IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 2, 2018, accepted September 1, 2018, date of publication September 24, 2018, date of current version October 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2871659

Adaptive Framework for Robust Visual Tracking

MOHAMED H. ABDELPAKEY[1], (Member, IEEE),
MOHAMED S. SHEHATA[1], (Senior Member, IEEE),
MOSTAFA M. MOHAMED[2,3], AND MINGLUN GONG[4]
[1]Faculty of Engineering and Applied Science, Memorial University of Newfoundland, St. John's, NL A1B 3X5, Canada
[2]Department of Electrical and Computer Engineering, University of Calgary, Calgary, AB T2N 1N4, Canada
[3]Biomedical Engineering Department, Helwan University, Helwan, Egypt
[4]Department of Computer Science, Memorial University of Newfoundland, St. John's, NL A1B 3X5, Canada

Corresponding author: Mohamed H. Abdelpakey (mha241@mun.ca)

This work was supported by the NSERC Discovery, Canada, under Grant 210410.

ABSTRACT Visual tracking is a difficult and challenging problem, for numerous reasons such as small object size, pose angle variations, occlusion, and camera motion. Object tracking has many real-world applications such as surveillance systems, moving organs in medical imaging, and robotics. Traditional tracking methods lack a recovery mechanism that can be used in situations when the tracked objects drift away from ground truth. In this paper, we propose a novel framework for tracking moving objects based on a composite framework and a reporter mechanism. The composite framework tracks moving objects using different trackers and produces pairs of forward/backward tracklets. A robustness score is then calculated for each tracker using its forward/backward tracklet pair to find the most reliable moving object trajectory. The reporter serves as the recovery mechanism to correct the moving object trajectory when the robustness score is very low, mainly using a combination of particle filter and template matching. The proposed framework can handle partial and heavy occlusions; moreover, the structure of the framework enables integration of other user-specific trackers. Extensive experiments on recent benchmarks show that the proposed framework outperforms other current state-of-the-art trackers due to its powerful trajectory analysis and recovery mechanism; the framework improved the area under the curve from 68% to 70.8% on OTB-100 benchmark.

INDEX TERMS Composite framework, trajectory analysis, virtual vectors, reporter, adaptive framework, visual tracking, unmanned aerial vehicle.

I. INTRODUCTION

Object tracking is very important in many applications such as image understanding, robotics, surveillance, and human-computer interaction [1]. In the last decade, the development of satellite and unmanned aerial vehicle (UAV) has significantly increased. Remote sensing videos, especially aerial ones, have been widely used in surveillance because of their ability to provide full coverage of ground areas of interest. However, objects tracking in these videos is very challenging due to many factors including small objects size, illumination changes, pose angle variations, occlusion, background clutter, and camera motion.

Conventional trackers (e.g., [2]–[5]) produce objects trajectories between successive frames using a specific model that updates the objects locations in new frames. Tracking drift is a common problem in tracking objects using conventional trackers, where the new locations estimated by the tracker being used start to drift away from the true objects locations. Tracking drift is more persistent in aerial videos in situations when there is consistent partial or full occlusions.

Generally, object trackers can be categorized into three categories [1], [6]: generative, discriminative, and composite trackers. Generative trackers track objects by searching for the image region that best matches a template or an appearance model [1]. For example, the histogram-based tracking method [7] relies on treating the weighted histogram of the current object image patch as a template, and the mean shift is used as an efficient search strategy. Black and Jepson [8] used a subspace as an offline template and tracked the object under the optical flow framework. He *et al.* [9] created a local sensitive histogram as a template, that is invariant to illumination, and an exhaustive search of the image patch with the similar local histogram is performed in the vicinity of the object. In [10], the distribution field is used to define the probability pixels over a grayscale image to construct a template for the object. Zoidi *et al.* [11] employed the similarity over a color histogram and texture descriptors to locate the target. Generally, the generative trackers discussed above fail in situations when there are occlusions and usually cannot be recovered from tracking drifts.

Discriminative trackers deal with the object tracking problem as a binary classification problem to separate the foreground from the background. Discriminative methods exploit the visual information from the target of interest and the background. Avidan [12] used the support vector machine classifier (SVM) with optical flow. CCOT [13] is based on discrimination correlation filter (DCF) and utilized multi-resolution deep feature map. Henriques *et al.* [14] used an analytic model of correlation filter for datasets of thousands of translated patches to utilize the circulant matrix and diagonlized it with the Discrete Fourier Transform. In [15], a neural network (MDNet) was used with shared layers and multiple branches of domain-specific layers, where each branch is responsible for binary classification and each domain corresponds to the training sequence.

Discriminative trackers have good performance over time, however, similar to generative trackers, they still suffer from the same drift problems when there is frequent occlusions. CFNet traker [16] used Siamese network and integrated a correlation filter layer to the network. In [17], a semantic branch added to Siamese network to capture more robust deep feature of the object of interest. In [18], a deconvolutinal network is used as a learnable upsampling layer to map the low resolution feature to enlarged feature map. Guan *et al.* [19] proposed event-triggered tracking framework, occlusion and drift identification module is used to identify if the drift has occurred. When drift event occurs, the target re-detection module is activated by the event-triggered decision module to recover the target again in short-term tracking. [20] used a lightweight convolutional network of two layers without offline training to extract a set of normalized patches from the target region. The extracted normalized patches are used as filters to integrate a series of adaptive contextual filters surrounding the target to define a set of feature maps in the subsequent frames. Song *et al.* [21] used self-similarity in visual tracking, the target image is divided into non-overlapped patches described by the histogram of gradient (HOG) features. Afterwards, a polynomial kernel feature map is constructed to extract the self-similarity information. A linear support vector machine is used as a classifier. Yang *et al.* [22] proposed a particle filter framework to handle the appearance changes. The framework used online Fisher discrimination boosting feature selection mechanism to enhance the discriminative capability between the target and background. Chen *et al.* [23] used a patch based tracker which adaptively integrates the kernel correlation filters with multiple effective features. The template patch is trained by kernel correlation filtering and particle filter framework and adaptively set the weight of each patch for each particle in a particle filtering framework. Zhang *et al.* [24] proposed a regularized correlation filter (CF) based tracking to capture the long-term spatio-temporally nonlocal superpixel appearance information to regularize the CF learning. Zhang *et al.* [25] proposed a boolean map based representation that exploits connectivity cues for visual tracking. The appearance model is described histogram of oriented gradients and raw color features. Boolean maps form together a target representation that can be approximated by an explicit feature map of the intersection kernel, which is fed into a logistic regression classifier. Song *et al.* [26] proposed a high-dimensional multi-scale spatio-color image feature vector to represent the target object. Afterwards, this feature vector is randomly projected onto a low-dimensional feature space. A feature selection technique is used to design an adaptive appearance model. Zhang *et al.* [27] developed appearance model based on features extracted from a multiscale image feature space with data-independent basis. A non-adaptive random projections is used along with a sparse measurement matrix to extract the features of the appearance model. In most scenarios where drift and significant appearance changes occurs, the above trackers cannot recover the object of interest.

Composite trackers are trackers that combine multiple trackers to track objects. The co-tracking algorithm in [28] used a support vector machine classifier to train with multiple different features and compined their results. The MEEM algorithm [3] used multiple trackers to memorize their past states, so that the tracker can ignore false positive. The unifying algorithm [29] used the relation among individual trackers by measuring the consistency of each tracker between two successive frames and the pair-wise correlation among different trackers. Kwon and Lee [30] decomposed the appearance model into multiple observation models and motion models (VTD) and exploited the results in a unifying tracker within a Baysian framework. In [31], a Struck tracker [32] was used based on three different feature descriptors; Haar-like features to represent texture information of a target object, color histograms to consider the local color distribution of the target object, and illumination invariant feature. In [33], a refined trajectory of an object is obtained by combining the trajectories of other conventional trackers in a benchmark. Kwon and Lee [34] used a tracker space, with multiple trackers, and adaptively sampled to run one at a time. Adaptive NormalHedge algorithm [35] proposed an adaptive framework based on a decision-theoretic online learning algorithm called NormalHedge. Adaptive NormalHedge used a set of weighted experts to predict the state of the target and overcomes the fixed percentage factor that is used in the standard NormalHedge. The HDT tracker [36] took feature maps from different CNN layers and used the parameter-free Hedge algorithm [37] to hedge multiple CNN-based trackers into a strong tracker. Qi *et al.* [38] proposed a CNN-based tracker to hedge deep features from different layers in the network. The correlation filter is applied to each feature maps from different layers to build up weak trackers which can be hedged into a strong tracker.

Generally, composite trackers will have a pre-set of trackers that can handle different scenarios but they cannot be extended or generalized. Therefore, their ability to handle challenging aerial videos with frequent occlusions and pose changes depend on the individual trackers performance. In most cases, this combination of challenges present in aerial videos causes tracking drift even in composite trackers

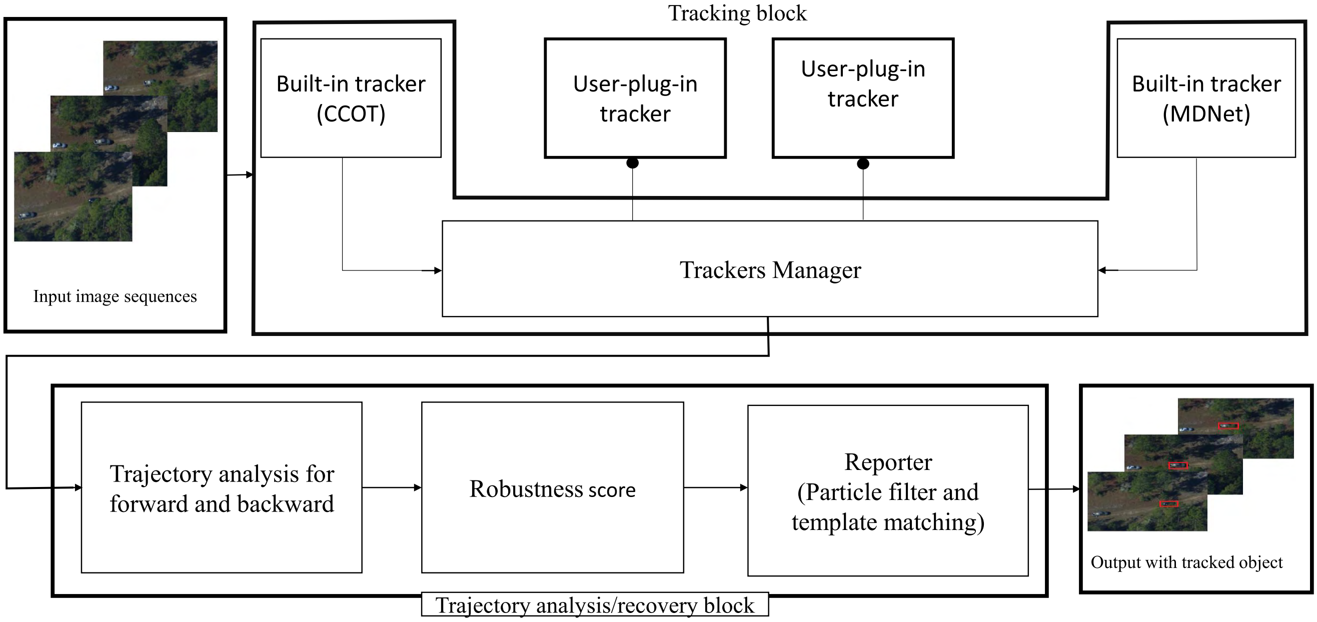

FIGURE 1. **Composite framework with the user-plugin trackers and reporter.**

which lack a mechanism to recover and correct the objects trajectories.

In this paper, we present an effective tracking framework that has the ability to track objects in challenging aerial videos. The proposed framework has the following novelties:

1) The framework contains two built-in trackers (MDNet [15] and CCOT [13]) and optional user trackers plugins that can be used by the user to include additional trackers in the framework to handle specific scenarios. This allows more flexibility and generalize the framework to be efficient in different applications.
2) A new mechanism, called the *reporter*, that intervene when there is a tracking drift and correct the object trajectory.
3) A new metric was developed, the virtual vector shown in Figure 7 to be combined with trajectory analysis to calculate a more accurate robustness score.

The rest of this paper is organized as follows. Section II details the proposed framework. In section III, we present the experimental results obtained on two UAV data-sets and compare them with current state-of-the-art relevant tracking algorithms. Finally conclusion and future work are provided in section IV.

II. PROPOSED FRAMEWORK

In this section, we present the proposed framework as shown in Figure 1. The framework mainly consists of two blocks: 1) the tracking block and 2) the trajectory analysis and recovery block. In the first block, a trackers manager controls, manage, and compile the results from the different built-in and user-plugin trackers, more explanation in next subsection. The second block consists of three steps: 1) the trajectory analysis of forward/backward tracklets, 2) a robustness score calculation, and 3) a reporter mechanism. Figure 1 shows the block diagram of the proposed framework. Initially the target is selected then, the framework is initialized. The trajectory analysis and recovery block will be explained in sub-section B. Each tracker in the framework is executed from frame$_{t-n}$ to frame$_t$ to get the forward trajectory and then, from frame$_t$ to frame$_{t-n}$to get the backward trajectory.

A. THE TRACKING BLOCK

Tracklet is a trajectory within a short period of time, it will be used throughout this paper. The input to this block is the video frames that contain the initial location of the target. The first tracker gives the forward trajectory-1 and backward trajectory-1, while the second tracker gives the forward trajectory-2 and backward trajectory-2. Both trackers work simultaneously to track objects. The optional user trackers plugins are added by user to include additional trackers to handle different scenarios. The tracking block outputs the location trajectories of the target overtime which will be delivered to the second block (trajectory analysis/recovery block) through the trackers manager.

B. THE TRAJECTORY ANALYSIS AND RECOVERY BLOCK

The second block receives the trackers results from the trackers manager and process them through trajectory analysis robustness score, and finally the reporter.

1) TRAJECTORY ANALYSIS AND ROBUSTNESS FOR FORWARD AND BACKWARD

The trajectory is the positions of the center of the bounding boxes through the tracking process. Suppose we have a set of frames, we will denote the first frame by frame$_{\text{t-n}}$ and the last frame by frame$_{\text{t}}$; where n is any number of frames. Suppose we do not have user-defined trackers, therefore, the composite framework consists of the first trackers that is CCOT and the second one is MD-Net. The framework is executed in two directions, forward trajectory for both trackers; from frame$_{\text{t-n}}$ to frame$_{\text{t}}$ (T$_{\text{t-1f}}$ and T$_{\text{t-2f}}$). The outcome of this execution is two forward trajectories. Another execution at the same time is made in reverse direction from frame$_{\text{t}}$ to frame$_{\text{t-n}}$ (T$_{\text{t-1b}}$ and T$_{\text{t-2b}}$); the outcome of this execution is two backward trajectories. Figure 7 shows the forward and backward trajectories, when they are cyclic, the robustness score will be very close to 1 and that indicates the current tracking result is very accurate and when they are non-cyclic the robustness score has a large value. To the end, we have two forward trajectories and two backward trajectories; in other words, a pair of forward trajectories and a pair of backward trajectories; the first pair trajectories from CCOT tracker and the second pair trajectories from MD-Net tracker. The robustness score is measured for each pair (forward and backward) trajectories; the trajectories with the highest robustness score are selected as the final trajectories. Consequently, the forward trajectory is the best choice to advance the tracking process within our adaptive framework.

In the trajectory analysis and robustness score, we use the geometric similarity, the cyclic weight, the appearance similarity, and the cosine similarity between virtual vectors. Virtual vectors are developed to calculate the angle between forward and backward trajectories through virtual vectors starting from the ending position and ending at the starting position as shown in Figure 7. We develop the virtual vector measure to get more accurate results in terms of robustness score. In Figure 7 we assume there are two virtual vectors start where the end of forward trajectory result and they end where the initial of the bounding box is located and the location of the object after backward analysis. Consequently, an angle between the two virtual vectors is called θ. A small θ indicates that, the initial location of the target object and the ending location of the object after backward tracking are very close to each other or might be identical. Thus we employ the cosine similarity to measure the angel between the paired virtual vectors as follows

$$Cos(\theta_{\text{t}}) = \frac{\vec{x}_{\text{t0:29}} \cdot \vec{x}_{\text{t29:t0}}}{\|\vec{x}_{\text{t0:29}}\| \, \|\vec{x}_{\text{t29:t0}}\|} \tag{1}$$

Suppose we have a video sequence of 30 frames, let $\vec{x}_{\text{t}}$ denotes the bounding box location at frame t, which is estimated by the built-in tracker-1 in the forward direction. The forward trajectory from t_0 to t_{29} can be described as follows

$$\vec{x}_{\text{t}_0:\text{t}_{29}} = \{\vec{x}_{\text{t}_0}, \vec{x}_{\text{t}_1}, \ldots, \vec{x}_{\text{t}_{29}}\} \tag{2}$$

where $\vec{x}_{\text{t}_0}$ is the bounding box position at the initial frame, $\vec{x}_{\text{t}_{29}}$ is the bounding box at the last frame. Similarly, The built-in tracker is initialized to trace the target object in the backward direction. The backward trajectory can be described as follows

$$\vec{x}_{\text{t}_{29}:\text{t}_0} = \{\vec{x}_{\text{t}_{29}}, \vec{x}_{\text{t}_{28}}, \ldots, \vec{x}_{\text{t}_0}\} \tag{3}$$

Similarly, the built-in tracker-2 is described in the same way. The geometric similarity can be computed from:

$$\lambda_{\text{t}} = \exp(-\frac{\|\vec{x}_{\text{t}_0:\text{t}_{29}} - \vec{x}_{\text{t}_{29}:\text{t}_0}\|^2}{\|\sigma^2\|}) \tag{4}$$

Where, λ is the geometric similarity and σ^2 is an empirically determined value equals to 500. When the difference between the trajectories is very small, the exponential gives a number very close to 1 and vice versa. In the ideal case, the forward and backward trajectories are identical therefore, the geometric similarity λ equals to 1. This equation will be used later to calculate Eq. 6.

In Figure 7, at cyclic virtual vectors block the blue trajectory is matched or very close to the red trajectory, therefore this trajectory is selected as a valid trajectory and called cyclic. In such a case the geometric similarity is very close to 1. In contrast, at non-cyclic virtual vectors block the blue forward trajectory does not match the red backward trajectory therefore, we discard this trajectory and we call it non-cyclic, because the initial object can not be accessed again from the backward direction. In such a case the geometric similarity decreases. To calculate the cyclic weight, we count the number of mismatched bounding boxes in the forward trajectory with their correspondences in the backward trajectory from the intersection over union (IoU) as follows:

$$\psi = \frac{\Delta(\vec{x}_{\text{t}_0:\text{t}_{29}}, \vec{x}_{\text{t}_{29}:\text{t}_0})}{\Delta(\vec{x}_{\text{t}_0:\text{t}_{29}}) + \Delta(\vec{x}_{\text{t}_{29}:\text{t}_0})} \tag{5}$$

Where the numerator $\Delta(\vec{x}_{\text{t}_0:\text{t}_{29}})$ and $\Delta(\vec{x}_{\text{t}_{29}:\text{t}_0})$ is the area of the bounding boxes overlap in the forward and backward trajectories. The denumerator is the area of the bounding boxes union in the forward and backward trajectories. Practically, we do not need to count the number of mismatched frames ν in the whole period; we consider the first four frames in the forward trajectory which corresponds to the last four frames in the backward trajectory. If the ψ is less than 0.33, a mismatch will be declared. Consequently, the forward and backward trajectories form a non-cyclic. To assign a weight to the cyclic or non-cyclic trajectories, we will use X where X is an arbitrary number to set the cyclic weight bases quit differently. If the number of mismatched frames ν is 0 or 1 within the the first four frames in the forward trajectory, the cyclic weight will be 10^5 otherwise it will be 1.

Now assume we have $\vec{x}_{\text{t}_0:\text{t}_{29}}$ and $\vec{x}_{\text{t}_{29}:\text{t}_0}$ from the composite framework. The first four frames in the forward trajectory will be denoted by $\vec{x}_{\text{t}}$ where $t = \{0, 1, 2, 3\}$ and its correspondence from the backward trajectory is $\overleftarrow{x}_{\text{t}}$ where $t = \{29, 28, 27, 26\}$.

Let $P(\overleftarrow{x}_{\text{t}})$ denotes the image patch centered at x position at frame t in the backward trajectory and $S_{\text{t}_0:\text{t}_3}$ denotes the first set of four patches in the forward trajectory as shown in Figure 2. The appearance similarity of $P(\overleftarrow{x}_{\text{t}})$ to the set $S_{\text{t}_0:\text{t}_3}$ can be calculated from Gaussian kernel between the set and

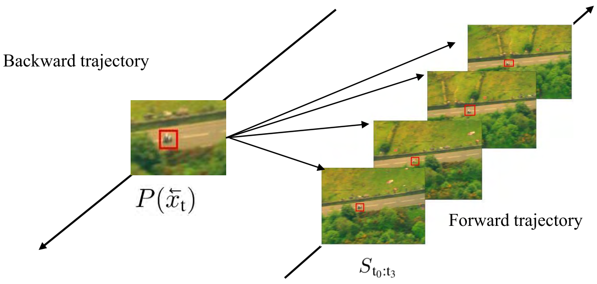

FIGURE 2. **Left: the backward trajectory and the centered patch, right: the forward trajectory and the first four frames (set).**

the patch. In general Gaussian kernel used to measure the similarity between two vectors or 2-D matrix. The appearance similarity can then be calculated as follows:

$$\phi_t = \exp\left(-\frac{\sum_{Q\in S_{t_0:t_3}} \|K\cdot(P(\overleftarrow{x_t})-Q)\|^2}{4\omega h\sigma_2^2}\right) \tag{6}$$

Where ϕ_t is the appearance similarity, $\sigma_2^2 = 900$ empirically determined, ω and h are the width and height of the bounding box, respectively. K is a Gaussian weight mask as shown in 5, and "$\cdot$" is the element-wise weight multiplication. Small ϕ_t indicates high changes in the bounding box appearance or a tracking error.

The large robustness score is, the more tracking results being accurate. We set the robustness score threshold to be 0.65, if one or both trackers scores are greater than the predefined threshold, then the framework will select the highest score. If both trackers scores are less than the predefined threshold, then the framework will active the reporter mechanism.

When the forward and backward trajectories are identical the similarity becomes 1. If the trajectories are not identical the similarity decreases. Finally, the robustness score for the composite framework can be calculated from :

$$\mu_1 = X\left(\sum_{t=t0}^{29} \lambda_t\cdot\phi_t\cdot\cos\theta_t\right) \tag{7}$$

Similarly for the built-in tracker-2 μ_2. The normalized score is required to compare both scores to each other. the normalized robustness score will be calculated as follows

$$\hat{\mu}_1 = \frac{\mu_1}{\mu_1+\mu_2} \tag{8}$$

$$\hat{\mu}_2 = \frac{\mu_2}{\mu_1+\mu_2} \tag{9}$$

Maximum score represents the best trajectory, as it tells how similar the forward and backward trajectories are to each other. The more similar the trajectories are to each other, the higher the value of robustness score.

2) THE REPORTER

The proposed reporter mechanism as shown in algorithm 1 consists of the particle filter and the template matching.

Algorithm 1 Reporter Mechanism Algorithm

```
Input : $\mu$: robustness score, $n$: number of particles.
Initialization : initialize particle filter with $n$ particles.
Precondition : If $\hat{\mu}_1$ & $\hat{\mu}_2 ... < 0.65$ goto :1 else get the score from Eq. 8 and 9.
Output : Recover the lost location of the target object.
1: foreach particle in next frame do:
2:     create bounding boxes around the $n$ particles
3:     update $n$ using linear motion model
4:     Measure the similarity among the the object in the previous frame and the object in the next frame using the template matching.
5:     If the matching score $< 0.50$, then the object still occluded/lost, go to 1 if no go to 6
6:     The highest score with the particle which associated the bounding box is the most likely the lost location.
7:     The recovered location and its bounding box is fed into the composite tracker
8: end
```

It only works when the robustness score is less than the predefined threshold which is 0.65 in our framework through this paper. When the the robustness score is less than the threshold, the particle filter will be initialized by 300 particles around the center of object, each particle is associated with five states as shown in Eq. 10. Therefore, the particle filter updates the states using the linear motion model of the previous object (which the object in frame$_{\text{t-n}}$) to the future states.

$$States = (x_t, y_t, s_t, \alpha_t, \theta_t) \tag{10}$$

where x_t, y_t, s_t, α_t, θ_t are x, y translations, scale, aspect ratio, and in-plane rotation angle respectively. At each particle, a bounding box is created around the location of particle; the size of the bounding box is 36 × 36 pixels since we work on very tiny objects in UAV. Afterward, template matching is used to calculate the similarity among the object in the previous frame$_{\text{t-n}}$ and all bounding boxes where all particles are located. The highest score of template matching is, the most likely the location of object is correct in the frame$_{\text{t}}$. Now the recovered location of the target object will be the input to composite trackers. Finally the framework takes the input images and calculates the forward/backward trajectories by the tracking block which has two trackers. The analysis of these trajectories is done in trajectory analysis and recovery block to give the final result which is the location of the object of interest.

III. EXPERIMENTAL RESULTS

In this section we will provide the parameters that were used in our experiment to make this approach reproducible with the same results. Also a qualitative discussion is provided by the end of this section. Each centered image of the target object is re-sized to be a 36 × 36 patch. To initialize the particle filter, 300 particles were used. The frame numbers n is set to $n = 30$

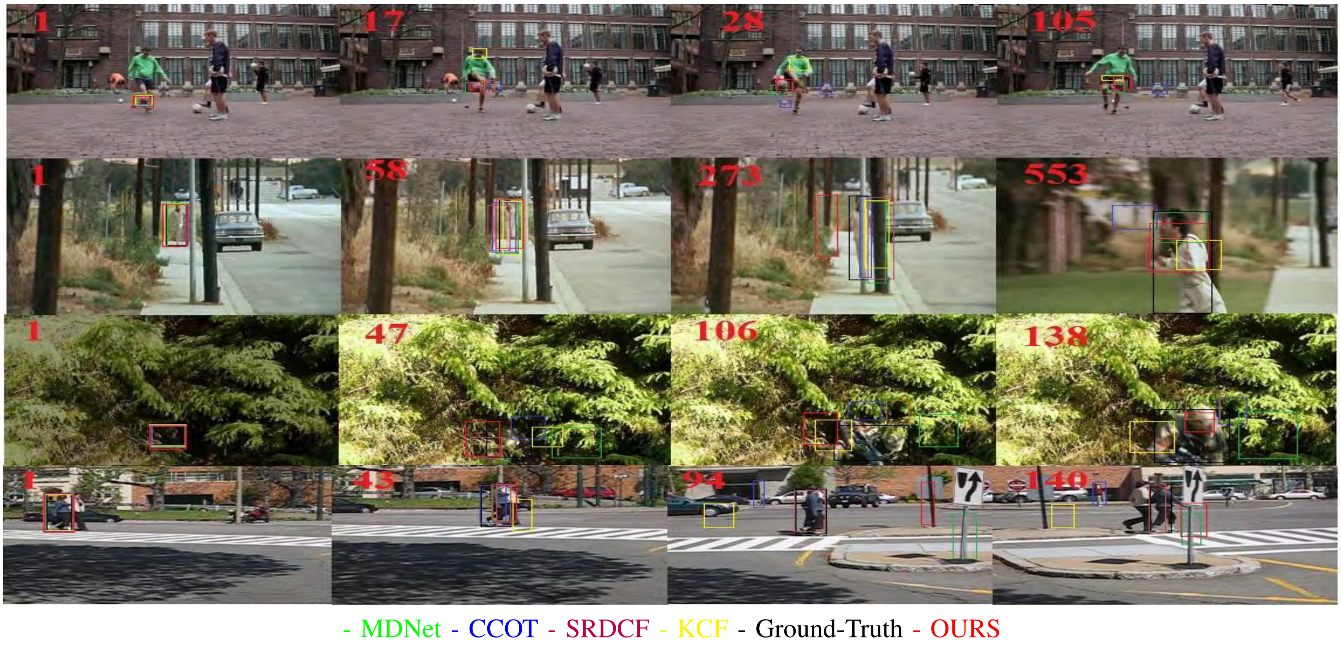

FIGURE 3. Visual results on VOT2016 data-set for four sequences.

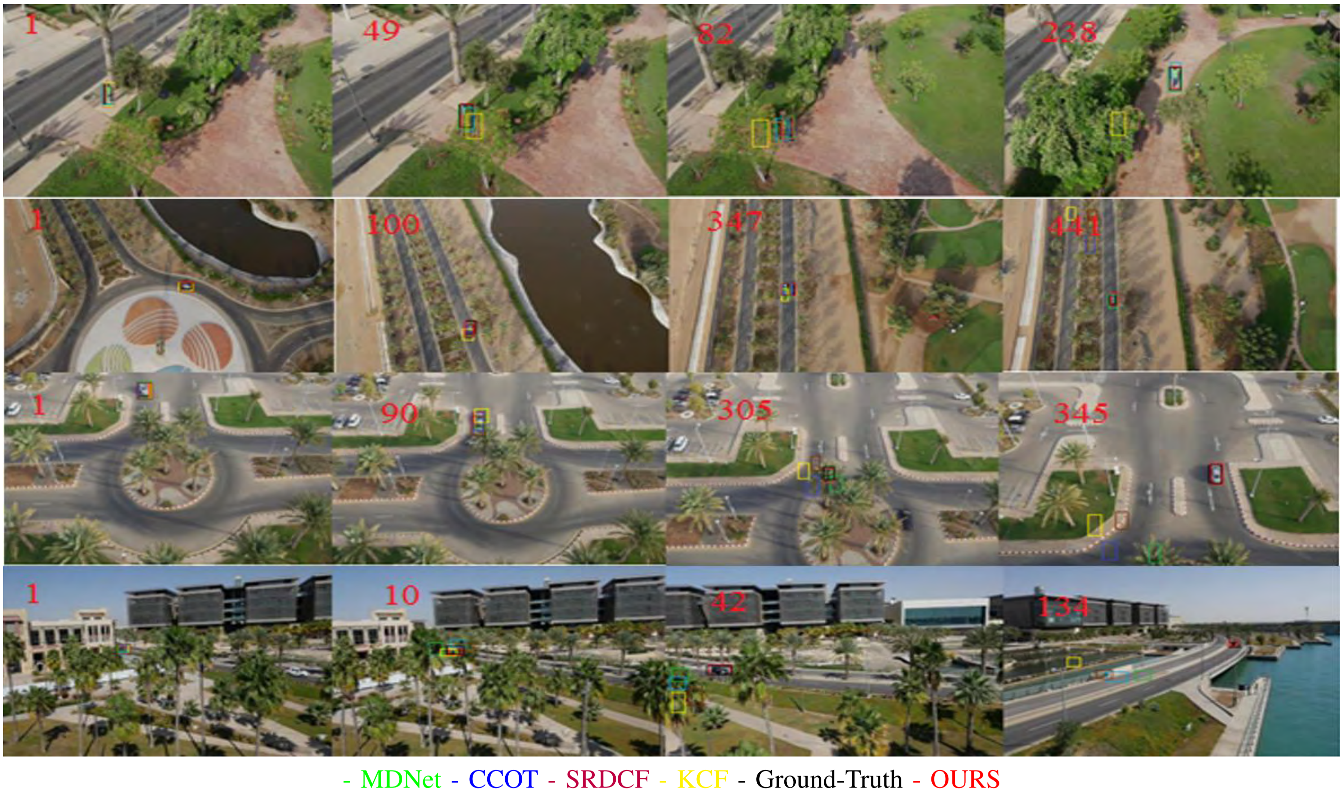

FIGURE 4. Visual results on UAV123 data-set for four sequences.

The implementation was performed using MATLAB-2017b, a computer with Core I7 CPU, 2.1 GHz processor with TITAN XP GPU, 64-GB RAM and no code optimization. We used VOT2016 [39], Vivid [40] and UAV123 [41] datasets to evaluate our proposed framework as shown in Figure 11. The overlap can be calculated as follows $S = \frac{r_t \cap r_a}{r_t \cup r_a}$, where r is the bounding box, $\cap$, and $\cup$ are the intersection and union of two bounding boxes, respectively.

TABLE 1. Overlap rate and the average time in each frame against state-of-the-art trackers on VIVID-EGTest and VOT data-sets.

Trackers	Sequences													
	01		02		03		04		05		VOT2016Road		Overall	
	OVR	Time(s)	OVR	Time(s)	OVR	Time(s)	OVR	Time(s)	OVR	Time(s)	OVR	Time(s)	OVR	Time(s)
MDNet	0.71	1	0.69	1.1	0.82	1.01	**0.81**	1	**0.84**	1.07	**0.78**	1.09	0.78	1.04
CCOT	0.79	0.05	**0.74**	0.06	**0.83**	0.05	0.77	0.08	0.83	0.07	0.77	0.089	**0.79**	0.07
SRDCF	**0.80**	0.2	0.62	0.2	0.79	0.8	0.73	0.3	0.79	0.7	**0.78**	0.88	0.75	0.51
KCF	0.72	0.005	0.65	0.002	0.77	0.008	0.76	0.005	0.76	0.007	0.75	0.008	0.73	0.006
MTA	0.73	0.08	0.69	0.10	0.83	0.09	0.73	0.07	0.82	0.12	0.80	0.08	0.77	0.09
OURS	**0.87**	0.25	**0.76**	0.22	**0.86**	0.17	**0.88**	1.23	**0.89**	1.20	**0.80**	1.24	**0.84**	0.72

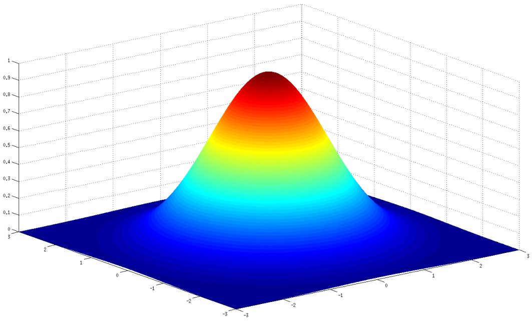

FIGURE 5. Normalized kernel Gaussian mask (K).

TABLE 2. Comparison with the state-of-the-art trackers on VOT2015.

Tracker	A	R	EAO	FPS
MDNet	0.60	0.69	0.38	1
DeepSRDCF	0.56	1.05	0.32	< 1
EBT	0.47	1.02	0.31	4.4
SRDCF	0.56	1.24	0.2	5
BACF	0.59	1.56	–	35
EAST	0.57	1.03	0.34	159
Staple	0.57	1.39	0.30	80
SamFC	0.55	1.58	0.29	86
Ours	0.612	0.67	0.39	0.72

TABLE 3. Comparison with the state-of-the-art trackers on VOT2016.

Tracker	A	R	EAO	FPS
ECOhc	0.54	1.19	0.3221	60
Staple	0.54	1.42	0.2952	80
STAPLE+	0.55	1.31	0.2862	> 25
SiamRN	0.55	1.36	0.2766	> 25
GCF	0.51	1.57	0.2179	> 25
Ours	0.55	1.15	0.3308	0.72

TABLE 4. Comparison with the state-of-the-art trackers on VOT2017.

Tracker	A	R	EAO	FPS
SiamDCF	0.500	0.473	0.249	60
ECOhc	0.494	0.435	0.238	60
CSRDCF++	0.453	0.370	0.229	> 25
SiamFC	0.502	0.585	0.188	86
SAPKLTF	0.482	0.581	0.184	> 25
Staple	0.530	0.688	0.169	> 80
ASMS	0.494	0.623	0.169	> 25
Ours	0.540	0.370	0.250	0.72

Based on ground truth data, Figure 3 visually compares the tracking results obtained using different state-of-the-art trackers against the proposed tracker. It shows that our approach is more robust and handles most of four sequences well although very fast motion in row 1 or occlusion in row 2 except the row 3 where mismatch occurs at frame106 in VOT2016 dataset [39]. In this sequence, the object is diffused with background; in such case the robustness score declares that the forward and backward trajectories are not similar or the object has been lost. The reporter starts to work and the particle filter will create 300 particles and its corresponding patches. Then the reporter mechanism successes to recover the object in the next frames. whereas many existing trackers have errors propagated. Also in row 4, frame #10 our approach reports that the robustness score is 0.22 which is less than the predefined threshold; In this case the reporter starts to work and it will create the 300-particle and unfortunately none of them is overlapped therefore, the framework reports that, the object is lost. we sample these particles randomly

TABLE 5. Comparison of state-of-the-art trackers on OTB-50 and OTB-100 without.

	Tracker	Ours	MDNet	CCOT	LMCF	CFNet	Staple	PTAV	SiamFC	ECOhc
OTB-50	AUC	**0.669**	0.645	0.614	0.533	0.530	0.507	0.581	0.516	0.592
	Prec.	**0.936**	0.890	0.843	0.730	0.702	0.684	0.806	0.692	0.814
OTB-100	AUC	**0.708**	0.678	0.671	0.580	0.568	0.578	0.635	0.582	0.643
	Prec.	**0.930**	0.909	0.898	0.789	0.748	0.784	0.849	0.771	0.856
Speed	FPS	0.86	1	0.3	85	75	80	25	86	60

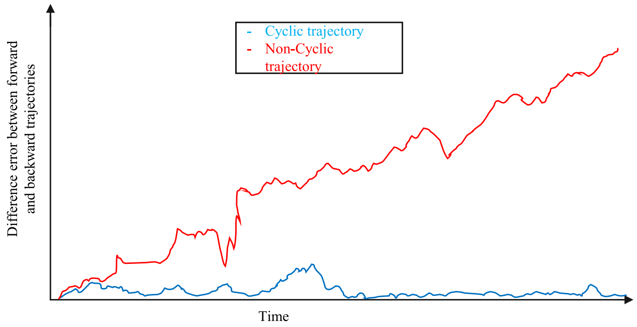

FIGURE 6. Cyclic and non-cyclic trajectories.

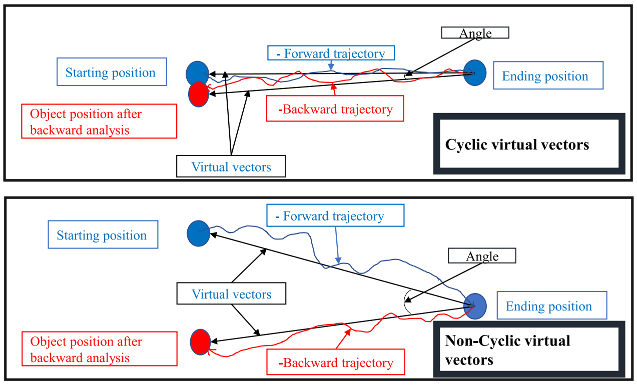

FIGURE 7. Virtual vectors representation.

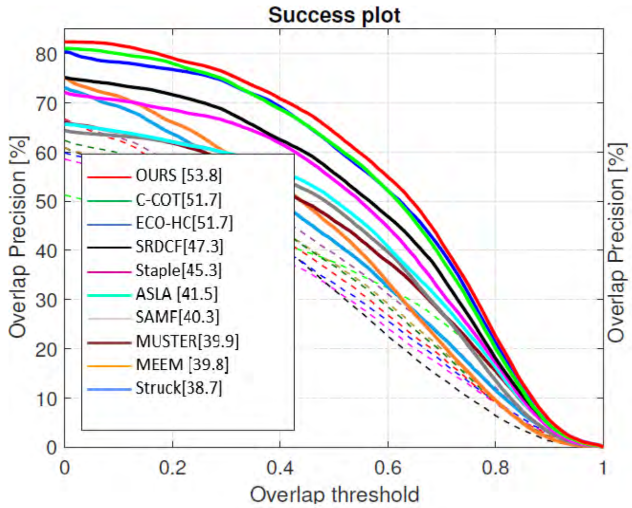

FIGURE 8. Success plot on UAV123 for top 10 trackers. Legend shows AUC.

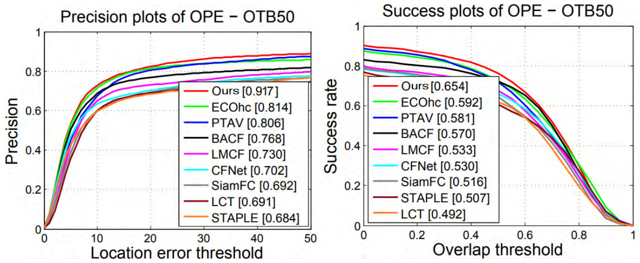

FIGURE 9. Precision and success plots on OTB-50 benchmark. All curves and numbers are generated from OTB toolkit.

from Gaussian distribution however, this case is very rare since the particles are randomly distributed around the object.

Table 1 summarizes the results in terms of overlap rates and the time spent in each frame for the five sequences in VIVId-EGTest dataset and VOT-2016 road sequence. It confirms that our approach is able to properly track the object with lowest tracking errors.

Figure 4 shows the visual results on UAV123 [41] dataset, we ran and evaluated 9 trackers in addition to ours on UAV123 data-set using success plot [33], and calculated the percentage of frames that is within a threshold with an intersection-over-union (IOU). Figure 8 ranks trackers acoording to their area-under-curve (AUC) score. CCOT runs at (0.30 FPS) and its AUC is 51.7%. Our tracker runs at (0.86 FPS) with an AUC score of 53.8% which is outperforming CCOT by 2.1%. At the first row all trackers fail to track the object of interest due to occlusion except CCOT tracker and ours. Moreover, when the object undergoes full occlusion such as row #3 all trackers drift off. However, our tracker still can track the object, in this case the reporter mechanism works and particle filter starts to propagate the particles to find the most similar patch to the object of interest. Figure 11 shows the results on VIVID-EGTest data-set and Vot2016-road sequenc. The first row shows that the object undergoes full occlusion in frame #17, all trackers fail to track the object however, our tracker can find the object after occlusion. We ran our tracker on VIDI-EGTest data-set, row #2 to row #6 show different challenges such as occlusion (row #2), very tiny objects (row #3), sudden discontinuity (jump forward) (row #4), illumination changes (row #5) and frequent occlusion by trees(row #6). Our tracker can handle all these scenarios compared to the other tracker.

TABLE 6. Ablation study of performance evaluation for adding user-plug-in trackers (ECOhc and SiamFC) to the propsed framework.

Proposed Tracker					Dataset				Speed
Built-in	User-plug-in		Components		OTB-50		OTB-100		FPS
Baseline	ECOhc	SiamFC	Reporter	Virtual-vector	AUC	Prec.	AUC	Prec.	-
✓					0.646	0.899	0.680	0.908	0.8926
✓			✓		0.650	0.909	0.688	0.910	0.8906
✓				✓	0.651	0.910	0.687	0.909	0.8920
✓			✓	✓	0.654	0.917	0.693	0.915	0.8898
✓	✓				0.649	0.909	0.689	0.913	0.8759
✓	✓		✓		0.653	0.912	0.695	0.917	0.8739
✓	✓			✓	0.655	0.923	0.693	0.916	0.8753
✓	✓		✓	✓	0.660	0.931	0.708	0.923	0.8730
✓	✓	✓			0.654	0.924	0.692	0.915	0.8590
✓	✓	✓	✓		0.657	0.928	0.698	0.922	0.8571
✓	✓	✓		✓	0.656	0.927	0.700	0.924	0.8584
✓	✓	✓	✓	✓	0.669	0.936	0.708	0.930	0.8563

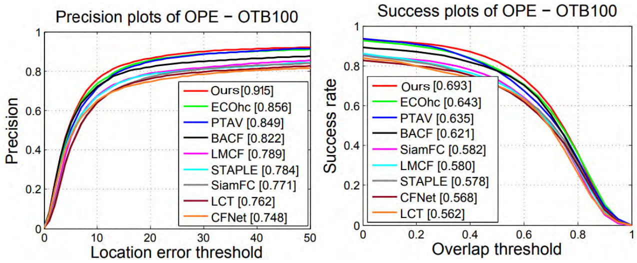

FIGURE 10. Precision and success plots on OTB-100 benchmark. All curves and numbers are generated from OTB toolkit.

Extensive experiments are conducted to evaluate our tracker against the state-of-the-art trackers on OTB-50, OTB-100, VOT2015, VOT2016 and VOT2017 benchmarks. All experiments in this section were done using only the built-in trackers except the experiment in Table 6. Table 2 shows the performance of our tracker against eight state-of-the-art trackers in terms of accuracy (A), robustness score(R) and expected average overlap (EAO). The first four trackers in Table 2 are non-real-time while other trackers are working in real-time. In terms of accuracy the proposed framework is outperforming other trackers especially MDNet by 1.2%. The robustness of the proposed framework is the best compared to all other trackers, the gap between the second best tracker (MDNet) is 2%. Consequently, the expected average overlap for the proposed framework has increased compared to MDNet by 1%. Table 3 and Table 4 show the performance of the proposed framework against five and eight state-of-the-art trackers respectively. The proposed framework outperforms all other trackers. Figure 5 and Figure 10 show the precision and area-under-curve (AUC) on OTB-50 and OTB-100 respectively, all curves are generated from OTB toolkit. Table 5 shows that the proposed framework outperforms all listed trackers on OTB-50 and OTB100.

The proposed framework outperforms other methods because it relies on the trajectory analysis from each tracker.

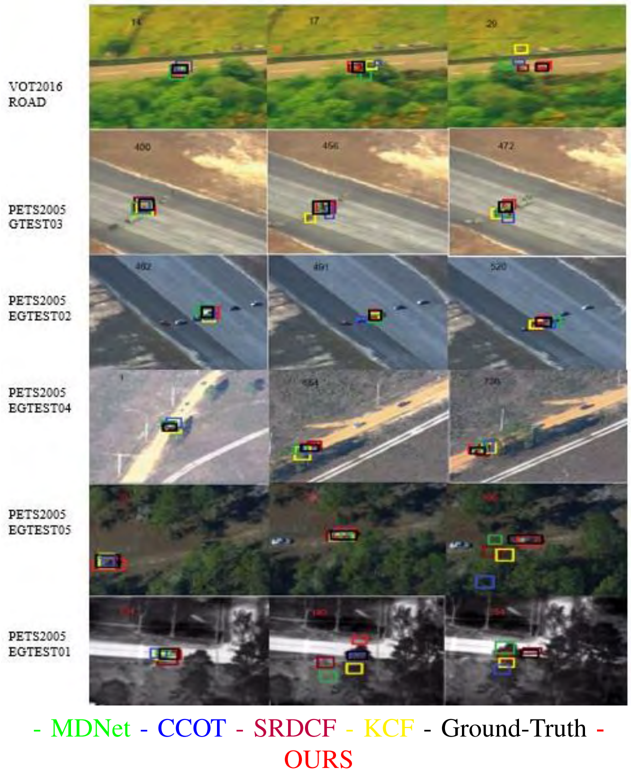

FIGURE 11. Visual results on two data-sets VOT2016-Road and VIVId-EGTest.

Th framework chooses the best trajectory pair (forward and backward) based on the highest score of robustness. On top of that, in case all trackers drift off, the framework detects that the object is lost and the reporter mechanism starts to work by creating 300 particles to find the most similar patch to the object of interest. The performance of the proposed framework on OTB benchmark better than VOT benchmark by 10% since the VOT has very challenging sequences. Figure 3 shows at the first row a very challenging sequence, the object (ball) moves very fast also in Road-sequence the object undergoes a full occlusion. However, the proposed tracker outperforms the state-of-the-art trackers.

A. ABLATION STUDY

In this experiment, we show the effect of adding more trackers (user-plug-in) to the framework. In addition, we show the effect of framework variation components such as the reporter and virtual vector. Table 6 lists the variation components of the proposed framework. Baseline means using only the framework with built-in trackers without the reporter and virtual vector. The first row shows that, using only the baseline hardly improves the performance. The second row shows that, adding the reporter to baseline improves the overall performance. This confirms the importance of the reporter. In the third row, adding virtual vector to baseline without the reporter improves the performance as reporter with almost the same performance. In the fourth row, adding the reporter and virtual vector to the framework improve the overall performance. Table 6 also lists the tested user-plug-in trackers(ECOhc and SiamFC). Obviously, adding user-plug-in trackers with reporter and virtual vector significantly improve the overall performance.

IV. CONCLUSION

In this paper, a composite framework for unmanned vehicle tracking is presented. The composite framework consists of two trackers with trajectory analysis and virtual vectors. The composite framework uses the forward and backward trajectories. A new mechanism called reporter is used to make the tracker more robust. The reporter uses the robustness score and particle filter to decide which trajectory will be selected from the forward pairs.

Extensive experiments were conducted on OTB-50, OTB-100, UAV123, VOT2015, VOT2016 and VOT2017. The experiments have shown that, adding user-plugins, reporter and virtual vector to the robustness score increased the robustness of the proposed framework. Future work includes a deep convolutionl reporter within the composite framework and using the moving horizon estimation instead of particle filter.

REFERENCES

[1] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, p. 13, 2006.

[2] 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.

[3] J. Zhang, S. Ma, and S. Sclaroff, "MEEM: Robust tracking via multiple experts using entropy minimization," in *Proc. Eur. Conf. Comput. Vis.*. Cham, Switzerland: Springer, 2014, pp. 188–203.

[4] 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 *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 749–758.

[5] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proc. Brit. Mach. Vis. Conf.*, Sep. 2014.

[6] Q. Wang, F. Chen, W. Xu, and M.-H. Yang, "Object tracking via partial least squares analysis," *IEEE Trans. Image Process.*, vol. 21, no. 10, pp. 4454–4465, Oct. 2012.

[7] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–577, May 2003.

[8] M. J. Black and A. D. Jepson, "EigenTracking: Robust matching and tracking of articulated objects using a view-based representation," *Int. J. Comput. Vis.*, vol. 26, no. 1, pp. 63–84, 1998.

[9] S. He, Q. Yang, R. W. H. Lau, J. Wang, and M.-H. Yang, "Visual tracking via locality sensitive histograms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2427–2434.

[10] L. Sevilla-Lara and E. Learned-Miller, "Distribution fields for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2012, pp. 1910–1917.

[11] O. Zoidi, A. Tefas, and I. Pitas, "Visual object tracking based on local steering kernels and color histograms," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 23, no. 5, pp. 870–882, May 2013.

[12] S. Avidan, "Support vector tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 8, pp. 1064–1072, Aug. 2004.

[13] 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.* Cham, Switzerland: Springer, 2016, pp. 472–488.

[14] 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.

[15] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4293–4302.

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

[17] A. He, C. Luo, X. Tian, and W. Zeng. (2018). "A twofold siamese network for real-time object tracking." [Online]. Available: https://arxiv.org/abs/1802.08817

[18] X. Lu, H. Huo, T. Fang, and H. Zhang, "Learning deconvolutional network for object tracking," *IEEE Access*, vol. 6, pp. 18032–18041, 2018.

[19] M. Guan, C. Wen, M. Shan, C.-L. Ng, and Y. Zou, "Real-time event-triggered object tracking in the presence of model drift and occlusion," *IEEE Access*, to be published, doi: 10.1109/TIE.2018.2835390.

[20] K. Zhang, Q. Liu, Y. Wu, and M.-H. Yang, "Robust visual tracking via convolutional networks without training," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1779–1792, 2016.

[21] H. Song, Y. Zheng, and K. Zhang, "Robust visual tracking via self-similarity learning," *Electron. Lett.*, vol. 53, no. 1, pp. 20–22, 2017.

[22] J. Yang, K. Zhang, and Q. Liu, "Robust object tracking by online Fisher discrimination boosting feature selection," *Comput. Vis. Image Understand.*, vol. 153, pp. 100–108, Dec. 2016.

[23] W. Chen, K. Zhang, and Q. Liu, "Robust visual tracking via patch based kernel correlation filters with adaptive multiple feature ensemble," *Neurocomputing*, vol. 214, pp. 607–617, Nov. 2016.

[24] K. Zhang, X. Li, H. Song, Q. Liu, and W. Lian, "Visual tracking using spatio-temporally nonlocally regularized correlation filter," *Pattern Recognit.*, vol. 83, pp. 185–195, Nov. 2018.

[25] K. Zhang, Q. Liu, J. Yang, and M.-H. Yang, "Visual tracking via Boolean map representations," *Pattern Recognit.*, vol. 81, pp. 147–160, Sep. 2018.

[26] H. Song, "Robust visual tracking via online informative feature selection," *Electron. Lett.*, vol. 50, no. 25, pp. 1931–1933, Dec. 2014.

[27] K. Zhang, L. Zhang, and M.-H. Yang, "Fast compressive tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 2002–2015, Oct. 2014.

[28] F. Tang, S. Brennan, Q. Zhao, and H. Tao, "Co-tracking using semi-supervised support vector machines," in *Proc. IEEE 11th Int. Conf. Comput. Vis. (ICCV)*, Oct. 2007, pp. 1–8.

[29] Y. Gao, R. Ji, L. Zhang, and A. Hauptmann, "Symbiotic tracker ensemble toward a unified tracking framework," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 24, no. 7, pp. 1122–1131, Jul. 2014.

[30] J. Kwon and K. M. Lee, "Visual tracking decomposition," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2010, pp. 1269–1276.

[31] D.-Y. Lee, J.-Y. Sim, and C.-S. Kim, "Multihypothesis trajectory analysis for robust visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 5088–5096.

[32] S. Hare *et al.*, "Struck: Structured output tracking with kernels," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 38, no. 10, pp. 2096–2109, Oct. 2016.

[33] 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.

[34] J. Kwon and K. M. Lee, "Tracking by sampling and integrating multiple trackers," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 7, pp. 1428–1441, Jul. 2014.

[35] S. Zhang, H. Zhou, H. Yao, Y. Zhang, K. Wang, and J. Zhang, "Adaptive NormalHedge for robust visual tracking," *Signal Process.*, vol. 110, pp. 132–142, May 2015.

[36] Y. Qi *et al.*, "Hedged deep tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 4303–4311.

[37] K. Chaudhuri, Y. Freund, and D. J. Hsu, "A parameter-free hedging algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2009, pp. 297–305.

[38] Y. Qi *et al.*, "Hedging deep features for visual tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published, doi: 10.1109/TPAMI.2018.2828817.

[39] M. Kristan *et al.*, "The visual object tracking vot2016 challenge results," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, Oct. 2016. [Online]. Available: http://www.springer.com/gp/book/9783319488806

[40] R. Collins, X. Zhou, and S. K. Teh, "An open source tracking testbed and evaluation Web site," in *Proc. IEEE Int. Workshop Perform. Eval. Tracking Surveill. (PETS)*, vol. 2, Jan. 2005, p. 35.

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

MOHAMED H. ABDELPAKEY (M'16) received the B.Sc. degree (Hons.) and the M.S. degree in computer engineering from Al-Azhar University, Egypt, in 2010 and 2014, respectively. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Memorial University, Canada. His research interests include computer vision, deep learning, and pattern recognition.

MOHAMED S. SHEHATA received the B.Sc. degree (Hons.) and the M.Sc. degree in computer engineering from Zagazig University, Egypt, in 1996 and 2001, respectively, and the Ph.D. degree from the University of Calgary, Canada, in 2005.

He was a Post-Doctoral Fellow with the University of Calgary, on a joint project between the University of Calgary and the Canadian Government, called Video Automatic Incident Detection. He was a Vice-President with the Research and Development Department, Intelliview Technologies Inc. In 2013, after seven years in the industry, he joined the Faculty of Engineering and Applied Science, Memorial University of Newfoundland, as an Assistant Professor of computer engineering. His research activities include computer vision, image processing, and software design.

MOSTAFA M. MOHAMED received the B.Sc. and M.Sc. degrees in system and biomedical engineering from Cairo University, Egypt, in 1999 and 2005, respectively, and the Ph.D. degree in software engineering from the University of Calgary, Canada, in 2012. He was an Assistant Professor of biomedical engineering with Helwan University, Egypt. After that, he returned to Canada to lead his start-up company Smart Labs Ltd. He is currently an Adjunct Professor with the University of Calgary. His research interests include embedded systems, accelerations, Internet of Things, data analytics, and image processing.

MINGLUN GONG received the B.Engr. degree from Harbin Engineering University, the M.Sc. degree from Tsinghua University, and the Ph.D. degree from the University of Alberta in 1994, 1997, and 2003, respectively. He was a Faculty Member at Laurentian University for four years before joined the Memorial University of Newfoundland. He is currently a Professor and the Head of the Department of Computer Science, Memorial University of Newfoundland and an Adjunct Professor at the University of Alberta. His research interests cover various topics in the broad area of visual computing (including computer graphics, computer vision, visualization, image processing, and pattern recognition).