Augmented Memory for Correlation Filters in Real-Time UAV Tracking

Yiming Li\textsuperscript{1}, Changhong Fu\textsuperscript{1,∗}, Fangqiang Ding\textsuperscript{1}, Ziyuan Huang\textsuperscript{2}, and Jia Pan\textsuperscript{3}

\textbf{Abstract}—The outstanding computational efficiency of discriminative correlation filter (DCF) fades away with various complicated improvements. Previous appearances are also gradually forgotten due to the exponential decay of historical views in traditional appearance updating scheme of DCF framework, reducing the model’s robustness. In this work, a novel tracker based on DCF framework is proposed to augment memory of previously appeared views while running at real-time speed. Several historical views and the current view are simultaneously introduced in training to allow the tracker to adapt to new appearances as well as memorize previous ones. A novel rapid compressed context learning is proposed to increase the discriminative ability of the filter efficiently. Substantial experiments on UAVDT and UAV123 datasets have validated that the proposed tracker performs competitively against other 26 top DCF and deep-based trackers with over 40fps on CPU.

I. INTRODUCTION

Unmanned aerial vehicle (UAV) object tracking has many applications such as target tracing [1], robot localization [2], mid-air tracking [3], and aerial cinematography [4]. It aims to locate the object in the following frames given the initial location, in sometimes difficult situations such as fast motion, appearance variation (occlusion, illumination and viewpoint change, etc.), scale changes, and limited power capacity.

In UAV tracking tasks, the speed has been a key issue besides its performance. It was because of its ability to track objects at hundreds of frames per second (fps) that discriminative correlation filter (DCF) is widely applied to perform UAV tracking in the first place. Unfortunately, the pioneering works [5]–[7], despite their incredible speed, have inferior tracking performances. Therefore, strategies like part-based methods [8], [9], spatial punishment [10]–[12] and robust appearance representation [13]–[15] are used to improve their robustness and accuracy. However, the speed of DCF is sacrificed in pursuit of better performances.

In order to adapt to appearance changes of tracked objects, an appearance model is maintained and updated at each frame for most DCF trackers. Due to its updating scheme, historical appearance decays exponentially with the number of subsequent frames. The appearance in the latest 2 seconds in a 60fps video has a similar weight to all appearances before these 2 seconds in the model. This makes the trackers prone to forget objects’ early appearances and focus on more recent ones. Therefore, when the tracker has a false result, when the object is occluded, or when it is out of the view, it is very likely that the tracker learns appearances of the background using this scheme, which will further lead to the loss of the object in the following frames.

Additionally, traditional DCF framework has a low discriminative power because of the lack of negative training samples. Spatial regularization [10]–[12] and target cropping [16]–[19] were used to expand the search region and extract background patches as negative samples. Introducing context and repressing the response to it can also help discriminate objects from complex scenes [20], [21]. These methods all propose effective solution to this problem, but not efficient ones.

This work proposes an augmented memory for correlation filters (AMCF) to perform efficient object tracking with strong discriminative power, which can easily be implemented on UAVs with only one CPU. Augmented memory is used to better memorize previous appearances of objects, with a novel application of image similarity criteria pHash [22] to carefully select what to memorize. Views in the memory and current view are simultaneously used in training the correlation filter so that it has suitable responses to both previous and current object appearances. Compressed context learning is proposed to rapidly learn the background so that discriminative power is efficiently raised. AMCF is evaluated extensively on the authoritative UAVDT and UAV123 datasets. The results show its competitive performance on CPU at over 40fps compared with top trackers.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{fig1.png}
\caption{Comparison between traditional DCF with appearance models and DCF with the proposed augmented memory. Multiple historical views are selected and stored to be used in training so that it contains more historical appearance information. Traditional DCF uses a gradually decaying appearance model and is prone to drift when drastic appearance variation happens. \textbf{Red} and \textbf{Blue} boxes denote the tracking results of our tracker and others respectively. Ground truth is displayed as a \textbf{Green} box.}
\end{figure}
II. RELATED WORKS

A. Real-time tracking for UAV using DCF

UAV object tracking has flourishing applications [23–27]. Different from generic object tracking operating in videos shot by stationary cameras, tracking for UAV requires trackers to perform robustly even with the drastic motion of on-board cameras. The motion of drone cameras combined with that of objects makes precise and robust tracking difficult. Since in most cases, the movement of the drone is going to take largely depends on its perception result, tracking for UAV also requires a high processing speed of the tracker.

Generative and discriminative methods have been applied to visual object tracking. One of the discriminative methods, DCF, has been widely adopted to perform the task. Although the original works [5]–[7] showed its exceeding performance in speed, they can hardly meet the requirement for accurate as well as robust tracking, and most current trackers have sacrificed their speed for better performances. A novel tracker that can balance the speed and performance is therefore called for.

B. Model update scheme of DCF framework

One significant difference between DCF and deep learning based tracking methods is that DCF can track objects online without any pre-training on appearances of the objects. This is achieved because DCF framework usually maintains an appearance model that is updated on a frame-to-frame basis [7], [10], [17]–[19], [28]. To do that, a mostly adopted scheme is that a new model in the new frame is composed of around 99% of the previous model and around 1% of the new appearance. This 1% is treated as the learning rate. Some trackers use only a new appearance as the new model [28]. Unfortunately, the updating scheme causes a model decay, which means the appearance in early frames only takes up a small weight in this model. Therefore, when occlusion, out-of-view, or lost track of the object happens, trackers tend to learn appearances of the background. Apparently, this is not robust enough for more challenging UAV tracking scenarios.

C. Negative samples and background noise in DCF

The efficiency of DCF stems from its transformation of correlation operations into the frequency domain. To do that, the object image is cyclically shifted and extracted as samples implicitly. For traditional DCF framework, the search area is limited to prevent the correlation filter to learn too much from the background. However, only the positive sample is essentially exploited in this manner. Several measures are taken to expand the search region and feed background patches to training as negative samples [10], [17], [19], [20]. Typically, [10] uses spatial punishment to suppress background learning, and [17] crops the target and background separately. [20], [21] proposes to introduce the context of the object and suppress the response to it. Despite their effectiveness, in order to be applied in UAV tracking, the efficiency of these methods is not sufficient.

D. Tracking by deep learning

Deep learning is demonstrating its outstanding performance in various tasks. In DCF-based tracking, deep features are extracted by convolutional neural networks (CNN) in [13]–[15] to further improve performance by strengthening object appearance representation. In addition to DCF-based tracking methods, end-to-end learning [29], deep reinforcement learning [30], multi-domain network [31] and recurrent neural networks (RNNs) [32] directly use deep learning to perform tracking tasks. Despite their slightly superior performance, a high-end GPU is required for them to be trained and implemented. Even with that condition, most of them still run at low fps. Therefore, deep learning is not as suitable as DCF for aerial tracking tasks.

III. REVIEW OF TRADITIONAL DCF

Learning traditional discriminative correlation filters [7] is to optimize the following function:

$$f(w) = \|Xw - y\|^2_2 + \lambda\|w\|^2_2,$$

where $X$ is the sample matrix produced by circulating feature vector $x \in \mathbb{R}^{M \times N}$, $y$ is the desired response and $w$ is the trained filter. $\lambda$ is for regularization. It uses a model update scheme as follows:

$$m_t = (1 - \lambda)m_{t-1} + \lambda x_t,$$

where $m_t$ and $x_t$ denote the model for training and object appearance feature in the $t$-th frame respectively. Early appearances of the object are decaying exponentially.

IV. AUGMENTED MEMORY OF DCF

In this section, learning augmented memory correlation filters for real-time UAV tracking is presented. The main structure of AMCF tracker is illustrated in Fig. 2, and the objective function is as follows:

$$f_t(w) = \|X_c w - y_c\|^2_2 + \lambda\sum_{k=0}^{K} \|X_k w - y_k\|^2_2 + \lambda\|w\|^2_2,$$

where $X_c$, $X_k$ and $X_b$ present the sample matrix generated by circulating feature maps $x_c$, $x_k$ and $x_b$. $x_c$ represents the extracted patch in current frame, $x_k$ is the training patch sampled at the first frame, $x_k \ (k \in \{1, 2, ..., K\}, K << E)$ is the $k$-th view from the memory queue introduced in IV-A, where $E$ is the length of the sequence. $y_c$ and $y_k$ are distinct desired response of current frame and the ones in the memory (explained in IV-B). $x_b$ denotes compressed context patch, $H$ refers to a suppression matrix generated by circulating $h \in \mathbb{R}^{M \times N}$ and its function is to remove the object from the compressed context (explained in IV-C). $\lambda_k \ (k = 1, 2, 3)$ are adjustable parameters that determine the importance of the corresponding term.
A. Augmented memory

1) Memory queue: Basically, a first-in-first-out queue is maintained with a total length of \( K \) to store \( K \) historical views, so that they can be exploited by the training of a correlation filter in each frame. Before the memory queue is full, i.e., \( L < K \) (with \( L \) being the number of views currently stored), only \( L \) views are fed in the training process. Otherwise, all \( K \) historical views are used to train the correlation filter. For efficiency reasons, the value of \( K \) is significantly smaller than the total number of the sequence.

2) View selection: Since the number of views that can be stored is limited, it is important that different views contain different appearance details. Therefore, only when the appearance of the object is significantly different than the last selected view is this appearance allowed into the memory queue. Perceptual hashing algorithm (PHA) [22] is adopted to determine the level of difference between these two appearances. Specifically, the gray images are firstly transformed into the frequency domain by discrete cosine transform (DCT). Only the low-frequency region with high energy density is retained, denoted as \( \mathbf{B} \in \mathbb{R}^{S \times S} \). Every element \( b_{ij} \) (\( i \) and \( j \) denote the index of the element) in \( \mathbf{B} \) is then compared with the average value of \( \mathbf{B} \) to generate respective element of \( p_{ij} \) in image’s hashing matrix \( \mathbf{P} \in \mathbb{R}^{S \times S} \):

\[
\text{if } b_{ij} > \frac{1}{S} \sum_{i=1}^{S} \sum_{j=1}^{S} b_{ij} / S^2 \text{ then } p_{ij} = 1; \text{ else } p_{ij} = 0. \tag{4}
\]

Difference score of the last view and the current view is calculated using respective hashing matrices \( \mathbf{P}^l \) and \( \mathbf{P}^c \):

\[
\text{score} = \frac{\sum_{i=1}^{S} \sum_{j=1}^{S} (p_{ij}^l \oplus p_{ij}^c)}{S^2}, \tag{5}
\]

where \( p_{ij}^l \) and \( p_{ij}^c \) respectively denote the element of hashing matrix \( \mathbf{P}^l \) and \( \mathbf{P}^c \). \( \oplus \) is the XOR operator. If the score is more than threshold \( \tau \), two appearances are considered different and current one is selected into the memory queue.

B. Different desired responses

Earlier selected views generally have a lower similarity to the current appearance than the later selected ones. Therefore, different desired responses are assigned to different views in the memory queue. By altering the maximum and the variance of Gaussian function, lower maximum and larger variance of desired responses are generated for early views:

\[
y_k(\sigma_k) = \nu y_c(\sigma_c) \\
y_k(\sigma_k) = \nu y_{k+1}(\mu \sigma_{k+1}) \quad (k = 1, 2, ..., K - 1), \tag{6}
\]

where \( y_c(\sigma_c) \) denotes the desired response of the current frame with the maximum of 1 and the variance of \( \sigma_c \). The subscript of \( y \) represents the index value of response target in memory queue (lower values correspond to earlier views). Parameters \( \nu < 1 \) and \( \mu > 1 \) make sure that maximum is decreasing and variance is increasing with the index decreasing. For the first frame of the sequence, the desired response is calculated as follows:

\[
y_0(\sigma_1) = \phi y_c(\varphi \sigma_c), \tag{7}
\]

where \( \phi < 1 \) and \( \varphi > 1 \) are used to adjust the maximum and variance of the Gaussian distribution of the desired response.

C. Compressed context learning

In order to increase the discriminative power of DCF, compressed context learning is proposed. Unlike traditional context learning [20], [21] which extracted multiply context patches, this work proposes to extract single patch for context learning by enlarging the search region and compressing it to the size of the correlation filter. Then the pixel value where the object is located is lowered by applying a quadratic suppressing function, so as to remove the object from this patch. This compressed context is assigned zero response so that the response to the surrounding area of the object can be minimized and discriminative power can thus be enhanced.
D. Learning and detection in AMCF

1) Learning process: The optimization result of \( f_t(w) \) for the \( d \)-th (\( d \in \{1, \ldots, D \} \)) channel is calculated as follows:

\[
\hat{w}_d^t = \frac{M_d^t + \lambda_2 \sum_{k=0}^{K} M_k^d}{\sum_{d=1}^{D} D_t^d + \lambda_1},
\]

where \( M = \hat{x}^t \odot \hat{y}, \hat{A} = \hat{x}^t \odot \hat{x} \) and \( \hat{E} = \hat{m}_b^t \odot \hat{m}_b \). \( \hat{m}_b = h \odot x, x^t \) denotes the complex-conjugate of \( x \) and the operator \( \odot \) stands for the element-wise product. Hat mark is the discrete Fourier transform (DFT) value.

The learning rate \( \gamma \) is used to update the numerator \( \hat{N}^d_t \) and the denominator \( D_t^d \) of the filter \( \hat{w}_d^t \) in the \( t \)-th frame:

\[
\hat{N}_t^d = (1 - \gamma) \hat{N}_{t-1}^d + \gamma \left( M_{t-1}^d + \lambda_2 \sum_{k=0}^{K} M_{t-1}^k \right),
\]

\[
D_t^d = (1 - \gamma) D_{t-1}^d + \gamma \left( \hat{A}_{t-1}^d + \lambda_2 \sum_{k=0}^{K} \hat{A}_{t-1}^k + \lambda_3 \hat{E}_{t-1}^d \right).
\]

In order to make sure reliable channels contribute more to the final result, a channel weight aggregate \( C = \{c_d^t\} (d \in \{1, \ldots, D\}) \) is assigned to each channel and updated as follows:

\[
c_d^t = (1 - \eta) c_{d-1}^t + \eta \frac{\max(\hat{w}_d^t \odot \hat{x}_b^t)}{\sum_{d=1}^{D} \max(\hat{w}_d^t \odot \hat{x}_b^t)}.
\]

2) Detection in AMCF: In detection phase, the following formula is used to generate the final response map \( R_t \) and update the object location by searching the maximum value:

\[
R_t = \mathcal{F}^{-1} \left( \sum_{d=1}^{D} c_d^t \hat{w}_d^t \odot z_t^d \right),
\]

where \( \hat{w}_d^t \) and \( z_t^d \) are respectively learned filter and current feature of search region in the frequency domain, and \( \mathcal{F}^{-1} \) denotes the inverse discrete Fourier transformation.

V. EXPERIMENTS

In this section, the presented AMCF tracker is comprehensively evaluated on two different datasets, i.e., UAVDT [33] and UAV123 [34], with 173 image sequences covering over 140,000 frames captured by UAV in various challenging scenarios. It is noted that the videos from both datasets are recorded at 30fps. 11 real-time trackers (CPU-based) are used to compare with AMCF, i.e., ECO_HC [35], STRCF [28], MCCT_H [36], STAPLE_CA [20], BACF [17], DSST [37], fDSST [38], STAPLE [39], KCC [40], KCF [7], DCF [7]. Furthermore, 15 deep-based trackers are compared with AMCF to further demonstrate its performance, i.e., ASRCF [41], ECO [35], C-COT [42], MCCT [36], DeepSRDCF [28], ADNet [30], CFNet [29], MCPRF [43], IBCCF [15], CF2 [14], CREST [44], HDT [45], FCNT [46], PTA [47], TADT [48]. The evaluation criteria are strictly according to the protocol in two benchmarks [33], [34].

Algorithm 1: AMCF tracker

**Input:** Groundtruth in the first frame

**Output:** Predicted location of target in frame

1. if \( t = 1 \) then
2. Extract \( x_1 \) centered at the groundtruth
3. Use Eq. (8) to initialize the filters \( w_1 \)
4. Initialize channel weight model \( \{ c_d^t \} = 1/D \)
5. else
6. Extract \( z_t \) centered at location on frame \( t - 1 \)
7. Use Eq. (11) to generate the response map
8. Find the peak position of map and output
9. Extract \( x_t \) and \( x_0 \) centered at location on frame \( t \)
10. Calculate the score between \( P^l \) and \( P^c \)
11. if \( \text{score} > \tau \) then
12. Update FIFO memory queue
13. end
14. Use Eq. (8) to update the filters \( w_t \)
15. Use Eq. (9) to update channel weight \( C_t \)
16. end

A. Implementation details

All the experiments of all trackers compared as well as AMCF are conducted on a computer with an CPU of i7-8700K (3.7GHz), 32GB RAM and an NVIDIA GTX 2080 GPU. All trackers are implemented in MATLAB R2018a platform, and their original codes without modification are used for comparison. Memory length \( K = 5 \), and the threshold for memory view selection is set to \( \tau = 0.5 \). STAPLE [39] proposed to solve two optimal problems for learning two models respectively robust to deformations and color changes. Due to its high efficiency and greater accuracy than traditional DCF trackers like [7], [37], this work chooses it as the baseline tracker.

B. Quantitative study

1) Effectiveness study: AMCF tracker is firstly compared with itself with different modules enabled. The effectiveness evaluation result can be seen in Table I. With each module (channel weight CW, augmented memory AM, and compressed context CC) added to the baseline, the performance is steadily being improved.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>UAV123</th>
<th>UAVDT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subject</td>
<td>Prec.</td>
<td>AUC</td>
</tr>
<tr>
<td>AMCF</td>
<td>69.5</td>
<td>49.3</td>
</tr>
<tr>
<td>Baseline+CW+CC</td>
<td>68.5</td>
<td>48.3</td>
</tr>
<tr>
<td>Baseline+CC+AM</td>
<td>67.4</td>
<td>47.8</td>
</tr>
<tr>
<td>Baseline+CW</td>
<td>66.8</td>
<td>46.8</td>
</tr>
<tr>
<td>Baseline</td>
<td>65.5</td>
<td>46.8</td>
</tr>
</tbody>
</table>
2) **Overall performance:** In comparison with other top real-time trackers, AMCF has shown superiority in terms of precision and accuracy on both benchmarks. Fig. 3 shows a separate performance evaluation of AMCF on two benchmarks. AMCF has achieved satisfactory performances on both benchmarks. Specifically, AMCF performs the best on UAVDT [33], with an improvement of 0.6% and 1.2% on precision and AUC score respectively. On UAV123, AMCF has achieved the second-best performance. Since the object size in UAV123 is generally much larger than that in UAVDT [33] because of the flying height of UAVs, many trackers with good performance on one benchmark can rank low on the other. One typical example is ECO_HC [35]. It ranks first on UAV123 but only comes out at fifth place on UAVDT [33]. AMCF, on the other hand, has a better generalization ability compared to most trackers. Overall evaluation for both benchmarks combined can be seen in Table II. AMCF has a slightly better overall performance than ECO_HC [35], ranking the first place. But ECO_HC [35] has a relatively large variance. Therefore, it can be concluded that in terms of overall performance, AMCF performs favorably against other top real-time trackers. In terms of speed, AMCF, solely running on CPU, can also meet the requirement of real-time tracking (>30fps on an image sequence captured at 30fps).

3) **Attribute-based performance:** Attribute-based evaluation results on both benchmarks are shown in Fig. 4. It can be seen that the reason behind our satisfactory generalization ability originates from the combination of two of our core modules, i.e., augmented memory and compressed context learning. On UAVDT [33], when the object is small, the feature of the object decreases and positive samples are thus not enough for robust tracking. Compressed context learning simultaneously expands the search region and brings more negative samples. Therefore, when there are camera motion and background clutter, AMCF performs satisfactorily. On UAV123 [34], the object is significantly closer, so viewpoint change and aspect ratio change can result in more drastic appearance changes. Augmented memory provides more appearance information on previous objects so that the desired
response can be obtained when the current view has some resemblance to previous views. Therefore, thanks to both modules, AMCF can handle both near objects with large viewpoint changes and distant objects with a small size on the image.

C. Comparison with deep-based trackers

On UAVDT, extra 15 deep-based trackers (trackers using deep learning methods or DCF-based trackers using deep features) are compared with AMCF. Precision and success plots can be seen in Fig. 5. Surprisingly, AMCF still succeeds to exceed most of the deep-based trackers in terms of precision and AUC scores. More specifically, AMCF achieved the best performance in precision, with 0.1% slightly better than the second-place ASRCF [41], while in AUC evaluation, AMCF achieved the second, falling behind ECO [35] by only 0.9%. In terms of tracking speed, AMCF is the fastest among the evaluated deep-based trackers. To sum up, in tracking distant objects, AMCF demonstrates superior tracking performance against both real-time trackers and deep-based trackers.

D. Qualitative study

Figure 6 intuitively demonstrates the aforementioned results. The first two sequences show the ability to track distant objects and adapt to view changes on UAVDT respectively. The third and fourth sequences show those abilities of AMCF on UAV123. Capability of resisting occlusion is demonstrated in the fifth sequence. The last one shows two modules can work smoothly together.

VI. CONCLUSIONS

In this work, augmented memory for correlation filters is proposed. Essentially, augmented memory maintains a FIFO
queue to store distinct previous views. Each stored view in
the memory will be assigned the desired response. Along
with the feature of the object in the current frame, all views
are simultaneously used to train a correlation filter that can
adapt to new appearances and has a response to previous
views at the same time. Compressed context learning pro-
vides more negative samples and suppresses responses to
surrounding areas of the tracked object. Extensive experiment
results proved that AMCF has competitive performance and
tracking speed compared to top real-time trackers. AMCF
also demonstrates an outstanding generalization power that
can track both near objects with large view change and
distant objects with small size in the image. Future work
can include introducing a confidence check to prevent false
tracking results to be selected as a view. This method can also
be applied to more powerful baselines in the replacement of
model updates.

ACKNOWLEDGMENT

The work is supported by the National Natural Science
Foundation of China under Grant 61806148 and the State
Key Laboratory of Mechanical Transmissions (Chongqing
University) under Grant SKLMT-KFKT-201802.

REFERENCES

environments using a quadrotor,” in Proceedings of IEEE/RSJ Inter-
national Conference on Intelligent Robots and Systems, 2016, pp. 446–
453.

“Localization from visual landmarks on a free-flying robot,” in Pro-
cedings of IEEE/RSJ International Conference on Intelligent Robots

P. C. Cervera, “Robust Real-time Vision-based Aircraft Tracking From
Unmanned Aerial Vehicles,” in Proceedings of IEEE Conference on

“Can a Robot Become a Movie Director? Learning Artistic Principles
for Aerial Cinematography,” in Proceedings of IEEE/RSJ International

object tracking using adaptive correlation filters,” in Proceedings of
IEEE Conference on Computer Vision and Pattern Recognition, 2010,
p. 2544–2550.

the Circulant Structure of Tracking-by-Detection with Kernels,” in
702–715.

tracking with kernelized correlation filters,” IEEE Transactions on
Pattern Analysis and Machine Intelligence, vol. 37, no. 3, pp. 583–596,
2015.

via adaptive correlation filters,” in Proceedings of the IEEE Conference
on Computer Vision and Pattern Recognition, 2015, pp. 4902–4912.

[9] C. Fu, Y. Zhang, R. Duan, and Z. Xie, “Robust Scalable Part-Based
Visual Tracking for UAV with Background-Aware Correlation Filter,”
in Proceedings of IEEE International Conference on Robotics and
Biomechatronics, 2018, pp. 2245–2252.

Spatially Regularized Correlation Filters for Visual Tracking,” in Pro-
cedings of IEEE International Conference on Computer Vision, 2015,
p. 4310–4318.

performance visual tracking for uav with automatic spatio-temporal
regularization,” in Proceedings of the IEEE/CVF Conference on
Computer Vision and Pattern Recognition, 2020, pp. 11 923–11 932.