Correlation Filter Tracking: Beyond an Open-loop System

Qingyong Hu[1]
huqingyong15@nudt.edu.cn
Yulan Guo[12]
yulan.guo@nudt.edu.cn
Yunjin Chen[3]
chenyunjin_nudt@hotmail.com
Jingjing Xiao[4]
shine636363@sina.com
Wei An[1]
nudtanwei@tom.com

[1] College of Electronic Science and Engineering
National University of Defense Technology
Changsha, China

[2] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China

[3] ULSEE Inc.

[4] Department of Medical Engineering, Third Military Medical University, Chongqing, China

Abstract

Most existing Correlation Filter (CF) based trackers do not use any feedback from tracking output and can be considered as open-loop systems. They are prone to drifting when the object endures occlusion and large appearance changes. In this paper, we propose a **generic** self-correction mechanism for CF based trackers by introducing a closed-loop feedback technique. Our mechanism first detects the abnormality in tracking output using the Gaussian shape prior of a response map, and then estimates the tracking error by minimizing the discrepancy of tracking output and the expected response. An optimal offset is finally given to regulate the tracking process and prevent tracking drifting through a feedback loop. Extensive experiments have been conducted on four large-scale benchmarks, including OTB-2013, OTB-2015, TC-128, and UAV123@10fps. Experimental results show that our self-correction mechanism can be used to improve the overall performance of most CF based trackers by a large margin. Besides, our proposed Enhanced Dual Correlation Filters (EDCF) tracker outperforms the state-of-the-art methods and runs at a high speed nearly 80 fps.

1 Introduction

Visual object tracking is a very popular topic in computer vision for its numerous applications, such as intelligent vehicles, human-machine interaction, and surveillance [20, 21]. Although object tracking has been investigated for several decades, it remains a challenging problem due to the limited number of labeled training samples and appearance variations of objects.

In recent years, Correlation Filter (CF) based trackers have attracted significant attention for its high efficiency and robustness [4, 5, 8, 9, 10, 18]. The success of CF based trackers

are mainly attributed to two facts. First, this kind of tracker can generate a large number of training samples by shifting the image patch. Therefore, the lack of training data faced by most trackers can be well addressed. Second, fast training and detection can be achieved by circular correlation via Fast Fourier Transform (FFT). Bolme et al. [3] introduced correlation filters to visual tracking for the first time by minimizing the output sum of squared error. Subsequently, a series of CF based trackers have been proposed to further enhance the tracking performance [1, 7, 11, 14, 15]. For instance, [4, 8, 12, 14] address scale and rotation variations, [1, 5, 6, 17] incorporate more information to the appearance model, while [7, 13] reduce boundary effects.

Although remarkable progress has been achieved by CF based tracker, there is an important issue that has been overlooked for a long time by the community. That is, most CF based trackers implicitly consider tracking as an open-loop process as they neither check the tracking output nor generate any feedback to regulate the tracking process. They are therefore, unable to detect the abnormity in tracking output and correct the localization error during tracking. Consequently, they are prone to drifting, especially when the object under tracking encounters large appearance changes and severe occlusion. To address this problem, some methods [18, 25] train an additional detector to re-initialize the tracking process when a tracking failure is found. However, training an additional detector significantly increases the computational burden, and the detection results are not always reliable as only a limited samples are available for the training of this detector.

Different from these existing algorithms, we design a closed-loop feedback tracking system for CF based trackers. Specifically, a self-correction mechanism is proposed to directly estimate the localization error by minimizing the discrepancy between the tracking output and the expected output, and an optimal offset is generated to compensate for the localization error and to regulate the tracking process through a feedback loop. Our mechanism is generic, and it can help CF based trackers to effectively reduce error accumulation and tracking drifting, with negligible computational cost increase.

The contributions of this paper can be summarized as follow:

- We interpret object tracking as a closed-loop tracking problem, and add a feedback loop to the tracking process by introducing an efficient method to estimate the localization error.
- We propose an effective and generic self-correction mechanism for CF based trackers to reduce tracking drifting. The proposed mechanism can be easily incorporated into any CF based tracker to improve its robustness and long-term tracking ability.
- Extensive experiments have been conducted on four large-scale benchmarks, including OTB-2013 [22], OTB-2015 [23], TC-128 [16], and UAV123@10fps benchmark [19]. Experimental results have demonstrated the efficiency and effectiveness of our proposed trackers as compared to the state-of-the-art.

2 The Proposed Approach

In this section, we first analyze the open-loop problem existed in CF based trackers and present the motivation of our work. We then illustrate our self-correction mechanism in details. Finally, we propose a new tracker by applying our self-correction mechanism to the Dual Correlation Filters (DCF) tracker.

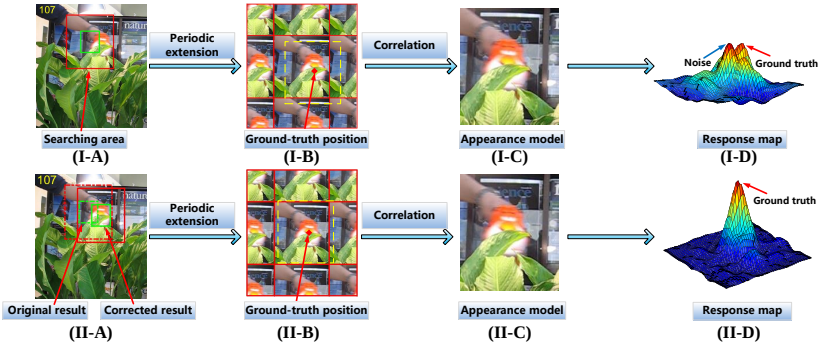

Figure 1: An illustration of our motivation. The top row shows the process of a traditional open-loop CF tracker, and the bottom row shows the process of a CF tracker with self-correction.

2.1 CF Tracker Dissection: From a Viewpoint of Open-loop System

Here, we first revisit the detection process of CF based trackers, and then explain how the tracking drift occurs in CF based trackers from a viewpoint of open-loop system.

For CF based trackers, the process to detect and localize an object in a new frame includes three steps: (1) crop a sub-image x_t (i.e., define a search area) in the current frame. Since CF based trackers implicitly assume that the object is unlikely to have a large displacement in two consecutive frames, the search area is defined as a sub-image centered at the last estimated position; (2) calculate the correlation response map between this sub-image x_t and the trained model h_t using FFT. This operation is equivalent to the process that the sub-image x_t is firstly extended in a periodic manner, and then the filter h_t slides from the upper-left pixel to the lower-right pixel of the sub-image x_t (as shown in Fig. 1(I-B)). The correlation score (i.e., detection score) at each position is calculated as the inner product between the filter h_t and the shifted sub-image at that position; (3) use the position with maximal correlation response to localize the object.

Generally, the output response map approximately follows a Gaussian distribution as CF based trackers are trained with Gaussian shaped regression labels. However, if a large localization error exists in the last predicted position or the object has fast motion in the current frame (Fig. 1(I-A)), the shifted patch at the ground-truth position in an extended image contains a large area with periodic repetitions (Fig. 1(I-B)), making the detection score generated at this position far below the expected value. In this case, the correlation response map (Fig. 1(I-D)) dose not follow Gaussian distribution any more, it has an abnormal shape even with multiple peaks. For an open-loop CF tracker, the position with maximal correlation response (i.e., Step 3) is still used to determine the object location. As a consequence, the tracker is easy to be drifted to the background. Besides, it is also very difficult for such an open-loop tracker to recover from drifting by itself. Therefore, a tracking failure is inevitable. We argue that most tracking failures start from this step.

In contrast, if the tracker can timely detect the abnormality in a response map, and repeat the detection process after reducing or correcting the localization error (Fig. 1(II-A)), the output response map will behave as expected (Fig. 1(II-D)), i.e., Gaussian shaped and energy-concentrated. Consequently, the object can be correctly tracked as there is no ambiguity or noise interference in the response map.

Motivated by these observations, we argue that most tracking failures can be prevented if

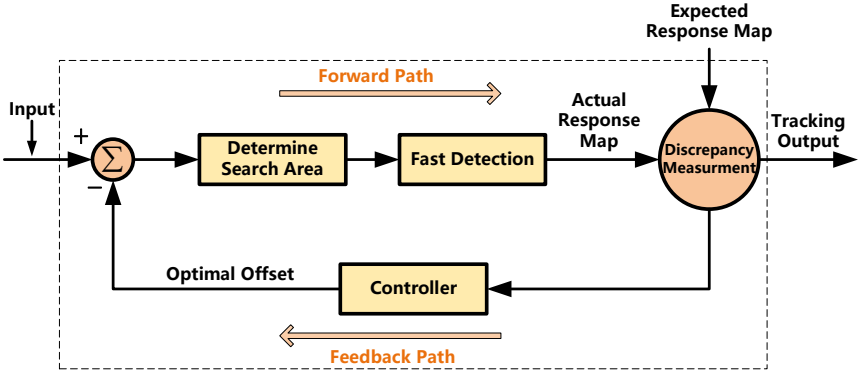

Figure 2: The pipeline of our self-correction mechanism.

the abnormality in tracking output can be timely detected and the error in a tracking system can be corrected. It is widely known that closed-loop feedback has been successfully used in many automatic control systems to correct errors and maintain system stability. Therefore, we aim to modify the open-loop structure of CF based trackers using a closed-loop feedback technique.

2.2 The Proposed Self-correction Mechanism

In this section, we design a self-correction mechanism to monitor the tracking output and automatically correct the localization error during detection, as shown in Fig. 2. Our mechanism is based on a closed-loop feedback technique. By minimizing the discrepancy between the tracking output and the expected response, our mechanism generates an optimal offset to regulate the system input (i.e., correct the localization error in last predicted position) through a feedback path. This can help the tracker to re-define the search area and correct the tracking output to meet our expectation.

Here, our mechanism is based on the assumption that the response map produced by the sub-image centered at the ground-truth position has the minimum discrepancy to the expected response. Therefore, the objective function of our mechanism can be written as:

$$\arg\min_{\mathbf{X}_p} \mathcal{D}(\mathbf{R}(\mathbf{X}_p, h_t), \mathbf{N}) \tag{1}$$

where $\mathbf{X}_p$ is a sub-image located at position p, h_t is the filter trained from the past frames, $\mathbf{R}(\mathbf{X}_p, h_t)$ denotes the filter response between the sub-image $\mathbf{X}_p$ and the filter h_t, $\mathbf{N}$ is an expected response map with a single peak, and $\mathcal{D}$ is a specific function to measure the discrepancy between the actual response map $\mathbf{R}$ and the expected response map $\mathbf{N}$. Note that, we have many choices for the measurement function $\mathcal{D}$. In our work, in order to keep the simplicity of the proposed self-correction mechanism, we propose a heuristic approach to measure the discrepancy:

$$\mathcal{D}((\mathbf{X}_p, h_t), \mathbf{N}) = w_P(P_{expected} - P(\mathbf{X}_p, h_t)) + w_S(S_{expected} - S(\mathbf{X}_p, h_t)) \tag{2}$$

where $P(\mathbf{X}_p, h_t)$ and $P_{expected}$ are the peak value of the filter response map $\mathbf{R}$ and expected response map $\mathbf{N}$, respectively. $S(\mathbf{X}_p, h_t)$ and $S_{expected}$ are the Peak-to-Sidelobe Ratio (PSR) of the filter response map $\mathbf{R}$ and expected response map $\mathbf{N}$, respectively. w_P and w_S are the

weights for the two terms. In our work, PSR is defined as:

$$S(\mathbf{X}) = \frac{\max(\mathbf{R}(\mathbf{X})) - \mu_{\Phi}(\mathbf{R}(\mathbf{X}))}{\sigma_{\Phi}(\mathbf{R}(\mathbf{X}))} \tag{3}$$

where sidelobe Φ represents the pixels in the sub-image $\mathbf{X}$ excluding the central region[1] around the peak, μ_{Φ} and σ_{Φ} denote the mean and deviation of the sidelobe, respectively. A higher value of $S(\mathbf{X})$ means that more energy is included in the area around the peak. Since both $P(\mathbf{X}_p, h_t)$ and $S(\mathbf{X}_p, h_t)$ are positive, the optimization problem in Eq. 1 is equivalent to solving the following maximization problem:

$$\arg\max_{\mathbf{X}_p} \mathcal{G}(\mathbf{X}_p, h_t) = w_P \cdot P(\mathbf{X}_p, h_t) + w_S \cdot S(\mathbf{X}_p, h_t) \tag{4}$$

Here, both $P(\mathbf{X}_p, h_t)$ and $S(\mathbf{X}_p, h_t)$ are normalized to the range of [0,1] before weighted summation in Eqs. (2) and (4). Since directly optimizing the problem defined in Eq. 4 is time-consuming, we further simplify this problem by adopting an efficient alternative to obtain an approximate solution in this paper. Considering that the time interval between two neighboring frames is very small and the speed of a moving object is limited, the object is likely to appear in the area near the last estimated position. Therefore, instead of exhaustively searching the whole image for the optimal results, we only evaluate the $\mathcal{G}$ score for a set of predefined sub-images, and then select the best one to maximum the $\mathcal{G}$ score. Specifically, our controller first set the last estimated position (x_0, y_0) as the center of a set of concentric circles, and then select a set of locations to generate candidate patches. To maintain high computational efficiency, a step-wise searching approach is used in this paper, that is:

$$x_{i,j} = x_0 + i \cdot \Delta T \cdot \cos(j \cdot \Delta\theta + \frac{(-1)^i + 1}{4} \cdot \Delta\theta) \tag{5}$$

$$y_{i,j} = y_0 + i \cdot \Delta T \cdot \sin(j \cdot \Delta\theta + \frac{(-1)^i + 1}{4} \cdot \Delta\theta) \tag{6}$$

where ΔT and $\Delta\theta$ respectively denote the distance step and angle step, i ranges from $[1, m]$ and $m = \frac{R_d}{\Delta T}$, R_d is the radius of the circle, it is dynamically determined by the target size and the peak value of the response map, j ranges from $[1, n]$ and $n = \frac{2\pi}{\Delta\theta}$.

Finally, $m \times n$ candidate sub-images around the previous estimated location are extracted. Our controller then calculates the correlation score (i.e., response map) between the appearance model and each extracted sub-image. The sub-image with the maximum $\mathcal{G}$ is considered as the optimal search region, and the offset between the center of this sub-image and the last predicted position is feed to the system input to correct the localization error.

In practice, existing CF based trackers can work well in simple scenarios (e.g., with slowly moving objects or homogeneous background). Therefore, it is unnecessary to optimize this problem in each frame for the sake of computational efficiency. For this reason, our proposed self-correction mechanism will be activated only when the output response map is far from the expected one.

[1]The central region is set as 15% of the response map area in this paper.

2.3 The Proposed Tracker

To further improve the tracking performance, we propose an Enhanced DCF (EDCF) tracker by augmenting the base tracker with scale estimation, multiple feature integration and our self-correction mechanism. Since DCF tracker can achieve an impressive performance with high efficiency, it is selected as our base tracker. Following [8], our EDCF tracker also learns an independent scale filter for accurate scale estimation, and uses principle component analysis to achieve dimensionality reduction. In addition, we use a combination of HOG and Color-Names (CN) as the feature representation of our EDCF tracker, the same as [14]. Finally, we integrate our self-correction mechanism into the EDCF tracker to further improve its robustness and long-term tracking performance.

3 Experimental Analysis

In this section, we first present the dataset, evaluation metrics, and parameters used in our experiments, and then conduct rigorous experiments to demonstrate the effectiveness of our proposed mechanism. Finally, we compare our proposed EDCF tracker with the state-of-the-art.

3.1 Experimental Setup

We implemented the proposed algorithm in MATLAB and conducted all experiments on a PC with an 3.2GHz Intel Core I5 CPU and a 8GB RAM. To achieve rigorous comparison, the parameters of our tracker are fixed for all experiments.

Dataset: We conducted extensive experiments on four large-scale datasets, including OTB-2013 (with 51 sequences), OTB-2015 (with 100 sequences), UAV@10fps (with 123 low frame rate sequences), and Temple Color-128 (with 128 color sequences).

Evaluation Metrics: Following the OTB dataset [23], we use the precision rate and success rate as our main evaluation metrics. The precision rate is calculated as the fraction of tracking frames with Center Location Errors (CLE) less than 20 pixels. Here, CLE is defined as the Euclidean distance between the estimated and the ground-truth positions. The success rate is defined as the Area Under the Curve (AUC) of the success plot. Here, a success plot is obtained by calculating the overlap ratio between the estimated and ground-truth bounding boxes under different thresholds. The overlap ratio is defined as $S = \frac{Area(R_t \cap R_g)}{Area(R_t \cup R_g)}$, where R_t is the estimated bounding box and R_g is the corresponding ground-truth bounding box. $\cup$ and $\cap$ denote the intersection and union of two regions, respectively.

Trackers \ Key parameter	Response threshold	PSR threshold	m	n
SAMF_SC	0.16	6	6	3
KCF_SC	0.2	10	4	3
DCF_SC	0.2	10	5	9
CNT_SC	0.25	12	3	5
CSK_SC	0.25	12	3	8
EDCF	0.2	10	6	8

Table 1: Parameter settings of 6 trackers enhanced with our self-correction mechanism.

Parameters: For all baseline trackers, we use the default parameters provided by the authors. For fair comparison, the CF variants enhanced with our self-correction mechanism use the same parameters as their original trackers, except for the new parameters introduced by our mechanism. The parameter setting of our proposed trackers is shown in Table 1. Key parameters of our mechanism include the distance step ΔT, angle step $\Delta\theta$ (which are determined by the number of steps m and n, $m = \frac{R_d}{\Delta T}$, $n = \frac{2\pi}{\Delta\theta}$), response threshold and PSR threshold (which determines the activation of our self-correction mechanism).

3.2 Evaluation of the Self-correction Mechanism

To test the effectiveness and efficiency of the proposed self-correction mechanism, we first selected five well-known correlation based trackers as our base trackers, including SAMF [14], DCF [10], KCF [10], CNT [6], and CSK [9]. We then implemented five variants by applying our self-correction mechanism to these base trackers, namely, SAMF_SC, DCF_SC, KCF_SC, CNT_SC, and CSK_SC.

3.2.1 Overall Performance

Here, we compare these base trackers to the variants enhanced with our self-correction mechanism on the OTB-2013, OTB-2015, and TC-128 datasets. The overall performance are shown in Fig. 3. It can be seen that, by integrating our self-correction mechanism, the precision rate of these trackers has been improved by around 7% in average, and their success rate has been increased by 4.4% in average on the OTB-2013 dataset. Take the DCF_SC tracker for example, its precision rate is 81.6%, which has been improved by 8.8%. Moreover, it can still run at a high speed of 217.7 fps. Although the number of sequences is increased to 100 and more challenging sequences are incorporated in the OTB-2015 dataset, these trackers enhanced by our self-correction mechanism still achieve a remarkable improvement as compared to their corresponding base trackers. Our SAMF_SC tracker obtains the best performance on the OTB-2015 benchmark, with a precision score of 81.2%. It outperforms the state-of-the-art trackers on this benchmark, including SRDCF (78.8%) and MUSTer (77.4%), more results can be found in the supplementary. Besides, the speed of the SAMF_SC tracker is still comparable to the base SAMF, with only a slight decline (15.5 fps vs 18.5 fps). That is mainly because, the base SAMF tracker can achieve a relatively high accuracy and the self-correction module does not need to be activated frequently. Similarly, the tracking performance of all these trackers can also be improved by a large margin on the TC-128 dataset, despite the sequences in this dataset are color sequences. Note that, the improvement in success rate is smaller than the improvement in precision rates. That is because, scale variations are not considered by these trackers except for SAMF.

3.2.2 Performance under Different Attributes

Since OTB-2015 dataset contains sufficient challenging sequences, we conducted experiments on this dataset, the results are shown in Table 2. The 100 sequences in the OTB-2015 dataset were annotated with 11 different 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 clutter (BC), and low resolution (LR). We use the results obtained on different annotated sequences to analyze the performance of our mechanism on sequences with each attribute. It can be seen from Table 2 that, compared to their base trackers, the performance of almost

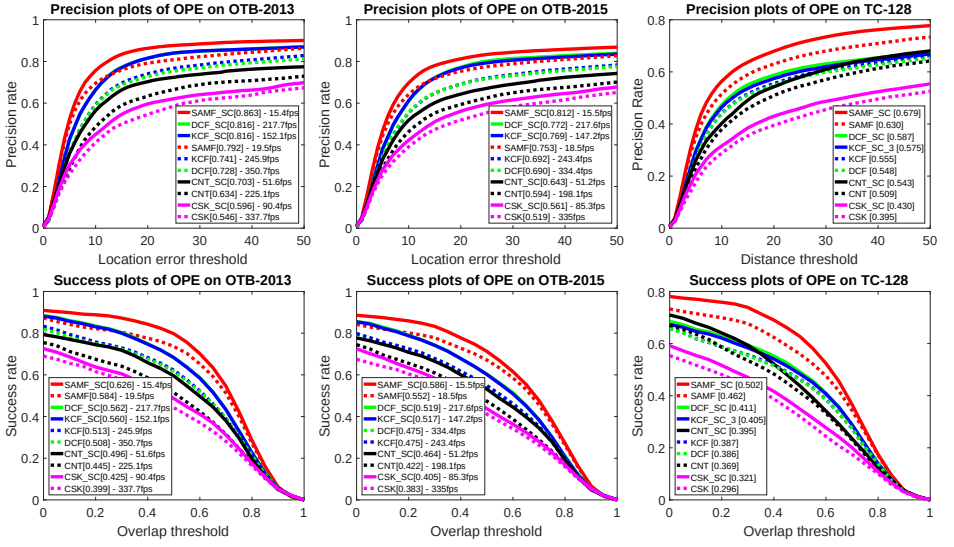

Figure 3: Experiment results achieved by 5 base trackers and their variants enhanced with our self-correction mechanism on the OTB-2013, OTB-2015, and TC-128 datasets.

all enhanced trackers have been improved on sequences with different attributes, except for CSK and CNT on sequences with few attributes. This is mainly because the features used in these two trackers are insufficient to construct a discriminative appearance model, and the response maps produced by these trackers are not reliable to activate our self-correction module. Besides, the overall precision rate of these five trackers has been significantly improved in precision rates (6.2% in average), especially when the object is under fast motion (11.1% in average) and motion blur (9.6% in average), this is mainly because the object localization error under fast motion can be corrected timely by our self-correction mechanism. In addition, since the proposed mechanism can help the trackers to define an accurate search area and prevent tracking drifting, the performance on images with occlusion and out-of-view are also improved remarkably.

3.2.3 Performance under Low Frame Rate Sequences

In addition, we also conducted experiments on the UAV@10fps dataset. These videos in this dataset are generated from the UAV123 dataset by downsampling each video to 10 fps. The movement of an object in adjacent frames in UAV@10fps is larger than that in the UAV123 dataset, that is, objects in the UAV@10fps dataset have fast motion across frames. It is clear from Fig. 4 that, these enhanced trackers achieve an average improvement of 10.6% in precision rate. Note that, CSK_SC with grayscale features even outperforms the sophisticated SAMF tracker in terms of precision rate. Besides, with the support of our self-correction mechanism, the SAMF_SC tracker achieves a comparable performance to the SRDCF tracker (57.5%) in precision rate and outperforms the MUSTer (52.6%) tracker by a large margin. Moreover, the precision rates achieved by KCF_SC and CSK_SC trackers are even better than those produced by their base trackers (52.3% and 48.8%, respectively) on the UAV123 dataset (i.e., without down-sampling). More experimental results on UAV123 can be found in [19]. This clearly show that, with the support of our self-correction mechanism, these CF based trackers can work well on sequences with low frame rate. This means that

Attribute	SAMF_SC	SAMF	KCF_SC	KCF	DCF_SC	DCF	CNT_SC	CNT	CSK_SC	CSK
LR(9)	82.7% ↑(6.00%)	76.7%	75.4% ↑(8.30%)	67.1%	74.7% ↑(5.30%)	69.4%	64.5% ↑(8.30%)	56.2%	61.0% ↑(16.4%)	44.6%
IPR(51)	79.0% ↑(5.90%)	73.1%	76.9% ↑(7.60%)	69.3%	77.1% ↑(8.50%)	68.6%	64.8% ↓(0.20%)	65.0%	56.4% ↑(4.80%)	51.6%
OPR(59)	79.1% ↑(5.30%)	73.8%	75.3% ↑(5.90%)	69.4%	76.3% ↑(7.50%)	68.8%	64.9% ↑(1.60%)	63.3%	54.6% ↑(3.50%)	51.1%
SV(61)	79.2% ↑(7.50%)	71.7%	73.9% ↑(9.80%)	64.1%	73.5% ↑(10.2%)	63.3%	55.8% ↑(0.60%)	55.2%	52.8% ↑(6.40%)	46.4%
OCC(44)	78.9% ↑(6.60%)	72.3%	71.3% ↑(6.20%)	65.1%	72.0% ↑(8.30%)	63.7%	61.9% ↑(1.80%)	60.1%	50.3% ↑(4.50%)	45.8%
DEF(39)	74.7% ↑(8.10%)	66.6%	72.2% ↑(7.30%)	64.9%	73.1% ↑(7.80%)	65.3%	59.4% ↑(1.10%)	58.3%	49.2% ↑(0.30%)	48.9%
BC(31)	76.7% ↑(6.60%)	70.1%	77.7% ↑(6.50%)	71.2%	77.4% ↑(8.80%)	68.6%	58.3% ↓(4.50%)	62.8%	53.0% ↓(4.50%)	57.5%
IV(35)	75.0% ↑(4.90%)	70.1%	75.6% ↑(5.90%)	69.7%	76.9% ↑(8.80%)	68.1%	54.4% ↓(2.30%)	56.7%	46.5% ↓(2.30%)	48.8%
MB(29)	71.4% ↑(5.50%)	65.9%	72.0% ↑(12.0%)	60.0%	70.7% ↑(13.1%)	57.6%	55.3% ↑(6.50%)	48.8%	46.6% ↑(10.9%)	35.7%
FM(37)	79.8% ↑(9.60%)	70.2%	76.2% ↑(13.2%)	63.0%	76.3% ↑(15.2%)	61.1%	60.3% ↑(8.60%)	51.7%	50.3% ↑(8.80%)	41.5%
OV(14)	71.3% ↑(6.00%)	65.3%	62.0% ↑(12.2%)	49.8%	59.9% ↑(11.2%)	48.7%	46.6% ↑(3.00%)	43.6%	33.1% ↑(5.50%)	27.6%
Total(100)	81.2% ↑(5.90%)	75.3%	76.9% ↑(7.70%)	69.2%	77.2% ↑(8.20%)	69.0%	64.3% ↑(4.90%)	59.4%	56.1% ↑(4.20%)	51.9%

Table 2: The precision rate of 5 base trackers and their variants enhanced with our self-correction mechanism under different attributes on the OTB-2015 dataset. The number in each brackets indicates the performance improvement (red) or decline (blue). The first column lists the abbreviations of 11 attributes, the number in each brackets gives the number of videos with this attribute in the OTB-2015 dataset.

some sophisticated trackers with high computational cost are now possible to be applied in real-time scenarios by down-sampling the input sequence and integrating our self-correction mechanism.

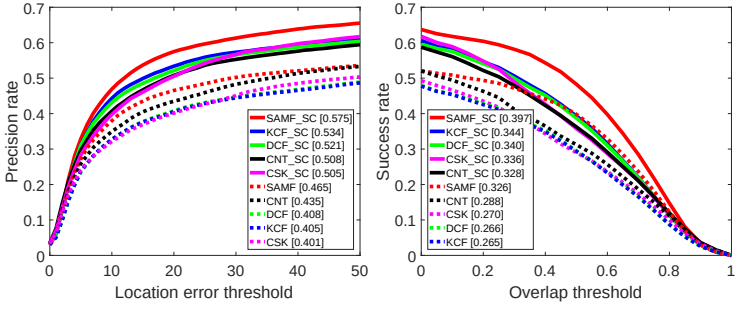

Figure 4: Experiment results of 5 base trackers and their variants enhanced with our self-correction mechanism on the UAV@10fps dataset.

3.3 Comparison to the State-of-the-Art

Finally, we compare our EDCF tracker to 42 state-of-the-art trackers on the OTB-2013 dataset, including 29 traditional trackers presented in [22], MEEM [24] and 12 CF based trackers, such as MUSTer [11], DeepSRDCF [6], LCT [18], SRDCF [7], SAMF_AT [2], RPT [15], Staple [1], SAMF [14], CCT [25], DSST [4], CNT [5], and DCF[10]. The average precision rates and success rates are reported in Fig. 5. It is clear that almost all of the top 12 trackers are CF based trackers except for MEEM. The MUSTer tracker obtains the second best performance, with an average precision score of 86.5% and a success score of 64.1%. It consists of a collaborative short-term store and a long-term store. Our proposed EDCF tracker achieves the best performance among these trackers, with an average precision

score of 87.2% and an average success score of 65.2%. Both of its average precision rate and success rate are improved by 14.4% as compared to the base DCF tracker. Besides, our proposed tracker can run at a speed of 77 fps, it is faster than the DeepSRDCF and MUSTer trackers by several times and is more suitable for real-time applications.

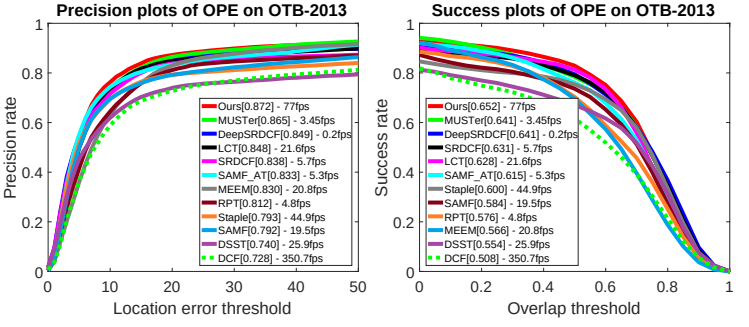

Figure 5: Experiment results of our EDCF tracker and 42 state-of-the-art trackers on the OTB-2013 dataset. For clarity, only the top 12 trackers are shown in the figures.

4 Conclusion

In this paper, we design a self-correction mechanism for CF based trackers by introducing a closed-loop feedback technique. Our mechanism can detect the abnormality in response map and automatically generates a feedback offset to compensate for the localization error. The optimal offset is estimated by searching the most possible location from a set of candidate patches. Extensive experimental results show that trackers equipped with our mechanism can achieve high tracking accuracy and robustness, while maintaining high efficiency. For future work, we aim to incorporate our proposed mechanism with more recent deep trackers, and investigate more effective discrepancy measures.

5 Acknowledgement

This work was partially supported by the National Natural Science Foundation of China (Nos. 61602499 and 61471371), the National Postdoctoral Program for Innovative Talents (No. BX201600172), and China Postdoctoral Science Foundation.

References

[1] Luca Bertinetto, Jack Valmadre, Stuart Golodetz, Ondrej Miksik, and Philip Torr. Staple: Complementary learners for real-time tracking. *arXiv preprint arXiv:1512.01355*, 2015.

[2] Adel Bibi, Matthias Mueller, and Bernard Ghanem. Target response adaptation for correlation filter tracking. In *European Conference on Computer Vision*, pages 419–433. Springer, 2016.

[3] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2544–2550, 2010.

[4] Martin Danelljan, Gustav Häger, Fahad Khan, and Michael Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference, Nottingham, September 1-5, 2014*. BMVA Press, 2014.

[5] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg, and Joost Van de Weijer. Adaptive color attributes for real-time visual tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1090–1097, 2014.

[6] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Convolutional features for correlation filter based visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 58–66, 2015.

[7] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Learning spatially regularized correlation filters for visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4310–4318, 2015.

[8] Martin Danelljan, Gustav Hager, Fahad Shahbaz Khan, and Michael Felsberg. Discriminative scale space tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016.

[9] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *European Conference on Computer Vision*, pages 702–715. Springer, 2012.

[10] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[11] Zhibin Hong, Zhe Chen, Chaohui Wang, Xue Mei, Danil Prokhorov, and Dacheng Tao. Multi-store tracker (MUSTer): A cognitive psychology inspired approach to object tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 749–758, 2015.

[12] Dafei Huang, Lei Luo, Zhaoyun Chen, Mei Wen, and Chunyuan Zhang. Applying detection proposals to visual tracking for scale and aspect ratio adaptability. *International Journal of Computer Vision*, pages 1–18, 2016.

[13] Hamed Kiani Galoogahi, Terence Sim, and Simon Lucey. Correlation filters with limited boundaries. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4630–4638, 2015.

[14] Yang Li and Jianke Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *European Conference on Computer Vision*, pages 254–265, 2014.

[15] Yang Li, Jianke Zhu, and Steven CH Hoi. Reliable patch trackers: Robust visual tracking by exploiting reliable patches. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 353–361, 2015.

[16] Pengpeng Liang, Erik Blasch, and Haibin Ling. Encoding color information for visual tracking: Algorithms and benchmark. *IEEE Transactions on Image Processing*, 24 (12):5630–5644, 2015.

[17] Chao Ma, Jia-Bin Huang, Xiaokang Yang, and Ming-Hsuan Yang. Hierarchical convolutional features for visual tracking. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3074–3082, 2015.

[18] Chao Ma, Xiaokang Yang, Chongyang Zhang, and Ming-Hsuan Yang. Long-term correlation tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5388–5396, 2015.

[19] Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for UAV tracking. In *European Conference on Computer Vision*, pages 445–461. Springer, 2016.

[20] Sayanan Sivaraman and Mohan Manubhai Trivedi. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, 2013.

[21] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[22] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2411–2418, 2013.

[23] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[24] Jianming Zhang, Shugao Ma, and Stan Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In *European Conference on Computer Vision*, pages 188–203. Springer, 2014.

[25] Guibo Zhu, Jinqiao Wang, Yi Wu, and Hanqing Lu. Collaborative correlation tracking. In *British Machine Vision Conference*, pages 184–1. BMVA Press, 2015.