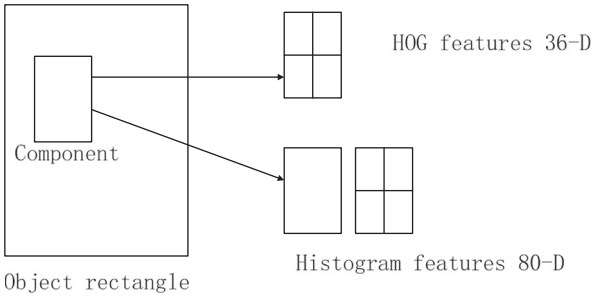


Fig. 3. Image representation.

components or the component that cover whole rectangle as Fig. 2(d). The image representation and component classifier updating are discussed in Section 3.2 and feature selection is discussed in Section 3.4. When the $t+1$ frame comes, we crop some samples $X^{\gamma} = \{x \| \|l_{t+1}(x) - l_t(x_0)\| < \gamma\}$ with a large radius surrounding the last object location as Fig. 2(e). Then, we apply the afore-trained classifier to these samples to find the sample with the maximum confidence as the new object location at $t+1$ frame. Based on the newly detected object location $l_{t+1}(x_0)$, our tracking system is to repeat the above-mentioned procedures.

3.2. Image representation and weak classifiers

Feature extractor plays the most important role in a tracker (Wang et al., 2015). Widely used Haar-like features are too simple to describe the object accurately. High dimensional features are required for better representation. We use HOG features like the work in Gra-MIL (Xie et al., 2012) and Histogram features. We generate M rectangles. For each rectangle, we obtain a HOG feature vector and a Histogram feature vector as Fig. 3. We obtain the HOG features by treating each rectangle as one block that contains 4 cells. For each cell, a 9 bins histogram of gradient magnitude at each orientation are computed, then we get a 36-D feature vector. We get the Histogram features by concatenating 16-bin intensity histograms from a spatial pyramid of 2 levels. At level L, the patch is divided into $L \times L$ cells, resulting in an 80-D feature vector.

For each rectangle and each feature representation, we learn a component classifier, the positive samples are extracted from the positive bag locations and the negative samples are extracted from the negative bag locations. Besides, we define a weight for each location in positive bag according to their distance to object location as Eq. (1) and treat each location in negative bag equal. For these samples, we pass the labels and weights from locations to the weak classifiers training examples.

$$C_j = \frac{1}{C} * \frac{1}{1 + \|l(x_j) - l(x_0)\|^2} \tag{1}$$

where C is a normalization constant, $l(x_0)$ and $l(x_j)$ are the locations of the object and jth example in positive bag respectively. The weighting function in WMIL is an exponential function, which gives too small weights for fringe positive samples. In our tracking system, a smoother weighting function is adopted.

To learn component classifiers in the online manner, we extend the Passive-Aggression algorithm to update several weighted examples one time. For each component classifier, we keep a vector w and the corresponding component classifier is $f(x) = w^T x$. The Object is to minimize the variety of vector w between two batches of continuous samples and predict the train examples with margins as follows

$$\begin{aligned} & w^{t+1} = \arg min_w \frac{1}{2} \|w - w^t\|^2 + \sum_{i=1}^{n} C_i \xi_i \\ & s.\ t. \qquad w^T x_i y_i \geq 1 - \xi_i,\ \xi_i \geq 0,\ i = 1, \ldots, n \end{aligned} \tag{2}$$

Note that the optimization problem is a standard quadratic programming problem, which has convex objective function and linear constraints. Therefore, we could use kernels rather than linear dot products by computing the dual form of the optimization problems. Here we use the linear kernel for fast computation. After applying the Lagrange dual, we can get the vector w^{t+1} and object function as Eq. (3) and Eq. (4).

$$w^{t+1} = w^t + \sum_{i=1}^{n} \alpha_i x_i y_i \tag{3}$$

$$\begin{aligned} L = & -\sum_{i,j=1}^{n} \alpha_i \alpha_j x_i^T x_j y_i y_j + \sum_{i=1}^{n} \alpha_i (w^{tT} x_i y_i - 1) \\ s.\ t. \qquad & 0 \leq \alpha_i \leq C_i,\ i = 1, \ldots, n \end{aligned} \tag{4}$$

To solve the dual form, we use the dual coordinate descent method. First, we select single α_i that do not satisfy the KKT conditions. Then optimize α_i by $\frac{\partial L}{\partial \alpha_i} = 0$ under the constraint $0 \leq \alpha_i \leq C_i$, i.e. $\alpha_i = \max\left(0, min\left(C_i, \frac{-2\sum_{j \neq i} \alpha_j x_i^T x_j y_i y_j + w^{tT} x_i y_i - 1}{2 x_i^T x_i}\right)\right)$. We optimize variables literately until the difference between the two iterations is less than a threshold.

3.3. Component filter

For each component we train two weak classifiers (one for HOG features and the other for Histogram features). It is not reasonable to update a weak classifier when the corresponding component contains cluttered background. In order to find out the components that contain the object, we introduce the component filter process. There are three steps in the component filter process, illustrated as Fig. 4. Firstly, we update background template using both current frame F_t and the last background template B_{t-1}. Then, we generate pixels mask M_t by comparing current tracking result $F_t^{L_t(x_0)}$ with corresponding region $B_t^{L_t(x_0)}$ of background template and the last object template O_{t-1}. At last, we use pixels mask M_t, tracking result $F_t^{L_t(x_0)}$ and the last object template O_{t-1} to obtain a new object template O_t. While updating the background template and object template we get pixels' mask M_t. For each component we calculate the rate of pixel belong to object and only update and select the components that have a rate higher than a threshold τ.

Before discussing the details of each step, we give the definition and initialization of some symbols. Let $B_t \in R^{W*H}$ and $F_t \in R^{W*H}$ represent the background template and image at frame t respectively, where W and H are the width and height of the frame. we use $G_t \in R^{W*H}$ to represent the copy of frame F_t with pixels in object location set to -1 that means we cannot get the value of these pixels. Let $O_t \in R^{w*h}$ and $M_t \in R^{w*h}$ denote the object template and pixels' mask at frame t respectively, where w and h are the width and height of the object rectangle. Let $L = (0, 0, W, H)^T$ and $L_t(x_0) = (l_t(x_0), w, h)^T$ represent frame rectangle and object rectangle. we use $v \in R^2$ to represent a movement and $F_t^{L \circ v}$ to denote the image crop at new location $L \bigcirc v$ from image F_t.

At the first frame, background template B_1 is initiated by the first frame F_1 with the values in object rectangle are set to -1. The object template O_1 is initialized by the object patch $F_1^{L_1(x_0)}$. The pixels' mask M_1 is initialized as ones means all pixels belong to object.

3.3.1. Update background template

At frame t, given the last background template B_{t-1} and current frame F_t, we obtain the new background template B_t. We assume background changes little and moves within a radius (25 in our experiments). Under the translation $v^* \in R^2$, we get a minimum average of absolute intensity difference. Base on this assumption, we first get the movement of background v^* as Eq. (5).

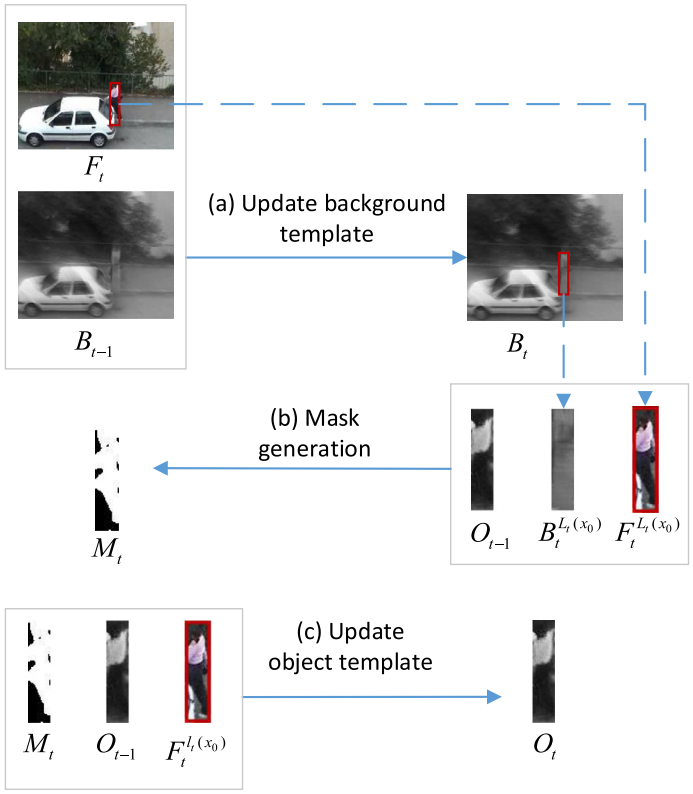


Fig. 4. The steps of update background template, target template and pixel mask: (a) update background template B_t, (b) generate pixels' mask M_t, (c) update object template O_t.

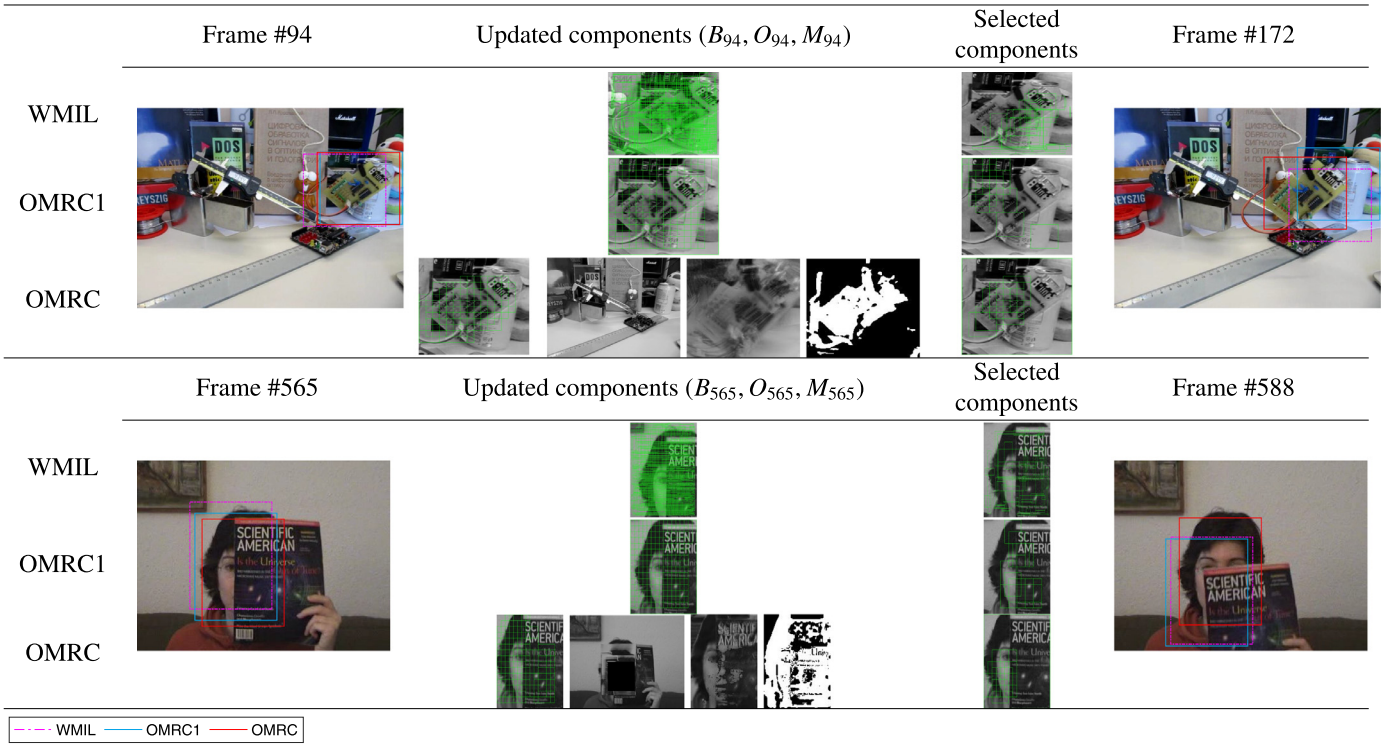


Fig. 5. Two illustration of how using component filter to deal with object patch polluted by cluttered background. The top case is from Board sequence due to irregular shape and the bottom case is from FaceOcc1 sequence caused by occlusion.

$$v^* = \arg min_v \sum_S \frac{1}{|S|} |B_{t-1}^{Lov} - G_t^L| \tag{5}$$

where B_{t-1}^{Lov} is the image crop from B_{t-1} at location $L \bigcirc v$ and G_t^L is the image crop from F_t at location L. S is the region where pixels have nonnegative pixel values. We do not count the pixels that we cannot get the pixel values. Consider calculation efficiency, we only use a region

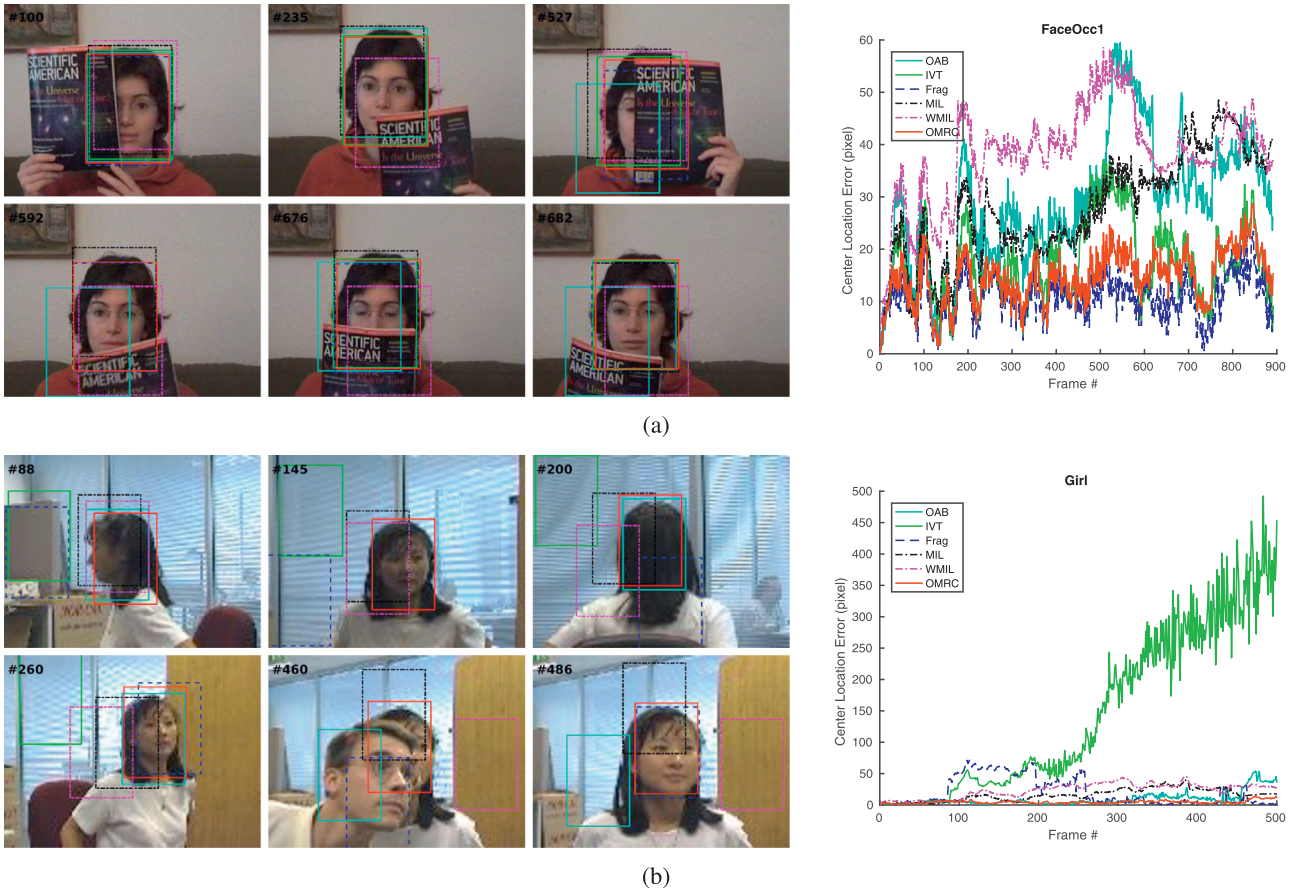


Fig. 6. Sampled tracking results and center location error plots for tested sequences. (a) FaceOcc1. (b) Girl.

centered at object location and the region size is 6^2 times the areas of object rectangle.

Secondly, we get the new background template B_t using G_t, $B_{t-1}$and v^* as Eq. (6).

$$B_t = (1-\lambda)B_{t-1}^{L\circ v^*} + \lambda G_t^L \tag{6}$$

where λ is the update rate. The new background B_t is alignment to current frame. There are some cases that the pixel values are missing. If the pixel value in B_{t-1} is -1, we use the value in G_t directly and if the pixel value in G_t is -1, we use the value in B_{t-1}.

3.3.2. Mask generation

We compute the pixels' mask M_t according to local average distance from $F_t^{L_t(x_0)}$ to $B_t^{L_t(x_0)}$ and O_{t-1} as Eq. (7). The value of a pixel in current frame close to object template indicated that the pixel is on the object and the pixel mask is 1.

$$M_t(i,j) = \begin{cases} 1 & if |F_t^{L_t(x_0)}(i,j) - B_t^{L_t(x_0)}(i,j)| \geq |F_t^{L_t(x_0)}(i,j) - O_{t-1}(i,j)| \\ 0 & otherwise \end{cases} \tag{7}$$

Note that the values of some pixels in background template may still miss. For these pixels, we divide the object rectangle into 4 cells and get the average and standard deviation of the absolute difference between object template and current location. The minimum sum of average and standard deviation is threshold to decide whether these pixels on object. That is based on the assumption at least $\frac{1}{4}$ region belong to the object.

3.3.3. Update object template

We update the target template O_t using current tracking result $F_t^{L_t(x_0)}$ and pixels' mask M_t as Eq. (8).

$$O_t(i,j) = \begin{cases} (1-\lambda)O_{t-1}(i,j) + \lambda F_t^{L_t(x_0)}(i,j) & if\ M_t = 1 \\ O_{t-1}(i,j) & otherwise \end{cases} \tag{8}$$

3.4. Feature selection

In our tracking system, we use J components and maintain a pool of $2J$ component classifier h_i, $i = 1, \cdots, 2J$. Each classifier is updated by the online learning method presented in Section 3.2, where the positive examples come from positive bag and the negative examples come from the negative bag. The weights of instances not only control the relative importance of each instance but also influence the update rate of component. We normalize the weights for all positive examples to 0.5, and the same for negative examples. The output of component classifiers is in the interval $(-\infty, +\infty)$. Then we model the example probability to be positive as $p(y = 1|x) = \sigma(f(x))$, where $\sigma(z) = 1/(1 + e^{-z})$ is a sigmoid function.

Boosting feature selection is widely used in MIL tracking methods. It chooses a number of classifiers from the weak classifier pool to construct the final tracking model. In our tracking system, we let N and L denote the number of instance in positive bag and negative bag respectively, then the training set is $\{(X^{\alpha}, 1), (X^{(\zeta,\beta)}, 0)\}$ containing two bags, where $X^{\alpha} = \{x_0, x_1, \cdots, x_{N-1}\}$ and $X^{(\zeta,\beta)} = \{x_N, x_{N+1}, \cdots, x_{N+L-1}\}$.

Here we use the feature selection method in WMIL (Zhang and Song, 2013) that assumes all instances in positive bag have different weights according to the distance to tracker location while all instances in negative bag contribute equally as Eqs. (9) and (10). Note that other feature selection methods such as the methods used in Babenko et al. (2011), Zhang et al. (2013) and Xu et al. (2015) also can be adopted. The instance weights in positive bag are defined as Eq. (1) in our tracking system.

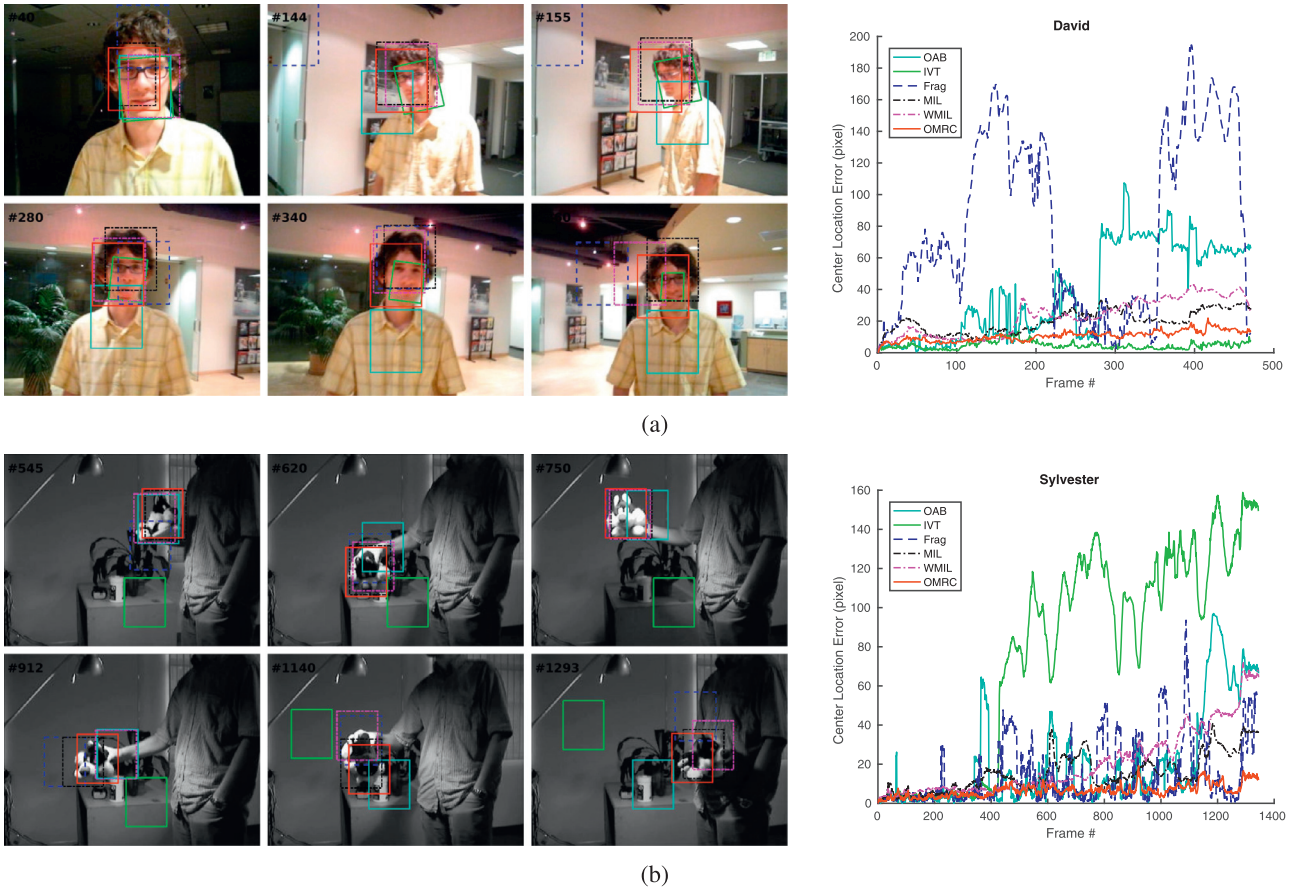


Fig. 7. Sampled tracking results and center location error plots for tested sequences. (a) David. (b) Sylvester.

$$p(y=1|X^{\alpha}) = \frac{1}{\sum_{j=0}^{N-1} C_j} \sum_{j=0}^{N-1} C_j p(y_j = 1|x_j) \tag{9}$$

$$p(y=0|X^{(\zeta,\beta)}) = \sum_{j=N+1}^{N+L-1} \frac{1}{L}(1 - p(y_j = 1|x_j)) \tag{10}$$

WMIL greedily selects K component classifiers by maximizing the log likelihood of bags and construct the strong instance classifier $H(x) = \sum_{k=1}^{K} h_k(x)$ as follows

$$h_k = \arg max_{h \in \{h_1, \cdots, h_M\}} L(H_{k-1} + h) \tag{11}$$

where the log likelihood of bags is defined as Eq. (12).

$$L = log(p(y=1|X^{\alpha})) + log(p(y=0|X^{(\zeta,\beta)})) \tag{12}$$

The OMRC tracker is similar to part based tracker by treating each component classifier as a part tracker. The component filter aims to remove the disturbance of background clutter and the feature selection is trying to find a discriminative subset of trackers. Inspired by the work of part based detection method (Felzenszwalb et al., 2009) and part based tracking methods (Yao et al., 2013; Zhao et al., 2016) where object models consist of a coarse global template for entire object and higher resolution part templates, our model always update and select the two component classifiers learned from the global template. For the rest classifiers, we select $K - 2$ classifiers.

3.5. Discuss

To illustrate how component filter can deal with object patch polluted by cluttered background, we compare the OMRC tracker with one of its simplify version named OMRC1 and the WMIL tracker. The OMRC1 method is the same as OMRC except that OMRC1 do not contain component filter and global classifiers. For the WMIL tracker, we plot all rectangles in the Haar-like features. The details are presented in Fig. 5. The top case is from the Board sequence where the object has irregular shape in a static background. It is obviously all trackers can catch up with object at frame #94. The OMRC1 tracker updates all component classifiers and selects some component classifiers from the background region, which makes it stay when object leaving at frame #170. The WMIL tracker updating all weak classifiers that cover whole object location and the selected classifiers also contain too much background region as the OMRC1 tracker. The OMRC tracker only updates the component classifiers that on the object by two templates and pixels' mask, and the selected classifiers still focus on the object. The bottom case is from the FaceOcc1 sequence where the object has partial occlusions. It is obviously both trackers can catch up with object at frame #565. The OMRC1 tracker updates all component classifiers and selects some component classifiers from the book region, which makes it drift away when the book leaves at frame #586. The selected weak classifiers of WMIL also contain too much occluded region as the OMRC1 tracker. Though the OMRC tracker also updates some component classifiers belong to the book here, it still prevents some component classifiers updating at every occluded frame, and it makes classifiers from occluded region hard to be selected.

4. Experiments

In this section, we mainly compare our MIL-based tracking method with five trackers. The Five tracker are Online Adaboost (OAB) tracker (Grabner and Bischof, 2006), IVT (Ross et al., 2008) tracker, fragments-based (Frag) (Adam et al., 2006) tracker, Online Multiple Instance Learning (MIL) tracker (Babenko et al., 2011), Online Weighted Multiple Instance Learning (WMIL) tracker (Zhang and Song, 2013). The

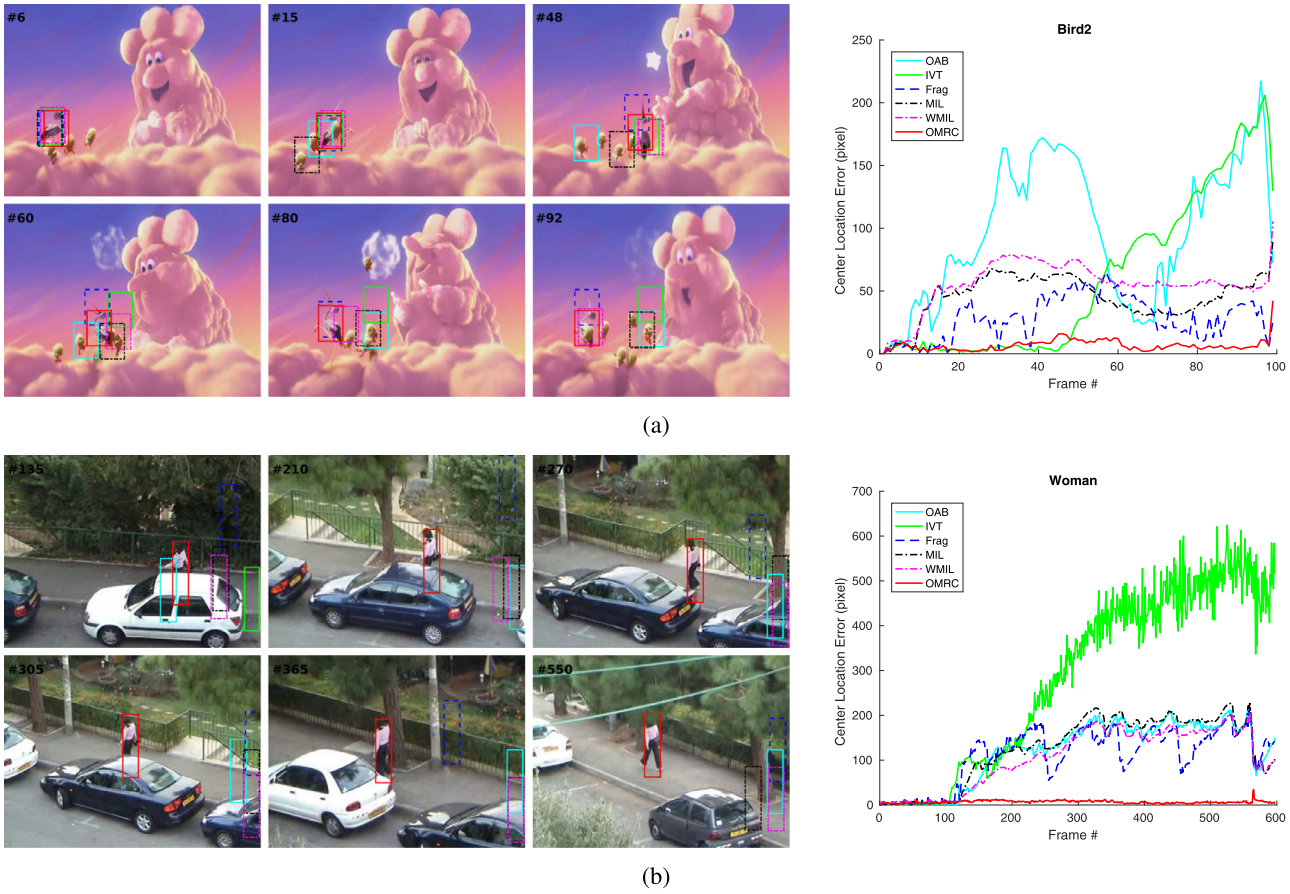


Fig. 8. Sampled tracking results and center location error plots for tested sequences. (a) Bird2. (b) Woman.

proposed OMRC tracker is implemented in MATLAB and runs at 3.6 fps on a PC with a 3.5GHz CPU. The experiments include three parts. Firstly, results on eight typical challenging sequences are analyzed. Secondly, we conduct the experiment on the OTB2015 benchmark (Wu et al., 2015) with 100 sequences. Not only one pass evaluation (OPE) results are presented, the robustness to initialization and the performances at different attributes are also discussed. In addition, we compare the OMRC tracker with four state of the art method on four attributes. Then, we compare OMRC with two simplified versions to verify the effectiveness of each component in our tracking system. Thirdly, we perform experiments on the Temple Color dataset with 128 videos.

4.1. Parameter configurations

In our method, the samples search radiuses α, ζ, β, γ are set to 4, 6, 50, 25 respectively. The number of samples in negative bag is limited to 50. The threshold of component on object τ is set to 0.5. The number of components J is set to 150 and the number of selected weak classifiers K is set to 15. Unlike other MIL trackers, we generate the component rectangle location through sliding window. At level L, the size of window is set to $\frac{1}{2}^L$ of object rectangle and step size is set to $\frac{1}{4}$ of window size. Besides, in order to avoid interference from small components, we limit the minimum length of the component rectangles to 12. For the compared trackers, we use the source codes by authors with default parameter settings.

4.2. Individual sequence analysis

We first compare the OMRC tracker with five trackers mentioned above on eight sequences, including FaceOcc1, Girl, David, Sylvester, Bird2, Woman, Board, Toy. These sequences contain several typical challenges in tracking and have been tested in many studies. For each video sequence, we present the screenshots and location error analysis. Screenshots give intuitive tracking results. Location error analysis plotting the center location error versus frame number. Because the randomness in sampling, we conduct the experiment on each sequence 10 times for the MIL, OAB, IVT, WMIL and OMRC trackers, and present the average result on location error analysis.

4.2.1. FaceOcc1 and Girl

These two sequences exhibit occlusion. In general, the OAB tracker and the MIL tracker and the WMIL tracker perform worse than other algorithms because these algorithms do not consider the occluded situation. They select the weak classifiers mainly based on current frame. The IVT tracker works well in FaceOcc1 sequence because the eigenbasis cover history information. But it drifts away at frame #88 in Girl sequence because the view of face changes significantly. The Frag tracker works well in FaceOcc1 sequence while drifts away at frames #88, #145, #200 and #460 in Girl sequence. Compared with these five trackers, our OMRC tracker works well in both sequences (Fig. 6).

4.2.2. David and Sylvester

These two sequences have illumination variation, in-plane rotation and out-of-plane Rotation. The IVT tracker performs best in David sequence due to its scale adaptation. For the other cases, our OMRC tracker performs better than other five trackers significantly (Fig. 7).

4.2.3. Bird2 and Woman

These two sequences comprise occlusion and deformation. For the Bird2 sequence, the main challenge is deformation. The OAB and MIL trackers are easy to drift since the target begin deformation (see frames

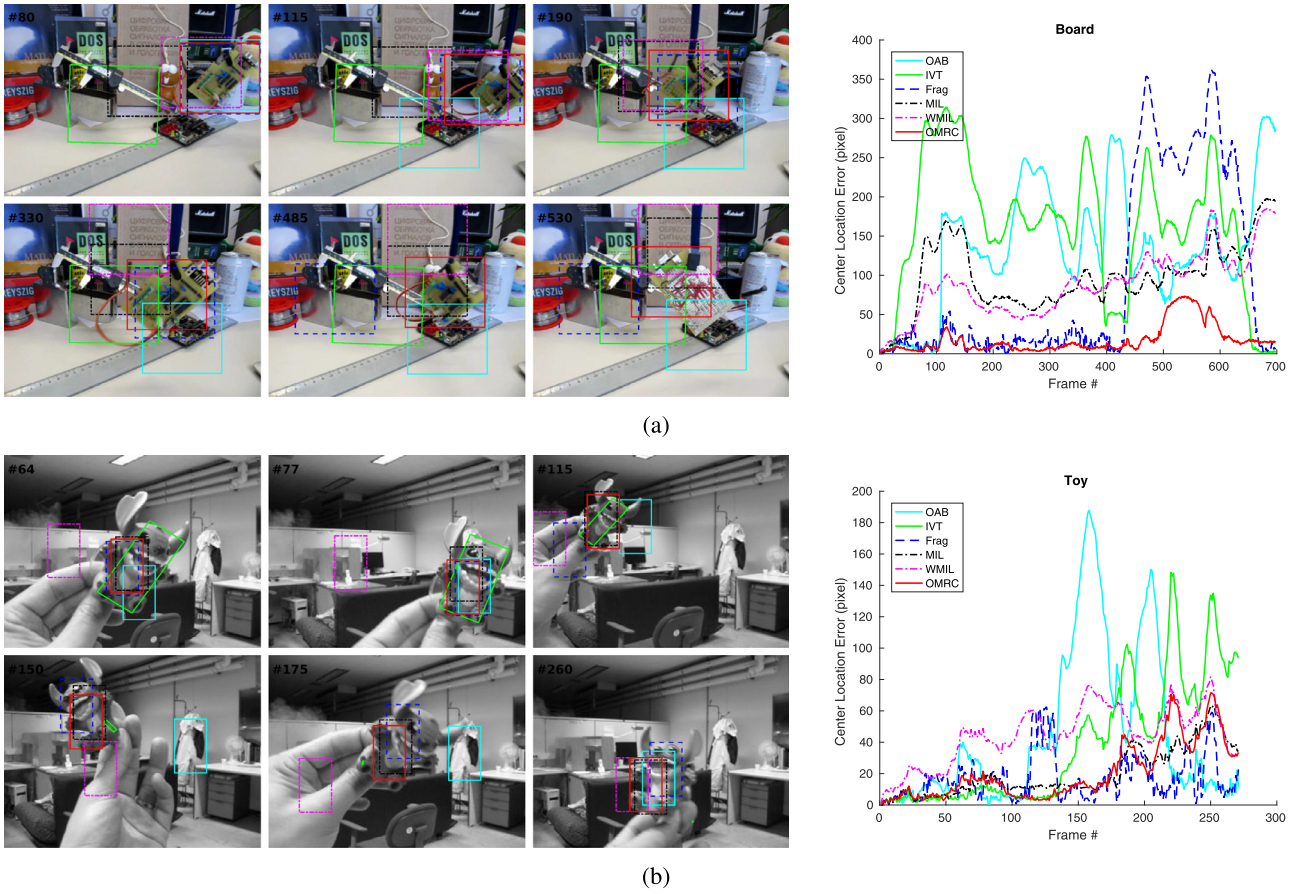


Fig. 9. Sampled tracking results and center location error plots for tested sequences. (a) Board. (b) Toy.

Table 1
Compare the OMRC tracker with five trackers on AUC, success rate score (SR) and precision score (P). The best results on each attribute are highlighted in bold.

	MIL	OAB	Frag	IVT	WMIL	OMRC
AUC	0.336	0.368	0.335	0.315	0.293	**0.406**
SR	0.333	0.407	0.363	0.366	0.276	**0.450**
P	0.450	0.488	0.426	0.432	0.384	**0.571**

#15, #48, #60, #80 and #92 of Bird2 sequence). The WMIL and Frag trackers drift away at frames #48, #60 and #92. The IVT tracker drifts away after the target turn around (see frames #60, #80 and #92 of Bird2 sequence). For the Woman sequence, there also exist motion blur and illumination variation. All of these five trackers perform worse (see all frames of Woman sequence in Fig. 8(b)). In general, our OMRC tracker performs better than above five trackers significantly.

4.2.4. Board and Toy

These two sequences contain scale variation, fast motion and out-of-plane rotation. For Board sequence, the IVT tracker models the background region as object since frame #28. The OAB, Frag, MIL and WMIL trackers often drift away. For Toy sequence, the object has obviously scale variation. The IVT tracker shrinks to a small region and drifts away (see frames #330 and #485 of Toy sequence). The OAB tracker and WMIL tracker often drift away. But the Frag tracker and the MIL tracker can keep up with the object. For these two sequences, our OMRC also performs better than above five trackers (Fig. 9).

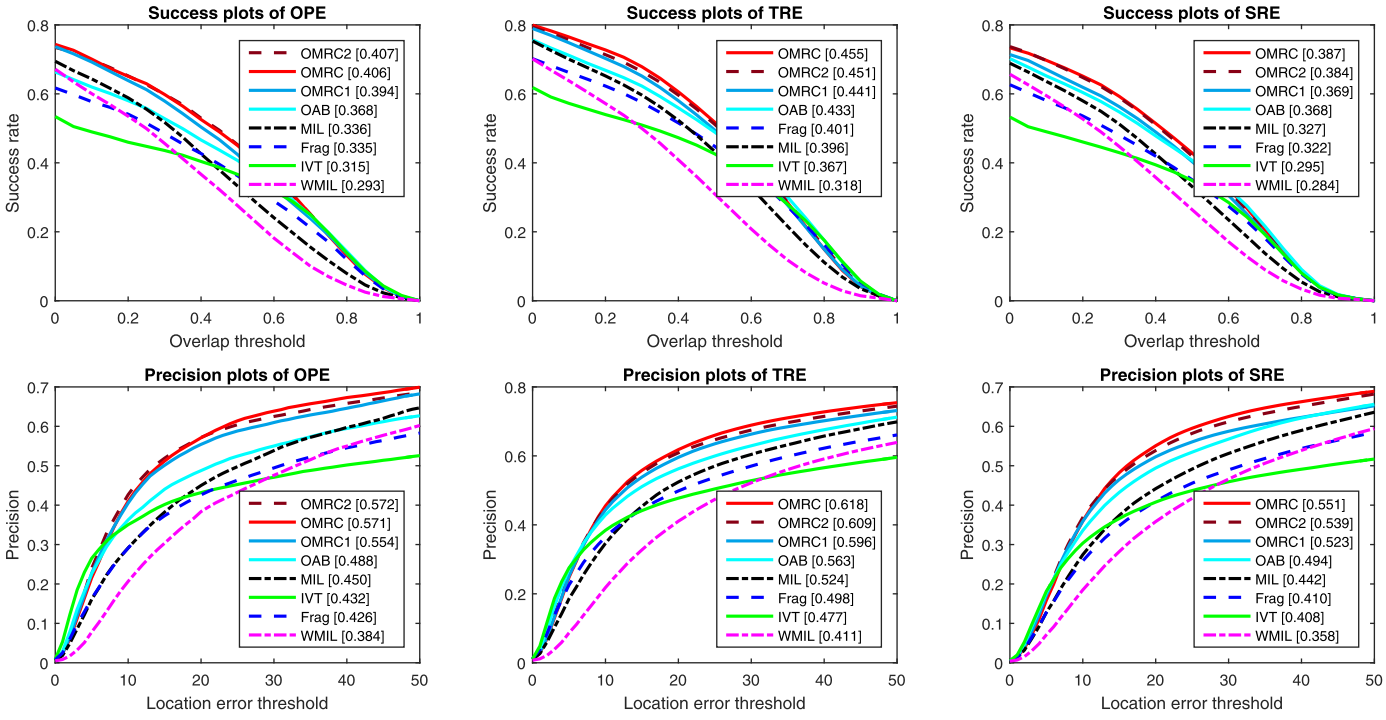
4.3. Experiment on OTB2015 dataset

We compare the OMRC tracker with five trackers mentioned above in OPE where the tracker is initialized in the first frame and track the object throughout a sequence. The quantitative results are summarized in Table 1, and plots are shown in the first column in Fig. 10. In Table 1, We report the Area Under the Curve (AUC), success rate score at 0.5 overlap rate and the precision score at 20 pixels. Among the compared methods, the OAB tracker gets the best performance. It obtains an AUC of 0.368, a success rate score of 0.407 and a precision score of 0.488. Our OMRC tracker significant outperforms the best compared tracker on three criteria by achieving an AUC of 0.406, a success rate score of 0.450 and a precision score of 0.571. All indicators are increased by more than 10.

In Fig. 10, the experimental results are illustrated by both success plots and precision plots. Success plots draw the percentages of frames for which the overlap ratio between estimated location and ground truth higher than some thresholds. Precision plots show the percentage of frames for which the estimated object location is within some threshold distance of the ground truth. The OPE plots in the first column in Fig. 10 show that our OMRC tracker outperforms the five compared trackers.

4.3.1. Robustness to initialization

To evaluate the robustness of the OMRC tracker to initialization, we adopt the temporal robustness (TRE) and spatial robustness evaluation (SRE) as Wu et al. (2015). TRE generates several subsequences from a test sequence at different time periods and runs the trackers in each subsequence with ground truth bounding box in the first frame. SRE runs the trackers in a whole sequence with different bounding box initialization by sifting or scaling the ground truth. Both criteria tally the



(a)

Fig. 10. The success and precision plots of the OTB2015 benchmark. Algorithms are ranked by the area under the curve and the precision score at 20 pixels.

Table 2
Precision score at 20 pixels on different attributes: illumination variation (IV), scale variation (SV), occlusion (OC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out-of-view (OV), background clutters (BC) and low resolution (LR). The best results on each attribute are highlighted in bold.

	OAB	IVT	Frag	MIL	WMIL	OMRC
IV	0.413	0.427	0.313	0.338	0.399	**0.557**
SV	0.471	0.410	0.388	0.455	0.386	**0.544**
OCC	0.433	0.407	0.388	0.426	0.357	**0.534**
DEF	0.410	0.332	0.403	0.464	0.386	**0.571**
MB	**0.450**	0.200	0.367	0.281	0.228	0.426
FM	0.475	0.222	0.380	0.359	0.279	**0.488**
IPR	0.481	0.438	0.415	0.477	0.389	**0.565**
OPR	0.469	0.430	0.436	0.475	0.397	**0.559**
OV	0.357	0.316	0.378	0.384	0.293	**0.502**
BC	0.420	0.437	0.376	0.409	0.389	**0.597**
LR	0.510	0.542	0.424	0.511	0.436	**0.678**

Table 3
Success rate score at 0.5 overlap rate on different attributes: illumination variation (IV), scale variation (SV), occlusion (OC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), out-of-view (OV), background clutters (BC) and low resolution (LR). The best results on each attribute are highlighted in bold.

	OAB	IVT	Frag	MIL	WMIL	OMRC
IV	0.339	0.375	0.287	0.263	0.286	**0.448**
SV	0.346	0.320	0.302	0.294	0.245	**0.364**
OCC	0.366	0.353	0.341	0.331	0.263	**0.439**
DEF	0.350	0.276	0.344	0.368	0.295	**0.447**
MB	**0.431**	0.191	0.378	0.215	0.193	0.375
FM	**0.426**	0.205	0.345	0.271	0.225	0.417
IPR	0.395	0.346	0.337	0.330	0.281	**0.449**
OPR	0.375	0.342	0.357	0.343	0.272	**0.449**
OV	0.334	0.275	0.346	0.328	0.255	**0.434**
BC	0.360	0.371	0.330	0.371	0.310	**0.537**
LR	0.218	**0.420**	0.231	0.212	0.158	0.232

overall statistics. The performances of the trackers are plotted in the middle column and right column in Fig. 10. Compared with the five trackers, the OMRC tracker achieves robust in both cases obviously.

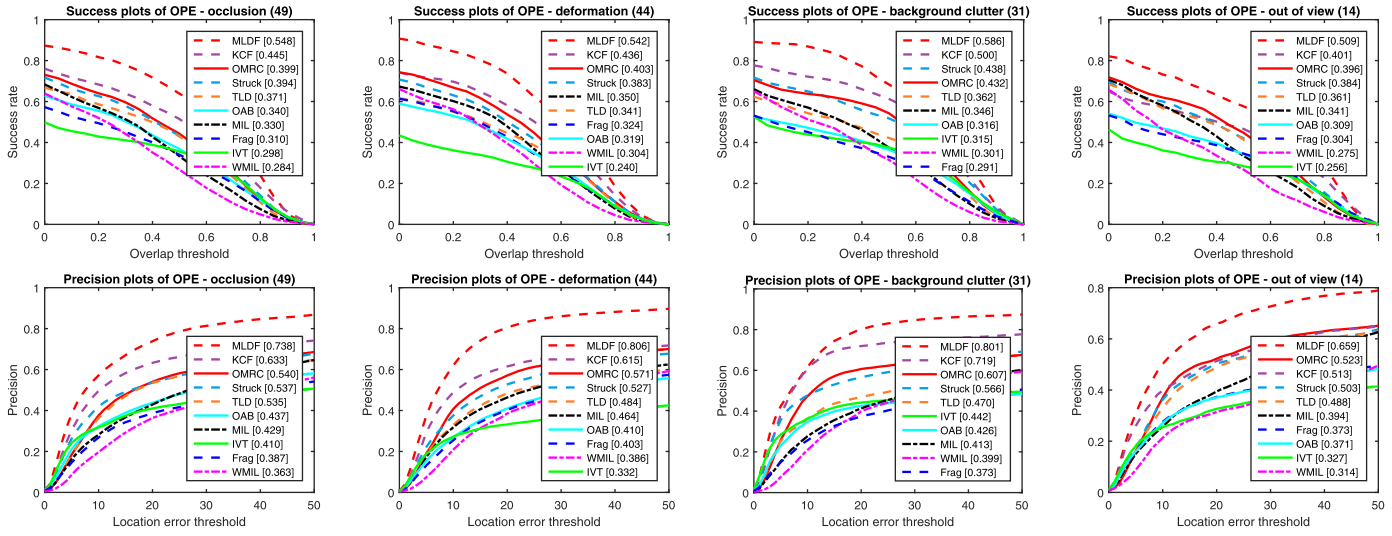
4.3.2. Attribute based analysis

The test sequences in OTB2015 are tagged with 11 attributes which represent the challenging aspects in object tracking. To have an overview how the trackers work with different challenges, we report the precision score at 20 pixels and success rate score at 0.5 overlap rate of 11 attributes in Table 2 and Table 3 respectively. These results are counted in OPE. In general, on precision score at 20 pixels in Table 2, the OMRC tracker gets the best performances at 10 attributes and is second to the OAB tracker at the motion blur attribute. On the success rate score in Table 3, the OMRC tracker gets the best performances at 8 attributes and get the second best results at the rest three attributes. The OMRC tracker focus on handle occlusions and irregular shape. The irregular shape is a common situation while tracking and is more obvious in the cases such as deformation, rotation, out-of-view and background clutters. But in the case of fast motion, the search radius in the motion model of OMRC limits its performance. These are in consist with the experimental results.

In addition, we compare our OMRC tracker with four state of the art methods at four attributes in OPE. The four trackers are the track-learning-detection (TLD) tracker (Kalal et al., 2012), Structure output (Struck) tracker (Hare et al., 2011), kernelized correlation filter (KCF) tracker (Henriques et al., 2015) and multi-level deep feature (MLDF) tracker (Kristan et al., 2016). We report the success plots and precision plots in Fig. 11. The OMRC tracker performs worse than the MLDF tracker and the KCF tracker, but it outperforms the Struck tracker and the TLD tracker. The MLDF tracker gets the best performance, which is mainly attributed to the deep features. It is worth to mention that our approach can also take the advantages of deep features to improve the performance.

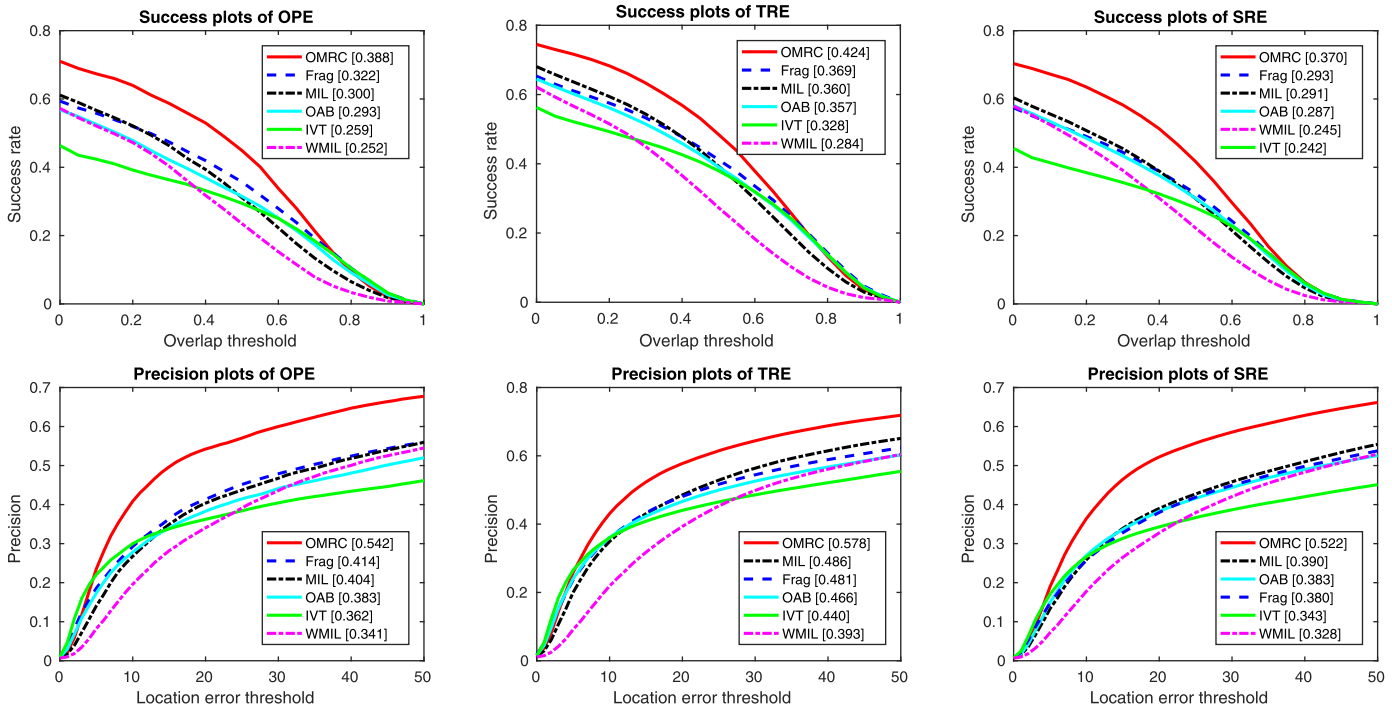
4.3.3. Compare OMRC with two simplified versions and WMIL

To prove the effectiveness of each component in OMRC, we



(a)

Fig. 11. Attribute-based analysis of our approach on the OTB-2015 dataset. Success plots are shown for four attributes where occlusion and irregular shape are a common problem. The title of each plot indicates the number of videos labeled with the respective attribute.



(a)

Fig. 12. The success and precision plots of the Temple Color benchmark. Algorithms are ranked by the area under the curve and the precision score at 20 pixels.

compare OMRC with its two simplified versions and the base tracker WMIL. The first simplified version named OMRC1 only use HOG features and Histogram features with WMIL feature selection as Section 3.5. The second simplified version named OMRC2 is constructed by adding component filter to OMRC1. It is intuitive to compare the precision plots and success plots of in Fig. 10. The OMRC1 tracker performs better than WMIL and the OMRC2 tracker outperforms OMRC1 in each plot. Although the OMRC tracker performs similarly to the OMRC2 in OPE, the OMRC tracker outperforms the OMRC2 tracker in SRE and TRE obviously. In general, each component of our OMRC tracker is effective.

4.4. Experiment on Temple Color dataset

Besides, we perform experiments on the Temple Color (Liang et al., 2015) dataset with 128 videos. The experimental results are shown in Fig. 12. Among the compared methods, the Frag tracker gets the best performance and provides an AUC of 0.322 and a precision score of 0.414 in OPE. OMRC tracker significant outperforms the best compared tracker by achieving an AUC of 0.388, and a precision score of 0.542. All indicators are increased by more than 20%. Meantime, our approach achieves robustness in SRE and TRE, leading to a consistent performance gain.

5. Conclusion

In this paper, we propose a novel online multi-instance learning with reliable components (OMRC) tracker. The OMRC tracker uses HOG features and Histogram features to represent each component and learn two component classifiers for each component. Besides, OMRC introduces a component filter process to find out unreliable components classifiers due to the irregular shape of object or occlusions. It maintains an object template and a background template. While updating two templates, a pixels' mask is generated which indicates whether the pixels belong to object. The component that has a low rate of pixels on object is a reliable component. Then, the OMRC tracker only updates and selects the weak classifiers from the reliable components except that the classifiers from the whole rectangle are always updated and selected. Experimental results on two challenging datasets have demonstrated our OMRC tracker performs better than the OAB, Frag, IVT, MIL and WMIL trackers and the effectiveness of each component in our OMRC tracker. Moreover, when almost all pixels close to background template, either the tracker drifts way or the object is occluded completely. This case will be explored in future work.

Acknowledgment

This research is supported by Science and Technology Planning Project of Guangdong Province, China under Grant No. 2017A020219008 and No. 2014B040401010, Guanddong Province Industry-Academia-Research Project under Grant No. 2017A020219008, South China University of Technology Project, Central Universities Basic Research Service Fees under Grant No. x2rjD2174870.

References

Adam, A., Rivlin, E., Shimshoni, I., 2006. Robust fragments-based tracking using the integral histogram. Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on. pp. 798–805.
Andriyenko, A., Schindler, K., 2011. Multi-target tracking by continuous energy minimization. Computer Vision and Pattern Recognition. pp. 1265–1272.
Babenko, B., Yang, M.H., Belongie, S., 2011. Robust object tracking with online multiple instance learning. IEEE Trans. Pattern Anal. Mach.Intell. 33 (8), 1619–1632.
Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M., 2010. Visual object tracking using adaptive correlation filters. Computer Vision and Pattern Recognition. pp. 2544–2550.
Chen, S., Fern, A., Todorovic, S., 2014a. Multi-object tracking via constrained sequential labeling. IEEE Conference on Computer Vision and Pattern Recognition. pp. 1130–1137.
Chen, S., Li, S., Su, S., Tian, Q., Ji, R., 2014b. Online mil tracking with instance-level semi-supervised learning. Neurocomputing 139 (139), 272–288.
Cruz-Mota, J., Bierlaire, M., Thiran, J.P., 2013. Sample and pixel weighting strategies for robust incremental visual tracking. IEEE Trans. Circuits Syst. Video Technol. 23 (5), 898–911.
Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D., 2009. Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. 32, 1627–1645.
Grabner, H., Bischof, H., 2006. On-line boosting and vision. IEEE Computer Society Conference on Computer Vision and Pattern Recognition. pp. 260–267.
Hare, S., Saffari, A., Torr, P.H.S., 2011. Struck: structured output tracking with kernels. IEEE International Conference on Computer Vision. pp. 263–270.
Henriques, J.F., Caseiro, R., Martins, P., Batista, J., 2015. High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. 37 (3), 583–596.
Kalal, Z., Mikolajczyk, K., Matas, J., 2012. Tracking-learning-detection. IEEE Trans. Pattern Anal. Mach.Intell. 34 (7), 1409–1422.
Kristan, M., Leonardis, A., Matas, J., 2016. The visual object tracking vot2016 challenge results. In: Hua, G., Jégou, H. (Eds.), Computer Vision – ECCV 2016 Workshops. Springer International Publishing, Cham, pp. 777–823.
Lee, D.Y., Sim, J.Y., Kim, C.S., 2014. Visual tracking using pertinent patch selection and masking. Computer Vision and Pattern Recognition. pp. 3486–3493.
Li, X., Hu, W., Shen, C., Zhang, Z., Dick, A., Hengel, A.V.D., 2013. A survey of appearance models in visual object tracking. ACM Trans. Intell. Syst. Technol. 4 (4), 58.
Li, Y., Zhu, J., Hoi, S.C.H., 2015. Reliable patch trackers: robust visual tracking by exploiting reliable patches. Computer Vision and Pattern Recognition. pp. 353–361.
Liang, P., Blasch, E., Ling, H., 2015. Encoding color information for visual tracking: algorithms and benchmark. IEEE Trans. Image Process. 24 (12), 5630–5644.
Ma, C., Yang, X., Zhang, C., Yang, M.H., 2015. Long-term correlation tracking. IEEE Conference on Computer Vision and Pattern Recognition. pp. 5388–5396.
Mei, X., Ling, H., 2009. Robust visual tracking using l1 minimization. IEEE International Conference on Computer Vision. pp. 1436–1443.
Ning, J., Yang, J., Jiang, S., Zhang, L., Yang, M.H., 2016. Object tracking via dual linear structured svm and explicit feature map. Computer Vision and Pattern Recognition. pp. 4266–4274.
Ross, D.A., Lim, J., Lin, R.S., Yang, M.H., 2008. Incremental learning for robust visual tracking. Int. J. Comput. Vision 77 (1-3), 125–141.
Wang, D., Lu, H., 2014. Visual tracking via probability continuous outlier model. Computer Vision and Pattern Recognition. pp. 3478–3485.
Wang, D., Lu, H., Yang, M.H., 2013. Online object tracking with sparse prototypes. IEEE Trans. Image Process. 22 (1), 314–325.
Wang, N., Shi, J., Yeung, D.Y., Jia, J., 2015. Understanding and diagnosing visual tracking systems. IEEE International Conference on Computer Vision. pp. 3101–3109.
Wu, Y., Lim, J., Yang, M.H., 2015. Object tracking benchmark. IEEE Trans. Pattern Anal. Mach.Intell. 37 (9), 1834–1848.
Xie, Y., Qu, Y., Li, C., Zhang, W., 2012. Online multiple instance gradient feature selection for robust visual tracking. Pattern Recognit. Lett. 33 (9), 1075–1082.
Xu, C., Tao, W., Meng, Z., Feng, Z., 2015. Robust visual tracking via online multiple instance learning with fisher information. Pattern Recognit. 48 (12), 3917–3926.
Yan, J., Chen, X., Deng, D., Zhu, Q., 2015. Visual object tracking via online sparse instance learning. J. Visual Commun. Image Represent. 26 (C), 231–246.
Yang, F., Lu, H., Yang, M.H., 2014. Robust visual tracking via multiple kernel boosting with affinity constraints. IEEE Trans. Circuits Syst. Video Technol. 24 (2), 242–254.
Yao, R., Shi, Q., Shen, C., Zhang, Y., Hengel, A.V.D., 2013. Part-based visual tracking with online latent structural learning. Computer Vision and Pattern Recognition. pp. 2363–2370.
Zhang, K., Song, H., 2013. Real-time visual tracking via online weighted multiple instance learning. Pattern Recognit. 46 (1), 397–411.
Zhang, K., Zhang, L., Yang, M.H., 2013. Real-time object tracking via online discriminative feature selection. IEEE Trans. Image Process. 22 (12), 4664–4677.
Zhao, C., Wang, J., Zhu, G., Wu, Y., Lu, H., 2016. Learning weighted part models for object tracking. Comput. Vision Image Underst. 143 (C), 173–182.