A Novel Joint Tracker Combined by Occlusion Detection

A, *Member, IEEE,* B, C*, *Senior Member, IEEE,* and D, *Fellow, IEEE*

***Abstract*—Visual object tracking methods can be roughly categorized into generative method or discriminative method. Each of the two methods has its advantages and negative factors which may result in occlusion or drifting problems. In order to deal with the occlusion and drifting problem better, we combine the two different methods together by the occlusion information, which is obtained from an occlusion detection mechanism. In this paper, first, we propose a novel mechanism which can predict occlusion accurately and sensitively with MIL and SVM classifiers. Second, we combine the discriminative method and the generative method in a joint-probability model and use the occlusion information to adjust the weights of the methods, which are complementary. Third, we propose a classified template updating method, in which we divide the templates into two groups according to occlusion information and use opposite probability distribution to update the two groups. The experiment results of our tracker on several challenging datasets demostrate that our approaches is effective and outperforms the state-of-the-art approaches.**

***Index Terms*—visual object tracking, occlusion prediction, template update, joint probability.**

I. Introduction

VISUAL object tracking is a basic task in computer vision. Many computer vision applications, such as vehicle navigation, video surveillance and automatic drive, cannot work without visual object tracking. Visual object tracking remains challenging, however, despite the considerable progress made in recent years, because of destabilising factors such as illumination and scale changes, complicated background, occlusions and pose changes in the video sequence. Occlusion and drifting problems are the core issues, and have not been fully resolved as yet.

Visual object tracking methods can be roughly divided into two categories: the discriminative method, which is implemented with classifiers, and the generative method, which aims to find the most similar regions. Discriminative methods classify the target object from the background to achieve the tracking procedure. Grabner and colleagues [1] proposed an on-line AdaBoost selection method for complex background models which exploits information on the background. Avidan [2] cast the tracking problem as a binary classification problem by combining a number of weak classifiers into a strong one, which is trained on-line to distinguish the object and the background by means of labelled pixels in the next frame. Later, Grabner and colleagues [3] upgraded the boosting method to an on-line semi-supervised boosting method to alleviate the drifting problem. They combined a given prior and an on-line classifier to do the tracking. In order to train the dataset more accurately, Babenko and colleagues [4] used multiple instance learning (MIL) instead of traditional supervised learning methods. With MIL, the training samples can be labelled more precisely, which can alleviate the drifting problem. In [5], Kalal and colleagues trained a binary classifier by means of labelled and unlabelled examples. They trained the classifier on the labelled data and improved it with the unlabelled data.

Generative methods hold the features by maintaining a template set, which is used to find the target object in the next frame, to do visual tracking. The templates can be used with pixel or patch format. Adam and colleagues [6] used arbitrary patches, which were not based on the object model, to represent the template object, and each patch voted on the possible positions and scales of the object, which were implemented with an integral histogram data structure [7]. Kwon and colleagues [8] decomposed the observation model into multiple basic observation models by sparse principal component analysis (SPCA). Each basic observation model covers a specific appearance of the object. Then spare presentation is used in visual tracking. Liu and colleagues [9] proposed a local sparse appearance model and sparse regularized mean-shift to do the tracking. The sparse dictionary is static, however, which may lead to a drifting problem. Wang and colleagues [10] proposed a tracking method from the perspective of mid-level vision with structural information captured in superpixels. Subsequently, they used a discriminative appearance model based on the superpixels to distinguish the target and the background. Besides the improvements in representation models, modelling methods also developed rapidly. Ross and colleagues [11] presented a tracking method that incrementally learns a low-dimensional subspace representation, which is adapted to the changes of the target appearance. To this end, an incremental algorithm for principal component analysis is implemented to deal with heavy changes in pose, scale, and illumination. Mei and Ling [12] cast tracking as a sparse approximation problem. The target candidate is sparsely represented by target templates and trivial templates, and the sparsity is achieved by solving an ℓ_1-regularized least squares problem. Another ℓ_1 tracker with minimum error bound and occlusion detection is proposed in [13]. The minimum error bound is calculated by a linear least squares equation and then serves as a guide for particle resampling in a particle filter framework; the occlusions are detected by investigating the trivial coefficients in the ℓ_1 minimization. Jia and colleagues [14] developed a robust tracking method based on the structural local sparse appearance model by applying an alignment-pooling method to exploit partial information and spatial information of the

A is with School of Computer Science and Technology, Harbin Institute of Technology Shenzhen Graduate School, China. (e-mail: A@hitsz.edu.cn).

B and C are with, China. (e-mail:).

D is with. (e-mail:).

target to handle occlusion. Although the above methods have made great progress in terms of object tracking, some serious problems have not been solved completely.

When partial or complete occlusion occurs, the tracker can use part of the target appearance and the occlusion information to find the best candidate bounding box. When the occlusion disappears, the tracker should avoid the influence of the occlusion and find the right target. In this paper, we propose an occlusion prediction mechanism with MIL and SVM methods. The observation of the target is sampled by patch, and we have trained the MIL classifier only on the templates without occlusion, to guarantee the accuracy of the occlusion ratio. Additionally, we present a joint probability model of the discriminative and generative methods to solve the occlusion and drifting problem. An occlusion ratio is exploited to adjust the weights of the two methods. The generative method is implemented with a local structural sparse representation model [14], solved by ℓ_1 minimization. The discriminative method uses MIL&SVM [15] as a classifier to locate the target from the background. Finally, we design a novel template updating method, which uses the occlusion information to update templates dynamically.

Summarily, our main contribution in this paper is threefold. The first contribution is developing a new prediction mechanism, which can predict occlusion precisely. The second contribution is combining the generative and discriminative methods in a joint probability model, which can handle the drift and occlusion problems effectively. This joint probability model is implemented with the distribution of positive and negative patches with the MIL classifier and the spare representation result from the template matching. The joint probability model can help us to locate the object more accurately. The third contribution is proposing a better template update method, which divides the templates into two groups according to occlusion information and updates them with different mechanisms. We can see the three pivotal stage pre-occlusion, occlusion and occlusion disappearing in Fig. 1 which shows the occlusion and drifting problems.

The rest of this paper is organized as follows. Section II is a short review of related research. Section III describes our occlusion detection method. Section IV introduces our joint probability model. Section V discusses our template update mechanism with occlusion information. Section VI shows the experimental details and results. Section VII concludes.

II. Related work

Sparse representation has natural advantages in visual categorization. Many works have been done to improve the performance of visual tracking based on sparse representation [9, 12, 13, 16]. Mei and Ling [12] exploited target templates and trivial templates to represent the candidate target with sparse coefficients. The sparsity is achieved by solving an ℓ_1-regularized least squares problem. With trivial templates, sparse representation can handle the occlusion problem. Jia and colleagues [14] proposed a structural local sparse appearance model, which can exploit the spatial information of the target with an alignment-pooling method. Additionally, they combined incremental subspace learning and sparse representation to update the template to alleviate the influence of occlusion and the probability of drifting. Thus, sparse representation with a local structure can handle occlusion well, although it may be influenced by the occlusion information from the occluded templates.

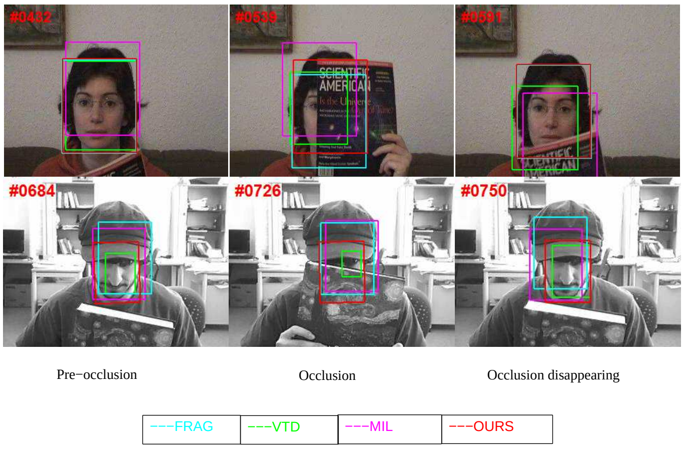


Fig. 1. Examples of occlusion tracking results. The first row is from dataset faceocc1 [6], and the second one is from dataset faceocc2 [6]. The numbers in red at the top left are frame numbers. The left, middle and right pictures correspond to the pre-occlusion case, occlusion case and occlusion disappearing case, respectively. The results of four trackers (FRAG[6], VTD[8], MIL[4] and ours) are shown by bounding boxes in different colors. The red bounding boxes are the results of our tracker. We can see that our tracker can locate the target accurately when severe occlusion happens and is not influenced by occlusion after it disappears.

Discriminative methods are also widely used for visual tracking [2–5] which exploit a trained classifier to classify the target from the background and focus on the differences between the target and the background. Babenko and colleagues [4] trained a classifier with MIL. The MIL method uses bags, which are composed of unlabelled instances. Bags which include one or more positive instances are labelled as positive bags. In contrast, bags which only contain negative instances are labelled as negative bags. In visual object tracking, a rectangular bounding box is commonly used to locate the target object, but the bounding box may also contain the background area, for the object may not be a standard rectangle and cannot fill the entire bounding box area. Hence, the patch sampling from the bounding box may also from part of the background. To deal with this problem, we can use MIL, in which the patches are treated as instances and a group of patches can be regarded as a bag. Thus, the MIL can train the classifier more precisely. Andrews and colleagues [15] cast the MIL problem as a maximum margin problem, which is solved by the support vector machine (SVM) learning approach. Here, we use the MIL&SVM classifier in a new format with image patches, which can accurately classify the target from the background. What is more, it can alleviate the influence of the background noise in the target bounding box.

The combination of discriminative and generative methods was proposed in [17], Zhong and colleagues devised a sparsity-based collaborative model to integrate the advantages of holistic discriminative and local generative modules for better results. They put the two methods together directly, however, whereas ours is combined with occlusion information and the weights of the methods can be dynamically adjusted according to the degree of occlusion. Occlusion information has been specially used for tracking [13, 18, 19]. In [13], Mei and colleagues proposed an occlusion detection method using an efficient ℓ_1 tracker with minimum error bound. They detected the occlusions by investigating the trivial coefficients in the ℓ_1 minimization. However, they performed tracking and occlusion detection with the same features and method, which are relevant to each other, as leads to no further information was obtained from the results. In the proposed algorithm, we implement the two works with different methods on different samples to exploit more information in different aspects.

III. Occlusion Prediction method

Occlusion happens when a near object obscures the target object and the complete appearance of the target object cannot be obtained from the two-dimensional image, e.g., the target object region is filled by the pixels of other objects in the picture. In essence, occlusion happens when the pixels around the target bounding box move into the bounding box region. In the visual tracking process, we regard the target object as foreground and other regions as the background. When the background around the foreground appears in the target object area, occlusion occurs in the frame sequence as shown in Fig. 2. Therefore, we can check the foreground region to find out whether it contains the surrounding background to predict occlusion. In order to detect the background, samples should be taken from both foreground and background. In the following paragraph, we introduce the sampling method in detail.

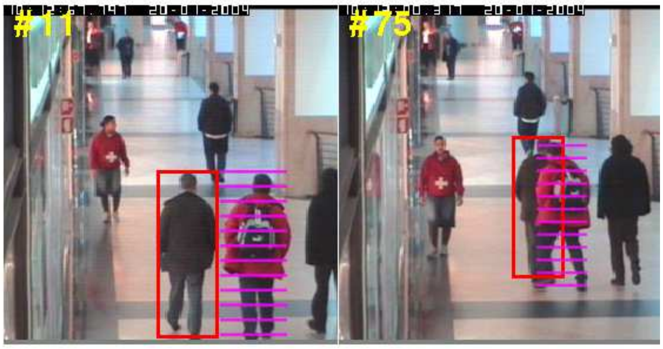


Fig. 2. An illustration of the occurrence of occlusion, where the two figures are picked from dataset Caviar3 [20]. The left picture is frame 11 and the second one is frame 75. Red bounding boxes are used to label the foreground region which contains the target object. We can see that the background around the foreground, which is marked with pink shadows, enters into the foreground area in the second picture and occlusion happens. As we know, occlusion is a gradual process and obstructions must come from the surrounding of the target object in the previous frames if the occlusion happens. Therefore, we can detect occlusion by detecting whether the surrounding backgrounds enter the bounding box.

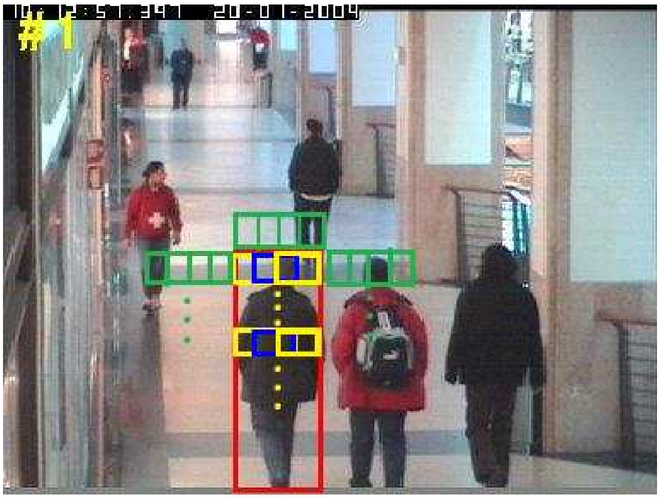


Fig. 3. An illustration of the sampling and labelling method; the picture comes from the Caviar3 dataset. The target object is located by a big red bounding box, which is also called as foreground. We use the patch as the sampling unit, which is shown as the smallest grid. The patches should overlap, as shown in the first row of the foreground area with yellow and blue patches. The foreground and the surrounding background should both be sampled. The foreground patches are regarded as positive samples and the background patches are regarded as negative samples, which are marked as green boxes. One row patches are exploited as the labelling unit instance, which uses the idea of MIL.

We use a patch that covers a small rectangular image region with dozens of pixels as the sampling unit. The patch retains the local structure features of the object. It also carries extra details about the relative position of the individual pixels. Hence, patches have more features than isolated pixels. In addition, the patch is more stable than pixels in terms of the representation of certain features. The sampling patches should overlap, i.e., the sampling step should be smaller than the length of the patch, to maintain the relationship between the neighboring patches. Overlapping patches can avoid sampling incomplete local object part. In other words, overlapping patches have a greater probability of covering a complete local part of the object in one patch other than dividing a local structure into several patches. We set the sampling step as half of the patch length and obtain satisfying results. In the generative method, the sampling patches can be used as the templates to compose a sparse dictionary, which is used to represent the candidate bounding box with patches to obtain the sparse coefficients. In the discriminative methods, sampling patches are used to train a classifier, which is used to distinguish the foreground from the background. The foreground is labelled as the positive sample and the background is labelled as the negative sample. There is a problem to be solved before labelling the patches, however: not all patches in the target bounding box region are on the target object because the bounding box is rectangular whereas the shape of the target object may not be a standard rectangle. The patches in the bounding box region should be trained as a positive sample, but some patches of the bounding box are background area which can be noise for the training of positive sample.

The multiple instance learning method is trained with instances, which are included in the bags. A bag is labelled as a positive sample if it includes at least one positive instance, and a negative sample only includes negative instances. Accordingly, in our method, we regard the image patch as the instance, and a group of patches as a bag. Generally, the height of the target bounding box equals that of the target object, to ensure that each row of the patches in the bounding box covers part of the object, i.e., one row of patches includes at least one patch of the target object other than the noise of the background. We treat one row of patches as a bag in order to ensure the property that a row patches in the bounding box includes at least one positive patch, which does not contains noise of the background. Hence, we can ensure all the bags sampled from the bounding box are positive samples.

We also sample negative bags in the background around the foreground, as shown in Fig. 3. A distance parameter $\mathcal{R}$, which measures the distance from the center-point of the target object to the edge of the negative sampling area, is used to set the range of the negative patch sampling area. We can adjust the sampling area in the background. In order to maintain the consistency, the negative samples are also grouped by a bag, which includes a row of patches. Here, we denote the positive patch, negative patch, positive bag and negative bag by p^+, p^-, b^+ and b^-, respectively.

After we obtain positive and negative samples, we use an SVM(Support Vector Machine) to train the samples and obtain the classifier. With this classifier, we can classify each patch of a candidate bounding box. The patches of a candidate bounding box are stored in a patch matrix $P_{M,N}$, where M is the row number of the patches and N the number of patch columns. Each patch can be classified as a positive sample, which means it belongs to the target object or as a negative sample, which means background area around the foreground. As each patch $p_{m,n}$ of P corresponds to a classification result $r_{m,n}$, we obtain the result matrix $R_{M,N}$. Then, we use Eq. (1) to detect occlusions.

$$O_{ratio} = \frac{|p^-|}{MN}. \tag{1}$$

In Eq. (1), $|p^-|$ is the number of negative patches and MN is the total patch number. O_{ratio} is the occlusion ratio which reflects the degree of occlusion. More importantly, we can obtain the occlusion position in the bounding box for each patch in the bounding box is matched to a classification result in R with the same position. We can check the distribution of the negative patches and find the specific occlusion location.

We train the classifier only with the frames without occlusion or with light occlusion. At the beginning, the initial frames are exploited to train the classifier and then the classifier selects the templates to do training which depends on the occlusion ratio. With these unoccluded templates, the classifier remains accurate even after occlusions as can alleviate the drifting problem significantly.

IV. Joint Probability Model

The procedure for visual object tracking is to use the previous template set $\mathcal{T}_{1:t-1}$, which can be initial bounding boxes or boxes of tracking results, to find the position of the target object in the current frame t. In the initial stage, the tracker extracts features from the template bounding box area or around the bounding box area. In the tracking stage, the tracker selects the most similar bounding box region $\hat{b}$ from a candidate bounding box set $\mathcal{B}$ as the tracking result. Candidate set $\mathcal{B}$ is generated in the current frame by random sampling around location of the latest tracking results. In each frame, one same action is done: the known information from $\mathcal{T}_{1:t-1}$ is used to find the most similar bounding box object region $\hat{b}_t$ in the current frame t. The tracking result is formulated as:

$$\hat{b}_t = \underset{b_t^i \in B_t}{\operatorname{argmin}} \left\| F(\mathcal{T}_{1:t-1}) - F(b_t^i) \right\|, \tag{2}$$

where $\mathcal{B}_t$ is the candidate bounding box set in the current frame t and b_t^i is the i-th candidate bounding box region in $\mathcal{B}_t$. $\mathcal{N}$ is the number of samples, and $i \in \{1, 2 \cdots, \mathcal{N}\}$. In the random sampling procedure to obtain $\mathcal{B}_t$, the velocity of the target object can be used to predict the central position of the sampling area, and the number of samples $\mathcal{N}$ can be set experimentally, which is a balance between efficiency and accuracy. $\mathcal{T}_{1:t-1}$ includes the target bounding box regions in the previous frames, which can be initial frames or frames of the tracking results. $F(\mathcal{X})$ is a measure function, which obtains the feature of a bounding box area set $\mathcal{X}$, and $\mathcal{X} \neq \emptyset$. The tracking result $\hat{b}_t$ is obtained by minimizing the feature distance of F between regions of template set $\mathcal{T}_{1:t-1}$ and regions of set $\{b_t^i\}$. Function F can be implemented with two different methods: one is the generative method, which extracts the features from $\mathcal{T}_{1:t-1}$, then uses the features to find the most similar bounding box area $\hat{b}_t$ in frame t; the other is the discriminative method, which uses the data $\mathcal{T}_{1:t-1}$ to train a classifier. The classifier can classify an image patch in the bounding box region b_t^i as the target object area or background area and the bounding box with the most object patches is selected as the tracking result. In our algorithm, the two methods are combined together for tracking.

In order to exploit the advantage of the two different methods, we fuse the two methods with a joint probability model, which applies probability to measure the similarity between $F(\mathcal{T}_{1:\,t-1})$ and $F(b_t^i)$. In the feature extraction stage, we divide the target region into overlapping patches and use patch as the basic unit to represent the features of the picture. Thus the set theory can be used to simplify the formulas. We obtain the probability $P(b_t^i|\mathcal{T}_{1:t-1})$ of b_t^i being the most similar candidate bounding box to $\mathcal{T}_{1:t-1}$ by

$$\begin{aligned} P(b_t^i|\mathcal{T}_{1:t-1}) &= \frac{\mathcal{S}_{similar}}{\mathcal{S}_{total}}, \\ &= \frac{|F(b_t^i) \cap F(\mathcal{T}_{1:t-1})|}{|F(\mathcal{T}_{1:t-1})|}, \end{aligned} \tag{3}$$

where $\mathcal{S}_{similar}$ denotes similar area, $\mathcal{S}_{total}$ the total foreground area, $|F(b_t^i) \cap F(\mathcal{T}_{1:t-1})|$ the number of similar patches and $|F(\mathcal{T}_{1:t-1})|$ the total patch number. We use the ratio of the two area sizes as the probability. In addition, $|F(\mathcal{T}_{1:t-1})|$ is a determined non-zero value. With Eq. (2) and Eq. (3), we can obtain

$$\begin{aligned} F(\mathcal{T}_{1:t-1}) - F(b_t^i) &= F(\mathcal{T}_{1:t-1}) - F(b_t^i) \cap F(\mathcal{T}_{1:t-1}), \\ &\propto \frac{F(\mathcal{T}_{1:t-1})}{F(\mathcal{T}_{1:t-1})} - \frac{F(b_t^i) \cap F(\mathcal{T}_{1:t-1})}{F(\mathcal{T}_{1:t-1})}, \\ &= 1 - P(b_t^i|\mathcal{T}_{1:t-1})). \end{aligned} \tag{4}$$

Thus, Eq. (2) can be written as:

$$\begin{aligned}\hat{b}_t &= \underset{b_t^i \in B_t}{\operatorname{argmin}}(1 - P(b_t^i|\mathcal{T}_{1:t-1})),\\ &= \underset{b_t^i \in B_t}{\operatorname{argmax}}\, P(b_t^i|\mathcal{T}_{1:t-1}).\end{aligned} \tag{5}$$

It is proved that Eq. (2) and Eq. (5) can obtain the same $\hat{b}_t$. And the joint probability can be written as $P(b_t^i|\mathcal{T}_{1:t-1}^G, \mathcal{T}_{1:t-1}^D)$, where $\mathcal{T}_{1:t-1}^G$ stands for the features obtained by the generative methods and $\mathcal{T}_{1:t-1}^D$ stands for the features obtained by discriminative methods from $\mathcal{T}_{1:t-1}$. The generative methods use $\mathcal{T}_{1:t-1}$ to hold the features of the target object and change the features according to the change of the target object by dynamical template updating. When occlusion happens, the best candidate bounding box can be represented by unoccluded and occluded templates. The variation of the target object itself, however, especially the occlusion, may mislead the templates, which will result in a drifting problem. Here, we use the discriminative method to deal with this problem. Discriminative methods focus on classifying the bounding box as target object area from the background. That is, they not only use the feature of the target in the bounding box, but also use the information of the background. Additionally, in our method, the discriminative method only uses the templates without occlusion to train the classifier, which ensures that the classifier is not affected by occlusion. Therefore, the two different methods are complementary. To this end, we define the fused probability as

$$P(b_t^i|\mathcal{T}_{1:t-1}^G, \mathcal{T}_{1:t-1}^D) = (1-\alpha)P(b_t^i|\mathcal{T}_{1:t-1}^G) + \alpha P(b_t^i|\mathcal{T}_{1:t-1}^D). \tag{6}$$

where $P(b_t^i|\mathcal{T}_{1:t-1}^G, \mathcal{T}_{1:t-1}^D)$, the joint probability, is the weighted sum of the discriminative method and generative method, and $\alpha \in [0, 1]$ is the control parameter, which can adjust the weights of the two methods. With this joint probability, we can easily assign a bigger weight to the more suitable method for a special case. We assign a greater weight to the discriminative features when the target object area is discriminative from the background, and assign a greater weight to the generative features when the target object area is similar to the background. Hence, we can evaluate the two features and assign a greater weight to the more discriminative features before the tracking step. That means the tracking result depends on the more distinguishable features. Furthermore, the joint probability model can be used to handle occlusion and drifting problems. In ordinary cases without occlusion, the two methods can both track the target object accurately. Therefore, the parameter α is set near 0.5. When occlusion occurs, we can use the occlusion prediction information to adjust the weights and guide the template updating mechanism, as explained in the following paragraph.

The core idea of the combination methods is that the generative method uses all the templates and the discriminative method only uses the templates without occlusion, which is an implement based on the occlusion detection results. Here, the generative method is implemented with a local structure sparse representation [14], which can track the target accurately even when occlusion occurs as the template set contains occluded templates that are used to represent the occluded target. In this case, the weight of the discriminative method α will reduce to a smaller value via the occlusion ratio O_{ratio}, which is obtained by means of the occlusion detection mechanism. When the occlusion disappears, the generative method may still use the templates with occlusion to represent the target, which may result in the drifting problem. In this case, the weight of the discriminative method α will increase to a larger value through the occlusion ratio O_{ratio} obtained by the occlusion detection. Since the discriminative method is trained only with the templates without occlusion, it can track the target accurately when the occlusion disappears. In conclusion, the weight of the discriminative method α is proportional to the occlusion ratio O_{ratio}, i.e.,

$$\alpha \propto O_{ratio}. \tag{7}$$

When O_{ratio} is close to zero, the two methods get the same weight and α is set to 0.5. When the object is heavily occluded, i.e., O_{ratio} is close to one, α should be set to one. So, we define α as

$$\alpha = \frac{1}{2}(1 + O_{ratio}). \tag{8}$$

With dynamic adjustment of the weights, the generative method can track the target along the occlusion and the discriminative method track the target when the occlusion is disappearing. The discriminative method holds the real feature of the target without any occlusion, which can tell us when the occlusion appears or disappears so we can adjust the weights of the two methods to prevent drifting problems.

The generative method is implemented with adaptive structural local sparse appearance model proposed in [14], in which each patch of the target object region can be represented by the corresponding patches of the templates. With ℓ_1 sparse minimization, we obtain the sparse coefficients $A = [a_1, \cdots, a_n]$ $a_i \in [0, 1]$ for each patch. The value of a_i indicates the similarity between the templates and the target object. And we can use this value as the probability of a patch being in the target region. Thus, the probability can be written as

$$P(b_t^j|\mathcal{T}_{1:t-1}^G) = \overline{a}_i, \tag{9}$$

where $\overline{a}_i$ is the mean of all the coefficients.

We use MIL&SVM [15] as the classifier to implement the discriminative methods given that the region in the bounding box contains the background area in the edge of the bounding. Hence, not all the patches in the bounding box are positive samples, which contain the feature of the target object. Multiple instance learning can handle this situation well. In our method, we also use patch as the unit for training. The difference is that we sampling both in the bounding box region and the regions around the bounding box. The patches in the bounding box are labelled as positive samples and those around the bounding box are labelled as negative samples. The aim of this method is to classify the patches in the current bounding box into positive or negative patches. We select the bounding box with the largest positive patches number as the tracking result. Accordingly, the probability of a bounding box being the tracking result can be defined as

$$P(b_t^j|\mathcal{T}_{1:t-1}^D) = \frac{|p^+|}{|p^-| + |p^+|}, \tag{10}$$

where $|p^+|$ is the number of positive patches, and $|p^-|$ is the negative patch number. Eq. 10 indicates that a bounding box with more positive patches is more likely to be an accurate tracking result. The procedure of our joint tracking is described in algorithm 1.

Algorithm 1 Joint Tracking

```
Input:
    The $t$-th frame $f_t$, templates $\mathcal{T}_{t-1} = \{f_1, ...f_K\}$, $K$ is the number of templates. Number of samples $\mathcal{N}$, sample radius $\mathcal{R}$, sampling patch size$(m_1, m_2)$
Output:
    Tracking result $\hat{b}_t$, the updated templates set $\mathcal{T}_t$
1: Sample uniformly on the search area with radius $\mathcal{R}$ to get the candidate set $\mathcal{B}_t = \{b_t^i\}, i \in [1, ..., \mathcal{N}]$;
2: For each candidate $b_t^i$, cut the bounding box area to patches to do classification and get $O_{ratio}^i$ of them via Eq. (1) with occlusion prediction mechanism;
3: Compute the joint probabilities $p_t^i$ of all the candidates $b_t^i$ with Eq. (6) and Eq. (8);
4: The tracking result $\hat{b}_t = \mathrm{argmax}\{p_t^i\}, i \in [1, ..., \mathcal{N}]$;
5: $\mathcal{T}_t$ ←templates update with $\mathcal{T}_{t-1}$ and $\hat{b}_t$ as Algorithm 2.
```

V. Template Update

Template update is a key step in tracking process and directly influences the results. Fixed templates cannot cope with the change of the target object because the tracker cannot accurately track the object when the illumination or the pose changes, as is inevitable in the real applications. Therefore, template update is necessary for tracking. Updating the template frequently, however, will result in the drifting problem, especially when the occlusion occurs. Many methods [11, 12, 14, 21] have been proposed to improve the mechanism for template update to get better results. Ross and colleagues [11] proposed an incremental principal component analysis algorithm to update the sample mean and adjust the weight of templates with a forgetting factor. In their algorithm, however, the reconstruction error is supposed to be a Gaussian distributed with a small variance, which cannot cope with the partial occlusion well. Jia and Lu [14] generated a cumulative probability sequence, which leads to a slow update for old templates and a quick update for new ones to alleviate the drifting problem. However, when the templates contain occlusion, old templates with occlusion may be less important than the new ones. In this paper, we design the joint tracker with the sparse representation and the MIL classifier and update the template with the subspace learning. When the target recoveries from the occlusion, we use the MIL classifier to track the target and adjust the probability of template update to avoid the drifting problem.

Many tracking methods agree that old templates are more accurate than new ones. Hence, the old templates are likely to stay for a longer time. When occlusion occurs, however, the new occluded templates are more similar to the current occluded target, because the occlusion and recovery are all gradual processes, and an old occlusive template has little value in terms of tracking the current target whereas the new occluded template may be useful for the current tracking. Hence, we classify the templates into two groups according to occlusion. The group without occlusion is denoted as $\mathcal{T}_{unocc} = \{f_1, \cdots, f_n\}$, where n is the number of unoccluded templates, and the occluded template set is denoted as $\mathcal{T}_{occ} = \{f'_1, \cdots, f'_{n'}\}$, where n' is the number of occluded templates. The templates in $\mathcal{T}_{unocc}$ are ordered by time and the templates in $\mathcal{T}_{occ}$ are ordered reversely by time. We give an increasing interval sequence $\mathcal{I}$, corresponds to the template sequence, which is defined as

$$\mathcal{I} = \left\{0, \cdots, \frac{i^2 + i}{K^2 + K}, \cdots, 1\right\}, \tag{11}$$

where K is the total number of the template sequence and i is the sequence number. Hence, the generated interval corresponding to f_i is $[\frac{i-1^2+(i-1)}{K^2+K}, \frac{i^2+i}{K^2+K}]$, where $i \in \{1, 2, \cdots, K\}$ and the i-th interval length is $\frac{2i}{K^2+K}$. Then, we generate a rand number r in interval $[0, 2]$. When $r \leq 1$, we discard one template from the unoccluded template group; otherwise, from the occluded template group. We can formulate this in Eq. (12)

$$y(r) = \begin{cases} i, & r \in [\frac{i-1^2+(i-1)}{K_{unocc}^2+K_{unocc}}, \frac{i^2+i}{K_{unocc}^2+K_{unocc}}], 0 < r \leq 1 \\ j, & r \in [1 + \frac{j-1^2+(j-1)}{K_{occ}^2+K_{occ}}, 1 + \frac{j^2+j}{K_{occ}^2+K_{occ}}], 1 < r \leq 2, \end{cases} \tag{12}$$

where K_{unocc} is the number of unoccluded template sequences $\mathcal{T}_{unocc}$ with time order, and K_{occ} the number of occluded template sequence $\mathcal{T}_{occ}$ with reverse time order. Eq. 12 indicates that the old templates without occlusion will stay longer, and conversely, the old occluded templates will be discarded with a high probability. With this mechanism, occlusion can be tracked by the new occluded templates and the drifting problem caused by the target recovering from occlusion can be solved by the old templates without occlusion and the MIL classifier.

After selecting the template to discard, we use the combination of sparse representation and subspace learning, which is proposed in [14] to update the templates. It should be noted that, in our method, the occluded or corrupted pixels, which are denoted as e', are obtained through the occlusion detection procedure, whereas in [14], the occluded or corrupted pixels are unknown. The incremental method proposed in [11] is used on the tracking results, which can adapt to the appearance change and maintain the constant feature of the target. The tracking result b can be modelled by a linear combination of the PCA basis vectors and additional trivial templates, which are employed in [12] as

$$b = Vx + e' = [V \quad I], \tag{13}$$

where b is the candidate bounding box, V is the matrix of eigenbasis vectors, x is the coefficients of V, and e' is the error vector. Eq. (13) is solved as an ℓ_1 regularized least squares problem

$$\min_c \|b - Dc\|_2^2 + \lambda\|c\|_1, \tag{14}$$

where $||.||_1$ and $||.||_2$ denote the ℓ_1 and ℓ_2 norms, respectively, $D = [V\ I]$, $c = [x\ e']^T$ and λ is the regularization parameter. The template update algorithm is described in algorithm 2.

```
Algorithm 2 Template update
Input:
    Old template set $\mathcal{T}_{t-1} = \{f_1, \cdots, f_K\}$, tracking result $\hat{b}_t$,
    $O^t_{ratio}$ occlusion ratio of $\hat{b}_t$, occlusion patches $O^t_{patch}$ of
    $\hat{b}_t$, eigenbasis vectors $V$
Output:
    New template set $\mathcal{T}_t$
    Generate a random number uniformly in $[0, 2]$;
    Get template $y(r)$ to be discarded with Eq. (12), then
        $\mathcal{T}^{tmp}_t = \mathcal{T}_{t-1} - f^{y(r)}_{t-1}$;
    Solve equation (14) with occluded patches $O^t_{patch}$, and
        get x;
    $f_t = Vx$ and $\mathcal{T}_t = \mathcal{T}^{tmp}_t \cup f_t$.
```

VI. Experiments

Implement details: the proposed algorithm is implemented in MATLAB on one PC with an Intel 2.9GHz Dual Core CPU and 2GB memory. The discriminative method MIL&SVM is implemented with a package at site http://www.kyb.mpg.de/bs/people/pgehler/mil/milsvm.zip. The ℓ_1 sparse minimization problem is solved by SPAMS package [22] and the regularization constant λ is set to 0.01. For each dataset, the first ten frames are used as the initial frames in which targets are labelled. In the sampling stage, we resize all the foregrounds as $(32, 64)$ in pixels in order to keep unity. The size of the sampling patch is $(16, 16)$ and the sampling step is 8 pixels. Thus, each foreground is cut into 21 overlapping patches. In initial tracking, the occlusion ratio O_{ratio} is set to 0, and α is set to 0.5. In the tracking step, we set the sample number $\mathcal{N}$ as 200, i.e., in each frame, the candidate set has 200 samples. Our tracker has been tested many times on many datasets and gives promising results.

Datesets:We use eight challenging sequences to evaluate our tracking system. These sequences include Faceocc1 [6], Faceocc2 [6], DavidIndoor [4], Singer, Caviar, Woman, Board [23] and Stone. Four main challenges of object tracking are partial and full occlusion, the change of illumination, pose and scale variation and confused background respectively, and each of the test datasets has its own focus on these challenges. For example, Faceocc1 and Faceocc2 focus on the partial or full occlusion, and their target objects are the people's heads, which are occluded by a hat or books. DavidIndoor focuses on the illumination and posture change. Girl and Board focus on the pose and scale variation. Caviar and Woman focus on complicated occlusion and change of pose and scale. Stone focuses on the confused background challenge.

Trackers: in order to examine the performance of our tracking algorithm, we use seven state-of-the-art trackers with the same initial position of the tracking target for comparison with ours. These tracking algorithms are the fragment-based (FragTrack) tracking methods [6], incremental visual tracking method [11], ℓ_1 tracker [12], multiple instance learning (MIL) tracker [4], P-N learning (PN) tracker [5], visual tracking decomposition (VTD) method [8] and the ASLA algorithm [14]. We obtain their results by running the trackers with the source codes provided by the authors with the adjusted parameters or finding the results from their papers or websites.

Evaluation: two widely used metrics are exploited to evaluate the trackers' performance. The first metrics is relative center position error (in pixels). The results are obtained by calculating the distance from the tracking result center position to the center position of the ground truth. The distance error of each frames is shown in Fig. 5 and the average error of all the frames is shown in Table 1. In Fig. 5, the comparison results on six datasets are demonstrated by seven different colors(the red line is ours). From this figure, we can see that our results are the best of all the six datasets. What is more, our tracker remains stable along the frame sequence from start to end. In Table 1, we can see the average center error of the eight trackers on eight datasets. Our results are marked in bold font. In order to show the stability of our tracker, we run it several times on each datasets and give the average results which are given in the last column of the table. The average error of all the datasets is shown in the last row of the table, and we can see our result is the best.

The center location error only checks the deviation of the center point which cannot detect the variation of pose and scale. Hence, we also use Pascal VOC overlap ratio as the second metrics to evaluate our results. This is defined as $R_{overlap} = (S_R \cap S_{GT})/(S_R \cup S_{GT})$, where S_R is the result bounding box area and S_{GT} is the ground truth bounding box area. Generally, it is considered to be a successful tracker if the VOC overlap ratio is greater than 0.5. The VOC overlap ratio of the trackers is shown in Table 2 and our results are marked in bold font. The average results are also given as in Table 1. We can see our results are the best and most of the VOC overlap ratios are greater than 0.8. The bounding box area of seven trackers in the original images is shown in different colors in Fig. 4 and Fig. 6. In Fig. 4 comparison results on Faceocc1, Faceocc2 and Caviar2 are shown. These datasets focus on occlusion. We select the frames which are pre-occlusion case, occlusion case and occlusion disappearing case. Our results are marked by red bounding boxes, and we can see the proposed tracker performs well no matter in the occlusion frames or in the frames after the occlusion. Fig. 6 includes Singer, Car11 and DavidIndoor. These datasets focus on the variation of illumination, pose and scale which may have the same influence as occlusion and blur. We select five representative frames from the result set and we can see our tracker performs best during the variation. These test results verify that our joint tracker can be adjusted automatically by the occlusion detection mechanism. When occlusion happens the generative method can track the object well with occluded templates and when occlusion disappears, the discriminative method trained with unoccluded samples can track the object accurately which can avoid the influence of occlusion.

VII. Conclusion

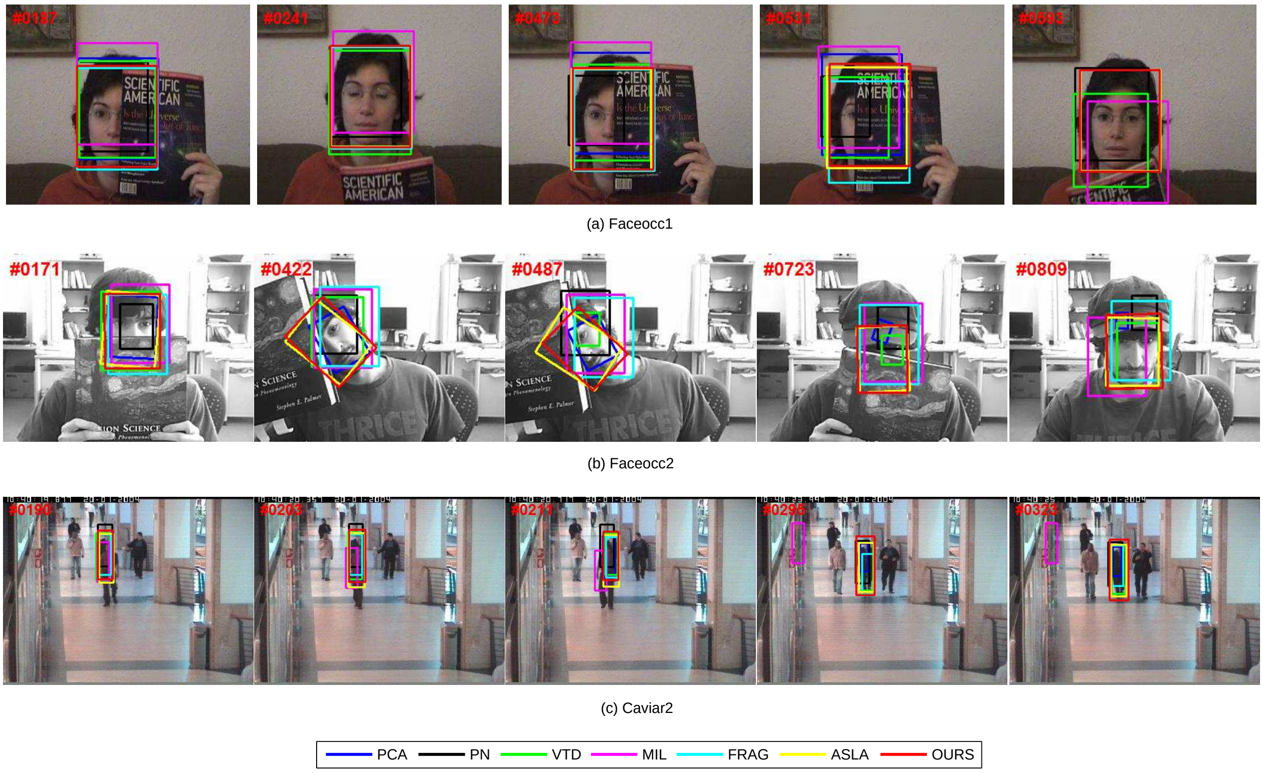
In this paper, we propose a joint probability tracker adjusted by an accurate occlusion detection method, and a new tem-

	IVT [11]	ℓ_1 [12]	PN[5]	VTD [8]	MIL[4]	FRAG[6]	ASLAS[14]	Ours_best	Ours_ave
Faceocc2	10.2	11.1	18.6	10.6	14.1	15.5	3.8	**3.8**	3.8
Caviar	66.2	65.9	53.0	60.9	83.9	94.2	2.3	**1.4**	1.4
Woman	167.2	131.6	9.0	136.6	122.4	113.6	2.8	**2.5**	2.6
Car11	2.1	33.3	25.1	27.1	43.5	63.9	2.0	**1.4**	1.5
David	3.6	7.6	9.7	13.6	16.1	76.7	3.6	**3.2**	3.2
Singer	8.5	4.6	32.7	4.1	15.2	22.0	4.8	**2.6**	2.7
Board	165.4	177.0	97.3	96.1	60.1	31.9	7.3	**9.1**	9.1
Stone	2.2	19.2	8.0	31.4	32.3	65.9	1.8	**1.1**	1.6
Average	52.9	56.3	31.7	47.6	48.5	60.4	3.6	**3.1**	3.2

TABLE I
THE CENTER POSITION ERROR IN PIXELS(CPE) COMPARING WITH OTHER SEVEN TRACKERS ON EIGHT SEQUENCES.

	IVT [11]	ℓ_1 [12]	PN[5]	VTD [8]	MIL[4]	FRAG[6]	ASLAS[14]	Ours_best	Ours_ave
Faceocc2	0.59	0.67	0.49	0.59	0.61	0.60	0.82	**0.82**	0.82
Caviar	0.21	.020	0.21	0.19	0.19	0.19	0.84	**0.90**	0.89
Woman	0.19	0.18	0.60	0.15	0.16	0.20	0.78	**0.84**	0.83
Car11	0.81	0.44	0.38	0.43	0.17	0.09	0.81	**0.86**	0.85
David	0.72	0.63	0.60	0.53	0.45	0.19	0.79	**0.81**	0.81
Singer	0.66	0.70	0.41	0.79	0.33	0.34	0.81	**0.87**	0.86
Board	0.17	0.15	0.31	0.36	0.51	0.73	0.74	**0.74**	0.74
Stone	0.66	0.29	0.41	0.42	0.32	0.15	0.56	**0.66**	0.64
Average	0.50	0.41	0.43	0.43	0.34	0.31	0.77	**0.81**	0.81

TABLE II
THE OVERLAP RATIO COMPARING WITH OTHER SEVEN TRACKERS ON EIGHT SEQUENCES.



(a) Faceocc1

(b) Faceocc2

(c) Caviar2

Fig. 4. Tracking comparison with other six state-of-the-art trackers on the original image. The frames selected for this comparison are one frame in pre-occlusion case, one frame in severely occlusion case, and one frame in occlusion disappearing case. It is evident that our tacker performs best.

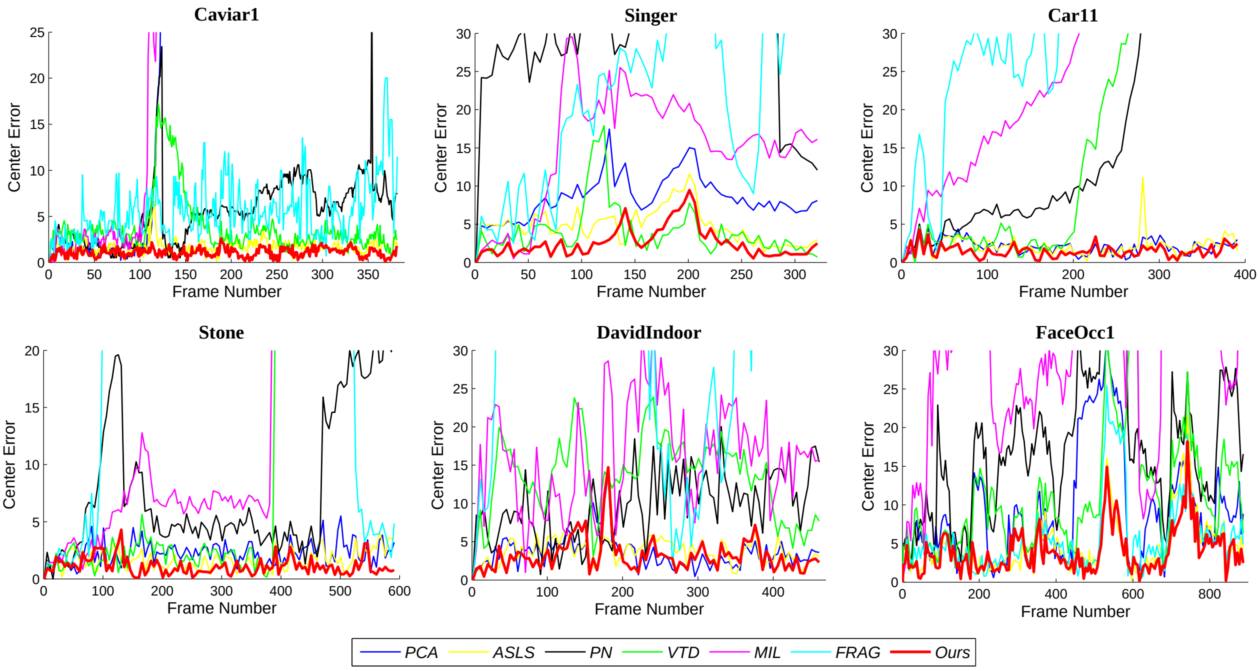
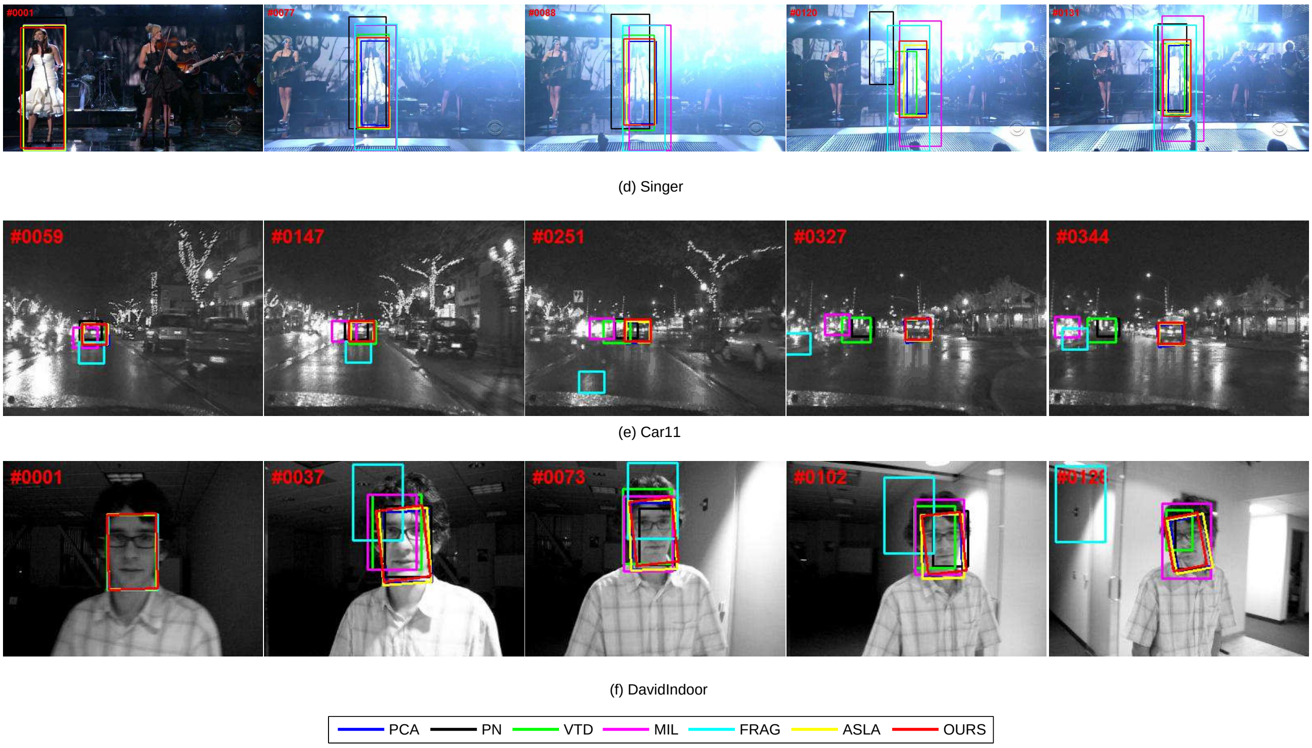


Fig. 5. Results for center errors of our tracker and six state-of-the-art trackers on six datasets. The red line shows the results of our tracker. It is evident that our track obtains the best results and performs stably.



(d) Singer

(e) Car11

(f) DavidIndoor

Fig. 6. Tracking comparison of another 3 datasets(Singer, Car11 and Davidindoor). These datasets focus on the variation of illumination, pose and scale, respectively. It is evident that our tracker performs best.

plate update mechanism with an occlusion ratio. The novel occlusion prediction method can accurately detect occlusion and outputs an occlusion ratio of the current frame. Then, the joint tracker fused with the generative and discriminative methods exploits the occlusion information for training and template update. The generative method uses the occlusion information to obtain more precise templates, and the discriminative methods are only trained on the unoccluded frames to maintaining correctness, which can prevent the influence of occlusion and avoid the drifting problem. In addition, the occlusion information is used to guide the template update. Experimental results and comparisons with other state-of-the-art trackers on large datesets show the superiority of our method.

References

[1] H. Grabner and H. Bischof, "On-line boosting and vision," in *Conference on Computer Vision and Pattern Recognition*, 2006, pp. 260 – 267.

[2] S. Avidan, "Ensemble tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, pp. 261–271, 2007.

[3] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *European Conference on Computer Vision*, 2008, pp. 234–247.

[4] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Conference on Computer Vision and Pattern Recognition*, 2009, pp. 983 – 990.

[5] Z. Kalal, J. Matas, and K. Mikolajczyk, "P-n learning: Bootstrapping binary classifiers by structural constraints," in *Conference on Computer Vision and Pattern Recognition*, 2010, pp. 49–56.

[6] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Conference on Computer Vision and Pattern Recognition*, 2006, pp. 798–805.

[7] F. Porikli, "Integral histogram: a fast way to extract histograms in cartesian spaces," in *Conference on Computer Vision and Pattern Recognition*, 2005, pp. 829 – 836.

[8] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Conference on Computer Vision and Pattern Recognition*, 2010, pp. 1269 – 1276.

[9] B. Liu, J. Huang, L. Yang, and C. Kulikowsk, "Robust tracking using local sparse appearance model and k-selection," in *Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1313 – 1320.

[10] S. Wang, H. Lu, F. Yang, and M.-H. Yang, "Superpixel tracking," in *International Conference of Computer Vision*, 2011, pp. 1323 – 1330.

[11] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International journal of computer vision*, vol. 77, pp. 125–141, 2008.

[12] X. Mei and H. Ling, "Robust visual tracking using l1 minimization," in *International Conference on Computer Vision*, 2009, pp. 1436 – 1443.

[13] X. Mei, H. Ling, Y. Wu, and E. Blasch, "Minimum error bounded efficient l1 tracker with occlusion detection," in *Conference on Computer Vision and Pattern Recognition*, 2011, pp. 1257 – 1264.

[14] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1822 – 1829.

[15] S. Andrews, I. Tsochantaridis, and T. Hofmann, "Support vector machines for multiple-instance learning," in *Advances in neural information processing systems*, 2002, pp. 561–568.

[16] B. Liu, L. Yang, J. Huang, P. Meer, L. Gong, and C. Kulikowski, "Robust and fast collaborative tracking with two stage sparse optimization," in *European Conference on Computer Vision*, 2010, pp. 624–637.

[17] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Conference on Computer Vision and Pattern Recognition*, 2012, pp. 1838–1845.

[18] E. B. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky, "Distributed occlusion reasoning for tracking with nonparametric belief propagation," in *Advances in Neural Information Processing Systems*, 2004, pp. 1369–1376.

[19] Y. Yang and G. Sundaramoorthi, "Modeling self-occlusions in dynamic shape and appearance tracking."

[20] "Ec funded caviar project," http://www.dai.ed.ac.uk/homes/rbf/CAVIAR/.

[21] L. Matthews, T. Ishikawa, and S. Baker, "The template update problem," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 6, pp. 810–815, 2004.

[22] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online learning for matrix factorization and sparse coding," *The Journal of Machine Learning Research*, vol. 11, pp. 19–60, 2010.

[23] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof, "Prost: Parallel robust online simple tracking," in *Conference on Computer Vision and Pattern Recognition*, 2010, pp. 723–730.