IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 10, 2019, accepted October 25, 2019, date of publication November 4, 2019, date of current version November 13, 2019.

*Digital Object Identifier 10.1109/ACCESS.2019.2951056*

# Bidirectional Tracking Scheme for Visual Object Tracking Based on Recursive Orthogonal Least Squares

**ZHIYONG HUANG, YUANLONG YU, AND MIAOXING XU**

College of Mathematics and Computer Science, Fuzhou University, Fuzhou 350002, China

Corresponding author: Yuanlong Yu (yu.yuanlong@fzu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant 61873067.

**ABSTRACT** Visual object tracking in unconstrained environments is a challenging task in computer vision. How to design an efficient discriminative feature representation is one challenging issue. To improve the adaptability of the tracker to large object appearance changes, the observation model needs to be updated online. However, a bad model update using inaccurate training samples can lead to model drift problem. Therefore, how to design an efficient online observation model and a model update strategy are two other challenging issues. This paper proposes the concatenation of histogram of oriented gradients variant (HOGv) and color histogram as the feature representation to balance discriminative power and efficiency. The single-hidden-layer feedforward neural network (SFNN) is used as an observation model, and the recursive orthogonal least squares (ROLS) algorithm is used to update the model online. A bidirectional tracking scheme is designed to alleviate the model drift problem during online tracking. The proposed bidirectional tracking scheme consists of three modules: the forward tracking module, the backward tracking module and the integration module. The forward tracking module first finds all the candidate regions, and then, the backward tracking module calculates the respective confidence of each candidate region according to historical information. Finally, the integration module integrates both of the first two modules' results to determine the final tracked object and the model update strategy for the current frame. Extensive evaluations of the existing tracking benchmarks have shown that the proposed tracking framework results in significant performance improvements compared with the base tracker, and it outperforms most of the state-of-the-art trackers.

**INDEX TERMS** Visual object tracking, bidirectional tracking scheme, recursive orthogonal least squares, model update mechanism.

## I. INTRODUCTION

Visual object tracking, which is used to estimate the trajectory of a target specified in the initial frame, is a fundamental topic in computer vision [1], [2]. Visual object tracking has numerous applications, such as intelligent video surveillance, intelligent transportation, human-computer interactions and so on. Despite significant progress in recent decades, visual object tracking is still a challenging problem due to irregular changes in appearance that are caused by partial or full occlusion, cluttered backgrounds, fast motion, deformation and illumination changes.

Feature representation is one of the important factors for visual object tracking. Numerous hand-crafted features have been utilized for visual object tracking, such as color name [3], histograms of oriented gradient (HOG) [4], local binary pattern (LBP) [5] and so on. These hand-crafted features have relatively high computational efficiency but have been demonstrated to be less effective on the complex scene. Recently, convolutional neural networks (CNNs), with strong capabilities to learn feature representations, have demonstrated state-of-the-art performance in various computer vision tasks [6]–[8]. However, in visual object tracking, it is difficult to straightforwardly adopt CNNs, since they require a large number of training samples, and there is only one labeled positive sample that is extracted from the initial frames. One possible way is to utilize CNNs that have been trained on other tasks with a large-scale training dataset. Unfortunately, those transferring pre-trained CNN based tracking methods cannot attain ideal results

The associate editor coordinating the review of this manuscript and approving it for publication was Yudong Zhang.

because features that are learned offline sometimes cannot adapt well to a specific target. Although deep siamese networks based trackers [9]–[13] have achieved extremely compelling results in different benchmarks, these trackers heavily rely on large numbers of external training videos. Furthermore, the computational costs and storage requirements of CNNs are also expensive, so CNN-based trackers face difficulties to implement under some resource-constrained environments. Both discriminative power and computational speed should be addressed by real-time applications of visual object tracking. Thus, how to improve computational efficiency while retaining a high discriminative power and robustness is a challenging issue for designing feature representations for visual object tracking.

In visual object tracking, the observation model also plays an important role. Currently, state-of-the-art trackers are typically based on the tracking-by-detection framework. A binary classifier is adopted as the observation model of those trackers. This binary classifier aims to determine the decision boundary for discriminately separating the tracked target from the background and should have the capability to update online to better handle appearance variations. Zhang *et al.* [14] utilize probabilistic classifier to update the observation model, but it has difficultly estimating the class conditional probability due to the lack of a large number of training samples in the visual object tracking task. An online support vector machine (SVM) is also utilized for visual object tracking [15]. However, the expensive computational costs due to the quadratic programming problem limit real-time applications of this method. The iterative stochastic gradient descent method (SGD) is often employed to conduct online optimization of the parameters in the CNN-based models [16], but it is also time-consuming and often makes the model tends to overfitting. Thus, how to design an observation model (i.e., a binary classifier) that can obtain effective and efficient online update solutions for visual object tracking is the second challenging issue.

Currently, model updaters that control the updating strategy for the observation model have attracted increasing attention [17]–[24]. The observation model should be online updated in order to adapt to variations in appearance. However, the process of online updating the appearance model using potentially inaccurate training examples often results in the model drift problem. It is a common phenomenon that the tracker's observation model often leads to ambiguous inferences owing to occlusion and similar object disturbance in the current frame. These two issues are shown in Fig. 1, where the true target can be detected by the classifier (red rectangle). However, here, the highest confidence score is given to the wrong region (green rectangle). Model drift will occur if the wrong region is chosen as the final tracked object in the current frame, and then, the model is updated using inaccurate training samples. However, if the tracker can recognize both issues and does not update the observation model, the tracker will avoid drift and subsequently re-detect the target in a short period. Thus, how to design a model update strategy to maintain a good balance between model adaptation and drift is the third challenging issue for visual object tracking.

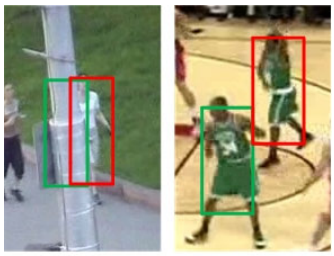



**FIGURE 1.** Two issues cause the current tracker's observation model to give ambiguous inferences. the red bounding box is the true target. the green bounding box is the wrong target where the highest confidence score is given by classifier. (a) Occusion. (b) Similar object disturbance.

This paper proposes an efficient framework for visual object tracking. The proposed tracking framework also treats tracking as a binary classification problem. The HOG variant (HOGv) [25], [26] and color histogram are utilized as the feature representation. The single-hidden-layer feedforward neural network (SFNN) is used as the classifier (observation model), and then, the SFNN conducts online learning using the recursive orthogonal least squares (ROLS) algorithm to distinguish the object from the background. The whole tracking framework consists of three modules: the forward tracking module, the backward tracking module and the integration module. A bidirectional tracking scheme is designed to determine the final tracked target and model update strategy for every frame.

This paper proposes concatenating the shape and color descriptors as feature representations because these two complementary features are the most effective features for the tracking task. The HOGv is used as the shape descriptor, and the color histogram is used as the color descriptor in this framework. the HOGv descriptor can maintain a good balance between redundancy and local details and has achieved great success in many fields, e.g., object detection [26] and traffic sign recognition [25]. Inspired by the extraction of HOGv descriptor, this paper proposes a color histogram descriptor that can also achieve both computational efficiency and high discriminative power. Therefore, the concatenation of the HOGv and color histogram descriptors can address the aforementioned first issue.

ROLS [27] is the online learning algorithm for the SFNN. Inspired by the theory [28] that the SFNN with a wide type of randomly generated hidden nodes is actually universal approximators, the input weights between input and hidden layers are randomly assigned. Since only the output weights between hidden and output layer are trained, layer-by-layer back-propagated tuning is not required. Therefore, the use of ROLS can address the aforementioned second issue.

The proposed bidirectional tracking scheme has the abilities to minimize the adverse effects of a bad model update as much as possible and subsequently re-detect the target

in a short period. This proposed tracking scheme first uses the forward tracking module to find all candidate regions. Then, the backward tracking module is activated to calculate the respective confidence of each candidate region according to historical information. Although the region has maximum confidence can be chosen as the final tracked object in the current frame, in order to further improve the robustness, this proposed bidirectional tracking scheme finally uses the integration module to integrate both the first two modules' results and the spatial prior to determine the final tracked object. Finally, a simple yet efficient model update strategy is also determined by the bidirectional tracking scheme to account for the appearance change of the target.

Many works focus on model update strategies [17]–[22], [29]–[34]. Most of these methods try to prevent bad updates from happening. In other words, these methods focus on designing a robust learning mechanism but do not consider a mechanism to correct for past mistakes. Once the learning mechanism fails, these trackers will miss the chance to evolve. Different from these methods, Zhang *et al.* [22] designs a multi-experts tracking framework, and the minimum entropy criterion is applied to choose the best expert to identify the model drift. However, this framework does not consider the strategy of updating each expert. Once all experts are polluted, the tracker will drift. Compared with these methods [17]–[22], [29]–[34], the superiority of the proposed bidirectional tracking scheme is that the integration module of the scheme can determine the better model update strategy and the final tracked target for the current frame by fully considering both of the first two modules' results and the spatial prior. This bidirectional tracking scheme is slightly similar to like ensemble learning techniques that combine the results that are achieved by weak trackers (i.e., the forward tracking module, the backward tracking module and the spatial prior) to produce a strong tracker (i.e., integration module) that is better than either of the weak trackers. Extensive evaluations of the existing tracking benchmarks [35]–[38] have demonstrated that significant performance improvements can be obtained by using the bidirectional tracking scheme in comparison with the base tracker (without the bidirectional tracking scheme) and this proposed framework also outperforms most of state-of-the-art methods. Therefore, the bidirectional tracking scheme can give a better solution for the aforementioned third challenging issue.

The remainder of this paper is organized as follows. Section II reviews related work on visual object tracking. Section III introduces the frameworks of this proposed method. Section IV and Section V present details about the extraction of features and ROLS-based classifier, respectively. Experimental results are shown in Section VI.

## II. RELATE WORK

There are mainly two categories of methods for visual object tracking depending on appearance model [1]. The first category is generative methods [39]–[43] which is to search for the most similar region to the tracked target in the consecutive frames by some generative process. To account for appearance changes of object, many online version of generative models have been proposed. The subspace is learned incrementally to model target appearance which has been introduced to tracking task [39], [40]. A tracking method based on sparse representation has been proposed [41] (called $\ell_1$-tracker), where the target is reconstructed by a sparse linear combination of the target and trivial templates, and it has been further improved recently [42], [43]. Although these $\ell_1$-trackers can effectively handle the corrupted appearance, expensive computational cost of $\ell_1$ minimization limits its applications in real-time scenarios. Despite much demonstrated success of these generative methods on the simple background, only the appearance of object is modeled while ignoring the influence of the background information resulting in failure in the complicated background.

The second category is discriminative methods [17]–[22], [29]–[34], [44]–[46] which treats the tracking problem as a binary classification task and aims to determine the decision boundary for separating the tracked target from the background discriminately. Numerous adaptive discriminative trackers [44]–[46] have been proposed so as to better handle appearance variations. However, the process of online updating the appearance model with potentially inaccurate training examples often brings the model drift problem. Various strategies have been introduced to alleviate drift problem [17]–[22], [29]–[33]. A novel online semi-supervised boosting tracking method is proposed [17] and the update process of this method depends on combined decision of a given prior and an on-line classifier. The multiple instance learning framework is introduced for online tracking and the model is trained with positive and negative examples bags [18]. Training-learning-detection (TLD) [19] employs two independent structural constraints to guide the sampling process so as to reduce inaccurate training samples. Online structured SVM is introduced for tracking and the model is trained using training samples with structured labels which can alleviate the effect of inaccurate training samples [20]. Self-paced learning is introduced for tracking [21], which is based on the selection of the most confident frames for learning appearance. Recently, Zhang *et al.* [22] propose a multi-expert tracking framework which maintains a collection of historical snapshots and the minimum entropy criterion is applied to expert selection for tracking. Ou *et al.* [32] and Liu *et al.* proposed a simple score function to predict the optimal candidate directly instead of learning a classifier. The coefficient constrained model [32] and sparsity-constrained model [33] are proposed respectively to select representative samples.

Feature representations play a great importance for visual object tracking. Hand-crafted features which are incapable to capture the appearance changes effectively during tracking so that they have an inherent limitation in complex background. Deep learning have demonstrated their superior representation power in various computer vision applications [6]–[8]. Hence, some deep learning based tracking methods [47]–[51]

have been proposed recently and they aim to replace hand-crafted features with high-level and robust features by training multilayer networks on external large-scale datasets. However, the pre-trained deep model sometimes cannot adapt well to a specific target so that these methods cannot attain ideal results in the field of visual object tracking. References [52]–[54] propose a multi-domain learning strategy to train the CNNs on a set of annotated video sequences, and showed that the CNNs trained on video sequences are more robust. Lin *et al.* [55] propose a novel localization-aware meta tracker (LMT) guided with adversarial features to better deal with various appearance variations. Long short-term memory (LSTM) also be used to learn long-range time dependencies [56]. Overall, deep learning based trackers are not efficient but very effective. We refer the readers to a comprehensive review on deep learning based tracking methods in [57].

Correlation filter based tracking methods have become increasingly popular recently due to its promising performance and low computational cost [58]. In essence, they are related to tracking-by-detection methods, since they can learn a discriminative regressor from foreground and background samples. Many attempts have been done to improve the original correlation filter model in terms of scale estimation [23], re-detection [24], kernelized correlation [59], complementary cues [60], deep feature integrations [11], [12], [61], spatial regularization [62]–[65], to name a few.

Currently, siamese networks based tracking methods [9]–[13], [66] gain more and more attention thanks to the outstanding performance compared with other state-of-the-art methods in some existing tracking benchmarks [35]–[38]. This category of tracking method formulates visual object tracking as a verification problem and aims to learn a similarity metric off-line with a large number of external tracking videos. However, it's a bit unfair to compare this category of tracking methods with those methods that do not require external tracking videos.

## III. FRAMEWORK OF THIS PROPOSED METHOD

The main idea of this proposed tracking framework is also to formulate the tracking as a binary classification problem. In this framework, the HOGv [25], [26] feature descriptor concatenates with the color histogram feature descriptor as the final feature representation. For the observation model, an SFNN is developed, and the ROLS algorithm [27] is used to online update the model (denoted as ROLS classifier). Different from the traditional methods that directly treat the region with the highest classification score as the final tracked target, this paper designed a bidirectional tracking scheme to determine the final tracked target. The framework of this proposed tracking method is depicted in Fig. 2. This framework includes three successive modules: the forward tracking module, the backward tracking module and the integration module. The basic principle of bidirectional tracking scheme is that the forward tracking framework aims to find all candidate regions, and the backward tracking module aims to calculate the respective confidence of each candidate region according to historical information. Finally, the integration module is used to integrate both of the first two modules' results and the spatial prior to determine the final tracked result. In other words, the forward tracking module aims to reduce the false negative regions, and the backward tracking module aims to further reduce the false positive regions. The integration module is used to boost the performance of the first two modules. Compared with the traditional methods, this proposed bidirectional tracking scheme can achieve more stable tracking results. Furthermore, a simple yet efficient model update strategy is also formed by the integration module to alleviate model drift.

### A. FORWARD TRACKING MODULE

This module is designed to implement three tasks. The first task is to find all candidate regions $\mathbf{C}$. The second task is to calculate the corresponding forward confidence values $\mathbf{F\_V}$. The last task is to determine whether the observation model needs to be updated according to the forward classifier's results (denoted as $F\_update$). This module contains six steps. The first step generates the samples using the sliding window approach that is centered on the previous predicted state. Then, the HOGv and color histogram are extracted as the feature representation of these samples. The third step puts the feature vector into the trained forward ROLS classifier (denoted as F_ROLS) to determine whether the window is the background or a target. The details of feature extraction and ROLS classifier can be viewed in Section IV and Section V, respectively. The fourth step selects those samples whose classification labels are the target category as positive samples, ranks these positive samples according to confidence values that are given by F_ROLS and selects the top several positive samples (the number of positive samples and selected positive samples are denoted as $P\_num$ and $SP\_num$, respectively, and $SP\_num$ is 10). The fifth step directly uses the clustering algorithm to merge $SP\_num$ positive samples together to form $N$ candidate targets (e.g. two candidates are formed, as showed in Fig. 2). The *groupRectangles* function on Opencv[1] is used to effectively and efficiently cluster the candidate rectangles. The idea of this function was proposed by David [67], and it clusters all the input rectangles using the rectangle equivalence criteria. The last step calculates $\mathbf{F\_V}$ and determines $F\_update$. The principle of calculating $\mathbf{F\_V}$ is that the more selected positive samples that a specified candidate contains, the higher the corresponding forward confidence value of this candidate is. In order to alleviate the model drift, we avoid using inaccurate samples to update the classifier as much as possible. Therefore, we do not update the observation model when the results that are obtained by the current F_ROLS are confusing, i.e., $F\_update = 0$. In this framework, we determine the state of confusion using $N$ and $SP\_num$. That is, confusion occurs when $N > 3$ (more than three similar object disturbances) or $SP\_num < 5$

[1] https://docs.opencv.org/3.4/d5/d54/group__objdetect.html#

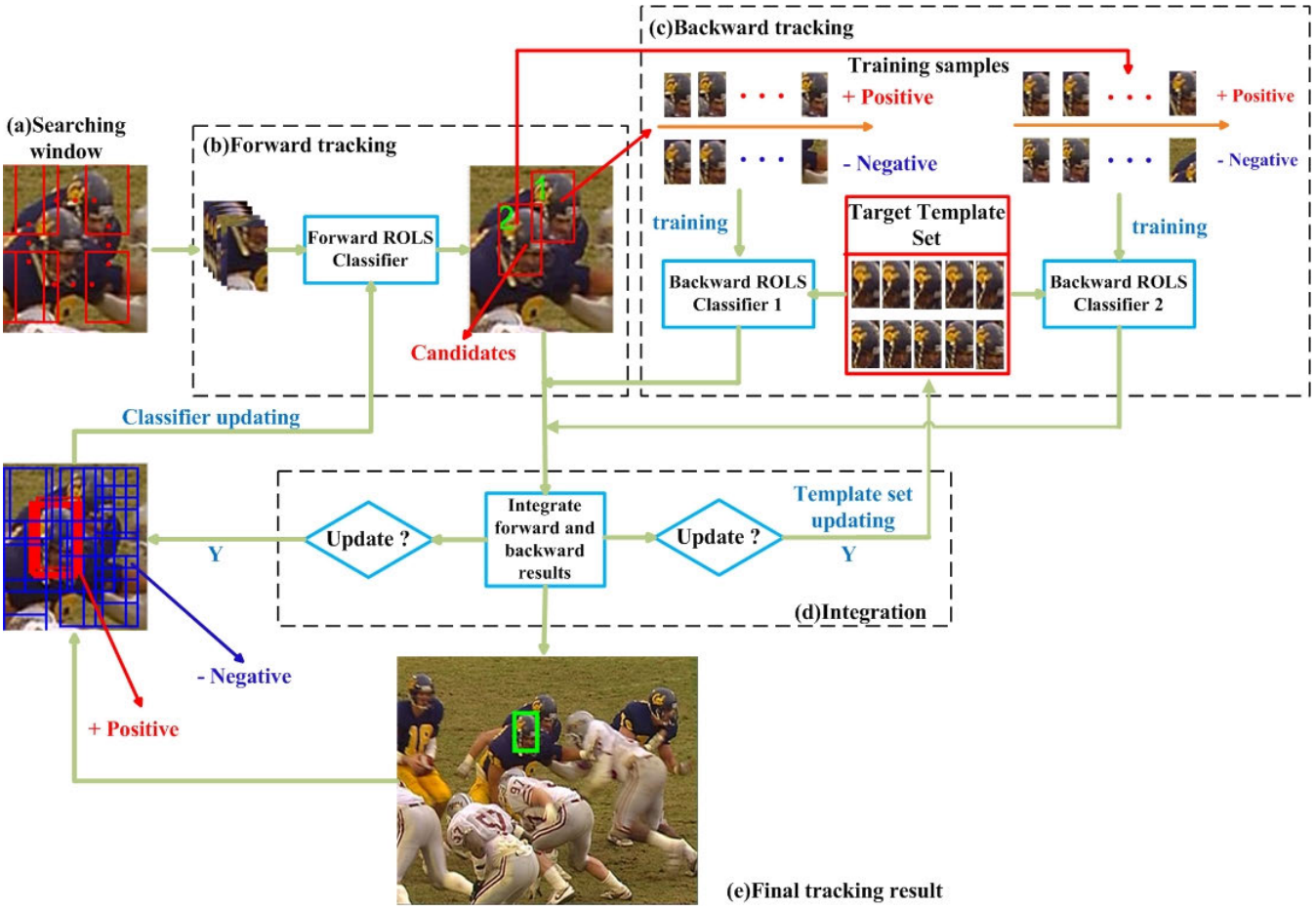



**FIGURE 2.** Framework of this proposed online tracking method.

(severe occlusion or out-of-view). More details of the forward tracking module are given in Algorithm 1.

### B. BACKWARD TRACKING MODULE

This module needs to implement two tasks. The first task is to calculate the corresponding backward confidence values **B_V**. The second task is to determine whether the observation model needs to be updated according to only the backward results (denoted as *B_update*). This module maintains a target template set and holds ten different modes of the target (i.e., historical information of the target). In the first ten frames, we only use the basic ROLS classifier (without the bidirectional tracking scheme) to track the target and collect each tracked result to initialize the target template set. The main idea of the backward tracking module is that if the specified candidate is the true tracked target, a new classifier that is initialized using this candidate as the positive class can correctly determine the labels of the target templates. Therefore, we treat $N$ candidates as the true tracked target and train the corresponding $N$ backward ROLSs (denoted as $B\_ROLS_i$). The principle of calculating **B_V** is that the more target templates that are correctly recognized by $B\_ROLS_i$, the higher the corresponding backward confidence value of $B\_ROLS_i$ (i.e., the higher the corresponding backward confidence value the $i$th candidate achieves). The principle of determining *B_update* is the same as the forward module (i.e., $B\_update = 0$ when the results that are obtained by current B_ROLS are confusing). The condition that determine the state of confusion is $max(\mathbf{B\_V})$. As long as $max(\mathbf{B\_V}) \geq 60\%$, we set $B\_update = 1$. The idea of the backward module is slightly similar to [68], [69]. Different from our method, [68] utilizes this idea to facilitate unsupervised feature learning, [69] focus on online evaluation of the tracking performance. More details of the backward tracking module are given in Algorithm 2.

### C. INTEGRATION MODULE

The integration module integrates both of the first two modules' results and the spatial prior to determine the final tracked target. First, we use the formula $(F_{max}, F_{index}) = \max(\mathbf{F\_V})$ to compute the confidence value of the best candidate and the corresponding index according to the results of the forward module, and we use the formula $(B_{max}, B_{index}) = \max(\mathbf{B\_V})$ to compute the confidence value of the best candidate and the corresponding index according to the results of the backward module. The results of the first two modules are mostly

**Algorithm 1** Forward Tracking Module

**Require:** Frame $\mathbf{I}^t$
**Ensure:** Candidate targets ($\mathbf{C}$) and corresponding forward confidence values ($\mathbf{F\_V}$), $F\_update$
Crop out the searching window and generate samples;
Feature extraction and put feature matrix into F_ROLS to classify;
Initialize $F\_update = 0$;
**if** $P\_num = 0$ **then**
  $C_1 \longleftarrow$The region of the same location as $\mathbf{I}^{t-1}$ tracked target, $F\_V_1 = 0\%$, $F\_update = 0$;
**else**
  Rank and select positive samples;
  Merge selected positive samples to form $N$ candidates;
  **for** $i = 0$ to $N$ **do**
    $Num(C_i) \longleftarrow$the number of selected positive samples belonging to $C_i$;
    $F\_V_i \longleftarrow Num(C_i)/SP\_num \times 100\%$;
  **end for**
  **if** $N \leq 3$ and $SP\_num \geq 5$ **then**
    $F\_update = 1$;
  **end if**
**end if**

**Algorithm 2** Backward Tracking Module

**Require:** Candidate targets ($\mathbf{C}$), feature matrix of searching window
**Ensure:** Corresponding backward confidence values of $\mathbf{C}$ ($\mathbf{B\_V}$), $B\_update$
**for** $i = 0$ to $Num(\mathbf{C})$ **do**
  Take $C_i$ as positive category and collect Corresponding positive and negitive features to train $B\_ROLS_i$;
  Judge the labels of target templates using $B\_ROLS_i$;
  $Num(C_i) \longleftarrow$the number of templates being correctly judged by $B\_ROLS_i$;
  $B\_V_i \longleftarrow Num(C_i)/10 \times 100\%$;
**end for**
$max\_value = \max(\mathbf{B\_V})$;
**if** $max\_value \geq 60\%$ **then**
  $B\_update = 1$;
**else**
  $B\_update = 0$;
**end if**

consistent (i.e., $B_{index} = F_{index}$). The $B_{index}$th (= $F_{index}$th) candidate can be directly chosen as the current tracked target in this case. However, it is difficult to determine the current tracked target when the results of the first two modules are inconsistent (i.e., $B_{index} \neq F_{index}$). When the confidence of the $B_{index}$th candidate is high and much better than the $F_{index}$th candidate (i.e., $B_{max} \geq 60\%$ and $B\_V_{F_{index}} < 60\%$), we can directly choose the $B_{index}$th candidate as the current tracked target. However, confusion occurs when the results of both modules are high or low. In this case, it is difficult to directly determine the final tracked target based on confidence. Therefore, the spatial prior is chosen as the final determinant, and the candidate (from $C_{B_{index}}$ and $C_{F_{index}}$) that has the shortest distance to the previous tracked target is chosen as the current tracked target. Prior works [22], [54] also integrate the spatial prior to determine the final tracked target. However, those methods define the spatial prior as a 2D Gaussian distribution map that is centered on the previously predicted location of the STD $\sigma$ and then the confidence map is weighted by the spatial prior map is directly used to determine the final tracked target. The experimental results in Section VI validated that the use of spatial prior can slightly improve the performance.

This module also ultimately determine whether the F_ROLS and the $\mathbf{T}_{target}$ need to be updated by synthetically analyzing the results of the first two modules. To ensure that the candidate set contains the true target as much as possible (even if an ambiguous inference owing to bad $F\_update$ update occurs), we need to update the F_ROLS as soon as possible so that it can account for appearance changes of the object. Therefore, we set two conditions: $F\_update = 1$ and $B\_update = 1$. As long as one of these two conditions is met, we start to update F_ROLS.

The target template set $\mathbf{T}_{target}$ holds ten different modes of the target, and it also needs to be updated online. $\mathbf{T}_{target}$ should be more carefully updated in order to avoid target templates from being wrongly replaced. Therefore, compared to the F_ROLS update condition, the $\mathbf{T}_{target}$ update needs to meet another condition: $F_{index} = B_{index}$. When meeting all of the update conditions, we add the current tracked target into the target template set and discard the oldest tracked target. To avoid all of the target templates being wrongly replaced, we fix five of these ten templates. More details of the integration module are given in Algorithm 3.

## IV. EXTRACTION OF HOGV AND COLOR HISTOGRAM DESCRIPTOR

This proposed tracking framework uses the HOGv [25], [26] descriptor and color histogram descriptor as the feature representation. The HOGv descriptor consists of two improvements compared with the original HOG descriptor [4]. First, both contrast sensitive and contrast insensitive orientations of gradients are included such that more detailed local information of objects can be involved into the accumulated histograms. Second, after each cell's oriented histogram is normalized over four of its neighboring blocks, respectively, these normalized histograms of this cell are dimensionally reduced based on a principal component analysis (PCA) like strategy [26] so as to remove redundant information. The HOGv descriptor is a good shape feature widely used for object detection [26]. We refer the readers to a comprehensive review on the HOGv in [25], [26].

In order to better represent the object, color information also plays a key role. This proposed color histogram refers

**Algorithm 3** Integration Module

**Require:** Outputs of the first two modules
**Ensure:** Tracked target $O^t$ at frame $\mathbf{I}^t$

```
(F_max, F_index) = max(F_V);
(B_max, B_index) = max(B_V);
if F_index = B_index then
    O^t ⟵ C_{B_index} (or C_{F_index})
else
    if B_V_{F_index} < 60% and B_max ≥ 60% then
        O^t ⟵ C_{B_index}
    else
        dB ⟵ The distance between C_{B_index} and the I^{t-1} tracked target;
        dF ⟵ The distance between C_{F_index} and the I^{t-1} tracked target;
        O^t ⟵ (dB < dF)?C_{B_index} : C_{F_index}
    end if
end if
if F_update = 1 or B_update = 1 then
    Online update F_ROLS;
end if
if (F_update = 1 and B_update = 1) and F_index = B_index then
    T_target ⟵ O^t, discard the oldest mode.
end if
```

to the idea of extracting HOGv features. The entire picture is first divided into non-overlapping grids. The histograms were calculated for each grid and all histograms are concatenated to form a color histogram descriptor. To reduce the disturbance by various light and weather conditions, the RGB space was converted to Lab space.

In this paper, we directly concatenate the HOGv descriptor with the color histogram descriptor as the final feature representation. We carefully set the configuration parameters (e.g., the number of bins and cells) of both two descriptors in order to better balance discriminative power and efficiency. Details of the parameters setting can be shown in Section VI. The final concatenate feature embedded in the proposed tracker has low dimensions (below 400), which is a relatively lightweight feature representation, and can meet the efficiency requirement of real-time tracking. However, how to integrate data from different modalities more effectively and efficiently requires further exploration in future work.

## V. ROLS BASED CLASSIFIER

### A. STRUCTURE OF ROLS

ROLS [27] is basically a machine learning algorithm for online training SFNN. Considering the unified framework of SFNN, The equation for calculating the output value is given by

$$f_L(\mathbf{x}) = \sum_{i=1}^{L} \theta_i h_i(\mathbf{x}; \mathbf{w}_i, b_i). \tag{1}$$

where $\mathbf{x}$ is the input feature vector of an image patch. The dimension number of $\mathbf{x}$ is denoted as P. $h_i(\mathbf{x}; \mathbf{w}_i, b_i)$ is the output of $i$th hidden node with respect to the input $\mathbf{x}$, where $\mathbf{h}$ is the activation function. This paper uses the sigmoid function as the activation function. $\mathbf{w}_i$ is the weight vector between the input nodes and the $i$th hidden node. $b_i$ is the bias of the $i$th hidden node and L is the number of hidden nodes. $\theta_i$ is the output weight between the $i$th hidden node to the output node. Given total training samples $\{\mathbf{x}_i\}_{i=1,\ldots,N}$ and its corresponding labels $\{\mathbf{t}_i\}_{i=1,\ldots,N}$, if the network can fit these N samples exactly, we have the following compact formulation:

$$\mathbf{H}\Theta = \mathbf{T} \tag{2}$$

where $\mathbf{H} \in \mathbb{R}^{N\times L}$ is called the hidden output matrix of the neural network. $\Theta \in \mathbb{R}^{L\times C}$ and $\mathbf{T} \in \mathbb{R}^{N\times C}$ are corresponding matrices of the output weights and targets, respectively. $C$ is the number of output nodes, for tracking task, $C = 2$.

Inspired by the theory [28] that the SFNN with a wide type of randomly generated hidden nodes are actually universal approximators, so the input weights and biases are randomly assigned in this paper.

### B. ONLINE UPDATE PROCESS

Model online update means that the training data are sequentially (one-by-one or chunk-by-chunk) presented to the learning algorithm. considering (2) for a set of training samples $\mathcal{X}_t = \{\mathbf{x}_i\}_{i=N_{t-1}+1}^{N_t}$ and its corresponding labels $\mathcal{T}_t = \{\mathbf{t}_i\}_{i=N_{t-1}+1}^{N_t}$, at iteration t, we have:

$$\mathbf{T}^{(t)} = \mathbf{H}(\mathbf{X}^{(t)})\Theta^{(t)} + \mathbf{E}^{(t)} \tag{3}$$

where

$$\mathbf{X}^{(t)} = [\mathcal{X}_1, \mathcal{X}_2, \cdots, \mathcal{X}_t] = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_{N_{t-1}}, \cdots, \mathbf{x}_{N_t}]^T \tag{4}$$

$$\mathbf{T}^{(t)} = [\mathcal{T}_1, \mathcal{T}_2, \cdots, \mathcal{T}_t] = [\mathbf{t}_1, \mathbf{t}_2, \cdots, \mathbf{t}_{N_{t-1}}, \cdots, \mathbf{t}_{N_t}]^T \tag{5}$$

$$\mathbf{E}^{(t)} = [\mathbf{e}_1, \mathbf{e}_2, \cdots, \mathbf{e}_{N_{t-1}}, \cdots, \mathbf{e}_{N_t}]^T \tag{6}$$

$\mathbf{E}^{(t)}$ is the error matrix. We need to solve an optimal $\Theta^{(t)}$ at iteration t that minimizes the following error cost function:

$$\boldsymbol{J}(t) = \|\mathbf{E}(t)\|_F^2 = \left\| \begin{bmatrix} \mathbf{T}^{(t-1)} \\ \mathcal{T}_t \end{bmatrix} - \begin{bmatrix} \mathbf{H}^{(t-1)} \\ \mathcal{H}_t \end{bmatrix} \Theta^{(t)} \right\|_F^2. \tag{7}$$

where $\| * \|_F$ is the Frobenius norm defined as $\|X\|_F = \sqrt{\text{trace}(X^T X)}$,

$$\mathbf{H}^{(t-1)} = [\mathbf{h}_1, \mathbf{h}_2, \cdots, \mathbf{h}_{N_{t-1}}]^T \tag{8}$$

$$\mathcal{H}_t = [\mathbf{h}_{N_{t-1}+1}, \mathbf{h}_{N_{t-1}+2}, \cdots, \mathbf{h}_{N_t}]^T \tag{9}$$

One can obtain the ROLS applying orthogonal decomposition.

Then the formula of (7) can be written as:

$$\boldsymbol{J}(t) = \|\mathbf{E}(t)\|_F^2 = \left\| \begin{bmatrix} \widehat{\mathbf{T}}^{(t-1)} \\ \mathcal{T}_t \end{bmatrix} - \begin{bmatrix} \mathbf{R}^{(t-1)} \\ \mathcal{H}_t \end{bmatrix} \Theta^{(t)} \right\|_F^2. \tag{10}$$

To find the update for $\mathbf{R}^{(t-1)}$ and $\widehat{\mathbf{T}}^{(t-1)}$, one can compute another orthogonal decomposition:

$$\begin{bmatrix} \mathbf{R}^{(t-1)} \\ \mathcal{H}_t \end{bmatrix} = \mathbf{Q}^{(t)} \begin{bmatrix} \mathbf{R}^{(t)} \\ \mathbf{0} \end{bmatrix}, \tag{11}$$

$$\begin{bmatrix} \widehat{\mathbf{T}}^{(t)} \\ \widetilde{\mathcal{T}}^t \end{bmatrix} = (\mathbf{Q}^{(t)})^T \begin{bmatrix} \widehat{\mathbf{T}}^{(t-1)} \\ \mathcal{T}_t \end{bmatrix}. \tag{12}$$

Hence, the optimal $\Theta^{(t)}$ in (10) can be easily solved from (13) by backward substitution.

$$\mathbf{R}^{(t)}\Theta^{(t)} = \widehat{\mathbf{T}}^{(t)} \tag{13}$$

In summary, the procedure of the ROLS algorithm is as following: Let $\alpha\mathbf{I}$ and 0 be initial values for $\mathbf{R}^{(0)}$, $\widehat{\mathbf{T}}^{(0)}$, where $\alpha$ is a small positive number. We set $\alpha = 0.01$ in the tracking framework. For solving $\Theta^{(t)}$ in (13), we calculate $\mathbf{R}^{(t)}$ in (11) and $\widehat{\mathbf{T}}^{(t)}$ in (12) at iteration t.

## VI. EXPERIMENTS

In this section, we provide the implementation details of the proposed tracker and analyze the effects of the modules in the tracker by ablation studies. We denote the proposed tracker as BiROLS for clarity. Extensive experiments are conducted to evaluate the BiROLS tracker against the state-of-the-art trackers on four benchmarks: OTB-2013 [35], OTB-2015 [36], VOT-2015 [37], and VOT-2017 [38]. All the tracking results are using the reported results to ensure a fair comparison.

### A. EXPERIMENTAL SETUP

The proposed tracking framework is implemented using Matlab and C++ with OpenCV library and runs at around 12 fps on a PC with Inter i7-4790 CPU (3.6 GHz).

To balance the computational efficiency and capability of the feature representation, we first scale the images of each video to the fixed size of w × h pixels according to the target's size in the first frame and the smallest of w and h is set to 36 pixels. For color video, the HOGv descriptor and color histogram descriptor are utilized. To extract HOGv descriptor, each window is divided into 4 × 4 non-overlapping cells and 2 × 2 cells are grouped into a block. We set 12 orientation bins over 0°∼360° and 6 orientation bins over 0°∼180°. So the dimension of the color image's HOGv descriptor is 88. To extract color histogram descriptor, each window is divided into 5 × 5 non-overlapping cells. Three spaces are respectively voted into the corresponding histograms (the number of the bin is set to 4) for each cell according to the pixel's value. All histogram are concatenated to form a 300 dimensional vector as the color descriptor. For grayscale video, we only use HOGv descriptor, in order to represent more local details, the images of each video are divided into 6 × 6 non-overlapping cells. So the dimension of the gray image's HOGv descriptor is 352. Search for the target is conducted within a radius of $\sqrt{wh}$ of the previous prediction (the ratio is set to 1.5 in our method). To generate samples in each frame, the window stride is set to [3,3]. The number of hidden layer nodes is set to L = 500. For model initialization, we collect 50 positive and 120 negative samples from the first frame, where negative samples do not significantly overlap the prediction (IOU < 0.5). For the model online update, we collect 10 positive and 40 negative samples every frame. We fix all the parameters for all experiments.

### B. DATASETS AND EVALUATION METRICS

#### 1) OTB BENCHMARK

The OTB-2013 [35] and OTB-2015 [36] datasets are composed of 51 and 100 sequences, respectively. We report the results of the one-pass evaluation (OPE) based on average success and precision rate. The success plot illustrates the percentages of successfully tracked frames at the threshold of intersection over union (IOU) in the range of 0 to 1. The area under the curve (AUC) is used to rank the trackers. The precision plot illustrates the percentage of successfully tracked frames at the threshold of center location error (CLE), and the representative precision score at Threshold = 20 pixels is used to rank the trackers. For more thorough evaluations and analysis of the performance of the trackers, the providers propose to classify the sequences by annotating them with the 11 attributes including illumination variation (IV), scale variation (SV), occlusion (OCC), 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).

#### 2) VOT BENCHMARK

The visual object tracking (VOT) benchmark has many versions, and we use VOT-2015 [37] and VOT-2017 [38]. Both datasets comprise 60 videos showing various objects in challenging backgrounds. Different from OTB benchmarks, the VOT benchmarks evaluate a tracker by applying a reset-based methodology. Whenever a tracker has no overlap with the ground truth, the tracker will be restarted by the ground-truth five frames after the failure. The performance is measured in terms of expected average overlap (EAO), accuracy and robustness. For VOT-2017, the toolkit also carried out the OTB [35] no-reset (unsupervised) experiment. The tracking performance of this experiment was evaluated in terms of the average overlap (AO).

### C. ABLATION STUDIES

In this subsection, we first compared the full tracking framework (BiROLS) with a baseline tracker in order to evaluate the performance of the proposed bidirectional tracking scheme. We then carry out some ablation studies to better understand the contributions of each component of the proposed tracker. All ablation studies are performed on the OTB-2013 dataset [35].

To evaluate the performance of this proposed bidirectional tracking scheme, we compare the BiROLS tracker with a baseline tracker, which is denoted as ROLS. The ROLS tracker is only based on one base classifier (i.e., one SFNN is

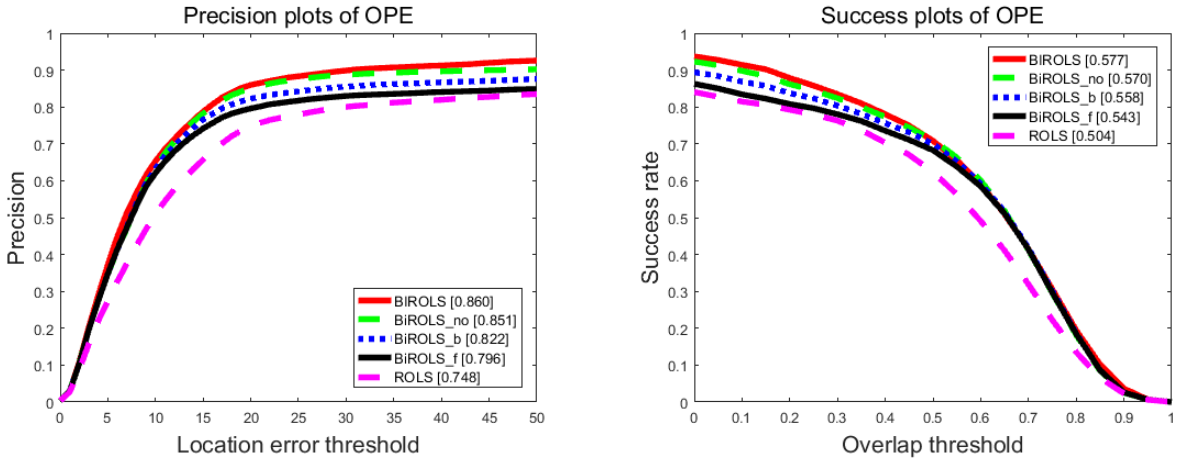



**FIGURE 3.** The precision and success plots of BiROLS tracker with different configurations on the OTB-2013 dataset [35].

developed as the observation model, and the model is updated every frame by simply using the ROLS algorithm without the bidirectional tracking scheme). There are two main differences between these two methods: 1) The ROLS tracker directly treats the region that has the highest classification score as the final tracked object. In addition, the BiROLS tracker first searches for more than one region to used as candidates, and it then chooses the best result from these candidates as the final tracked object according to the bidirectional tracking scheme. 2) The ROLS tracker is updated online at every frame while the model update strategy of the BiROLS tracker for the current frame is determined by the bidirectional tracking scheme. For a fair competition, the other parameters of the ROLS tracker are the same as those of the BiROLS tracker.

Fig. 3 illustrates the success and precision plots of the BiROLS tracker and ROLS tracker on the benchmark dataset. It can be seen that the BiROLS tracker achieves a noticeable performance increase. The BiROLS tracker outperforms the ROLS tracker by approximately 11.2% and 7.3% in terms of the precision plot and success plot, respectively. Based on this observation, we can confirm the importance of the model update strategy in the tracker and the effectiveness of the proposed bidirectional tracking scheme.

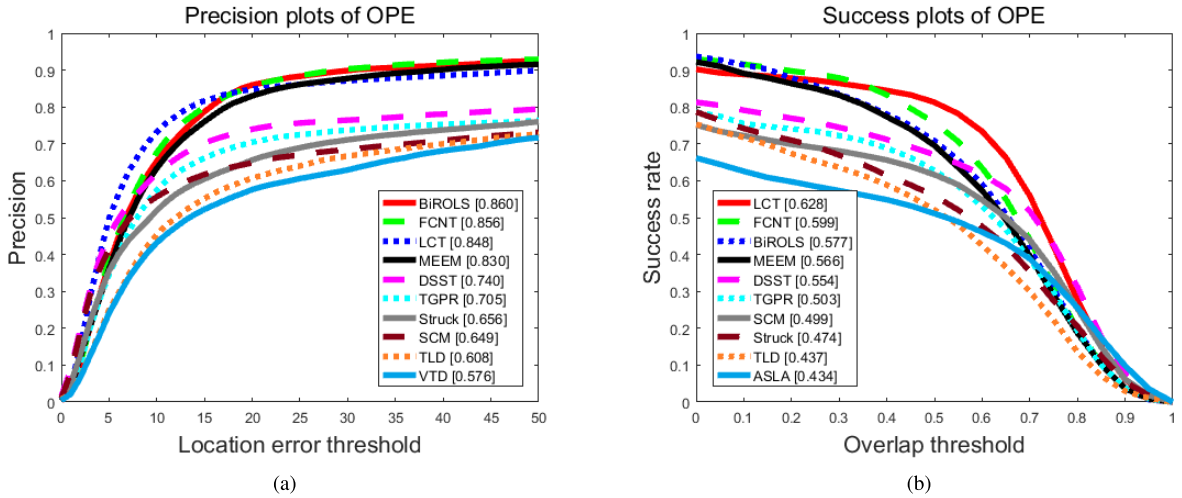
To evaluate the contributions of each component of the proposed tracker, we compare the BiROLS tracker with three variants: BiROLS_f (BiROLS without both the backward and integration modules), BiROLS_b (BiROLS without the integration module) and BiROLS_no (BiROLS without the spatial prior in the integration module). BiROLS_f tracker directly chooses the $F_{index}$th candidate as the final tracked target according to **F_V** and choose *F_update* to determine whether to update the F_ROLS. BiROLS_b tracker chooses the $B_{index}$th candidate as the final tracked target according to **B_V** and choose *B_update* to determine whether to update the $\mathbf{T}_{target}$. BiROLS_no tracker also chooses the $B_{index}$th candidate as the final tracked target according to **B_V** but the integration module is used to determine whether to update the F_ROLS and $\mathbf{T}_{target}$.

From Fig. 3, it can be seen that BiROLS_f tracker significantly improves the performance compared with the ROLS tracker. This improvement can be attributed to the simple update strategy in the forward module benefitting the tracking performance.In addition, the BiROLS_b tracker outperforms the BiROLS_f tracker by about 2.6% and 1.5% in terms of the precision plot and success plot, respectively. This result indicated that the backward module has a better capability to determine the final tracked target from the candidate targets than the forward module. From the performances of BiROLS_no tracker and BiROLS tracker, one can see that the performance of the BiROLS tracker obtains an obvious improvement by using the integration module. This result demonstrates that the integration module indeed can integrate both the first two modules' results and the spatial prior to boost the performance. By further comparing the performances of BiROLS_b tracker, BiROLS_no tracker and BiROLS tracker, it can be confirmed that the spatial prior helps to slightly boost the performance, but the update strategy that is determined by the integration module helps to more significantly improve the capability of tracker. To summarize, all three components are helpful in improving the tracking accuracy, which together explain the favorable performance of our BiROLS tracker.

### D. QUANTITATIVE EVALUATION

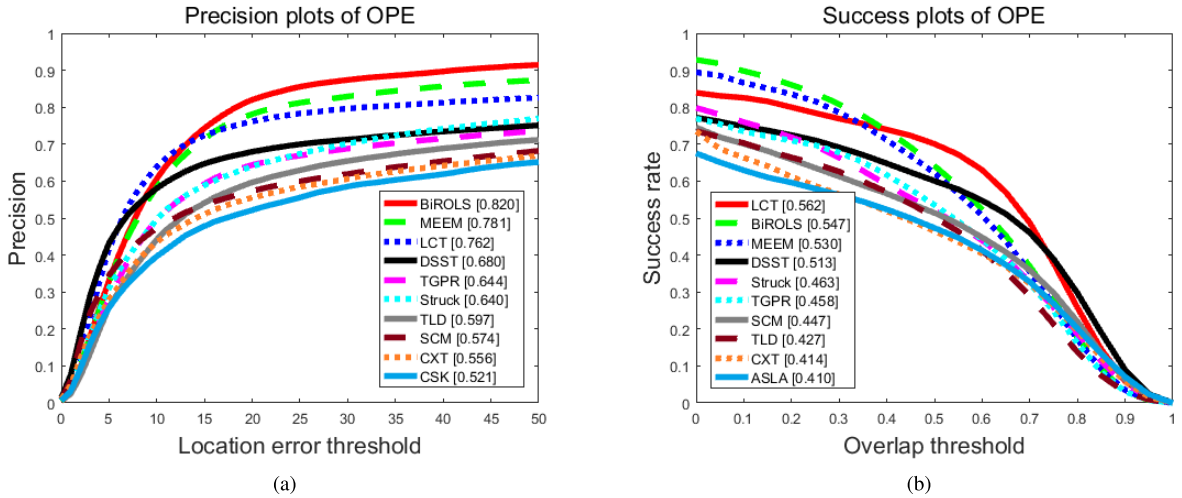
#### 1) COMPARISONS ON OTB BENCHMARKS

On OTB-2013, OTB-2015 benchmarks, we compare the proposed BiROLS tracker against 29 trackers that were reported by [35]. We also compare our method with recent state-of-the-art methods: FCNT [50], MEEM [22], LCT [24], TGPR [72], and DSST [23]. Among these five methods, the FCNT [50] is the CNN based tracker, and the




(a) (b)

**FIGURE 4.** The precision plot and success plot of OPE for our tracker and the compared trackers on OTB-2013 dataset [35].




(a) (b)

**FIGURE 5.** The precision plot and success plot of OPE for our tracker and the compared trackers on OTB-2015 dataset [37].

LCT tracker [24] and DSST tracker [23] are correlation filter based trackers. It should be mentioned that the LCT tracker [24] also contains an online random fern classifier to re-detect objects in case of tracking failure. The MEEM tracker [22] uses a multi-expert mechanism to alleviate the model drift problem. The TGPR tracker [72] treats the tracking problem as a Gaussian process regression task. Fig. 4 and Fig. 5 presents the precision and success plot for OTB-2013 and OTB-2015, respectively. For presentation clarity, only the top 10 trackers are presented in each plot.

With respect to the OTB-2013, BiROLS tracker achieves performance that is competitive with that of the state-of-the-art methods in both metrics. It can be seen from Fig. 4 that the BiROLS tracker ranks the first position in terms of the precision rate while it ranks third in terms of the success rate among all of these state-of-the-art methods. The reason for achieving a relatively lower success score is that BiROLS tracker does not handle scale variations while the FCNT tracker [50] and LCT tracker [24] contain schemes to estimate the change in the scale. It can also be seen that the performance of the BiROLS tracker is close to the performance of the FCNT tracker. However, it is worth mentioning that the BiROLS tracker only uses specific target information of the first frame to initialize the model while the FCNT tracker uses CNN model that is pre-trained using auxiliary training data.

The competitive performance of the BiROLS tracker can be further demonstrated using OTB-2015 dataset. It can be seen from Fig. 5 that the BiROLS tracker ranks first in terms of the precision rate while it ranks second in terms of the success rate among all of these state-of-the-art methods (We did not compare BiROLS tracker with the FCNT tracker on the OTB-2015 benchmark, since the results of the FCNT tracker were not published). Note that the BiROLS tracker

**TABLE 1.** Average precision scores and average success scores in terms of individual attributes on the OTB-2013 dataset. The red fonts indicate the best performance, the blue fonts indicate the second best ones, and the green fonts indicate the third best ones.

| Attribute | VTD [70] | TLD [19] | SCM [71] | Struck [20] | TGPR [72] | DSST [23] | MEEM [22] | LCT [24] | FCNT [50] | **BiROLS** |
|---|---|---|---|---|---|---|---|---|---|---|
| IV | 0.557/0.420 | 0.537/0.399 | 0.594/0.473 | 0.558/0.428 | 0.671/0.484 | 0.730/0.561 | 0.766/0.533 | 0.792/0.588 | 0.830/**0.598** | **0.837**/0.557 |
| OPR | 0.620/0.434 | 0.596/0.420 | 0.618/0.470 | 0.597/0.423 | 0.678/0.485 | 0.736/0.536 | 0.840/0.557 | **0.850**/**0.624** | 0.831/0.581 | 0.846/0.557 |
| SV | 0.597/0.405 | 0.606/0.421 | 0.672/0.518 | 0.639/0.425 | 0.620/0.418 | 0.738/0.546 | 0.785/0.498 | 0.758/0.553 | 0.830/**0.558** | **0.841**/0.512 |
| OCC | 0.545/0.403 | 0.563/0.402 | 0.640/0.487 | 0.564/0.413 | 0.675/0.484 | 0.706/0.532 | 0.799/0.552 | 0.845/**0.627** | 0.797/0.571 | **0.846**/0.572 |
| DEF | 0.501/0.377 | 0.512/0.378 | 0.586/0.448 | 0.521/0.393 | 0.691/0.510 | 0.658/0.506 | 0.846/0.560 | 0.873/**0.668** | **0.917**/0.644 | 0.881/0.602 |
| MB | 0.375/0.309 | 0.518/0.404 | 0.339/0.298 | 0.551/0.433 | 0.537/0.434 | 0.544/0.455 | 0.715/0.541 | 0.664/0.524 | 0.789/0.580 | **0.810**/**0.594** |
| FM | 0.352/0.302 | 0.551/0.417 | 0.333/0.296 | 0.604/0.462 | 0.493/0.396 | 0.513/0.428 | 0.742/0.553 | 0.665/0.534 | 0.767/0.565 | **0.792**/**0.575** |
| IPR | 0.599/0.430 | 0.584/0.416 | 0.597/0.458 | 0.617/0.444 | 0.675/0.479 | 0.768/0.563 | 0.800/0.535 | 0.802/**0.592** | **0.811**/0.555 | 0.796/0.528 |
| OV | 0.462/0.448 | 0.576/0.457 | 0.429/0.361 | 0.539/0.459 | 0.505/0.442 | 0.511/0.462 | 0.727/0.606 | 0.728/0.594 | 0.741/0.592 | **0.865**/**0.663** |
| BC | 0.571/0.425 | 0.428/0.345 | 0.578/0.450 | 0.585/0.458 | 0.717/0.522 | 0.694/0.517 | 0.797/0.569 | 0.796/**0.587** | **0.799**/0.564 | 0.780/0.546 |
| LR | 0.168/0.177 | 0.349/0.309 | 0.305/0.279 | 0.545/0.372 | 0.538/0.370 | 0.497/0.408 | 0.490/0.360 | 0.352/0.286 | **0.765**/**0.514** | 0.667/0.463 |
| Overall | 0.576/0.418 | 0.608/0.437 | 0.649/0.499 | 0.656/0.474 | 0.705/0.503 | 0.740/0.554 | 0.830/0.566 | 0.848/**0.628** | 0.856/0.599 | **0.860**/0.577 |

achieves a precision score of 0.820, which outperforms the suboptimal tracker (i.e., MEEM) by a large margin.

To gain insight into the performance of BiROLS tracker, we further evaluate the top 10 trackers' performances on sequences with 11 attributes using the OTB-2013 benchmark. Table 1 shows the average precision scores and average success scores for the different attributes, respectively. It can be seen that the BiROLS tracker performs favorably in almost challenges and achieves the highest average precision score for 6 attributes. However, as shown in Table 1, the BiROLS tracker only achieves the highest average success score for 3 attributes due to lack of the scale estimation strategy. As shown in Table 1, for FM, MB and OV attributes, the BiROLS tracker performs the best against the other trackers in terms of both metrics. Specifically, the BiROLS tracker achieved extremely compelling performances for the OV as it outperforms the suboptimal tracker by approximately 12.4% and 6.9% in terms of the precision plot and success plot, respectively. It can be attributed that BiROLS tracker can redetect the object.

### 2) COMPARISONS ON VOT BENCHMARKS

On the VOT-2015 benchmark, we compare the BiROLS tracker with state-of-the-art trackers, including MDNet [52], DeepSRDCF [63], Struck [20], MEEM [22], TGPR [72], KCFv2 [59], DSST [23], MIL [18], and IVT [39]. Table 2 shows the results of our BiROLS tracker and state-of-the-art trackers. We can observe that the BiROLS tracker ranks third in terms of EAO and robustness metrics. However, the top performing trackers (MDNet and DeepSRDCF) are far from meeting the real-time requirements. In addition, MDNet tracker employs a large number of external tracking videos for training, while our tracker does not need offline training. It is worth noting that the BiROLS tracker performs much worse than the top performance trackers in terms of accuracy metric, which is ascribed to the lack of scale estimation strategy. However, the BiROLS tracker achieve comparable performance with DeepSRDCF tracker in terms of robustness metric.

On VOT-2017 benchmark, we compare BiROLS tracker with state-of-the-art trackers, including ECO [64], SiamDCF [12], MEEM [22], SiamFC [11], Staple [60], KCF [59], MIL [18], Struck [20], and DSST [20]. It can be seen from the Table 3 that BiROLS tracker also ranks the 3rd in terms of EAO and robustness metrics among all these state-of-the-art methods. Specifically, the performance of our BiROLS tracker is similar to the SiamDCF tracker in terms of robustness.

We further analyze the performance of BiROLS tracker on VOT-2017 benchmark by no-reset experiments. Table 4 shows the results of these trackers on videos with different attributes based on the average overlap. The annotated attributes include camera change (Cam.), illumination change (Illu.), motion change (Mot.), occlusion (Occ.), size change (Size), and not assigned (N/A). It can be seen that the BiROLS

**TABLE 2.** Comparison with the state-of-the-art trackers on the VOT-2015 dataset. The results are presented in terms of expected average overlap (EAO), accuracy (A), and robustness (R).

| Tracker | EAO | A | $R^2$ |
|---|---|---|---|
| MDNet [52] | **0.379** | **0.608** | **0.214** |
| DeepSRDCF [63] | 0.319 | 0.566 | 0.280 |
| **BiROLS** | 0.271 | 0.507 | 0.294 |
| Struck [20] | 0.245 | 0.470 | 0.418 |
| MEEM [22] | 0.220 | 0.499 | 0.499 |
| TGPR [72] | 0.193 | 0.475 | 0.627 |
| KCFv2 [59] | 0.192 | 0.484 | 0.569 |
| DSST [23] | 0.172 | 0.547 | 0.760 |
| MIL [18] | 0.170 | 0.421 | 0.729 |
| IVT [39] | 0.122 | 0.445 | 1.147 |

[2]It should be noted that the values of robustness achieved by the latest vot toolkit are different from the reported VOT-2015 results [36] (a new normalization approach may be used in the latest vot toolkit), but the ranking of these trackers is consistent.

**TABLE 3.** Comparison with the state-of-the-art trackers on the VOT-2017 dataset. The results are presented in terms of expected average overlap (EAO), accuracy (A), and robustness (R).

| Tracker | EAO | A | R |
|---|---|---|---|
| ECO [64] | **0.280** | 0.483 | **0.276** |
| SiamDCF [12] | 0.249 | 0.500 | 0.473 |
| **BiROLS** | 0.209 | 0.475 | 0.473 |
| MEEM [22] | 0.192 | 0.463 | 0.534 |
| SiamFC [11] | 0.188 | 0.502 | 0.585 |
| Staple [60] | 0.169 | **0.530** | 0.688 |
| KCF [59] | 0.135 | 0.447 | 0.773 |
| MIL [18] | 0.118 | 0.393 | 1.011 |
| Struck [20] | 0.097 | 0.418 | 1.297 |
| DSST [23] | 0.079 | 0.395 | 1.452 |


**FIGURE 6.** Representative tracking results of the BiROLS tracker compared with other 9 trackers on some challenging sequences.

**TABLE 4.** The average overlap (AO) for no-reset experiments on VOT-2017 dataset. Attributes include camera change (Cam.), not assigned(N/A), illumination change (Illu.), motion change (Mot.), occlusion (Occ), and size change (Size).

| Tracker | Cam. | N/A | Illu. | Mot. | Occ. | Size | All |
|---|---|---|---|---|---|---|---|
| ECO | **0.419** | **0.413** | **0.425** | 0.370 | 0.280 | **0.370** | **0.403** |
| SiamDCF | 0.348 | 0.360 | 0.380 | 0.334 | 0.213 | 0.338 | 0.340 |
| **BiROLS** | 0.404 | 0.370 | 0.214 | **0.380** | **0.316** | 0.314 | 0.375 |
| MEEM | 0.337 | 0.357 | 0.319 | 0.311 | 0.243 | 0.269 | 0.327 |
| SiamFC | 0.360 | 0.349 | 0.387 | 0.336 | 0.239 | 0.331 | 0.343 |
| Staple | 0.397 | 0.288 | 0.354 | 0.378 | 0.228 | 0.325 | 0.333 |
| KCF | 0.282 | 0.243 | 0.320 | 0.270 | 0.263 | 0.277 | 0.267 |
| MIL | 0.169 | 0.202 | 0.177 | 0.150 | 0.133 | 0.162 | 0.179 |
| Struck | 0.178 | 0.221 | 0.200 | 0.189 | 0.125 | 0.181 | 0.196 |
| DSST | 0.156 | 0.208 | 0.251 | 0.131 | 0.129 | 0.132 | 0.172 |

tracker ranks second on the overall performance. We notice that the BiROLS tracker performs worse than some trackers under size change attribute due to lack of the scale estimation strategy. But regard to the occlusion, motion change, camera change and not assigned attributes, our method performs better than the most of trackers. Specifically, the BiROLS tracker exhibits obvious advantages for the occlusion attribute, outperforms the ECO tracker by about 3.6%.

Overall, the experimental results on VOT benchmarks are consistent with those on the OTB benchmarks, which further prove the validness of the proposed bidirectional tracking scheme.

### E. QUALITATIVE EVALUATION

To better visualize the tracking performance of the proposed framework, Fig. 6 compares the tracking results of the proposed BiROLS tracker with other 9 state-of-the-art trackers, i.e., FCNT [50], LCT [24], MEEM [22], DSST [23], TGPR [72], Struck [20], SCM [71], TLD [19] and VTD [70],

on 8 representative video sequences in the OTB-2013 benchmark [35].

It can be seen from Fig. 6 that the proposed BiROLS tracker performs well on all challenging sequences. The FCNT tracker and LCT tracker perform well in the most sequences, but they are less robust to fast motion, e.g., FCNT tracker fails in the *Matrix* sequence and LCT tracker fails in the *Skiing* sequence. In addition, the LCT tracker fails to handle fast deformation, e.g., it loses track of the target after frame #14 in the *MotorRolling* sequence. It also can be seen from Fig. 6 that the performances of other trackers (i.e., DSST, TGPR, Struck, SCM, TLD, VTD) are far worse than the proposed BiROLS tracker except for MEEM tracker (those trackers lose track of the target in most of the challenging situations).

For visual tracking, background clutters is the most common challenge. Fig. 6 presents some sampled results in two sequences (i.e., *Matrix* and *Soccer*) in which the target objects undergo background clutters. Most of the trackers gradually lose the target objects in these sequences during tracking, but the BiROLS tracker reliably tracks target objects which performs slightly better than the FCNT tracker. It demonstrates that the proposed feature representation has a good discriminative capability to distinguish the tracked target from the backgrounds.

During the tracking process, rotation and deformation will result in appearance changes. e.g., in *MotorRolling* sequence and in *FleetFace* sequence. In the *MotorRolling* sequence, the mountain bike has the in-plane rotations due to its acrobatic actions in the arena. In the *FleetFace* sequence, the man' face has significant out-of-plane rotations due to the poses of his head changes a lot. In order to deal with these challenges, the observation model of tracker needs to online update to account for appearance change of the target. It can be seen from Fig. 6 that most of the trackers could not adapt to the serious rotation and gradually drift away in the *MotorRolling* sequence. Only the FCNT tracker and BiROLS tracker can track the target well. It can be attributed to the effective and efficient learning algorithm (i.e., ROLS) of BiROLS tracker.

Fig.6 shows the sampled results of *Jogging*2 and *SUV* sequences where the targets undergo heavy occlusions. In the *Jogging*2 sequence, the tracked person is almost fully occluded by the traffic light (i.e., #53, #57). In the *SUV* sequence, the tracked vehicle is frequently occluded by dense tree branches(i.e., #524, #787). It can be seen from Fig.6 that most of the traditional trackers drift away to the distracters, but the BiROLS tracker is able to re-detect the target after occlusion (e.g., after frame #69 in *Jogging*2 sequence). It can be attributed that the BiROLS tracker can determine an appropriate update strategy to avoid degradation of the observation model performance. The multi-expert mechanism of the MEEM tracker also has the ability to handle occlusion, but this mechanism is not stable. It can be seen from the results of *SUV* sequence that the MEEM tracker drifts after the second occlusion (i.e., frame #787).

It can also be seen from Fig.6 that the BiROLS tracker lacks the ability to deal with the scale changes. In the *Matrix* sequence, although the BiROLS tracker can reliably track man's head, it cannot calculate the accurate bounding box of tracked target. In the *MoterRolling* sequence, since only one fixed scale is used, when the target is zoomed out (i.e., frame #62 and #149), a lot of background information included in the bounding box so that the discriminative power of feature representation degrades, so the tracker slightly drifts away.

## VII. CONCLUSION

This paper proposes an efficient framework for visual object tracking. This framework presents concatenation of HOGv and color histogram descriptor for building feature representation and develops an SFNN that can be online trained using ROLS algorithm for observation model. A bidirectional tracking scheme is designed to alleviate the model drift problem during online tracking. The whole framework consists of three modules: the forward tracking module, the backward tracking module and the integration module. The main idea of this framework is slightly similar to like ensemble learning techniques that combine the results that are achieved by weak trackers to produce a strong tracker that is better than either of the weak trackers. Extensive evaluations demonstrate the effectiveness of the proposed bidirectional tracking scheme. Experimental results also show that this proposed framework outperforms most of state-of-the-art methods. Future work focuses on designing an effective and efficient scale estimation strategy to further improve the performance.

## REFERENCES

[1] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. Van Den Hengel, "A survey of appearance models in visual object tracking," *ACM Trans. Intell. Syst. Technol.*, vol. 4, no. 4, 2013, Art. no. 58.

[2] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah, "Visual tracking: An experimental survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1442–1468, Jul. 2014.

[3] J. van de Weijer, C. Schmid, J. J. Verbeek, and D. Larlus, "Learning color names for real-world applications," *IEEE Trans. Image Process.*, vol. 18, no. 7, pp. 1512–1524, Jul. 2009.

[4] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 886–893.

[5] X. Wang, T. X. Han, and S. Yan, "An HOG-LBP human detector with partial occlusion handling," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 32–39.

[6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 580–587.

[8] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, Oct. 2017, pp. 2961–2969.

[9] R. Tao, E. Gavves, and A. W. Smeulders, "Siamese instance search for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1420–1429.

[10] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 5000–5008.

[11] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "Fully-convolutional siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2016, pp. 850–865.

[12] Q. Wang, J. Gao, J. Xing, M. Zhang, and W. Hu, "DCFNet: Discriminant correlation filters network for visual tracking," 2017, *arXiv:1704.04057*. [Online]. Available: https://arxiv.org/abs/1704.04057
[13] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with siamese region proposal network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8971–8980.
[14] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2012, pp. 864–877.
[15] S. Zhang, X. Yu, Y. Sui, S. Zhao, and L. Zhang, "Object tracking with multi-view support vector machines," *IEEE Trans. Multimedia*, vol. 17, no. 3, pp. 265–278, Mar. 2015.
[16] H. Li, Y. Li, and F. Porikli, "Robust online visual tracking with a single convolutional neural network," in *Proc. Asian Conf. Comput. Vis.*, 2014, pp. 194–209.
[17] H. Grabner, C. Leistner, and H. Bischof, "Semi-supervised on-line boosting for robust tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2008, pp. 234–247.
[18] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 983–990.
[19] Z. Kalal, K. Mikolajczyk, and J. Matas, "Tracking-learning-detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1409–1422, Jul. 2012.
[20] S. Hare, A. Saffari, and P. H. S. Torr, "Struck: Structured output tracking with kernels," in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 263–270.
[21] J. S. Supancic, III, and D. Ramanan, "Self-paced learning for long-term tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2379–2386.
[22] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: Robust tracking via multiple experts using entropy minimization," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 188–203.
[23] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, 2014, pp. 1–11.
[24] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5388–5396.
[25] Z. Huang, Y. Yu, J. Gu, and H. Liu, "An efficient method for traffic sign recognition based on extreme learning machine," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 920–933, Apr. 2017.
[26] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.
[27] D. L. Yu, J. B. Gomm, and D. Williams, "A recursive orthogonal least squares algorithm for training RBF networks," *Neural Process. Lett.*, vol. 5, no. 3, pp. 167–176, 1997.
[28] G.-B. Huang, L. Chen, and C.-K. Siew, "Universal approximation using incremental constructive feedforward networks with random hidden nodes," *IEEE Trans. Neural Netw.*, vol. 17, no. 4, pp. 879–892, Jul. 2006.
[29] W. Huang, J. Gu, X. Ma, and Y. Li, "Correlation-filter based scale-adaptive visual tracking with hybrid-scheme sample learning," *IEEE Access*, vol. 6, pp. 125–137, 2017.
[30] W. Huang, J. J. Gu, X. Ma, and Y. Li, "Self-paced model learning for robust visual tracking," *J. Electoron. Imag.*, vol. 26, no. 1, pp. 13–26, 2017.
[31] H. Liu, F. Sun, and Y. Yu, "Multitask extreme learning machine for visual tracking," *Cogn. Comput.*, vol. 6, no. 3, pp. 391–404, 2014.
[32] W. Ou, D. Yuan, Q. Liu, and Y. Cao, "Object tracking based on online representative sample selection via non-negative least square," *Multimedia Tools Appl.*, vol. 77, pp. 10569–10587, May 2018.
[33] Q. Liu, X. Ma, W. Ou, and Q. Zhou, "Visual object tracking with online sample selection via lasso regularization," *Signal, Image Video Process.*, vol. 11, no. 5, pp. 881–888, 2017.
[34] Q. Zhou, B. Zhong, Y. Zhang, J. Li, and Y. Fu, "Deep alignment network based multi-person tracking with occlusion and motion reasoning," *IEEE Trans. Multimedia*, vol. 21, no. 5, pp. 1183–1194, May 2018.
[35] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.
[36] Y. Wu, J. Lim, and M. H. Yang, "Object tracking benchmark," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 9, pp. 1834–1848, Sep. 2015.
[37] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder, "The visual object tracking VOT2015 challenge results," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2015, pp. 564–586.
[38] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. C. Zajc, T. Vojir, G. Hager, A. Lukezic, A. Eldesokey, and G. Fernandez, "The visual object tracking VOT2017 challenge results," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Oct. 2017, pp. 1949–1972.
[39] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *Int. J. Comput. Vis.*, vol. 77, nos. 1–3, pp. 125–141, 2008.
[40] D. Wang, H. Lu, and M.-H. Yang, "Least soft-threshold squares tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2371–2378.
[41] X. Mei and H. Ling, "Robust visual tracking using $\ell_1$ minimization," in *Proc. IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 1436–1443.
[42] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust $\ell_1$ tracker using accelerated proximal gradient approach," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1830–1837.
[43] D. Wang, H. Lu, Z. Xiao, and M.-H. Yang, "Inverse sparse tracker with a locally weighted distance metric," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2646–2657, Sep. 2015.
[44] H. Grabner and H. Bischof, "On-line boosting and vision," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 2006, pp. 260–267.
[45] S. Avidan, "Ensemble tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 2, pp. 261–271, Feb. 2007.
[46] R. T. Collins, Y. Liu, and M. Leordeanu, "Online selection of discriminative tracking features," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, no. 10, pp. 1631–1643, Oct. 2005.
[47] S. Hong, T. You, S. Kwak, and B. Han, "Online tracking by learning discriminative saliency map with convolutional neural network," 2015, *arXiv:1502.06796*. [Online]. Available: https://arxiv.org/abs/1502.06796
[48] H. Li, Y. Li, and F. Porikli, "DeepTrack: Learning discriminative feature representations online for robust visual tracking," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1834–1848, Apr. 2016.
[49] N. Wang and D.-Y. Yeung, "Learning a deep compact image representation for visual tracking," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 809–817.
[50] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 2127–3119.
[51] L. Wang, T. Liu, G. Wang, K. L. Chan, and Q. Yang, "Video tracking using learned hierarchical features," *IEEE Trans. Image Process.*, vol. 24, no. 4, pp. 1424–1435, Apr. 2015.
[52] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4293–4302.
[53] H. Fan and H. Ling, "SANet: Structure-aware network for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jul. 2017, pp. 42–49.
[54] W. Liu, Y. Song, D. Chen, S. He, Y. Yu, T. Yan, G. P. Hancke, and R. W. H. Lau, "Deformable object tracking with gated fusion," *IEEE Trans. Image Process.*, vol. 28, no. 8, pp. 3766–3777, Aug. 2019.
[55] Y. Lin, B. Zhong, G. Li, S. Zhao, Z. Chen, and W. Fan, "Localization-aware meta tracker guided with adversarial features," *IEEE Access*, vol. 7, pp. 99441–99450, 2019.
[56] B. Zhong, B. Bai, J. Li, Y. Zhang, and Y. Fu, "Hierarchical tracking by reinforcement learning-based searching and coarse-to-fine verifying," *IEEE Trans. Image Process.*, vol. 28, no. 5, pp. 2331–2341, May 2019.
[57] P. Li, D. Wang, L. Wang, and H. Lu, "Deep visual tracking: Review and experimental comparison," *Pattern Recognit.*, vol. 166, pp. 71–81, Aug. 2019.
[58] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2544–2550.
[59] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 3, pp. 583–596, Mar. 2015.
[60] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1401–1409.
[61] Q. Liu, X. Lu, Z. He, C. Zhang, and W.-S. Chen, "Deep convolutional neural networks for thermal infrared object tracking," *Knowl.-Based Syst.*, vol. 134, pp. 189–198, Oct. 2017.

[62] M. Zhang, Q. Wang, J. Xing, J. Gao, P. Peng, W. Hu, and S. Maybank, "Visual tracking via spatially aligned correlation filters network," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 469–485.
[63] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4310–4318.
[64] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jul. 2017, pp. 6638–6646.
[65] K. Dai, D. Wang, H. Lu, C. Sun, and J. Li, "Visual tracking via adaptive spatially-regularized correlation filters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 4670–4679.
[66] X. Li, Q. Liu, N. Fan, Z. He, and H. Wang, "Hierarchical spatial-aware siamese network for thermal infrared object tracking," *Knowl.-Based Syst.*, vol. 166, pp. 71–81, Feb. 2019.
[67] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Dec. 2001, pp. 511–518.
[68] N. Wang, Y. Song, C. Ma, W. Zhou, W. Liu, and H. Li, "Unsupervised deep tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2019, pp. 1308–1317.
[69] H. Wu, A. C. Sankaranarayanan, and R. Chellappa, "In situ evaluation of tracking algorithms using time reversed chains," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.
[70] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1269–1276.
[71] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparsity-based collaborative model," in *Proc. IEEE Int. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1838–1845.
[72] J. Gao, H. Ling, W. Hu, and J. Xing, "Transfer learning based visual tracking with Gaussian processes regression," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 188–203.

**ZHIYONG HUANG** received the bachelor's degree in computer science and technology from Fuzhou University, Fuzhou, China, in 2014, where he is currently pursuing the Ph.D. degree. His current research interests include computer vision and machine learning.

**YUANLONG YU** received the Ph.D. degree in electrical engineering from the Memorial University of Newfoundland, St. Johns, NL, Canada, in 2010. Since 2011, he has been a Postdoctoral Fellow with the Memorial University of Newfoundland and Dalhousie University, Halifax, NS, Canada. Since 2013, he has been a Professor with Fuzhou University, Fuzhou, China. His main interests include computer vision, machine learning, visual attention, autonomous mental development, and cognitive robotics.

**MIAOXING XU** received the bachelor's degree in computer science and technology with Fuzhou University, Fuzhou, China, in 2017, where he is currently pursuing the master's degree. His research interests include computer vision and machine learning.