IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received May 16, 2019, accepted June 2, 2019, date of publication June 7, 2019, date of current version July 15, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2921562

Trajectory Smoothing Constraint and Hard Negative Mining for Distractor-Aware Regression Tracking

WEICHUN LIU[1,2], XIAOAN TANG[1], AND XIAOYUAN REN[1]
[1]College of Electronic Science, National University of Defense Technology, Changsha, China
[2]College of Information Engineering, Shaoyang University, Shaoyang, China

Corresponding author: Weichun Liu (liuweichun17@nudt.edu.cn)

ABSTRACT Recently, convolutional regression networks have drawn great attention in the tracking community. Convolutional regression trackers formulate the regression network as one convolutional layer and take advantages of end-to-end learning. However, existing convolutional regression trackers regress the input feature to Gaussian-like soft labels, which still assign a large label to semantic backgrounds in online model fine-tuning. As a result, in the presence of background distractors, convolutional regression trackers tend to drift toward regions, which exhibit a similar appearance compared to the object of interest. In this paper, we propose to achieve distractor-aware regression tracking with trajectory smoothing constraint and hard negative mining. The trajectory smoothing constraint measures motion distance and motion direction between the adjacent frames to discard distractors that fail to meet trajectory smoothness. On this basis, distractors are fed into convolutional regression networks as hard negative samples, which boosts the discriminability of convolutional regression trackers against background distractors. The experimental results on the OTB100 and VOT2016 benchmarks show that the proposed algorithm performs favorably in terms of both accuracy and robustness when compared with state-of-the-art trackers.

INDEX TERMS Trajectory smoothing constraint, hard negative mining, distractor-aware, convolutional regression trackers.

I. INTRODUCTION

Visual tracking is a fundamental task for a wide range of high-level visual understanding problems such as motion analysis, event detection and activity recognition. Generally speaking, the task of visual tracking is to estimate the trajectory of a target in an image sequence, given only its initial location (a rectangular bounding box) in the first frame. Despite significant progress made in recent years, accurate and robust tracking is still challenging in complicated scenarios due to illumination variation, background clutter, partial/full occlusion and background distractors. In this paper, we only focus on single-camera, single-target, short-term and model-free tracking. The interested readers are referred to [1] and [2] for a thorough review of existing tracking algorithms.

Most tracking-by-detection trackers can be categorized into generative trackers [6]–[8] and discriminative trackers [9], [10]. Generative trackers model target appearance and ignore the background information, which leads to tracking drift in complex scenes. On contrast, discriminative trackers treat the tracking task as a binary classification objective that discriminates the object from its surrounding background. Generally, discriminative trackers achieve superior accuracy and robustness on numerous tracking benchmarks [11]–[14]. Discriminative trackers can be further classified into two-stage trackers and one-stage trackers. Two-stage trackers first draw a large number of samples around target objects in the previous frame (first stage) and then classify each sample as the target object or as the background (second stage). Numerous two-stage trackers have been introduced in the tracking community, such as structure support vector machine [9] and online multiple-instance learning [10]. Different from two-stage trackers, one-stage trackers directly learn a mapping from a regularly dense sampling of target objects to soft labels generated by a Gaussian function to estimate target positions. One-stage trackers have recently received increasing attention due to their potential to be much faster and simpler than two-stage trackers. One representative category of one-stage trackers are based on discriminative correlation filters which

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

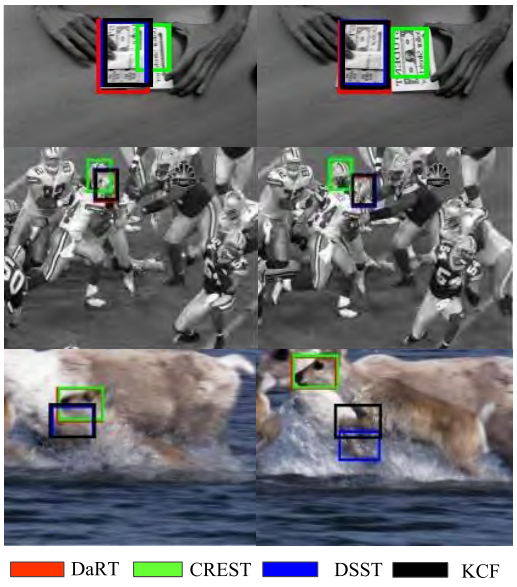

FIGURE 1. Tracking snapshots of our tracker (DaRT), CREST [3], DSST [4] and KCF [5] in presence of background distractors on the *Coupon*, *football1* and *deer* sequences from OTB100.

regress all the circularly shifted versions of input image into soft labels. Different variants of correlation filters have been proposed to boost tracking performance using multi-dimensional features [15], robust scale estimation [16], non-linear kernels [5], long-term memory components [17], target response adaptation [18] and complementary cues [19]. Despite increasing performance on benchmarks, discriminative correlation filters take few advantages of end-to-end training in the deep learning era. The other representative category of one-stage trackers are based on convolutional regression networks. The recent FCNT [20], CRT [21] and CREST [3] trackers belong to this category. Among them, convolutional regression trackers, such as CRT and CREST, reformulate correlation filters as a convolutional layer which is fully differentiable and can be trained end to end.

Due to the straight-forward end-to-end training scheme, convolutional regression trackers achieve balanced accuracy and robustness during online tracking. However, in presence of background distractors, convolutional regression trackers tend to drift towards regions which exhibit a similar appearance compared to the object of interest (see Fig.1). This is because background distractors are still regressed to nonzero labels in the expected Gaussian-like output map during online model finetuning (see Fig.2). Therefore, convolutional regression networks have a big response to semantic backgrounds, which leads to severe tracking drift in real tracking scenarios with background clutter and distractors.

To tackle this problem, in this paper, we propose a Distractor-aware Regression Tracker (DaRT) for accurate and robust tracking. During online tracking, our DaRT tracker tackles background distractors from two perspectives: trajectory smoothing constraint and hard negative minining. A peak to sidelobe ratio (PSR) value is proposed to detect potential background distractors. In presence of background

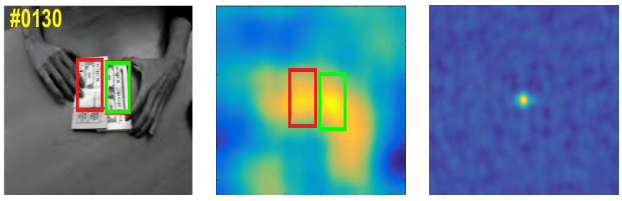

FIGURE 2. (a) Ground truth (red) and background distractor (green); (b) real regression map; (c) expected output.

distractors, there would be several peaks on the regression map. The real peak corresponding to the target of interest will be picked out according to the trajectory smoothing constraint. The background distractors corresponding to the other peaks are treated as hard negative samples and fed into the convolutional regression network for online finetuning. The main contributions of our work are summarized below:

1. A peak to sidelobe ratio (PSR) value is proposed to detect potential background distractors.
2. Trajectory smoothing constraint is applied to convolutional regression tracking to discard background distractors with temporal smoothness of the trajectory.
3. Different from traditional convolutional regression trackers which only use the positive sample for model finetuning, our method exploits background distractors for hard negative mining.
4. Extensive experiments performed on the tracking benchmarks demonstrate the superiority of the proposed algorithm against many other state-of-the-art trackers in comparison.

II. RELATED WORKS

There have been many advances in the object tracking literature in recent years. Due to the space limitation, here we focus on those that are most relevant to our work.

A. CORRELATION FILTER BASED TRACKING

Compared with traditional tracking-by-detection methods [6]–[10], discriminative correlation filters draw much attraction in the tracking community due to dense training samples and high computational efficiency. The pioneer MOSSE tracker [22] achieves an impressive tracking speed of over 600 fps. Later, based on the standard DCF formulation, different variants of correlation filters have been proposed to boost tracking performance using multi-dimensional features [15], robust scale estimation [16], non-linear kernels [5], long-term memory components [17], complementary cues [19], target adaptation [18] and spatial regularization [23]. To achieve further performance gain, correlation filter based trackers extract convolutional features from convolutional neural networks to enhance target representation. DeepSRDCF [24] and CCOT [25] replace handcraft features with convolutional features and achieve top rank on the VOT challenge [26], [27] respectively. Despite top performance on tracking challenges, discriminative correlation filters hardly benefit from end-to-end training.

B. CONVOLUTIONAL REGRESSION TRACKING

Recently, the tracking community leads a fashion of end-to-end training for visual tracking. Following this trend, convolutional regression trackers incorporate feature extraction and correlation filter learning into a unified convolutional neural network. The overall network architecture consists of a feature extraction network and a one-channel-output convolution layer. Different from traditional correlation filter-based trackers, deep-regression trackers try to obtain an approximate solution via gradient descent in spatial domain. Chen and Tao [21] first introduced a single-layer regression model for visual tracking. This model directly regresses the input feature to Gauss-like soft labels. Wang *et al.* [20] introduced a fully convolutional network to exploit multiple CNN features by leveraging a feature-map selection strategy. Both a top layer and a lower layer were jointly used with a switch mechanism during tracking. Song *et al.* [3] applied residual learning to take appearance changes into account on a single convolutional layer and formulated the tracking process in an end-to-end manner by integrating feature extraction, response-map generation and model updated into the neural networks. Despite remarkable tracking performance on benchmarks, existing convolutional regression trackers treat semantic backgrounds and non-semantic backgrounds equally in online model fine-tuning. Therefore, these trackers tend to drift towards background distractors which exhibit similar appearance as the foreground target.

C. DISTRACTOR-AWARE TRACKING

Background distractors are a big challenge for long-term robust tracking. Numerous trackers are proposed to achieve robustness against background distractors. Possegger *et al.* [28] proposed a color model based distractor-aware tracker which relies on standard color histograms for target representation. This tracker identifies and suppresses distracting regions in advance which significantly improves the tracking robustness. To further improve the robustness against background distractors, several trackers [29], [30] incorporated contextual information. Such approaches distinguish between context provided by supporting and distracting regions. Supporting regions have different appearance than the target but occur with it, providing valuable cues to overcome occlusions. Distractors, on the other hand, exhibit similar appearance and may therefore be confused with the target. However, the above context-aware trackers assume that distractors are of the same object class (e.g. pedestrians and cars) and need to track these distractors in addition to the target to prevent drifting, which results in a slow tracking speed. Recently, Mueller *et al.* [31] proposed a context-aware correlation filter based tracker which incorporates the global context into correlation filter learning. This context-aware correlation filter based tracker achieves strong robustness against background clutter and distractors while running at a real-time speed. Kuai *et al.* [32] and Li *et al.* [33] incorporate correlation filters with the fully-convolutional Siamese network to achieve distractor-aware tracking. The Siamese network is used to detect the object within a large search area and potential candidate distractors are screened out on the response map. Correlation filters are employed to identify the target location from these candidates.

III. OUR APPROACH

In this section, we introduce the basic building blocks for our Distractor-aware Regression Tracking (DaRT). It's worth to mention that DaRT is a very flexible tracking framework and our implementation is far from optimal. We believe there is still room for future improvement and generalization. In the following discussion, we will give a brief overview of three building blocks incorporated in our tracker.

A. DEEP REGRESSION TRACKING

Deep regression trackers regress a dense sampling of inputs to Gaussian-like soft labels. Here, we formulate the linear-ridge regression model as one convolutional layer. Given an image patch centered at the target, we can extract convolutional features X from a convolutional neural network and generate corresponding Gaussian-like labels Y ranging from 0 to 1. The kernel weights W of the convolutional layer for the regression function $Y = X * W$ are estimated by solving the following minimization problem:

$$\min_{W} \|W * X - Y\|^2 + \lambda \|W\|^2, \tag{1}$$

where $*$ denotes the convolution operation and λ is a regularization parameter that controls model overfitting. It's worth noting that there is no bias term as we set the bias parameter to 0.

Different from discriminative correlation filters which has a closed-form solution in the Fourier domain, deep regression trackers solve the regression problem in the spatial domain by reformulating the problem as the loss minimization of the convolutional neural network. The receptive-field size of the convolutional kernel in the regression layer is set to cover the target area. The convolutional weights in the convolutional kernel can be effectively calculated by iteratively optimizing W with the Statistical Gradient Descent (SGD) method. Once we have the convolutional kernel trained, target localization is simply finding the maxima on the response map.

B. TRAJECTORY SMOOTHING CONSTRAINT

In this subsection, we indicate the presence of background distractor by the temporal smoothness of the target trajectory. During tracking, we take the last n states before the current frame t to estimate the objects's velocity along the x and y axis. We calculate the object velocity along the x and y axis as follows:

$$\begin{cases} v_x^i = \|x^i - x^{i-1}\|, \\ v_y^i = \|y^i - y^{i-1}\|, \end{cases} \tag{2}$$

where $i \in [t-n-1, t-n, ..., t]$, x^i and y^i are the axis coordinates of the target along the x and y axis in frame i,

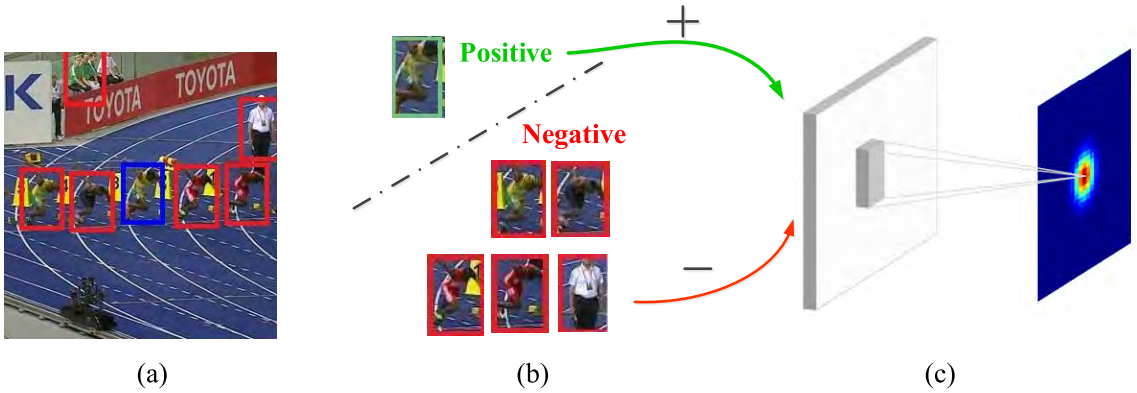

FIGURE 3. (a) Search area filled with the foreground target and background distractors; (b) positive samples and hard negative sample; (c) convolutional regression network.

v_x^i and v_y^i are the velocity of the target along the x and y axis in frame i. As the target motion tendency does not vary significantly at least in a short time period, we formulate the motion trend of the velocity of the target along the x and y axis via a Gaussian distribution which has mean value μ_t^l and standard deviation σ_t^l as following:

$$\begin{cases} \mu_x^t = \dfrac{1}{n-1}\sum_{i=t-n}^{t} v_x^i, \\ \sigma_x^t = \sqrt{\dfrac{1}{n-1}\sum_{i=t-n}^{t}(v_x^i - \mu_x^t)}, \end{cases} \tag{3}$$

$$\begin{cases} \mu_y^t = \dfrac{1}{n-1}\sum_{i=t-n}^{t} v_y^i, \\ \sigma_y^t = \sqrt{\dfrac{1}{n-1}\sum_{i=t-n}^{t}(v_y^i - \mu_y^t)}, \end{cases} \tag{4}$$

We measure the stability of the objects's velocity along the x and y axis at frame t as following:

$$\begin{cases} S_x^t = \dfrac{|v_x^t - \mu_x^t|}{\sigma_x^t}, \\ S_y^t = \dfrac{|v_y^t - \mu_y^t|}{\sigma_y^t}, \end{cases} \tag{5}$$

A smaller value of S_x^t or S_y^t means that the current tracking result tends to be acceptable and instead of a higher S_x^t or S_y^t shows that the tracking result of frame t deviates much from its history trajectories. Based on this principle, we calculate the temporal smoothness of the tracking result in frame t as follows:

$$S^t = S_x^t \cdot S_y^t. \tag{6}$$

A higher value of S^t means higher temporal smoothness of tracking result in frame t.

C. HARD NEGATIVE MINING

Note that the objective function defined in Equ.6 only involves features extracted from a positive training sample centered at the target of interest. During online tracking, a number of background distractors will emerge in the target search area. Generally, these background distractors will be ignored or discarded along with the non-semantic backgrounds. To achieve strong robustness against background distractors, we argue that it's necessary for the convolutional regression network to learn from these hard negative samples which exhibit similar appearance with the target and blur intra-class difference (see Fig.3).

During online tracking, the convolutional regression network is finetuned with both positive and hard negative training samples as following:

$$\min_W \|\sum_{i=0}^{n}(X_i * W - Y_i)\|^2 + \lambda\|W\|^2, \tag{7}$$

where X_0 is the positive training sample and $X_i(i = 1...n)$ is the hard negative training sample. Y_0 are Gaussian-like label while $Y_i(i = 1...n)$ are flat labels filled with zeros.

The kernel weights W in Equ.7 are optimized with the SGD (Stochastic Gradient Descent) algorithm.

D. PEAK TO SIDELOBE RATIO

The proposed distractor handling module including trajectory smoothing constraint and hard negative mining is computationally consuming and puts a big burden on the real-time capacity of convolutional regression trackers. In fact, it's unnecessary to activate this distractor-handling module in every frame. Most frames in a video are 'easy' frames where the target moves smoothly and its appearance changes slowly. In easy frames, there are no background clutters or background distractors and thus the response map has only one peak value in the real target position. Accurate and robust tracking can be achieved by traditional convolutional regression trackers without the distractor-handling module. By contrast, in a few 'hard' frames which are filled with background distractors, the response map will fluctuate intensely and include more peak values as shown in Fig.4. Therefore, the distractor-handling module should be activated occasionally in these 'hard' frames. Whether activating the distractor-handling module on or off depends on the tracking status of the convolutional regression tracker.

We point out that the tracking status of the convolutional regression tracker can be inferred from the peak to sidelobe

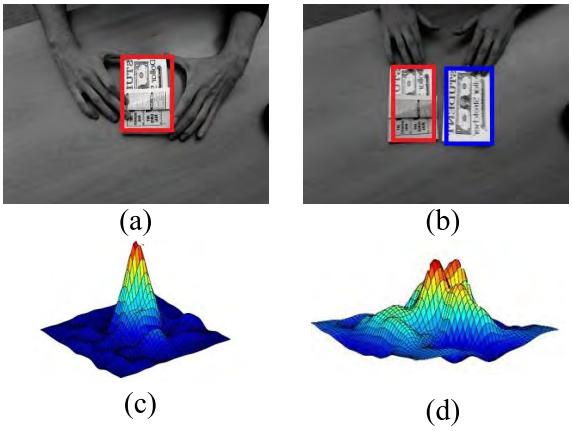

FIGURE 4. (a) Without background distractors; (b) with background distractors; (c) larger PSR value; (d) small PSR value.

ratio of the response map. In the general case, this response map peaks at the highest and damps fast from the peak to the boundary. However, in presence of background distractors, there will be several sub-peaks on the response map. In light of this observation, we design the peak to sidelobe ratio (PSR) measure to quantify the peakiness of the response peak and thus evaluate the tracking status. The PSR score with higher value means the detection confidence is higher.

Here we describe the PSR measure in detail. Let f_t be a $N \times M$ matrix representing the response map of frame t. The PSR value of the response map is defined by

$$PSR_t = \frac{max(f_t) - \mu(f_t)}{\sigma(f_t)} \tag{8}$$

where $\mu(f_t)$ and $\sigma(f_t)$ are the mean value and the standard deviation of the response map f_t.

The PSR criterion becomes larger when the response map has fewer noise and sharper peak. Otherwise, the PSR criterion will fall into a small value. We save the PSR values and calculate their historical average values as threshold:

$$PSR_{threshold} = \frac{\sum_{t=1}^{T} PSR_t}{T} \tag{9}$$

Online finetuning of the convolutional regression network is performed if only the PSR value is lower than the PSR threshold, which avoids unwanted model updating and alleviates the computational burden of convolutional regression trackers.

E. SCALE ESTIMATION

In the source codes provided in [3], scale variation is estimated by processing the search image at several scales with the a fixed aspect ratio. With no doubt, searching scale at multiple resolutions significantly increases the computational cost. To achieve real-time scale adaptive tracking, our DaRT tracker removes the scale estimation from CREST and exploits a common 1-dimensional scale correlation filter.

After we estimate the translation of target in the current frame, the next step is to estimate the target scale. Inspired by [4], the scale estimation is efficiently completed through a one-dimensional correlation filter. For convenient reasons, we adopt the same notation in [4]. The training example f for updating the scale filter is computed by extracting features using variable patch sizes centered on the target. Let $P \times R$ denote the target size in the current frame and S be the size of the scale filter. For each $n \in \{\left\lfloor \frac{-(s-1)}{2} \right\rfloor, \cdots, \left\lfloor \frac{s-1}{2} \right\rfloor\}$, we extract an image patch J_n of size $a^n P \times a^n R$ centered on the target. Here a denotes the scale factor between feature layers. The value $f(n)$ of the training example f at scale level n is set to the d-dimensional feature descriptor of J_n. The scale filter h_{scale} is computed and updated with the sample f. After the translation estimation of the target, an example z is extracted from this location using the same procedure as for f and the scale filter h_{scale} is applied to obtain the response value. The scale change corresponding to the largest value is picked out to compute the target scale.

Algorithm 1 Distractor-Aware Regression Tracking

Input:
Target state $X_{t-1} = (x_{t-1}, y_{t-1}, s_{t-1})$ in frame $t-1$.

Output:
Estimated target state $X_t = (x_t, y_t, s_t)$ in each frame.

Tracking:

1: Derive the response map with the convolutional regression tracker and calculate the PSL_t value from response map in frame t.
2: **if** $PSL_t > PSL_{threshold}$ **then**
3: Estimate the target location (x_t, y_t) as the maxima of the response map of the convolutional regression tracker around (x_{t-1}, y_{t-1}) in frame t.
4: **else**
5: Locate n local maximas on the response map and find the local maixma with the largest temporal trajectory smoothness as the target position.
6: Locate n Feed the other local maximas into the convolutional regression network as hard negative samples to online finetune the network parameters.
7: Feed the local maxima with the largest temporal trajectory smoothness into the convolutional regression network as a positive sample to online finetune the network parameters.
8: Estimate the target scale s_t with DSST.
9: Update the $PSL_{threshold}$ value.
10: Update the mean and standard deviation of the velocity of the target along the x and y axis.

IV. OVERALL TRACKING FRAMEWORK

In this subsection, we present an outline of our proposal in Algorithm 1 and show the diagram in Fig.5.

V. EXPERIMENTS

We validate our Distractor-aware Regression Tracker (DaRT) by performing comprehensive experiments on two tracking benchmarks: OTB100 [12] and VOT2016 [27].

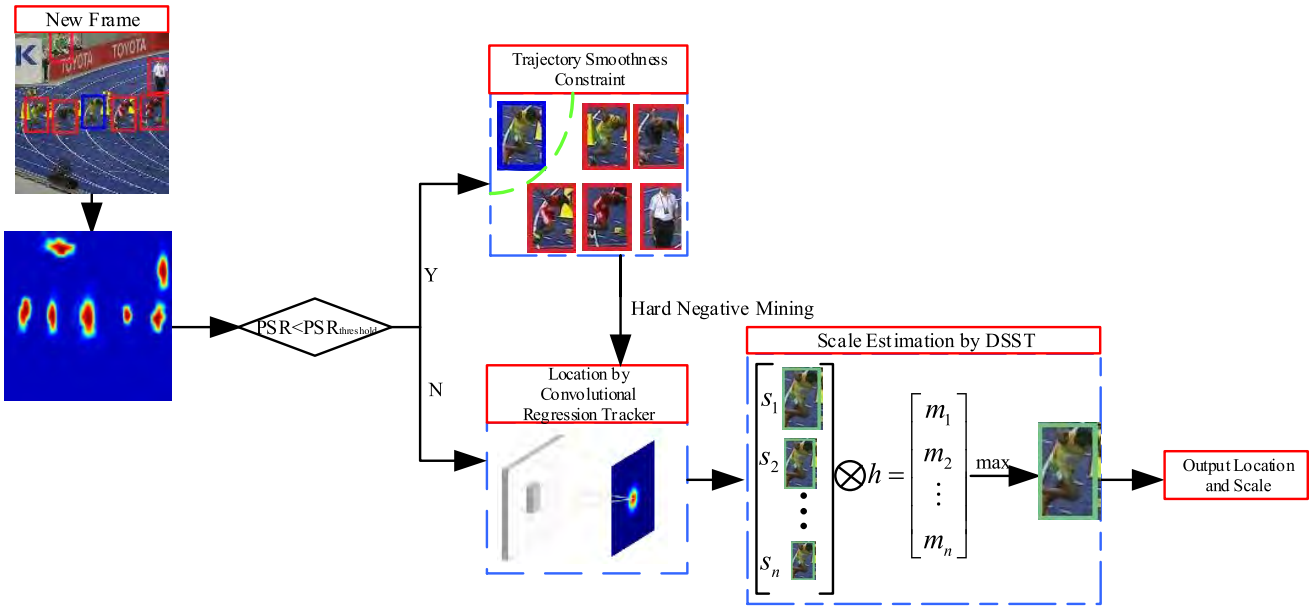

FIGURE 5. The overall flowchart of the proposed distractor-aware regression tracker.

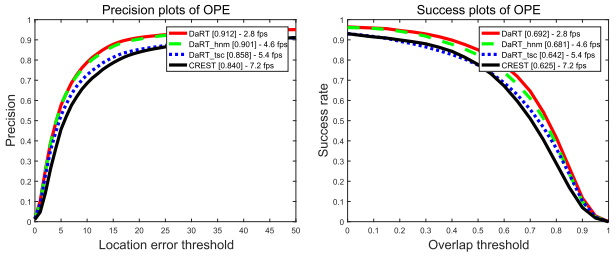

FIGURE 6. Precision plots (left) and success plots (right) of CREST, DaRT$_{tsc}$, DaRT$_{hnm}$ and DaRT on the OTB100 benchmark.

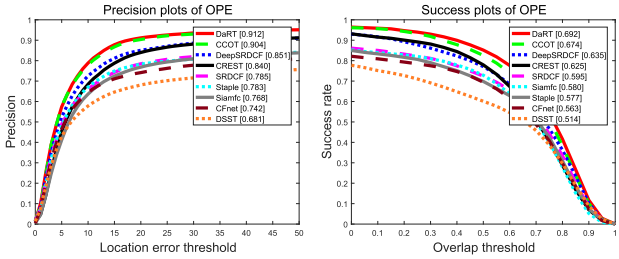

FIGURE 7. Precision plots (left) and success plots (right) of trackers in comparison on the OTB100 benchmark.

Evaluation Methodology: On OTB100, we use the precision plots and success plots in one-pass evaluation (OPE) [11] to rank all the trackers. The precision plots are computed as the percentage of frames in the sequences where Euclidean distance between the ground-truth and the estimated target position is smaller than a certain threshold. The success plots are plotted over the range of intersection over union (IoU) thresholds over all videos. For the VOT2016 dataset, tracking performance is evaluated in terms of both accuracy and robustness. The accuracy score is based on the overlap with ground truth, while the robustness is determined by failure rate. Different from OTB100, the trackers in VOT2016 are restarted at each failure.

Comparison Scenarios: An ablation study is done on OTB100 to demonstrate the effectiveness of trajectory smoothing constraint and hard negative mining respectively on distractor-aware tracking. On OTB100, we also compare DaRT with state-of-the-art trackers. On the VOT2016 dataset, we compare DaRT with the top 10 trackers in the challenge.

Implementation Details: In this paper, we choose CREST as the convolutional regression tracker. For convenience reasons, we follow the default parameter setting of CREST as reported in [3]. The CREST tracker employs the convolutional features extracted from the relu4_3 layer in the imagenet-vgg-verydeep-16 model for feature representation. This model can be downloaded from http://www.vlfeat.org/matconvnet/pretrained/. The target search area of CREST is set to be square and five times the target size. Parameters are fixed for all videos in each dataset. Our tracker is implemented in Matlab and uses Matconvnet [34] for deep feature extraction. The comparison experiments of DaRT are performed on a 4-core Intel Core -7-6700 CPU at 3.4GHz with a GeForce GTX TITAN GPU.

A. EVALUATION ON OTB

1) ABLATION STUDY

In this subsection, an ablation study on OTB100 is conducted to demonstrate the effectiveness of the trajectory smoothing constraint and hard negative mining respectively on distractor-aware tracking. We introduce two baseline trackers (DaRT_tsc and DaRT_hnm). DaRT_tsc equips CREST with the trajectory smoothing constraint while DaRT_hnm equips CREST with hard negative mining. On contrast, DaRT equips CREST with both the trajectory smoothing constraint and hard negative mining. Fig.6 shows the precision and success

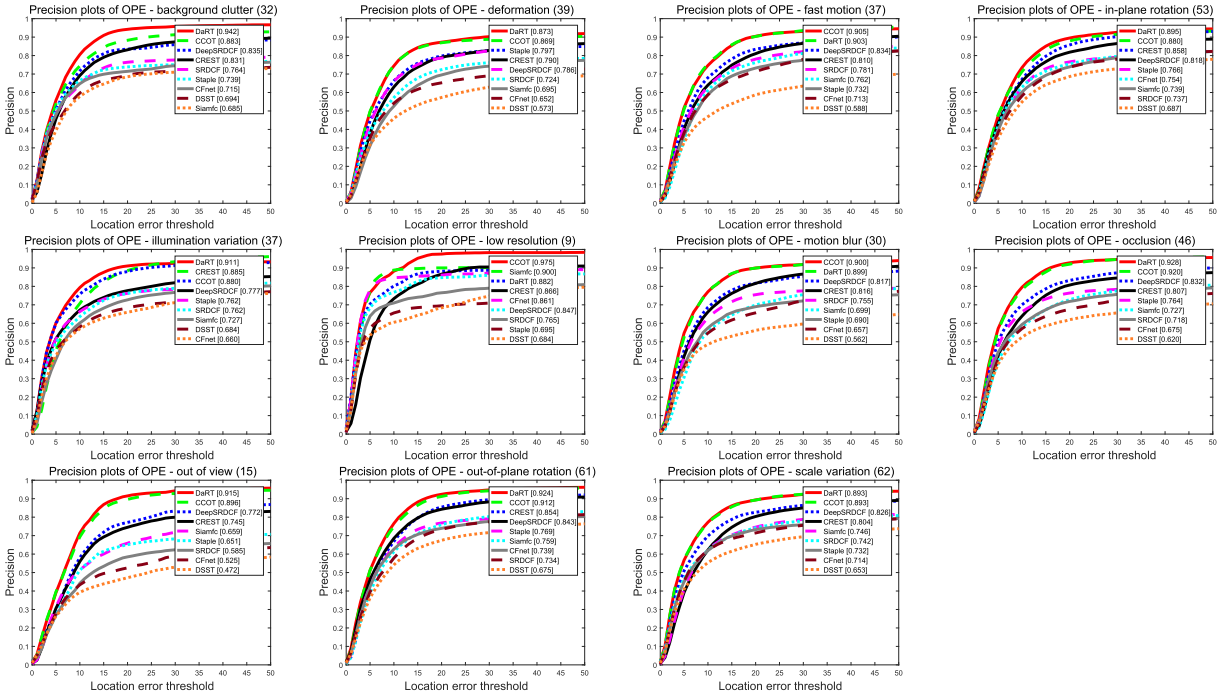

FIGURE 8. ***Precision*** **plots on 11 attributes of the OTB100 dataset. These trackers are ranked by their scores at the threshold of 20 pixels.**

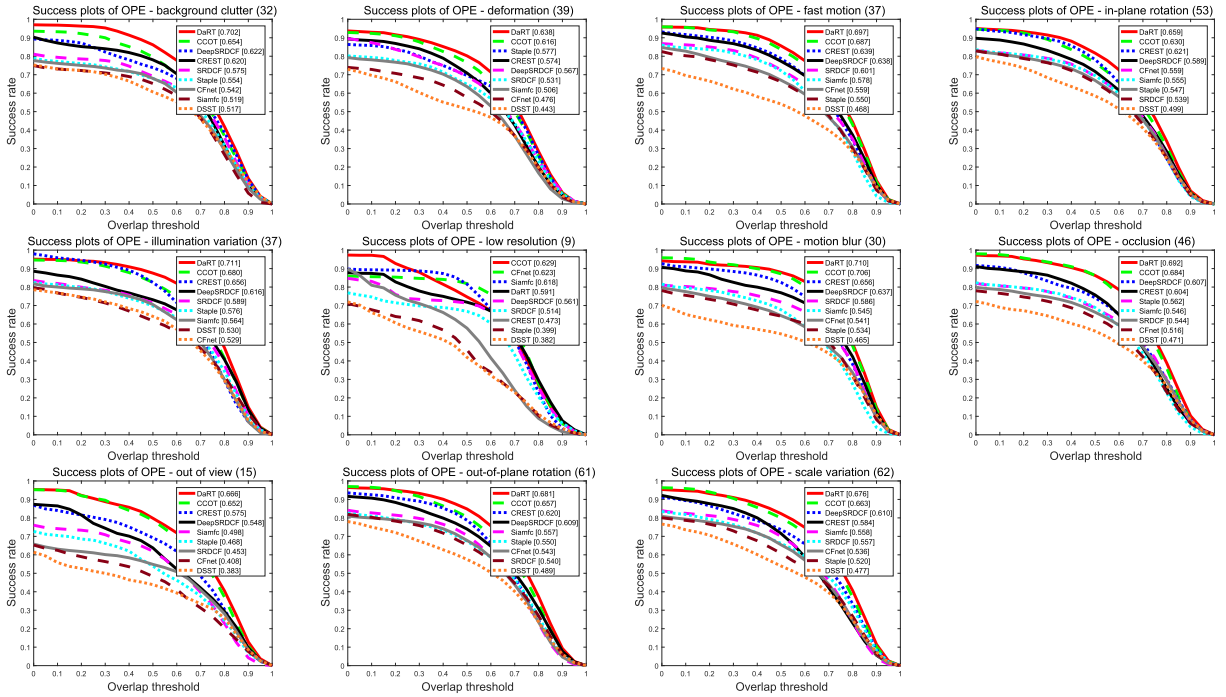

FIGURE 9. ***Success*** **plots on 11 attributes of the OTB100 dataset. These trackers are ranked by their AUC scores.**

plots of the three trackers (DaRT, DaRT_*tsc*, DaRT_*hnm*). Both DaRT_*tsc* and DaRT_*hnm* achieve performance improvement against CREST. Compared with DaRT_*tsc* and DaRT_*hnm*, DaRT achieves better performance in both the precision and success plots, which demonstrates the effectiveness of the trajectory smoothing constraint and hard negative mining respectively on distractor-aware tracking.

2) QUANTITATIVE RESULTS

In this subsection, we compare DaRT with eight state-of-the-art trackers on OTB100. The trackers in comparison include CREST [3], CFnet [35], Siamfc [36], CCOT [25], DeepSRDCF [24], SRDCF [23], Staple [37] and DSST [4]. Fig.7 shows the performance of trackers in comparison on the OTB100 benchmark. In general, our proposed algorithm

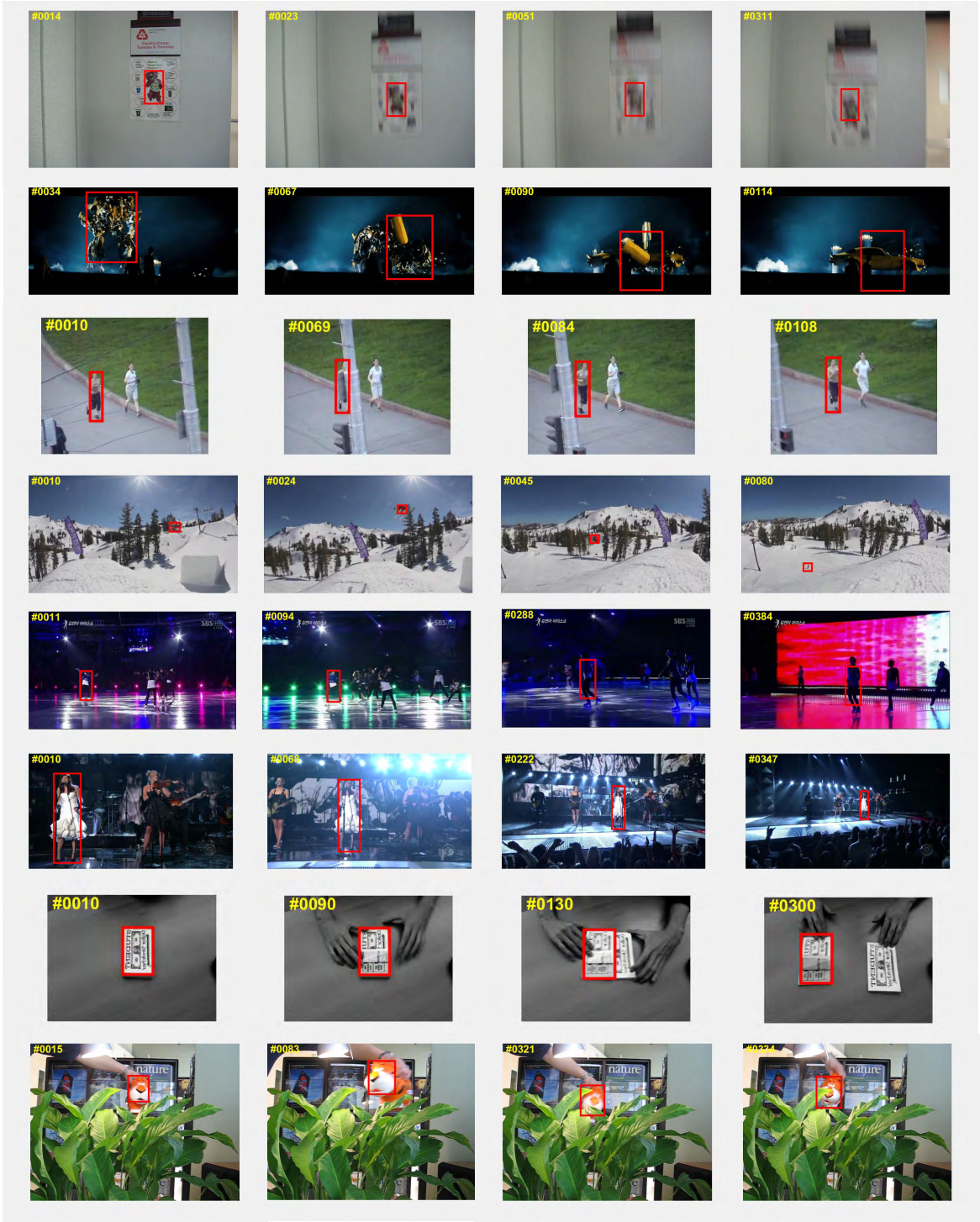

FIGURE 10. Tracking snapshots of our tracker. The videos (from top to bottom) are: ***BlurOwl***, ***Trans***, ***Jogging***, ***Skiing***, ***Skating1***, ***Singer1***, ***Coupon*** and ***Tiger1***.

performs superiorly against the trackers in comparison in both the precision and success plots.

Fig.8 and Fig.9 illustrate the attribute based evaluation of all trackers on the OTB100 dataset. All sequences in the OTB100 dataset are annotated by 11 different attributes, namely: illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutter and low resolution. In Fig.8 and Fig.9, we show the precision and success rates of our tracker and other trackers in comparison

TABLE 1. State-of-the-art comparison in terms of expected average overlap (EAO), robustness (failure rate), accuracy, and speed (in EFO units) on the VOT 2016 dataset. Only the top-10 best compared trackers are shown. The best and second best values are highlighted by red and blue fonts.

	DNT	Staple+	SRBT	EBT	DDC	Staple	MLDF	SSAT	TCNN	C-COT	**DaRT**
EAO	0.278	0.286	0.290	0.291	0.293	0.295	0.310	0.320	0.325	0.331	0.335
Failure rate	1.18	1.32	1.25	0.90	1.23	1.35	0.83	1.04	0.96	0.85	0.92
Accuracy	0.515	0.557	0.496	0.465	0.541	0.544	0.490	0.577	0.554	0.539	0.560
EFO	1.127	44.765	3.688	3.011	0.198	11.144	1.483	0.475	1.049	0.507	5.03

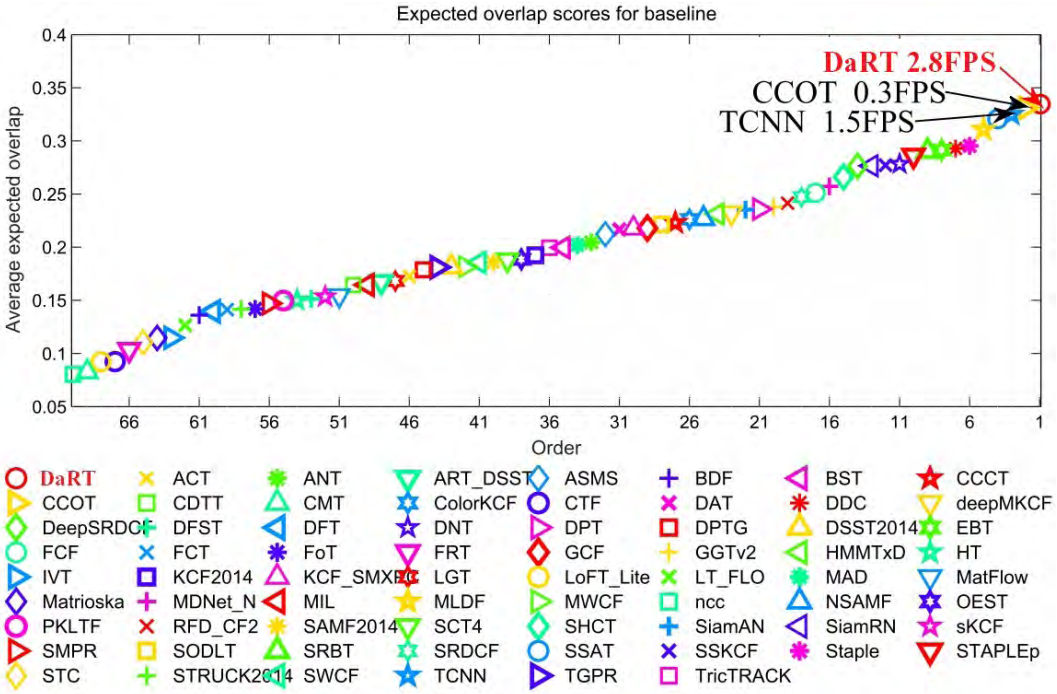

FIGURE 11. Expected average overlap ranking plots on the VOT2016 benchmark.

under 11 attributes: illumination variation, scale variation, occlusion, deformation, motion blur, fast motion, in-plane rotation, out-of-plane rotation, out-of-view, background clutter and low resolution. It can be seen that our proposed tracker achieves the best performance under most attributes.

3) QUALITATIVE RESULTS

To intuitively exhibit the superiority of the proposed algorithm, Fig.10 illustrates snapshots of the tracking results of DaRT on videos from the OTB100 dataset. The videos (from top to bottom) are:*BlurOwl*, *Trans*, *Jogging*, *Skiing*, *Skating1*, *Singer1*, *Coupon* and *Tiger1*. These sequences cover many challenging scenarios that a tracker may encounter, such as occlusion, fast motion, background clutter, illumination variation, dramatic scale change and et, al. Our tracker performs well in all the above videos which demonstrates the strong robustness of DaRT under complicated tracking scenarios.

B. EVALUATION ON VOT2016

The visual object tracking (VOT) challenge is a competition between short-term, model-free visual tracking algorithms. Different from OTB, for each sequence in this dataset, a tracker is restarted whenever the target is lost (*i.e.* at a tracking failure). Four primary measures are used to analyze tracking performance: accuracy (A), robustness (R), expected average overlap (EAO) and equivalent filter operation (EFO). A is calculated as the average IoU, while R is expressed in terms of the total number of failures. EAO represents the average IoU with no re-initialization following a failure. EAO reports the tracker speed in terms of a predefined filtering operation that the toolkit carries out prior to running the experiments. We refer readers to [27] for details.

Table 1 shows the comparison of our approach with the top 10 participants in the VOT2016 challenge. In Table 1, DeepTACF outperforms all the top 10 trackers at the

EAO score (0.335). Fig.11 shows a visualization of the accuracy and robustness ranking plot for the compared trackers on the VOT2016 dataset.

VI. CONCLUSION

In the paper, we propose a distractor-aware regression tracker frame inspired by the trajectory smoothing constraint and hard negative mining. The trajectory smoothing constraint measures motion distance and motion direction between adjacent frames to discard distractors which fail to meet trajectory smoothness. On this basis, distractors are fed into convolutional regression networks as hard negative samples, which boosts the discriminability of convolutional regression trackers against background distractors. To validate the effectiveness of the proposed method, experiments are performed to compare our algorithm with many other state-of-the-art trackers on the popular OTB100 benchmark and VOT2016 challenge. Experimental results demonstrate the superiority of our tracker. One interesting avenue for future work would be extending our tracker to multi-object and/or long-term tracking with these inspiring ideas.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, pp. 1–45, 2006.

[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] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. H. Lau, and M.-H. Yang, "CREST: Convolutional residual learning for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 2574–2583.

[4] 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.

[5] 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.

[6] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean shift," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, vol. 2, Jun. 2000, pp. 142–149.

[7] 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.

[8] X. Mei and H. Ling, "Robust visual tracking using $\ell 1$ minimization," in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 1436–1443.

[9] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M.-M. Cheng, S. L. Hicks, and P. H. S. Torr, "Struck: Structured output tracking with kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2096–2109, Oct. 2016.

[10] B. Babenko, M.-H. Yang, and S. Belongie, "Robust object tracking with online multiple instance learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1619–1632, Aug. 2011.

[11] 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.

[12] 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.

[13] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 445–461.

[14] P. Liang, E. Blasch, and H. Ling, "Encoding color information for visual tracking: Algorithms and benchmark," *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5630–5644, Dec. 2015.

[15] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1090–1097.

[16] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Discriminative scale space tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 8, pp. 1561–1575, Aug. 2017.

[17] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5388–5396.

[18] A. Bibi, M. Mueller, and B. Ghanem, "Target response adaptation for correlation filter tracking," in *Proc. 14th Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands, Oct. 2016, pp. 419–433.

[19] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1401–1409.

[20] L. Wang, W. Ouyang, X. Wang, and H. Lu, "Visual tracking with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Dec. 2015, pp. 3119–3127.

[21] K. Chen and W. Tao, "Convolutional regression for visual tracking," *IEEE Trans. Image Process.*, vol. 27, no. 7, pp. 3611–3620, Jul. 2018.

[22] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 2544–2550.

[23] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4310–4318.

[24] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proc. IEEE Conf. Comput. Vis. Workshop*, Dec. 2015, pp. 621–629.

[25] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 472–488.

[26] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernandez, T. Vojir, G. Hager, G. Nebehay, R. Pflugfelder, A. Gupta, A. Bibi, A. Lukezic, A. Garcia-Martin, A. Saffari, A. Petrosino, and A. S. Montero, "The visual object tracking VOT2015 challenge results," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Dec. 2015, pp. 564–586.

[27] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Čehovin, T. Vojír, G. Häger, A. Lukežič, G. Fernández, A. Gupta, A. Petrosino, A. Memarmoghadam, A. Garcia-Martin, and A. S. Montero, "The visual object tracking VOT2016 challenge results," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Oct. 2016, pp. 777–823.

[28] H. Possegger, T. Mauthner, and H. Bischof, "In defense of color-based model-free tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 2113–2120.

[29] T. B. Dinh, N. Vo, and G. Medioni, "Context tracker: Exploring supporters and distracters in unconstrained environments," in *Proc. 24th IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Colorado Springs, CO, USA, Jun. 2011, pp. 1177–1184.

[30] M. Yang, Y. Wu, and G. Hua, "Context-aware visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 7, pp. 1195–1209, Jul. 2009.

[31] M. Mueller, N. Smith, and B. Ghanem, "Context-aware correlation filter tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 2, Jul. 2017, pp. 1387–1395.

[32] Y. Kuai, G. Wen, and D. Li, "When correlation filters meet fully-convolutional siamese networks for distractor-aware tracking," *Signal Process., Image Commun.*, vol. 64, pp. 107–117, May 2018.

[33] D. Li, F. Porikli, G. Wen, and Y. Kuai, "When correlation filters meet siamese networks for real-time complementary tracking," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.

[34] A. Vedaldi and K. Lenc, "MatConvNet: Convolutional neural networks for MATLAB," in *Proc. 23rd ACM Int. Conf. Multimedia*, 2015, pp. 689–692.

[35] J. Valmadre, L. Bertinetto, J. F. Henriques, A. Vedaldi, and P. H. S. Torr, "End-to-end representation learning for correlation filter based tracking," 2017, *arXiv:1704.06036*. [Online]. Available: https://arxiv.org/abs/1704.06036

[36] L. Bertinetto, J. Valmadre, and J. F. Henriques, "Fully-convolutional siamese networks for object tracking," in *Proc. IEEE Conf. Comput. Vis.*, 2015, pp. 3119–3127.

[37] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. H. S. Torr, "Staple: Complementary learners for real-time tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 1401–1409.