Multi-Cue Correlation Filters for Robust Visual Tracking

Ning Wang[1], Wengang Zhou[1], Qi Tian[2], Richang Hong[3], Meng Wang[3], Houqiang Li[1]
[1]CAS Key Laboratory of GIPAS, University of Science and Technology of China
[2]University of Texas at San Antonio [3]HeFei University of Technology
wn6149@mail.ustc.edu.cn, {zhwg,lihq}@ustc.edu.cn, qi.tian@utsa.edu, {hongrc,wangmeng}@hfut.edu.cn

Abstract

In recent years, many tracking algorithms achieve impressive performance via fusing multiple types of features, however, most of them fail to fully explore the context among the adopted multiple features and the strength of them. In this paper, we propose an efficient multi-cue analysis framework for robust visual tracking. By combining different types of features, our approach constructs multiple experts through Discriminative Correlation Filter (DCF) and each of them tracks the target independently. With the proposed robustness evaluation strategy, the suitable expert is selected for tracking in each frame. Furthermore, the divergence of multiple experts reveals the reliability of the current tracking, which is quantified to update the experts adaptively to keep them from corruption.

Through the proposed multi-cue analysis, our tracker with standard DCF and deep features achieves outstanding results on several challenging benchmarks: OTB-2013, OTB-2015, Temple-Color and VOT 2016. On the other hand, when evaluated with only simple hand-crafted features, our method demonstrates comparable performance amongst complex non-realtime trackers, but exhibits much better efficiency, with a speed of 45 FPS on a CPU.

1. Introduction

Visual tracking is a fundamental task in computer vision with a wide range of applications. It is challenging since only the initial state of the target is available. Although substantial progress has been made in the past decades [52, 16, 35], there still remain many challenges [49].

In recent years, Discriminative Correlation Filter (DCF) based tracking methods [4, 16] have gained much attention thanks to their impressive performance as well as high speed. In DCF based trackers, the filter is trained through minimizing a least-squares loss for all circular shifts of a training sample. Since the correlation operation can be calculated in Fourier domain, DCF shows the advantage of high computational efficiency. Its performance is further enhanced by using multi-dimensional features [16, 26, 31], part-based strategies [29, 28], adaptive scale [9, 26], long-term framework [32, 17, 45], end-to-end learning [41, 40] and improved filter training methods [20, 30, 33, 14]. Specially, the combination of DCF and features from deep convolutional neural networks (CNNs) [23, 39] has demonstrated state-of-the-art results. Recently, Ma *et al.* [31] propose a HCF algorithm, which constructs multiple DCFs on low, middle and high-level features to capture both spatial details and semantics. It predicts the target location using multi-level DCF response maps in a coarse-to-fine fashion.

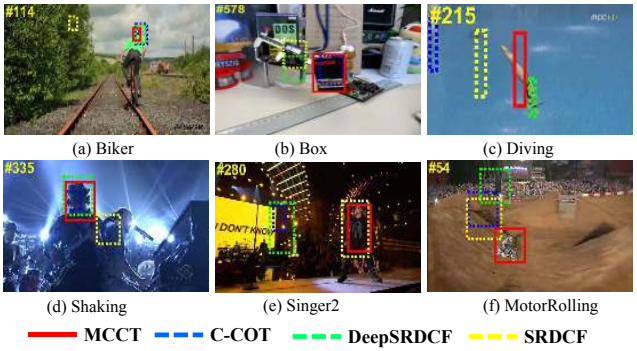

Figure 1. Comparison of the proposed algorithm (MCCT) with the state-of-the-art trackers: SRDCF [12], DeepSRDCF [11] and C-COT [13]. These trackers adopt various types of features and perform differently in challenging scenes. Our method maintains multiple cues for tracking and performs favorably against these trackers. All the videos are from OTB-2015 [48].

Although feature-level fusion methods [26, 31, 38] have been widely used or extended to boost the performance, there still leaves room for improvement. In HCF [31] or other methods [56, 38] that follow such a fusion strategy, the initial weight of high-level features is usually high such that semantic features play the dominant role in general. It is reasonable because of better effectiveness of high-level features compared to shallow features. However, due to the occasional misguidance of the semantic information, a transient drift or wrong prediction may be amplified by the inadequate online update. Therefore, the feature-level fusion

approach sometimes still fails to fully explore the relationship of multiple features. Furthermore, it is quite difficult to handle various challenging variations using a single model, and relying on a certain feature-level fusion strategy limits the model diversity to some extent.

To better illustrate the issues mentioned above, our proposed algorithm is compared with three DCF based methods. In Figure 1, SRDCF [12], DeepSRDCF [11] and C-COT [13] adopt different types of features, but none of them is able to handle various challenging factors, even for the C-COT algorithm with multiple features as well as the novel continuous convolution operators. Since it is quite difficult to design a satisfying feature-level fusion method that suits various challenging scenes, it is intuitive to design an adaptive switch mechanism to achieve better performance, which can flexibly switch to the reliable tracker depending on what kind of challenging factors it is expert at handling. In other words, the performance of a single tracker can sometimes be unstable but the decision-level fusion of the outputs from multiple trackers can enhance the robustness effectively.

In this paper, a novel Multi-Cue Correlation filter based Tracker (MCCT) is proposed. Different from the previous DCF based methods [4, 16] that usually lack the diversity of target appearance representations, our method maintains multiple experts to learn the appearance models from different views. Here, a certain combination of features constructs an individual expert and provides a reliable cue (predicted target state) for tracking. The main contributions of our work can be summarized as follows. (1) We propose an algorithm that maintains multiple cues for tracking. Through checking the robustness scores of multiple experts carefully, our method refines the tracking results by choosing the reliable expert for tracking in each frame. (2) By considering the divergence of multiple experts, we present an adaptive update strategy which can discriminate the unreliable samples (*e.g.,* occlusion or sever deformation) effectively and alleviate the contamination of training samples. (3) We implement two versions of the proposed method to validate the generality of the framework. The MCCT tracker with deep features demonstrates outstanding performance on several challenging benchmarks [47, 48, 27, 21]. The MCCT-H tracker with only two standard Hand-crafted features (HOG [7] and ColorNames [46]) achieves comparable performance with many complex deep-model based trackers, but operates about 45 frames per second on a single CPU, exceeding most competitive trackers by several times.

2. Related Work

In this section, we discuss two categories of trackers closely related to our algorithm: correlation tracking and ensemble tracking.

Correlation Tracking. Since Bolme *et al.* [4] propose a tracker using minimum output sum of squared error (MOSSE) filter, the correlation filters have been widely studied in visual tracking. Heriques *et al.* exploit the circulant structure of the training patches [15] and propose to train correlation filter in a kernel space with HOG features [16]. Zhang *et al.* [54] incorporate context information into filter learning. The SRDCF tracker [12] alleviates the boundary effects by penalizing correlation filter coefficients depending on spatial location, and is enhanced by reducing the influence of corrupted samples [10]. Qi *et al.* [38] propose a HDT algorithm that fuses several DCFs through adaptive hedged method. Luca *et al.* [2] propose a Staple tracker which combines DCF and color histogram based model while running in excess of real-time. The CSR-DCF algorithm [30] constructs DCF with channel and spatial reliability. The recent C-COT [13] adopts a continuous-domain formulation of the DCF, leading to the top performance on several tracking benchmarks. The enhanced version of C-COT is ECO [8], which improves both speed and performance by introducing several efficient strategies.

Different from the DCF based methods mentioned above, our algorithm considers not only feature-level fusion but also decision-level fusion to better explore the relationship of multiple features, and adaptively selects the expert that is suitable for a particular tracking task.

Ensemble Tracking. To enhance the tracking robustness and obtain reliable results, the ensemble approach treats the trackers as black boxes and takes only the bounding boxes returned by them as input [43]. In [1], a dynamic programming based trajectory optimization approach is proposed to build a strong tracker. In [44], Wang *et al.* propose a factorial hidden Markov model for ensemble-based tracking. The MEEM algorithm [51] exploits the relationship between the current tracker and its historical snapshots using entropy minimization, and Li *et al.* [25] extend it by introducing a discrete graph optimization framework. In [19], a partition fusion framework is proposed to cluster reliable trackers for target state prediction.

Although promising results are obtained through fusing multiple trackers, there still leave some limitations: (1) the overall speed is limited by the lowest tracker in the ensemble (*e.g.,* sparse representation based methods [57, 18], about 1 FPS), which restricts the real-time application. Specially, the fusion methods [19, 24] by analyzing the forward and backward trajectories require each tracker to run at least twice; (2) the trackers are just regarded as independent black boxes and their fusion result in each frame does not feedback to the trackers [1, 44, 19], which fails to make full use of the reliable fusion outputs; (3) if the fusion tracker number increases, the dynamic programming based fusion methods [25, 1] still bring obvious computational burden (e.g., $O(TN^2)$ for T frames and N trackers).

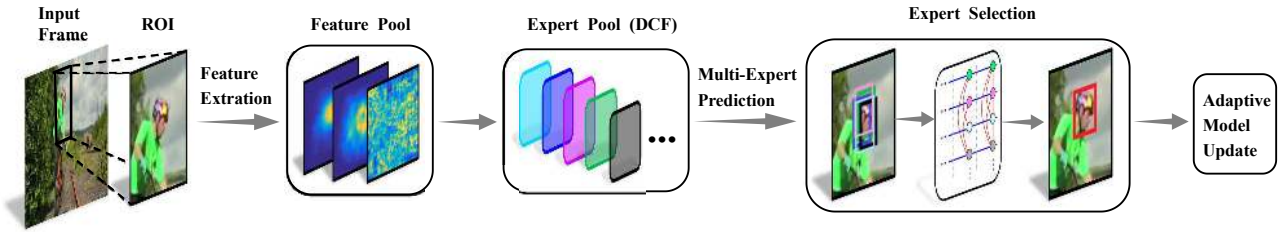

Figure 2. A systematic flowchart of the proposed tracking algorithm. First, different features of ROI are extracted and combined to train multiple experts following DCF framework (Sec. 3.1 and 3.2). Then, each expert gives an individually predicted cue and the most reliable expert is selected for the current tracking (Sec. 3.3). Finally, adaptive update helps keep the experts from corruption (Sec. 3.4).

Different from the above methods: (1) Our approach constructs all the experts in the DCF framework, and through the proposed ROI and training sample sharing strategy (Sec. 3.3), the efficiency is greatly ensured; (2) the refined tracking results are fed back to the experts to boost them further; (3) through a simple but effective robustness evaluation strategy, our method selects the reliable expert for tracking with a complexity of only $O(TN)$ for T frames and N experts.

3. Method

Our algorithm is composed of several experts which provide different levels of cues for tracking. A preview of DCF is introduced in Sec. 3.1. The component of the expert pool is described in Sec. 3.2. How to switch to the suitable expert in each frame is elaborated in Sec. 3.3. Finally, Sec. 3.4 introduces the adaptive update. The framework of our method is depicted in Figure 2.

3.1. Preview of DCF

A typical tracker based on DCF [4, 16] is trained using an image patch $\mathbf{x}$ of size $M \times N$, which is centered around the target. All the circular shifts of the patch $\mathbf{x}(m,n) \in \{0,1,...M-1\} \times \{0,1,...N-1\}$ are generated as training samples with Gaussian function label $y(m,n)$ in terms of the shifted distance. The filter $\mathbf{w}$ is trained by minimizing the following regression error:

$$\min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda\|\mathbf{w}\|_2^2, \tag{1}$$

where λ is a regularization parameter ($\lambda \geq 0$) and $\mathbf{X}$ is the data matrix by concatenating all the circular shifts. The filter solution on the d-th ($d \in \{1, \cdots, D\}$) channel is defined by

$$\hat{\mathbf{w}}_d^* = \frac{\hat{\mathbf{y}} \odot \hat{\mathbf{x}}_d^*}{\sum_{i=1}^{D} \hat{\mathbf{x}}_i^* \odot \hat{\mathbf{x}}_i + \lambda}, \tag{2}$$

where $\odot$ is the element-wise product, the hat symbol denotes the Discrete Fourier Transform (DFT) of a vector (*e.g.*, $\hat{\mathbf{x}} = \mathcal{F}(\mathbf{x})$) and $\hat{\mathbf{x}}^*$ is the complex-conjugate of $\hat{\mathbf{x}}$. In the next frame, a Region of Interest (ROI) is cropped out for tracking the target (*e.g.*, a patch $\mathbf{z}$ with the same size of $\mathbf{x}$). The response map R of $\mathbf{z}$ is calculated in Eq. (3) and the location of the target is identified by searching for the maximum value of R.

$$R = \mathcal{F}^{-1}(\sum_{d=1}^{D} \hat{\mathbf{w}}_d \odot \hat{\mathbf{z}}_d^*). \tag{3}$$

To avoid the boundary effects during learning, we apply Hann window to the signals [16]. Besides, inspired by [30, 5], color information is applied to the training sample in a simple way to enhance its spatial reliability: $\mathbf{X}' = \mathbf{X} \odot \mathbf{C}$, where $\mathbf{X}$ represents the data matrix and $\mathbf{C}$ is the color mask obtained by computing the histogram-based per-pixel score map [37, 2] of ROI. The online update of the numerator $\hat{\mathbf{A}}_d$ and the denominator $\hat{\mathbf{B}}_d$ of the filter $\hat{\mathbf{w}}_d^*$ is as follows,

$$\begin{aligned} \hat{\mathbf{A}}_d^t &= (1-\eta)\hat{\mathbf{A}}_d^{t-1} + \eta\, \hat{\mathbf{y}} \odot \hat{\mathbf{x}}_d^{*t}, \\ \hat{\mathbf{B}}_d^t &= (1-\eta)\hat{\mathbf{B}}_d^{t-1} + \eta \sum_{i=1}^{D} \hat{\mathbf{x}}_i^{*t} \odot \hat{\mathbf{x}}_i^t, \\ \hat{\mathbf{w}}_d^{*t} &= \frac{\hat{\mathbf{A}}_d^t}{\hat{\mathbf{B}}_d^t + \lambda}, \end{aligned} \tag{4}$$

where η is the learning rate and t is the index of the current frame. As for the target scale estimation, we follow the DSST tracker [9].

3.2. Feature Pool and Expert Pool

A variety of features can be adopted in DCF and different features have their own strength. Hand-crafted features are typically used to capture low-level details while deep features are semantic-aware. As discussed in HCF [31], the DCF constructed by each single layer of VGG-19 [39] is not accurate enough, so HCF performs coarse-to-fine search on several DCF response maps from different layers. In our MCCT tracker, HOG [7] is used as low-level features. Then, we remove the fully-connected layers and extract the outputs of the *conv*4-4 and *conv*5-4 convolutional layers of VGG-19 as middle-level and high-level features, respectively. Thus, the feature pool consists of three types of features: $\{Low, Middle, High\}$ and they are optionally combined into $C_3^1 + C_3^2 + C_3^3 = 7$ experts. As for the coarse-to-fine

Table 1. The component of the expert ensemble. Our MCCT and MCCT-H trackers both consist of seven experts. Each expert adopts different features and tracks the target via a different view.

	Tracker MCCT	Tracker MCCT-H
Expert Pool	Feature Type	Feature Type
Expert I	Low (HOG)	HOG_1
Expert II	Middle ($conv$4-4 of VGG-19)	HOG_2
Expert III	High ($conv$5-4 of VGG-19)	ColorNames
Expert IV	Middle, Low	HOG_1, ColorNames
Expert V	High, Low	HOG_2, ColorNames
Expert VI	High, Middle	HOG_1, HOG_2
Expert VII	High, Middle, Low	HOG_1, HOG_2, ColorNames

weighting parameters of different level DCF response maps, we follow the settings in HCF [31]. Although some experts (*e.g.,* Expert I, II and III) with single type of features may be less robust compared to Expert VII, they provide the diversity of tracking results, which is crucial in ensemble-based tracking [43].

Our fast variant, the MCCT-H tracker only utilizes standard hand-crafted features (HOG [7] and ColorNames [46]) to construct experts. Since HOG and ColorNames are both low-level features, we do not conduct coarse-to-fine fusion and just simply concatenate them to construct DCF. ColorNames provides an 11 dimensional color representation and HOG feature is 31-dim. To obtain more experts, we take the average grey value over all pixels in a patch as a 1-dim feature and concatenate it to the HOG feature into a 32-dim vector, which is further evenly decomposed into two 16-dim features, denoted as HOG_1 and HOG_2, respectively. The detailed information of the experts in MCCT and MCCT-H is shown in Table 1.

3.3. Multi-Cue Correlation Tracker

The tracking process of our proposed framework can be illustrated by Figure 3, where the multiple experts track the target in parallel and the nodes denote the generated cues (bounding boxes) of the experts. In each frame, the evaluation between different hypothesis nodes reveals the consistency degree between the experts, which is denoted as pair-wise evaluation. Besides, each expert has its own trajectory continuity and smoothness. Thus, given a hypothesis node, its robustness degree can be evaluated by both pair-evaluation and self-evaluation. After evaluating the overall reliability of each node, the expert with the highest robustness score is selected and its tracking result is taken for the current frame.

In the following, we elaborate the formulation of pair-evaluation, self-evaluation and expert selection strategy.

Expert Pair-Evaluation. Most experts in the ensemble are capable of tracking the target stably, and the cues produced by a good expert should be consistent with the cues from other experts as much as possible.

Let $E_1, \cdots, E_7$ denote Expert I, $\cdots$, Expert VII, respectively. In the t-th frame, the bounding box of Expert i is

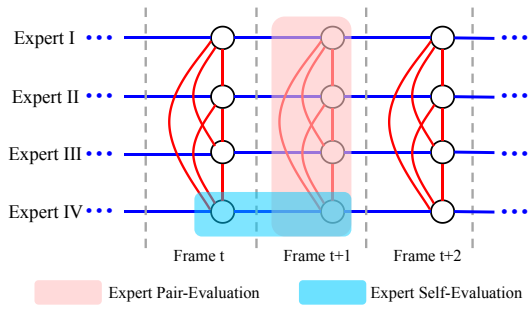

Figure 3. Graph illustration of the multi-expert framework. The node in the graph denotes the predicted bounding box. The robustness score of each expert is evaluated by both pair-evaluation and self-evaluation. For clarity, only four experts are displayed.

denoted as $B^t_{E_i}$. Through regarding all the experts as black boxes, the bounding box $B^t_{E_i}$ only contains the target state (*e.g.*, location and scale) without any context information, which reduces the computational and memory burden effectively.

First, we compute overlap ratios of the bounding boxes from different experts. The overlap ratio $O^t_{(E_i,E_j)}$ of Expert i and Expert j at frame t is calculated as follows,

$$O^t_{(E_i,E_j)} = \frac{Area(B^t_{E_i} \bigcap B^t_{E_j})}{Area(B^t_{E_i} \bigcup B^t_{E_j})}. \tag{5}$$

To reduce the gap between low and high overlap ratios, we adopt a nonlinear gaussian function to $O^t_{(E_i,E_j)}$ as follows, which gathers the expert scores.

$$O'^t_{(E_i,E_j)} = exp\left(-\left(1 - O^t_{(E_i,E_j)}\right)^2\right). \tag{6}$$

The mean value $M^t_{E_i} = \frac{1}{K}\sum_{j=1}^{K} O'^t_{(E_i,E_j)}$ of overlap ratios reveals the trajectory consistency between expert i and others, where K denotes the number of experts (in our experiment, $K = 7$). In general, the pair-wise comparison scores between two experts should be temporal stable. Thus, the fluctuation extent of overlap ratios in a short period Δt (*e.g.,* 5 frames) reveals the stability of overlap evaluation between E_i and other experts, which is given by Eq. (7).

$$V^t_{E_i} = \sqrt{\frac{1}{K}\sum_{j=1}^{K}\left(O'^t_{(E_i,E_j)} - \overline{O'^{t-\Delta t+1:t}_{(E_i,E_j)}}\right)^2}, \tag{7}$$

where $\overline{O'^{t-\Delta t+1:t}_{(E_i,E_j)}} = \frac{1}{\Delta t}\sum_{\tau} O'^{\tau}_{(E_i,E_j)}$ and the time index $\tau \in [t - \Delta t + 1, t]$.

Then, to avoid performance fluctuation of the experts, we further take temporal stability into account and introduce an increasing sequence $\mathbf{W} = \{\rho^0, \rho^1, \cdots, \rho^{\Delta t-1}\}, (\rho > 1)$ to give more weight to the recent scores. After considering the temporal context, the average weighted mean and standard

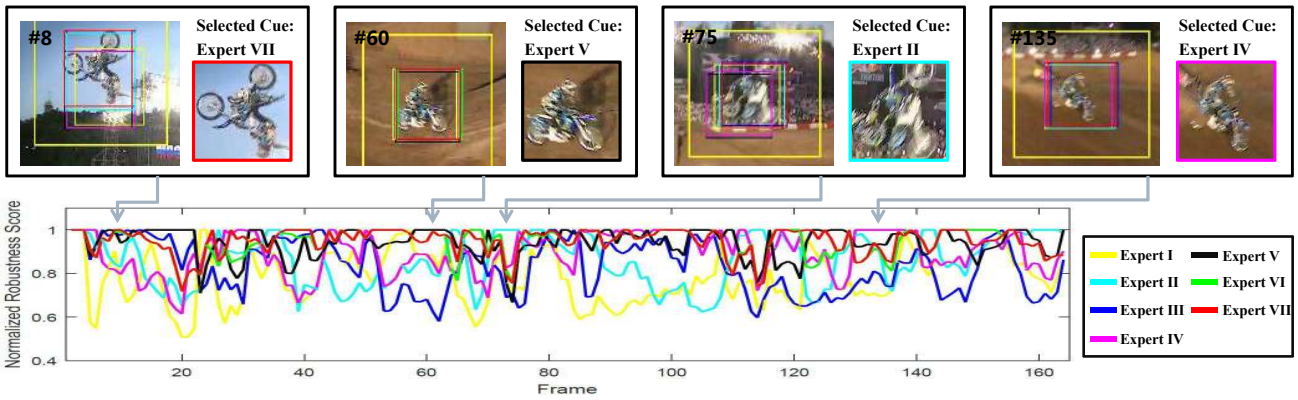

Figure 4. Expert selection process of our MCCT tracker in *MotorRolling* sequence. The bottom plots show the normalized robustness scores of different experts. In top figures, the expert with the highest robustness score is selected for tracking. All the experts share the same ROI (the largest yellow box) and are updated using the same selected results.

variance are calculated through: $M'^{t}_{E_i} = \frac{1}{N}\sum_{\tau} W_\tau M^{\tau}_{E_i}$ and $V'^{t}_{E_i} = \frac{1}{N}\sum_{\tau} W_\tau V^{\tau}_{E_i}$, respectively, where W_τ denotes the $(\tau - t + \Delta t)$-th element in sequence $\mathbf{W}$ and N is the normalization factor defined by $N = \sum_{\tau} W_\tau$.

Finally, the pair-wise expert robustness score of Expert i at frame t is defined as follows,

$$R^t_{pair}(E_i) = \frac{M'^{t}_{E_i}}{V'^{t}_{E_i} + \xi}, \tag{8}$$

where ξ is a small constant that avoids the infinite pair-evaluation score for a zero denominator. A larger $R^t_{pair}(E_i)$ means better consistency with other experts and higher stability of the target state prediction.

Expert Self-Evaluation. The trajectory smoothness degree of each expert indicates the reliability of its tracking results to some extent. The Euclidean distance measuring the shift between the previous bounding box $B^{t-1}_{E_i}$ and the current bounding box $B^t_{E_i}$ is computed by $D^t_{E_i} = \|c(B^{t-1}_{E_i}) - c(B^t_{E_i})\|$, where $c(B^t_{E_i})$ is the center of the bounding box $B^t_{E_i}$. In frame t, the trajectory fluctuation degree of Expert i is given by Eq. (9).

$$S^t_{E_i} = exp\left(-\frac{1}{2\sigma^2_{E_i}}(D^t_{E_i})^2\right), \tag{9}$$

where σ_{E_i} is the average length of the width $W(B^t_{Ei})$ and height $H(B^t_{Ei})$ of the bounding box provided by Expert i, *i.e.*, $\sigma_{E_i} = \frac{1}{2}[W(B^t_{Ei}) + H(B^t_{Ei})]$.

Similar to the pair-evaluation, we collect the previous movement information to consider the temporal stability. Finally, the self-evaluation score is defined by $R^t_{self}(E_i) = \frac{1}{N}\sum_{\tau} W_\tau S^{\tau}_{E_i}$. The higher $R^t_{self}(E_i)$ means the better reliability of the tracking trajectory.

Expert Selection. The final robustness score $R^t(E_i)$ of the Expert i in t-th frame is a linear combination of its pair-evaluation score $R^t_{pair}(E_i)$ and self-evaluation score $R^t_{self}(E_i)$:

$$R^t(E_i) = \mu \cdot R^t_{pair}(E_i) + (1-\mu)\cdot R^t_{self}(E_i), \tag{10}$$

where μ is the parameter to trade off the pair-evaluation and self-evaluation weights. In each frame, as shown in Figure 4, the expert with the highest robustness score is selected for the current tracking.

Sharing Strategy. In the tracking process, all the experts are updated using the same selected samples and share the same searching areas (ROI). This sharing strategy has two advantages: (1) It alleviates the drift and tracking failures of the weak experts effectively by sharing the refined results for target position prediction and model update; (2) It ensures the efficiency of our framework greatly. The main computational burden of DCF lies in the feature extraction process, especially the deep features. Although multiple experts ($K = 7$) are maintained in our method, the sharing strategy makes the feature extraction process only be carried out twice in each frame (instead of $7 \times 2 = 14$ times), one for the searching patch (ROI) and the other for the training patch (used for model update), which is the same as the standard DCF and independent of the expert number.

3.4. Adaptive Expert Update

Due to the training sample sharing strategy used in our framework, the selected tracking results should be carefully checked to avoid the corruption of the experts. The peak-to-sidelobe ratio (PSR) is widely used in DCF to quantify the reliability of the tracked samples [4]. PSR is defined as $P = (R_{max} - m)/\sigma$, where R_{max} is the maximum confidence, m and σ are the mean and standard deviation of the response, respectively. We compute the average PSR of different features: $P^t_{mean} = \frac{1}{3}(P^t_H + P^t_M + P^t_L)$ to evaluate the t-th tracking result, where P^t_H, P^t_M, P^t_L denote the PSR values of the High, Middle and Low-level response maps, respectively. However, in some cases when the unreliable

tracked results have similar features with the target, the PSR value may fail to make a difference.

In our experiment, we observe that when occlusion or severe deformation occurs, the average robustness score of the experts $R^t_{mean} = \frac{1}{K}\sum_{i=1}^{K} R^t(E_i)$ decreases significantly, which can be regarded as the divergence between multiple experts when facing an unreliable sample. Through considering the average PSR score as well as average expert robustness score together, a combined reliability score $S^t = P^t_{mean} \cdot R^t_{mean}$ is proposed, which can discover unreliable samples more effectively and better evaluate the quality of the current tracking.

Since the DCF learns both target and background information, it is not reasonable to simply discard unreliable samples. When the current reliability score S^t is significantly lower than the past average reliability score: $S^{1:t}_{mean} = \frac{1}{t}\sum_{i=1}^{t} S^i$, the learning rate η in Eq. (4) is decreased by Eq. (11).

$$\eta = \begin{cases} C & \text{if } S^t > \alpha \cdot S^{1:t}_{mean}, \\ C \cdot [S^t/(\alpha \cdot S^{1:t}_{mean})]^{\beta} & \text{otherwise,} \end{cases} \quad (11)$$

where C is the learning rate of the standard DCF, α is the reliability threshold, and β is the power exponent of the power function. The designed power function penalizes samples with low reliability scores severely to protect the experts from corruption.

4. Experiments

4.1. Experimental Setup

Implementation Details: In our experiment, we follow the parameters in standard DCF method [16] to construct experts. The parameter ρ in the weigh sequence $\mathbf{W}$ is set to 1.1. The weighting factor μ in Eq. (10) is set to 0.1. As for the adaptive update, α and β in Eq. (11) are set to 0.7 and 3 for MCCT tracker, and 0.6 and 3 for MCCT-H tracker, respectively. More details can be found in the source code[1].

We use the same setting of parameters for all the experiments. Our trackers are implemented in MATLAB 2017a on a computer with an Intel I7-4790K 4.00GHz CPU and 16GB RAM. The MatConvNet toolbox [42] is used for extracting the deep features from VGG-19 [39]. Our MCCT tracker runs at about 1.5 FPS on CPU. The GPU version of MCCT tracker runs at about 8 FPS, which is carried out on a GeForce GTX 1080Ti GPU. The speed of MCCT-H tracker is about 45 FPS on CPU.

Evaluation Benchmarks and Metrics: Our method is evaluated on three benchmark datasets by a no-reset evaluation protocol: OTB-2013 [47], OTB-2015 [48] and TempleColor [27]. All the tracking methods are evaluated by the overlap precision (OP) at an overlap threshold 0.5. For the better performance measure, we also use the average distance precision (DP) plots and overlap success plots over these datasets using one-pass evaluation (OPE) [47, 48].

Finally, we evaluate the performance of the proposed trackers on the VOT2016 [21, 22] benchmark, which consists of 60 challenging sequences and provides an evaluation toolkit which will re-initialize the tracker to the correct position to continue tracking when tracking failure occurs. In VOT2016, the expected average overlap (EAO) is used for ranking trackers, which combines the raw values of per-frame accuracies and failures in a principled manner [21].

[1] https://github.com/594422814/MCCT

Table 2. Effectiveness study of the proposed MCCT-H (top) and MCCT (bottom) trackers. The DP (@20px) and AUC scores are reported on the OTB-2013 [47] and OTB-2015 [48] datasets (DP/AUC) corresponding to the OPE. Expert VII (E_7) is the best individual expert (best baseline) in MCCT and MCCT-H.

	E_7 in MCCT-H Best Baseline	MCCT-H-NU Sec. 3.3	MCCT-H-PSR Sec. 3.3 + PSR	MCCT-H (Final) Sec. 3.3 + 3.4
OTB-2013	78.4 / 60.8	83.5 / 64.4	83.5 / 64.6	**85.6 / 66.4**
OTB-2015	79.2 / 60.5	83.3 / 63.3	82.7 / 63.1	**84.1 / 64.2**
	E_7 in MCCT Best Baseline	MCCT-NU Sec. 3.3	MCCT-PSR Sec. 3.3 + PSR	MCCT (Final) Sec. 3.3 + 3.4
OTB-2013	89.7 / 69.0	91.9 / 70.9	90.9 / 70.5	**92.8 / 71.4**
OTB-2015	87.1 / 66.5	88.3 / 67.6	88.1 / 67.4	**91.4 / 69.5**

4.2. Framework Effectiveness Study

To evaluate the effectiveness of the proposed methods, we compare the MCCT and MCCT-H trackers with their individual experts. By only fusing multiple experts, we obtain the MCCT-NU (MCCT-H-NU) without adaptive update (only Sec. 3.3). The MCCT-PSR (MCCTH-PSR) represents the tracker adopts the fusion method and uses PSR measurement for adaptive update (Sec. 3.3 + PSR). MCCT (MCCT-H) is our final algorithm which combines the expert selection mechanism and the proposed adaptive update strategy (Sec. 3.3 + Sec. 3.4).

From the results in Table 2, we can observe that our final methods outperform their corresponding best baseline Expert VII (the best individual expert) obviously, which illustrates the effectiveness of the proposed framework. On the OTB-2013 and OTB-2015 datasets, our final MCCT-H tracker outperforms its best baseline with a gain of about 6% (from 4.9%-7.2%) in DP and 4% (from 3.7%-5.6%) in AUC. For tracker MCCT, it should be noted that its Expert VII has already achieved a sufficient performance level, but our framework still boosts its performance further (about 4% DP and 3% AUC on challenging OTB-2015). Besides, for the baseline MCCT-NU with high performance, the traditional PSR method for reliability estimation [4] does not have obvious impact, but our adaptive update strategy by considering the divergence of multiple experts improves the performance further.

Table 3. A comparison of our methods using overlap precision (OP) (%) at an overlap threshold 0.5 with recent state-of-the-art trackers on the OTB-2013 [47], OTB-2015 [48] and Temple-Color [27] datasets. The average speed (*i.e.,* frames per second, FPS) is evaluated on the OTB-2013 dataset. The speed labeled with $*$ represents the corresponding tracker utilizes GPU in the experiment. The first and second highest values are highlighted by red and blue.

	DSST	MEEM	SAMF	KCF	LCT	HCF	SRDCF	DeepSRDCDF	SCT	HDT	Staple	SiamFc	SRDCFdecon	MDNet	C-COT	ACFN	CSR-DCF	ADNet	MCPF	ECO	**MCCT-H**	**MCCT**
When	2014	2014	2014	2015	2015	2015	2015	2015	2016	2016	2016	2016	2016	2016	2016	2017	2017	2017	2017	2017		
Where	BMVC	ECCV	ECCVW	TPAMI	CVPR	ICCV	ICCV	ICCVW	CVPR	CVPR	CVPR	ECCV	CVPR	CVPR	ECCV	CVPR	CVPR	CVPR	CVPR	CVPR		
OTB-2013	66.8	70.3	71.4	62.6	81.4	72.9	78.1	77.8	74.2	72.0	76.0	77.8	79.8	**89.6**	82.0	71.8	74.0	80.6	84.6	87.0	81.8	**90.7**
OTB-2015	61.6	62.5	66.4	55.3	70.1	65.4	72.6	76.4	63.0	65.0	70.8	73.0	75.8	82.9	81.5	68.1	68.3	77.2	77.5	**84.1**	78.9	**86.4**
Temple-Color	47.0	60.6	56.8	45.9	55.3	56.9	61.2	65.2	55.7	56.8	62.5	63.5	65.6	-	70.2	-	57.7	-	66.9	**73.5**	67.4	**74.4**
Speed (FPS)	26.3	19.2	18.3	**243.7**	26.1	10.8*	5.6	<1	41.5	10.5*	67.2	**86.0***	2.9	1.0*	0.3	15.0*	14.2	2.9*	0.6*	15.0*	44.8	7.8*

4.3. State-of-the-art Comparison

We evaluate the proposed approach with 20 recent state-of-the-art trackers including DSST [9], MEEM [51], SAMF [26], KCF [16], LCT [32], HCF [31], SRDCF [12], DeepSRDCF [11], SCT [5], HDT [38], Staple [2], SiamFc [3], SRDCFdecon [10], MDNet [35], C-COT [13], ACFN [6], CSR-DCF [30], ADNet [50], MCPF [56], ECO [8]. A comparison with these state-of-the-art trackers using OP metric on OTB-2013, OTB-2015 and Temple-Color datasets is shown in Table 3.

Evaluation on OTB-2013. On the OTB-2013 [47] benchmark, our proposed MCCT method achieves the best OP of 90.7% (Table 3) and the highest area-under-curve (AUC) score of 71.4% (Figure 5). The MDNet method also exhibits excellent results and performs slightly better than ours on DP metric (Figure 5), which is mainly due to the effectiveness of the multi-domain network trained using various similar tracking videos. Compared with other DCF based methods, the proposed MCCT method outperforms the recent ECO, C-COT and MCPF on various metrics, including on metrics OP, DP and AUC.

Evaluation on OTB-2015. On the OTB-2015 [48] dataset, our proposed MCCT tracker achieves the best results on OP and DP with scores of 86.4% and 91.4% (Table 3 and Figure 6), outperforming the second best method by 2.3% and 1.3%. Among all the trackers, only ECO slightly outperforms our method on AUC score. Our MCCT tracker performs much better than HCF, HDT and MCPF, which are also based on correlation filters with multiple types of features.

The proposed MCCT-H method which only incorporates simple hand-crafted features achieves impressive performance and operates at about 45 FPS on a single CPU (Table 3). It significantly outperforms the DCF based trackers with the same features (*e.g.,* SAMF, SRDCF and CSR-DCF) in both performance and speed. Furthermore, MCCT-H achieves comparable results with the recent complex trackers using deep features (*e.g.,* ADNet, MCPF and DeepSRDCF) but runs much faster than them. Compared with other frame-wise adaption trackers (*e.g.,* STC [5] and ACFN [6]) that use attention mechanism for tracker construction, our framework maintains several experts in parallel for decision-level selection and achieves better performance (Table 3, Figure 5 and 6).

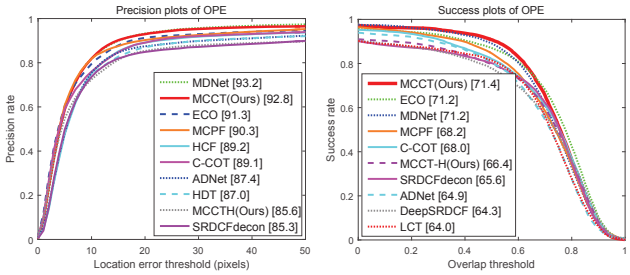

Figure 5. Precision and success plots on the OTB-2013 [47] dataset with 50 videos. Only the top 10 trackers are displayed for clarity. In the legend, the DP at a threshold of 20 pixels and area-under-curve (AUC) are reported in the left and right figures, respectively.

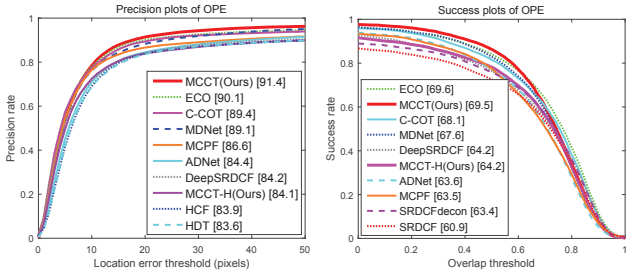

Figure 6. Precision and success plots on the OTB-2015 [48] dataset with 100 videos. In the legend, the DP (@20px) and AUC scores are reported in the left and right figures, respectively.

Evaluation on Temple-Color. For further evaluation, we compare the proposed MCCT-H and MCCT methods on the Temple-Color dataset [27] with the trackers mentioned in Sec. 4.3 excluding MDNet, ADNet and ACFN, which all need many external tracking videos for network training. In contrast, our approach is free of such necessity.

In Table 3 and Figure 7, the proposed MCCT-H tracker still achieves outstanding performance amongst real-time trackers and outperforms some complex non-realtime trackers. Our MCCT method achieves the OP, DP and AUC scores of (74.4%, 79.7%, 59.6%), while ECO and C-COT yield (73.5%, 79.7%, 60.7%) and (70.2%, 78.1%, 58.3%), respectively. Overall, our MCCT tracker shows comparable results compared to the recent performance leader ECO and outperforms other state-of-the-art methods (*e.g.,* C-COT) on various metrics.

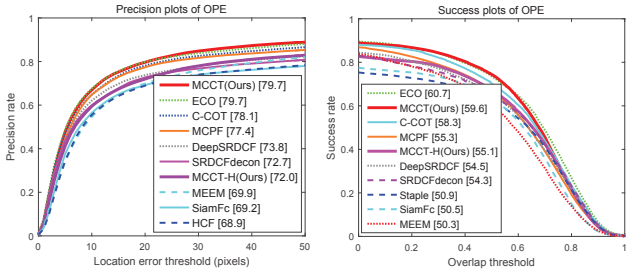

Figure 7. Precision and success plots on the Temple-Color [27] dataset with 128 color videos. In the legend, the DP (@20px) and AUC scores are reported in the left and right figures, respectively.

Evaluation on VOT2016. Figure 8 shows the ranking results in terms of expected average overlap (EAO) in VOT2016 [21], from which we can observe that our MCCT tracker outperforms the top performer (C-COT [13]) by a considerable margin. For presentation clarity, we only show some top ranked and baseline trackers for comparison. For more information, please refer to [21].

In Table 4, we list the detailed results of our approaches and the top ranked methods in VOT2016 (*e.g.,* C-COT [13], TCNN [34] and SSAT). Besides, two recently proposed methods: CSR-DCF [30] and ECO [8] are also put into comparison which do not participate in VOT2016. Among all the compared methods, the proposed MCCT tracker demonstrates advances in both accuracy and robustness. As a result, our MCCT tracker provides the best EAO score of 0.393, achieving a relative gain of 18.7% compared to the VOT2016 top performer C-COT.

Discussion: (1) *About Performance.* It should be noted that the ECO, C-COT and DeepSRDCF methods all take SRDCF as baseline, which can alleviate boundary effects effectively and performs much better than the standard DCF [4, 16]. Besides, ECO and C-COT adopt the novel continuous operator to better fuse the feature maps. However, all the experts in our MCCT-H and MCCT methods just take the simple DCF as baseline and all the strategies mentioned above and other novel techniques [14, 33] can also be integrated into our framework to further boost the performance. (2) *About Efficiency*. Our tracker achieves almost the same speed with expert VII because the same number of DCFs and features are utilized (*i.e.,* sharing strategy), but achieves much better performance than it. Our general framework provides a promising alternative for the multi-feature based DCF trackers with ignorable impact on efficiency. Furthermore, we believe that other tracking algorithms [53, 36, 55] with multiple types of features can also benefit from our multi-cue analysis framework.

5. Conclusion

In this paper, we propose a multi-cue analysis framework for robust visual tracking, which considers not only feature-level fusion but also decision-level fusion to fully explore the strength of multiple features. Our framework maintains multiple experts to track the target via different views and selects the reliable outputs to refine the tracking results. Moreover, the proposed method evaluates the unreliable samples through considering the divergence of multiple experts and updates them adaptively. Through extensive experiments on several challenging datasets, we show that after adopting our simple yet effective multi-cue analysis framework, without sophisticated models, only standard Discriminative Correlation Filter (DCF) with deep or handcrafted features is able to perform favorably against state-of-the-art methods in both accuracy and efficiency.

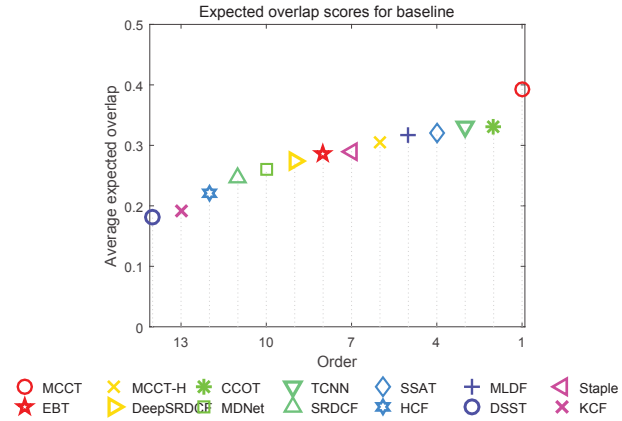

Figure 8. Expected Average Overlap (EAO) graph with trackers ranked from right to left evaluated on VOT2016 [21]. Our proposed MCCT tracker outperforms the top performer (C-COT [13]) by a considerable margin.

Table 4. The accuracy, robustness (failure rate) and EAO of state-of-the-art methods on the VOT2016 [21]. The proposed MCCT tracker achieves superior results compared to the top ranked methods in the challenge and recently proposed state-of-the-art algorithms (ECO [8] and CSR-DCF [30]). The first and second highest values are highlighted by red and blue.

	MCCT-H	MLDF	SSAT	TCNN	C-COT	CSR-DCF	ECO	MCCT
Accuracy	**0.57**	0.48	**0.57**	0.54	0.52	0.51	0.54	**0.58**
Failure Rate	1.24	0.83	1.04	0.96	0.85	0.85	**0.72**	**0.73**
EAO	0.305	0.311	0.321	0.325	0.331	0.338	**0.374**	**0.393**

Acknowledgement. This work was supported in part to Dr. Houqiang Li by 973 Pro-gram under contract No. 2015CB351803 and NSFC under contract No. 61390514, in part to Dr. Wengang Zhou by NSFC under contract No. 61472378 and No. 61632019, the Fundamental Research Funds for the Central Universities, and Young Elite Scientists Sponsorship Program By CAST (2016QNRC001), and in part to Dr. Qi Tian by ARO grant W911NF-15-1-0290 and Faculty Research Gift Awards by NEC Laboratories of America and Blippar. This work was supported in part by National Science Foundation of China (NSFC) 61429201.

References

[1] C. Bailer, A. Pagani, and D. Stricker. A superior tracking approach: Building a strong tracker through fusion. In *ECCV*, 2014. 2

[2] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. Torr. Staple: Complementary learners for real-time tracking. In *CVPR*, 2016. 2, 3, 7

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *ECCV*, 2016. 7

[4] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 1, 2, 3, 5, 6, 8

[5] J. Choi, H. Jin Chang, J. Jeong, Y. Demiris, and J. Young Choi. Visual tracking using attention-modulated disintegration and integration. In *CVPR*, 2016. 3, 7

[6] J. Choi, H. Jin Chang, S. Yun, T. Fischer, Y. Demiris, and J. Young Choi. Attentional correlation filter network for adaptive visual tracking. In *CVPR*, 2017. 7

[7] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2, 3, 4

[8] M. Danelljan, G. Bhat, F. Shahbaz Khan, and M. Felsberg. Eco: Efficient convolution operators for tracking. In *CVPR*, 2017. 2, 7, 8

[9] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 1, 3, 7

[10] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. In *CVPR*, 2016. 2, 7

[11] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCV Workshop*, 2015. 1, 2, 7

[12] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, 2015. 1, 2, 7

[13] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, 2016. 1, 2, 7, 8

[14] H. K. Galoogahi, A. Fagg, and S. Lucey. Learning background-aware correlation filters for visual tracking. In *ICCV*, 2017. 1, 8

[15] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012. 2

[16] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 37(3):583–596, 2015. 1, 2, 3, 6, 7, 8

[17] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): A cognitive psychology inspired approach to object tracking. In *CVPR*, 2015. 1

[18] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012. 2

[19] O. Khalid, J. C. SanMiguel, and A. Cavallaro. Multi-tracker partition fusion. *TCSVT*, 27(7):1527–1539, 2017. 2

[20] H. Kiani Galoogahi, T. Sim, and S. Lucey. Correlation filters with limited boundaries. In *CVPR*, 2015. 1

[21] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernández, T. Vojir, Hager, and et al. The visual object tracking vot2016 challenge results. In *ECCV Workshop*, 2016. 2, 6, 8

[22] M. Kristan, J. Matas, A. Leonardis, T. Vojíř, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin. A novel performance evaluation methodology for single-target trackers. *TPAMI*, 38(11):2137–2155, 2016. 6

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[24] D. Y. Lee, J. Y. Sim, and C. S. Kim. Multihypothesis trajectory analysis for robust visual tracking. In *CVPR*, 2015. 2

[25] J. Li, C. Deng, R. Y. D. Xu, D. Tao, and B. Zhao. Robust object tracking with discrete graph based multiple experts. *TIP*, 26(6):2736–2750, 2017. 2

[26] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *ECCV Workshop*, 2014. 1, 7

[27] P. Liang, E. Blasch, and H. Ling. Encoding color information for visual tracking: algorithms and benchmark. *TIP*, 24(12):5630–5644, 2015. 2, 6, 7, 8

[28] S. Liu, T. Zhang, X. Cao, and C. Xu. Structural correlation filter for robust visual tracking. In *CVPR*, 2016. 1

[29] T. Liu, G. Wang, and Q. Yang. Real-time part-based visual tracking via adaptive correlation filters. In *CVPR*, 2015. 1

[30] A. Lukezic, T. Vojir, L. Cehovin Zajc, J. Matas, and M. Kristan. Discriminative correlation filter with channel and spatial reliability. In *CVPR*, 2017. 1, 2, 3, 7, 8

[31] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015. 1, 3, 4, 7

[32] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *CVPR*, 2015. 1, 7

[33] M. Mueller, N. Smith, and B. Ghanem. Context-aware correlation filter tracking. In *CVPR*, 2017. 1, 8

[34] H. Nam, M. Baek, and B. Han. Modeling and propagating cnns in a tree structure for visual tracking. *arXiv preprint arXiv:1608.07242*, 2016. 8

[35] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, 2016. 1, 7

[36] J. Ning, J. Yang, S. Jiang, L. Zhang, and M.-H. Yang. Object tracking via dual linear structured svm and explicit feature map. In *CVPR*, 2016. 8

[37] H. Possegger, T. Mauthner, and H. Bischof. In defense of color-based model-free tracking. In *CVPR*, 2015. 3

[38] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, and J. L. M.-H. Yang. Hedged deep tracking. In *CVPR*, 2016. 1, 2, 7

[39] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 1, 3, 6

[40] Y. Song, C. Ma, L. Gong, J. Zhang, R. Lau, and M.-H. Yang. Crest: Convolutional residual learning for visual tracking. In *ICCV*, 2017. 1

[41] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. Torr. End-to-end representation learning for correlation filter based tracking. In *CVPR*, 2017. 1

[42] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *ACM MM*, 2014. 6

[43] N. Wang, J. Shi, D. Y. Yeung, and J. Jia. Understanding and diagnosing visual tracking systems. In *ICCV*, 2015. 2, 4

[44] N. Wang and D. Y. Yeung. Ensemble-based tracking: Aggregating crowdsourced structured time series data. In *ICML*, 2014. 2

[45] N. Wang, W. Zhou, and H. Li. Reliable re-detection for long-term tracking. *TCSVT*, 2018. 1

[46] J. V. D. Weijer, C. Schmid, J. Verbeek, and D. Larlus. Learning color names for real-world applications. *TIP*, 18(7):1512–1523, 2009. 2, 4

[47] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 2, 6, 7

[48] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *TPAMI*, 37(9):1834–1848, 2015. 1, 2, 6, 7

[49] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4):13, 2006. 1

[50] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *CVPR*, 2017. 7

[51] J. Zhang, S. Ma, and S. Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In *ECCV*, 2014. 2, 7

[52] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *ECCV*, 2012. 1

[53] K. Zhang, L. Zhang, and M.-H. Yang. Real-time object tracking via online discriminative feature selection. *TIP*, 22(12):4664–4677, 2013. 8

[54] K. Zhang, L. Zhang, M.-H. Yang, and D. Zhang. Fast visual tracking via dense spatio-temporal context learning. In *ECCV*, 2013. 2

[55] T. Zhang, A. Bibi, and B. Ghanem. In defense of sparse tracking: Circulant sparse tracker. In *CVPR*, 2016. 8

[56] T. Zhang, C. Xu, and M.-H. Yang. Multi-task correlation particle filter for robust object tracking. In *CVPR*, 2017. 1, 7

[57] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012. 2