remote sensing

MDPI

Article

Visual Object Tracking for Unmanned Aerial Vehicles Based on the Template-Driven Siamese Network

Lifan Sun [1,2], Zhe Yang [1], Jinjin Zhang [1], Zhumu Fu [1,*] and Zishu He [2]

[1] School of Information Engineering, Henan University of Science and Technology, Luoyang 471023, China; lifan.sun@haust.edu.cn (L.S.); 190319050238@stu.haust.edu.cn (Z.Y.); 200320050405@stu.haust.edu.cn (J.Z.)
[2] School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China; zshe@uestc.edu.cn
* Correspondence: Fuzhumu@haust.edu.cn

Abstract: Visual object tracking for unmanned aerial vehicles (UAV) is widely used in many fields such as military reconnaissance, search and rescue work, film shooting, and so on. However, the performance of existing methods is still not very satisfactory due to some complex factors including viewpoint changing, background clutters and occlusion. The Siamese trackers, which offer a convenient way of formulating the visual tracking problem as a template matching process, have achieved success in recent visual tracking datasets. Unfortunately, these template match-based trackers cannot adapt well to frequent appearance change in UAV video datasets. To deal with this problem, this paper proposes a template-driven Siamese network (TDSiam), which consists of feature extraction subnetwork, feature fusion subnetwork and bounding box estimation subnetwork. Especially, a template library branch is proposed for the feature extraction subnetwork to adapt to the changeable appearance of the target. In addition, a feature aligned (FA) module is proposed as the core of feature fusion subnetwork, which can fuse information in the form of center alignment. More importantly, a method for occlusion detection is proposed to reduce the noise caused by occlusion. Experiments were conducted on two challenging benchmarks UAV123 and UAV20L, the results verified the more competitive performance of our proposed method compared to the existing algorithms.

Keywords: unmanned aerial vehicles; visual object tracking; Siamese network; template library; feature-aligned

Citation: Sun, L.; Yang, Z.; Zhang, J.; Fu, Z.; He, Z. Visual Object Tracking for Unmanned Aerial Vehicles Based on the Template-Driven Siamese Network. *Remote Sens.* **2022**, *14*, 1584. https://doi.org/10.3390/rs14071584

Academic Editors: Józef Lisowski, Kouzou Abdellah, Haitham Abu-Rub and Piotr Szymak

Received: 16 February 2022
Accepted: 22 March 2022
Published: 25 March 2022

1. Introduction

Visual object tracking is a typical problem in computer vision. It has been studied for many years and applied in the fields of security monitoring, telemedicine, smart city, and so on [1–4]. Different from traditional video sequences, the object in the video shot by UAV has distinct characteristics: (1) smaller object size and more complex background, as shown in Figure 1a; (2) targets often disappear because of occlusion, illustrated in Figure 1b; (3) broader perspective and changeable appearance, shown in Figure 1c. Due to these problems, visual object tracking in videos shot by UAV is still challenging [5–7].

Most of the visual object tracking methods for UAV are divided into two categories: one is based on correlation filter, another is based on deep learning. The trackers [8–10] based on correlation filter calculate a filter to maximize their response to the region of the target. Due to their excellent speed performance, they have gradually become a popular research topic in the field of visual object tracking. Recently, the trackers based on deep learning have received extensive attention because of their superior performance [11–13]. The deep features are more robust than handcrafted features, leading to a considerably improved tracking accuracy. However, due to the large computation of backpropagation, even though many trackers train the network offline and fine-tune it in the tracking phase, their tracking speed is still far from meeting the requirements of working in real time.

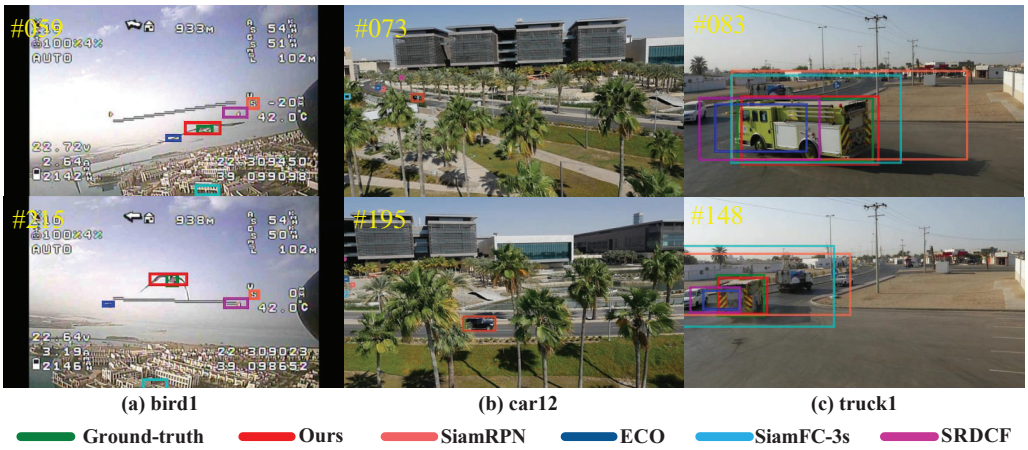


Figure 1. Comparison of proposed tracker with other existed methods (SiamRPN [14], ECO [15], SiamFC-3s [16], SRDCF [17]) on the bird1, car12 and truck1 from UAV123 [18].

Fortunately, the visual object tracking algorithms [14,16,19–23] based on Siamese networks can well balance the accuracy and speed simultaneously, which has gradually attracted people's attention. The concise Y-shaped structure simplifies visual object tracking into the template matching process. They train the network parameters in the offline stage and fix the parameters in the inference stage. Therefore, the performance of these template-based trackers depends mainly on the quality of the template. However, since the appearance of the target in the video captured by UAV often changes, a fixed template is obviously not enough to ensure the tracking stability of the whole video sequence. To solve this problem, many efforts have been made. Bertinetto et al. proposed the SiamFC [16] using a multiscale strategy to improve robustness. Li et al. combined the region proposal network with the Siamese network to remove the multiscale test and improve the tracking speed greatly [14]. Although significant success has been achieved, the performance of visual object tracking for UAV is still not satisfactory. As shown in Figure 1, the performance of SiamFC and SiamRPN will become worse in the case of complex background, occlusions, and appearance changes commonly seen in the field of view of UAV.

In view of this, this paper proposes a template-driven Siamese network (TDSiam) which is more suitable for UAV aerial video. The TDSiam can be divided into three main parts: feature extraction subnetwork, feature fusion subnetwork and bounding box estimation subnetwork. Additionally, the feature-aligned (FA) module is the key of the feature fusion subnetwork. The FA module can more effectively fuse features from multiple templates and the bounding box estimation subnetwork can more accurately regress the parameters of the bounding box. In addition, the average peak-to-correlation energy (APCE) [24] is applied to avoid distractors from polluting the template in the inference stage. The experimental results show that the proposed method has competitive performance. In addition, when AlexNet [25] is selected as the backbone, it reaches 180 fps, and when ResNet [26] is selected as the backbone, it reaches 35 fps, both exceeding the real-time requirements (speed > 20 fps). The main contributions of this work are as follows:

1. This paper proposes a template-driven Siamese network, which is more suitable for the tracking task under the UAV's vision. Compared with existing approaches, the proposed template-driven Siamese network has concise forms and favorable properties;
2. A template library structure is proposed, which can effectively improve the tracking performance combined with the new template updating strategy. In addition, a feature alignment module is proposed to fuse template features more efficiently;
3. Detailed experiments show that the proposed method has competitive performance, not only in accuracy but also in speed.

The structure of this paper is as follows: other works related to ours have been reviewed in Section 2. A description of the framework of TDSiam has been given in Section 3.

Tracking results obtained by applying the proposed method on UAV video benchmarks are presented in Section 4. Along with a discussion on these results, conclusions from this work are presented in Section 5.

2. Related Works

2.1. The Siamese Trackers

Thanks to outstanding performance, the trackers [16,19,20,22,23] based on the Siamese network have attracted people's increasing attention in recent years. These trackers simplify the target tracking problem into a template matching process and realize the tracking by matching the similarity between deep features. They train the network offline by using a large amount of data and distinguish whether a pair of images belong to the same category or not by learning a similarity map.

By learning the invariance of the object, the Siamese instance search for tracking (SINT) algorithm can robustly deal with the possible shape changes of the object in the form of template matching [19]. At the same time, Bertinetto et al. proposed the SiamFC algorithm, which has a two-branch architecture: one is a template branch, the other is the search branch [16]. The former contains information about the tracked object, the latter is used to provide a search area containing the target to be tracked. These features extracted from two branches are cross-correlated to generate a similarity map. Li et al. combined the region proposal network with a Siamese network to remove multiscale test steps and greatly improve the tracking speed [14]. Since then, many improvements have been made to SiamRPN. Zhu et al. introduced a series of strategies to enhance the generalization of deep features for the imbalance of training data, and introduced a long-time tracking strategy to improve the tracking performance after the target disappears [23]. SiamDW [27] and SiamRPN++ [28] were proposed to investigate the utilization of a deep network for Siamese tracking in different aspects. The SiamAttn [29] incorporates self-attention and cross-attention to help trackers adapt well to complex situations. Zhang et al. proposed UpdateNet [30], which trains a small network to learn the best template for the next frame prediction.

Recently, the anchor-free architecture has been combined with Siamese networks, e.g., Xu et al. provided four guidelines for designing the modern tracker and proposed the SiamFC++ [31] which has favorable performance. Guo et al. proposed a parallel centerness branch to reduce the impact of poor-quality bounding boxes far from the target [32]. Chen et al. proposed SiamBAN [33] to reduce hyperparameters by using the anchor-free framework and provided a new method of distinguishing positive and negative samples. These Siamese tracking algorithms have achieved great success because of their higher accuracy and faster speed. Although some invariance can be learned in the offline training process using a large amount of data, its performance is still not satisfactory due to frequent changes in the appearance of the target in the UAV video scene.

2.2. Trackers for UAV Videos

Recently, in visual object tracking, there have been many tracking algorithms that can achieve good results on UAV benchmarks. They have been optimized from different aspects and tested on UAV datasets. Huang et al. proposed a new tracking method which can effectively suppress the distortion while solving the boundary effect [34]. Zhang et al. proposed UpdateNet, which used three different templates and fused them to obtain a more powerful template [30]. Yao et al. proposed the Lucas–Kanade network and integrated it into the Siamese network to enable learning of aligned feature representations [35]. Li et al. proposed the SiamRPN++ [28], which has better robustness thanks to increasing its network depth. Zhu et al. proposed the DaSiamRPN [23] to reduce the influence of interferents via a distractor-aware module. They have been tested on UAV datasets and achieve relatively good performance. However, the performance in some UAV tracking scenarios is not satisfactory. As shown in Table 1, there is room for further improvement

and developing these algorithms under BC (Background Cluster), FOC (Full Occlusion), POC (Partial Occlusion) and SV (Scale Variation) attributes.

Table 1. Comparisons of the success rate between the proposed approach and the existing approaches including SiamRPN++, DaSiamRPN and SiamRPN on the UAV123 dataset.

Attributes / Algorithms	Overall	BC	FOC	POC	SV
SiamRPN++	0.618	0.448	0.411	0.539	0.595
DaSiamRPN	0.569	0.444	0.379	0.493	0.544
SiamRPN	0.535	0.427	0.332	0.441	0.512
Ours (TDSiam_Res50)	0.643	0.498	0.520	0.600	0.639

In this paper, we propose a new tracking approach based on a Siamese network for UAV. Firstly, the concept of template library is proposed to solve the problem of the appearance of the target changing frequently. Furthermore, to make more efficient use of the template feature, we construct the FA module. Finally, the APCE is used to prevent dictators from entering the template library in the inference stage.

3. Template-Driven Siamese Network

Figure 2 illustrates the framework of TDSiam—it consists of three main parts: feature extraction subnetwork, feature fusion subnetwork and bounding box estimation subnetwork. The feature extraction subnetwork is divided into two main branches: template library branch and searching branch. For more useful information, the template library contains strong template T_0, weak template T_{acc} and T_i. The feature fusion subnetwork is responsible for merging information from the template library. In particular, the FA module is the core of the template feature fusion subnetwork, which is responsible for template feature alignment. The bounding box estimation subnetwork adopts the RPN network proposed in Faster RCNN [36], which decomposes the tracking into two tasks: classification and regression. Based on the success of target and background classification, the bounding box is regressed more finely.

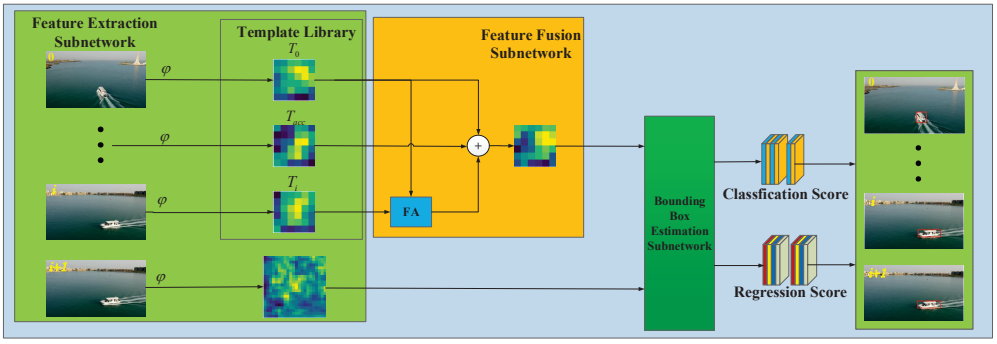


Figure 2. Schematic of the TDSiam framework. On the left is the feature extraction subnetwork. The feature fusion subnetwork is depicted in the middle, which is responsible for fusing the information from the template library. On the right is the bounding box estimation subnetwork, which has two branches, one is in charge of foreground–background classification, and the other is used for bounding box regression.

Next, the proposed TDSiam will be introduced in detail. Firstly, we will illustrate the feature extraction subnetwork in Section 3.1. Secondly, the feature fusion subnetwork is proposed in Section 3.2. Finally, we will introduce the bounding box estimation subnetwork in Section 3.3 and the inference stage in Section 3.4, respectively.

3.1. The Feature Extraction Subnetwork

In order to obtain diversified target appearance information, we designed the template library (TL) structure in Siamese network with reference to Updatenet [30]. The template library contains three templates: T_0, T_i and T_{acc}, where T_0 is the strong template cropped according to the ground-truth from the first frame, T_i is obtained from the prediction result of the current frame, and T_{acc} represents the last accumulated template.

$$T_{acc} = \sum_{i=1}^{N} T_i \tag{1}$$

where T_0 is the initial ground-truth template and contains highly reliable information. T_i is used to adapt to the changes of the target appearance during the tracking phase. Moreover, by integrating the historical appearance information of the object under consideration, the last accumulated template T_{acc} can increase the robustness against model drift.

The feature extraction is accomplished by a deep convolutional neural network. The AlexNet [25] and ResNet [26] are chosen as the backbone of TDSiam in the experiment. The former is more concise, which can make the tracking speed faster. The latter can stack the network deeper and obtain higher tracking accuracy. The tracking process is transformed into two tasks: classification and regression. The classification task is responsible for identifying whether the candidate area contains targets, and the regression task is in charge of refining the classification results. The detailed optimization process will be described in Section 3.3.

3.2. The Feature Fusion Subnetwork

The weighted sum is a simple but efficient way of fusing these templates. However, the object might not always be at the center of the template T_i because of tracking error. The direct fusion of the misaligned features will result in the aliasing effect [37]. Therefore, a feature-aligned (FA) module has been proposed to obtain the central-aligned features.

Figure 3a,b present the templates cropped from the predicted results during the tracking process. The red cross indicates the center of the image, but they belong to different targets. If we fuse them according to their linear weights, interference will be introduced into the template. Fortunately, we notice that features with the same appearance have high cosine similarity, as seen in Figure 3c, where the red region indicates a high value. Thus, we have proposed an FA module to reconstruct the misaligned template features based on their cosine similarity.

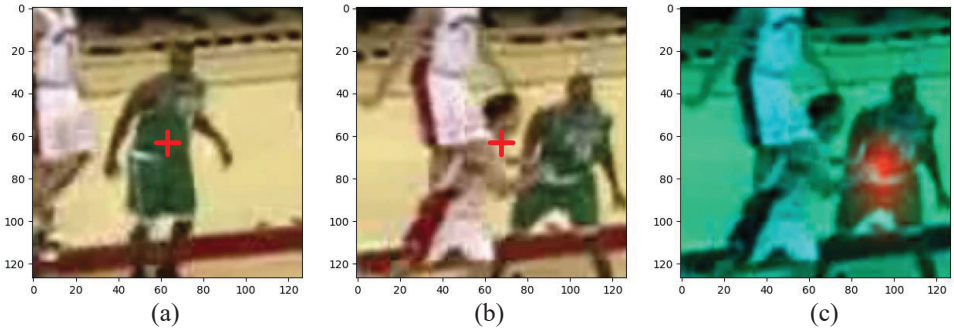


Figure 3. Visualization of the cosine similarity. (**a**) is the original template image, (**b**) is the image cropped by the tracking result in the subsequent, and (**c**) is the cosine similarity added into the image patch. The red region represents the high value.

The goal of the FA module is to ensure that the same spatial position on the reconstructed feature map belongs to the same part by reconstructing each misaligned feature map. It can be considered as a matching task between two feature maps. The detailed construction of the FA module is shown in Figure 4. The T_0 and T_i indicate the feature map extracted from the initial ground-truth and the predicted results, respectively, both their dimensions are $H \times W \times C$. The cosine similarity (CS) between each position of T_0 and T_i can be calculated as follows:

$$CS_{(m,n)} = \frac{T_{0(m,n)} \cdot T_{i(m,n)}}{\left\| T_{0(m,n)} \right\| \cdot \left\| T_{i(m,n)} \right\|} \tag{2}$$

where (m, n) is the position in the feature map. Then, the whole cosine similarity map can be calculated as follows:

$$CS = \frac{T_0 \cdot T_i}{\|T_0\| \cdot \|T_i\|} \tag{3}$$

The Equation (3) can be simplified as follows:

$$CS = l_2(T_0) \cdot l_2(T_1) \tag{4}$$

where the l_2 indicates the normalization process and the symbol $\cdot$ represents the dot product. The shape of the cosine similarity map is $HW \times HW$, each of its rows corresponds to the cosine similarity between each feature in T_0 and all features in T_i.

Since the scales of the same object in a video sequence can be different, one position in the template T_0 can have several corresponding pixels in the template T_i and vice versa. Therefore, we cannot simply shift the features to the corresponding position, all pixels that have the same appearance should be included. Therefore, we obtain new features $y_{(m,n)}$ by weighted summation of features with similar appearance, which can be formulated as follows:

$$y_{(m,n)} = \sum_{m=1}^{H} \sum_{n=1}^{W} sigmoid(\mathrm{CS}_{(m,n)}) T'_{i(m,n)} \tag{5}$$

where H and W are the size of the feature map, and the *sigmoid* function maps the weight parameters to a range of 0 to 1.

The object might become incomplete during the tracking process, in which case the feature map T_i cannot align well with T_0. Thus, in order to avoid errors caused in such a case, we use a spatial attention mask to determine the correctness of matching as shown in Figure 4. The *Conv* is implemented by a 1×1 convolution and $\odot$ is the element-wise multiplication. The mask produced by *Conv* indicates the semantic similarity between T_0 and the reconstructed feature.

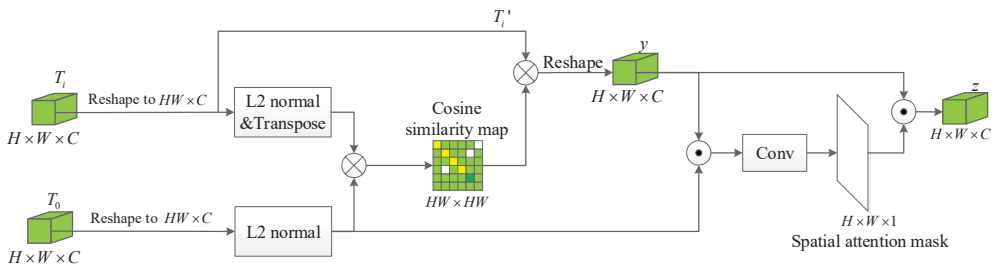


Figure 4. The feature-aligned (FA) module has two inputs: T_0 indicates the initial template feature and T_i is extracted from the predicted target location. The feature-reconstructed network is on the left and the spatial attention network is on the right.

3.3. The Bounding Box Estimation Subnetwork

The main part of the bounding box prediction subnetwork is the RPN network, which is composed of a classification and regression subnetwork. Firstly, a few anchors with different scales are defined. Inspired by SiamRPN [14], the number of anchors is set to 5. Since the target does not scale considerably between adjacent frames, we only adopt one scale with different ratios $[0.33, 0.5, 1, 2, 3]$. After feature extraction, we can obtain $\varphi(z)$ and $\varphi(x)$, which denote the template library branch feature and searching branch feature, respectively. The classification and regression scores can be calculated as follows:

$$A^{cls}_{w\times h\times 2k} = [\varphi(x)]_{cls} * [\varphi(z)]_{cls} \tag{6}$$

$$A^{reg}_{w\times h\times 4k} = [\varphi(x)]_{reg} * [\varphi(z)]_{reg} \tag{7}$$

where the $\varphi(z)_{cls}$ and $\varphi(x)_{cls}$ are applied to the classification subnetwork, which are from the template image and the search area, respectively. Similarly, the results of cross-correlation operation between $\varphi(z)_{reg}$ and $\varphi(x)_{reg}$ are used to regress the finer bounding box, and $*$ denotes the cross-correction operation.

After cross-correlation operation, we can obtain $A^{cls}_{w\times h\times 2k}$ and $A^{reg}_{w\times h\times 4k}$. The $A^{cls}_{w\times h\times 2k}$ represents the confidence score of target and background classification of each candidate box, and $A^{reg}_{w\times h\times 4k}$ represents the refined parameters of each candidate box. For classification subnetwork, a $softmax$ function is applied to monitor the classification loss.

IOU is used as an indicator to distinguish positive and negative samples. Positive samples indicate that the IOU between anchor boxes and their ground truth is greater than 0.6, and the negative samples indicate that the IOU is less than 0.3. Therefore, for a single sample, the loss function is defined as:

$$l(y, v) = log[(1 + exp(-yv)] \tag{8}$$

where y represents the label and v represents the classification score of network prediction. For a set of anchors, y and v will be vectors. Thus, the corresponding loss function is

$$L_{cls} = \frac{1}{|D|}\sum_{u\in D} log(1 + exp(-y(u)v(u))) \tag{9}$$

where D denotes a sample space, u is the sample serial number and $|D|$ is the size of sample space. The process of training the network is constantly searching for the appropriate θ to minimize the classification loss:

$$\arg\min_{\theta} \frac{1}{|D|}\sum_{u\in D} log(1 + exp(-y(u)v(u))) \tag{10}$$

For the regression task, the parameters of the bounding box need to be refined by the regression network. In order to obtain a more stable regression training process, the regression parameters are standardized:

$$\begin{aligned} &\delta[0] = \tfrac{T_x - A_x}{A_w},\ \delta[1] = \tfrac{T_y - A_y}{A_h} \\ &\delta[2] = \ln(\tfrac{T_w}{A_w}), \delta[3] = \ln(\tfrac{T_h}{A_h}) \end{aligned} \tag{11}$$

where A_x, A_y, A_w and A_h represent the central coordinates (x, y), width and height of bounding box predicted by the current frame, respectively. Similarly, T_x, T_y, T_w and T_h represent the central coordinates (x, y), width and height of annotation of the current frame, respectively.

The regression branch is supervised by the loss function $smoothL_1$, which can be formulated as follows:

$$smoothL_1(x,\sigma) = \begin{cases} 0.5\sigma^2 x^2, |x| < \frac{1}{\sigma^2} \\ |x| - \frac{1}{2\sigma^2}, |x| \geq \frac{1}{\sigma^2} \end{cases} \tag{12}$$

The L_{reg} can be written as:

$$L_{reg} = \sum_{i=0}^{3} smoothL_1(\delta[i],\sigma) \tag{13}$$

The overall loss function is:

$$loss = L_{cls} + \lambda L_{reg} \tag{14}$$

where the λ is the coefficient, it can be used to balance the loss of classification and regression.

3.4. Inference Phase: Template Updating

During the tracking phase, updating the template too frequently can not only lead to error accumulation, but also affect the tracking speed. Partial- and full-occlusion phenomena also occur sometimes. When the target in the template is blocked, adding it to the template library will introduce a lot of adverse noise. To address the problem that the target might be occluded during the tracking process, the APCE index was used to measure the tracking result. Following the SiamFC [16], the response map R was calculated by performing a cross-correction operation.

$$R = \varphi(z) * \varphi(x) \tag{15}$$

here, z and x represent the template image and the search area, respectively. The $*$ denotes the cross-correction operation.

As shown in Figure 5, when the target is not occluded, there is only a particularly high peak in the response graph, and the response value is several times greater than that in other places. Otherwise, the response map will fluctuate intensely when the target is occluded. Inspired by the LMCF algorithm [24], the APCE index is used to evaluate the occlusion degree of the target, which can be calculated as follows:

$$APCE = \frac{|F_{\max} - F_{\min}|^2}{mean\left(\sum_{w,h}(F_{w,h} - F_{\min})^2\right)} \tag{16}$$

where $F_{\max}$, $F_{\min}$ represent the maximum and minimum values in the response map, respectively. $F_{w,h}$ is the corresponding value at (w,h). The numerator part reflects the peak value in the response map and represents the reliability of the current response map, whereas the denominator represents the average degree of fluctuation of the response map. Once the target is occluded, the APCE will become smaller immediately, compared with its previous level.

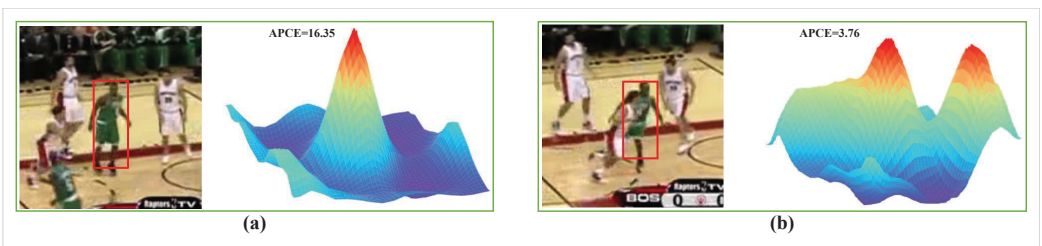


Figure 5. Visualization of the response map. (**a**) When the target is not occluded, the response map has only one peak and the value of APCE is larger. (**b**) When the target is occluded, the response map has some peak with low strength and the value of APCE is smaller.

In this paper, the historical average of APCE, i.e., $APCE_{avg}$, is used as the criterion to judge whether the target is occluded or not, i.e.,

$$APCE_{avg} = \frac{1}{m}\sum_{i=1}^{m} APCE_i \tag{17}$$

where the $APCE_i$ represents the APCE value of the ith frame image, the m is the number of frames that have been tracked. where the $APCE_i$ represents the APCE value of the ith frame image, m is the number of frames that have been tracked. When APCE is less than the $APCE_i$, the target is considered as occluded and the tracker should stop updating the template. The detailed tracking process is shown as Algorithm 1.

Algorithm 1: The proposed TDSiam algorithm

Input: The first frame I_0 and the ground truth (x_0, y_0, w_0, h_0) of the object;
Output: The predicted bounding box (x_i, y_i, w_i, h_i) of the ith frame;

```
Crop the initial template region and extract the feature T_0;
Initialization: T_f = T_0;
for i = 1, 2, ..., n do
    Crop the search region using the predicted result in the last frame and extract
     the feature S_i;
    Compute the classification scores and regression boxes using T_f and S_i;
    Compute the response map R_i by cross-correlation operation between T_0 and
     S_i using Equation (15), i.e., R = φ(z) ∗ φ(x) ;
    Compute the APCE of the current frame using Equation (16);
    if APCE >= APCE_avg then
        Crop the dynamic template image and extracted the feature T_i;
        Perform feature alignment for T_i using FA module;
        Fuse the features of T_0, T_i and T_acc by element-wise summation using
         T_f = T_0 + T_i + T_acc;
    else if T_acc exists then
        Fuse the features of T_0 and T_acc by element-wise summation using
         T_f = T_0 + T_acc;
    Update the APCE_avg using Equation (17).
```

4. Experiments

We selected three UAV video benchmarks, UAV123, UAV20L [18] and UAVDT, for qualitative and quantitative experiments. The UAV123 consists of 123 shorter UAV aerial videos, and UAV20L consists of 20 longer video sequences. Compared with UAV123, UAV20L has more changes in appearance due to its long-time series, and there are many scenes in which the target completely disappears or reappears.

Success plot and precision plot both are important indicators used in the evaluation of this experiment. The success plot mainly reflects the overlap rate, that is, the proportion of the overlap area between the bounding box predicted by the tracker and the ground truth in the total area. When the overlap ratio exceeds this threshold, the tracking can be considered successful. We count the proportion of successful tracking frames to the total frames and draw the curve with this proportion as the ordinate. The precision plot is responsible for measuring the distance between the prediction result and the center of ground truth. When the distance between the center coordinate of the prediction result and the real position is less than this threshold, the tracking can be considered successful.

The evaluation is divided into two parts: overall evaluation and attribute-based evaluation. Attribute-based evaluation is responsible for evaluating the performance of the algorithm under different attributes, including aspect ratio change (ARC), background

cluster (BC), camera motion (CM), fast motion (FM), full occlusion (FOC), illumination variation (IV), low resolution (LR), out of view (OV), partial occlusion (POC), similar object (SOB), scale variation (SV), and viewpoint change (VC).

The UAVDT [38] dataset was proposed by Du et al. in 2018 for UAV video object detection and tracking, in which 50 video sequences are used for single-object tracking. There are eight attributes in UAVDT that are different from UAV123, including Camera Rotation (CR), Large Occlusion (LO), Small Object (SMO), Object Motion (OM), Background Clutter (BC), Scale Variations (SV), Illumination Variations (IV) and Object Blur (OB). The evaluation methods and indicators are consistent with those of UAV123.

4.1. Implementation Details

Our tracker was implemented with the Pytorch toolkit, which runs on the hardware environment with Intel(R) Xeon(R) Sliver 4110 2.1 GHz CPU and a single NVIDIA RTX2080 GPU.

In the training phase, we adopted a two-stage training method. In the first stage, we only trained the Siamese tracker without the FA module and the template library. The sample pairs were picked from ImageNet VID [39], GOT10K [40] and YouTube-BB datasets [41]. We chose the template patch and search region from the same video with an interval of less than 100. The template was cropped to a size of 127 px $\times$ 127 px centered on the target, and the search region was cropped to a size of 255 px $\times$ 255 px in the same manner. In addition, the affine transformation was also adopted. In this stage, the Siamese tracker was trained using the Adam [42] optimization algorithm with a batch size of 32 pairs in 20 epochs. The parameters of the feature extraction network were pretrained on ImageNet, and the first three convolution layers were fixed. The initial learning rate was set to 0.01, and there was no fine-tuning. In the second stage, the FA module and the template library were integrated into the Siamese tracker and an end-to-end training process was adopted. In this stage, pairs of input triplets (T_0, T_i, T_{acc}) from the same video sequence were required. The T_0 and T_i can be obtained directly from the same video sequence. For T_i, the ground truth was too accurate to reflect the predicted location in practice. Therefore, we added a random shift to the object so that it is not exactly at the center of the image. T_{acc} could be obtained by a standard linear update, three historical frames before T_i were selected for feature extraction and fusion. This can be formulated as follows:

$$T_{acc} = T_{i-1} + T_{i-2} + T_{i-3} \tag{18}$$

In addition, we use two backbones, AlexNet and ResNet, to obtain more fair results. The backbones in the feature extraction networks of TDSiam_Res50 and TDSiam_Alex are ResNet and AlexNet, respectively.

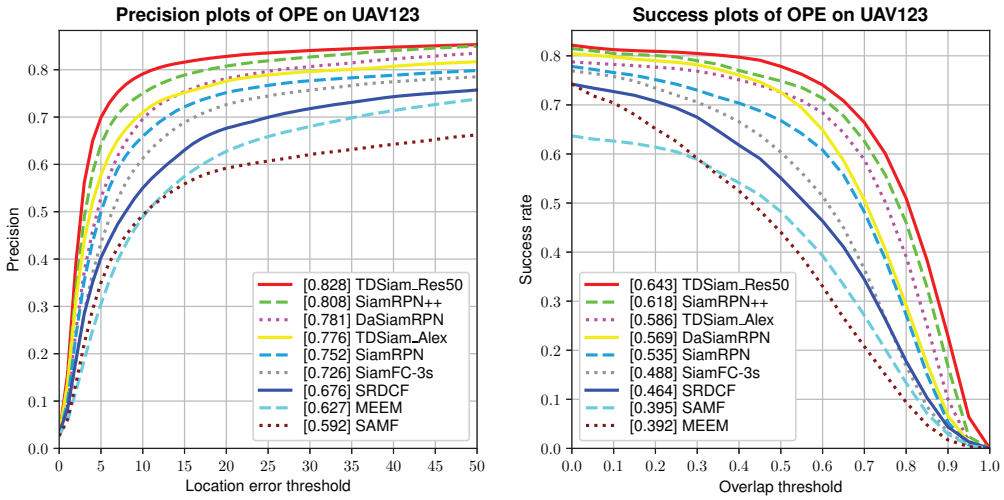
4.2. Experiments on the UAV123 Benchmark

We compared our method with SiamRPN [14], SiamRPN++ [28], DaSiamRPN [23], SiamFC-3s [12] and a few methods provided on the UAV123 benchmark, which include SRDCF [17], MEEM [43] and SAMF [44].

Overall Evaluation: The success plots and precision plots of the one-pass evaluation (OPE) are shown in Figure 6. TDSiam_Alex is observed to achieve a success score of 0.586 and a precision score of 0.776. Compared to the SiamRPN, TDSiam_Alex significantly outperforms in terms of the success score as well as precision score. In addition, we replaced the backbone of TDSiam with ResNet for performance comparison with SiamRPN++. The TDSiam_Res50 outperforms SiamRPN++ in terms of both metrics, TDSiam_Res50 achieves a substantial gain of 4% in terms of the success score.

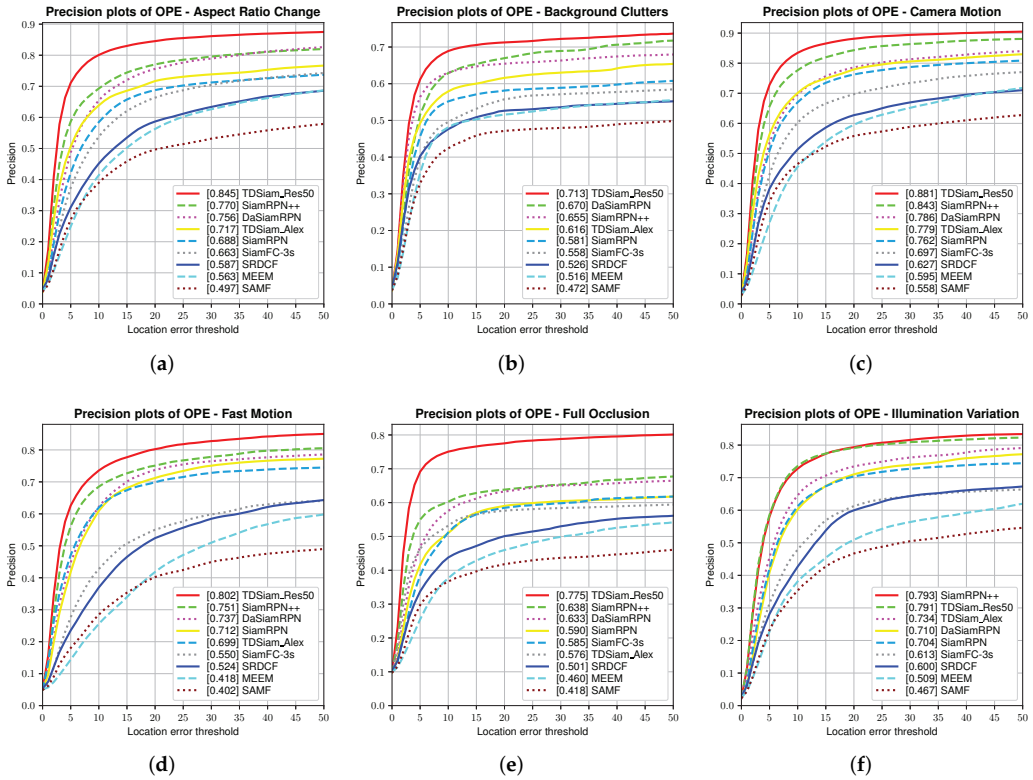
Attribute-Based Evaluation: The performance in the 12 attributes is shown in Figures 7 and 8, from which it can be seen that TDSiam_Res50 achieves superior performance. On the one hand, as for the precision score, our method achieves better results, including ARC (0.845), BC (0.713), CM (0.881), FM (0.802), FO (0.775), LR (0.701), OV (0.839), PO (0.806), SO (0.798), SV (0.821), VC (0.865). On the other hand, as for the success rate, our method also yields the

best performance, including ARC (0.652), BC (0.498), CM (0.682), FM (0.615), FO (0.520), IV (0.602), LR (0.485), OV (0.650), PO (0.600), SO (0.601), SV (0.639), VC (0.692). In particular, in several scenarios, ARC, CM and VC, our method achieved more significant performance gain, with 9.7%, 4.5%, 5.3% in precision score and 13%, 5.9%, 7.8% in success rate. To sum up, the attribute-based evaluation proves that the proposed method can significantly improve the performance in the scene where the appearance of the target changes.



(**a**) (**b**)

Figure 6. Experimental results of different methods on UAV123 dataset.The proposed algorithm TDSiam_Res50 performs favorably against other trackers. (**a**) Precision plots; (**b**) Success plots.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 7. *Cont.*

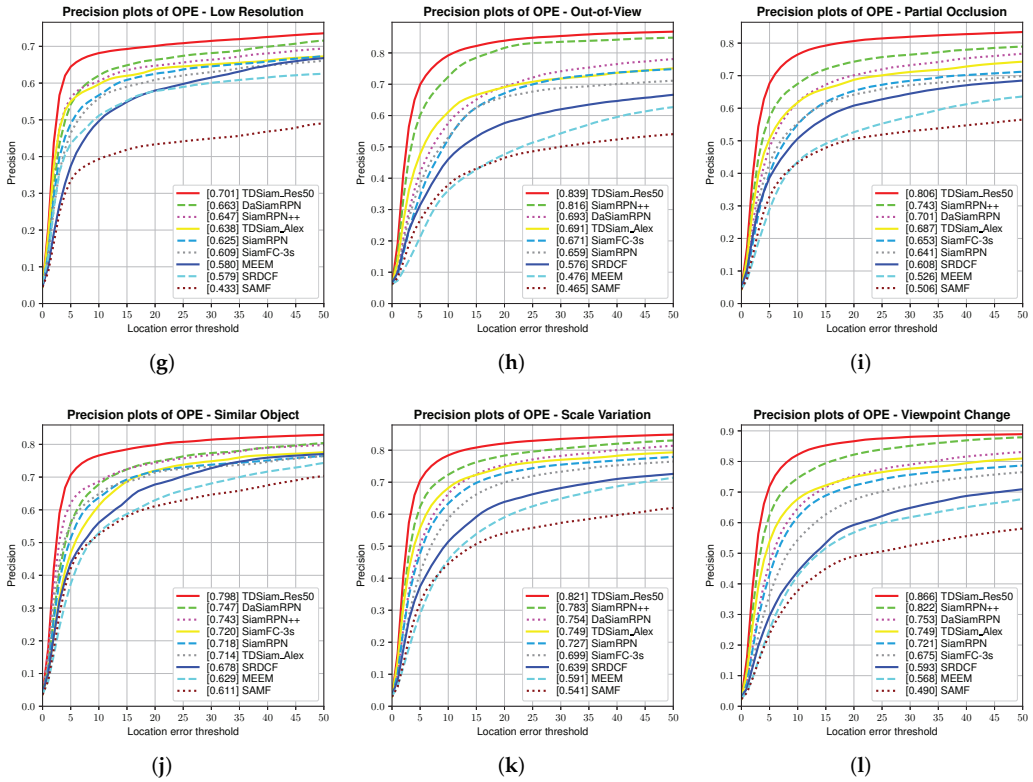


Figure 7. Precision plots corresponding to some typical scenarios. The UAV123 dataset was used for the evaluation. (**a**) Aspect Ratio Change; (**b**) Background Clutter; (**c**) Camera Motion; (**d**) Fast Motion; (**e**) Full Occlusion; (**f**) Illumination Variation; (**g**) Low Resolution; (**h**) Out of View; (**i**) Partial Occlusion; (**j**) Similar Object; (**k**) Scale Variation; (**l**) Viewpoint Change.

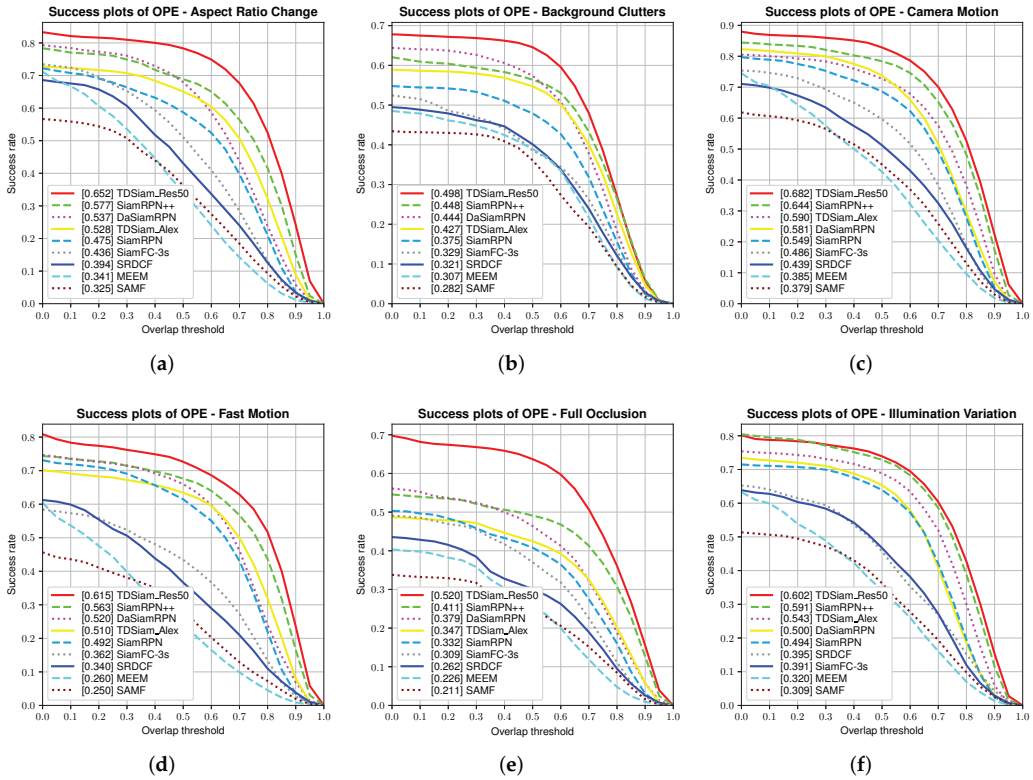


Figure 8. *Cont.*

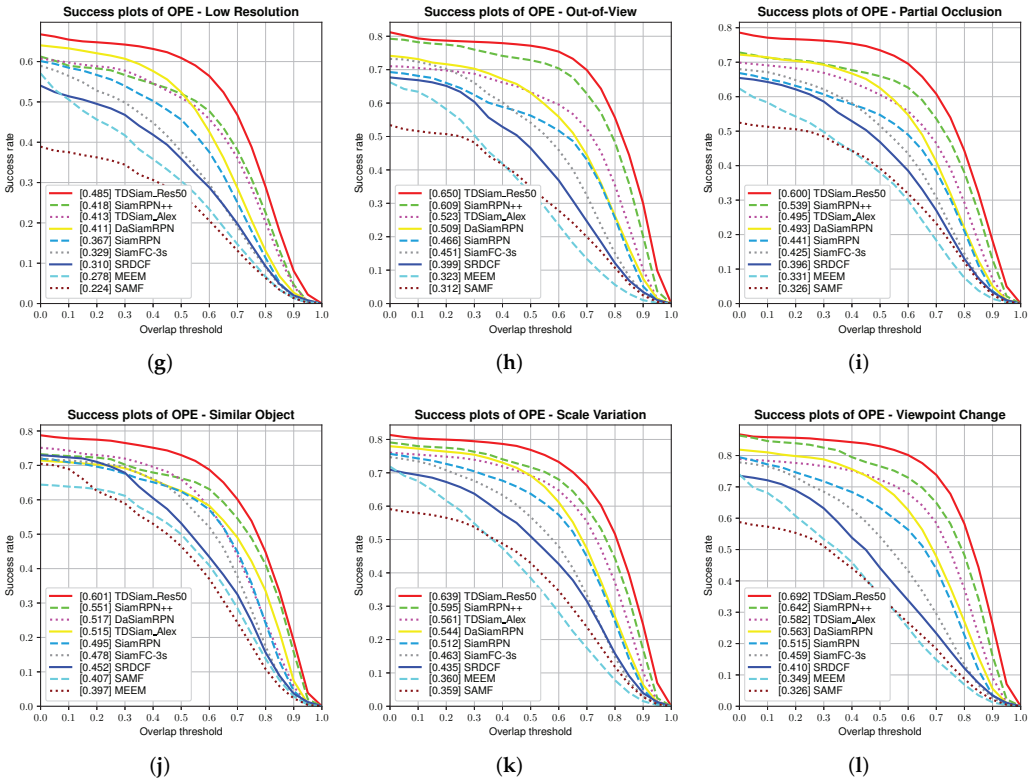


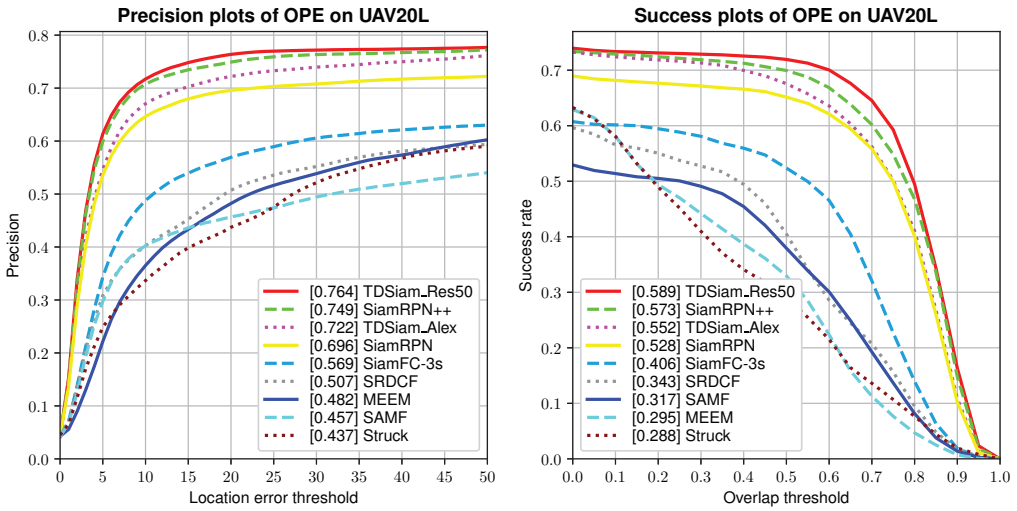
Figure 8. Success plots corresponding to some typical scenarios. The UAV123 dataset was used for the evaluation. (**a**) Aspect Ratio Change; (**b**) Background Clutter; (**c**) Camera Motion; (**d**) Fast Motion; (**e**) Full Occlusion; (**f**) Illumination Variation; (**g**) Low Resolution; (**h**) Out of View; (**i**) Partial Occlusion; (**j**) Similar Object; (**k**) Scale Variation; (**l**) Viewpoint Change.

4.3. Experiments on the UAV20L Benchmark

The UAV20L [18] dataset consists of 20 long-term challenging video sequences. There are more appearance changes in the whole video sequence, especially the accumulation of appearance changes for a long time. In the experiment, we compared our proposed method with SiamRPN++ [28], SiamRPN [14], SiamFC-3s [12], SRDCF [17], MEEM [43], SAMF [44] and Struck [45]. The indicators used in the evaluation are the same as those of UAV123.

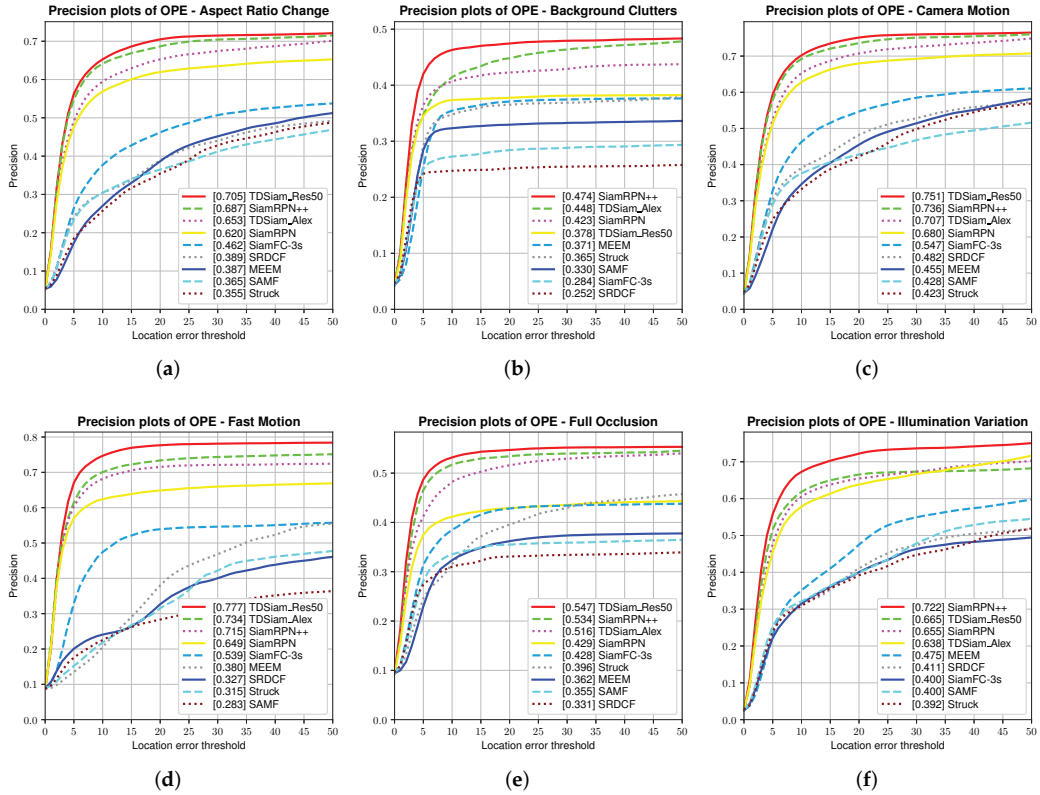
Overall Evaluation: As can be seen from Figure 9, the TDSiam_Res50 achieves the best performance in terms of precision score and success rate, which are 0.764 and 0.589. In particular, compared to the SiamRPN, our proposed method TDSiam_Alex achieves a substantial gain of 4.5% in terms of the success score.

Attribute-Based Evaluation: As shown in Figures 10 and 11, our proposed method TDSiam can performs better than other recent methods in some typical scenarios under aerial version. As for the precision score, our method TDSiam_Res50 also achieves better results, including ARC (0.705), CM (0.751), FM (0.777), FO (0.547), LR (0.590), OV (0.792), PO (0.738), SO (0.792), SV (0.751), VC (0.728). As for the success rate, our method also yields the best performance, including ARC (0.537), CM (0.576), FM (0.650), FO (0.360), LR (0.424), OV (0.611), PO (0.564), SO (0.643), SV (0.584), VC (0.589). However, in both scenarios, BV and IV, the performance of our method is still unsatisfactory. The reason for this may be that the template-matching methods such as SiamRPN++, SiamRPN and SiamFC do not have good tracking performance in complex backgrounds.



(**a**) (**b**)

Figure 9. Experimental results of different methods on the UAV20L dataset. The proposed algorithmTDSiam_Res50 performs favorably against state-of-the-art trackers. (**a**) Precision plots; (**b**) Success plots.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

Figure 10. *Cont.*

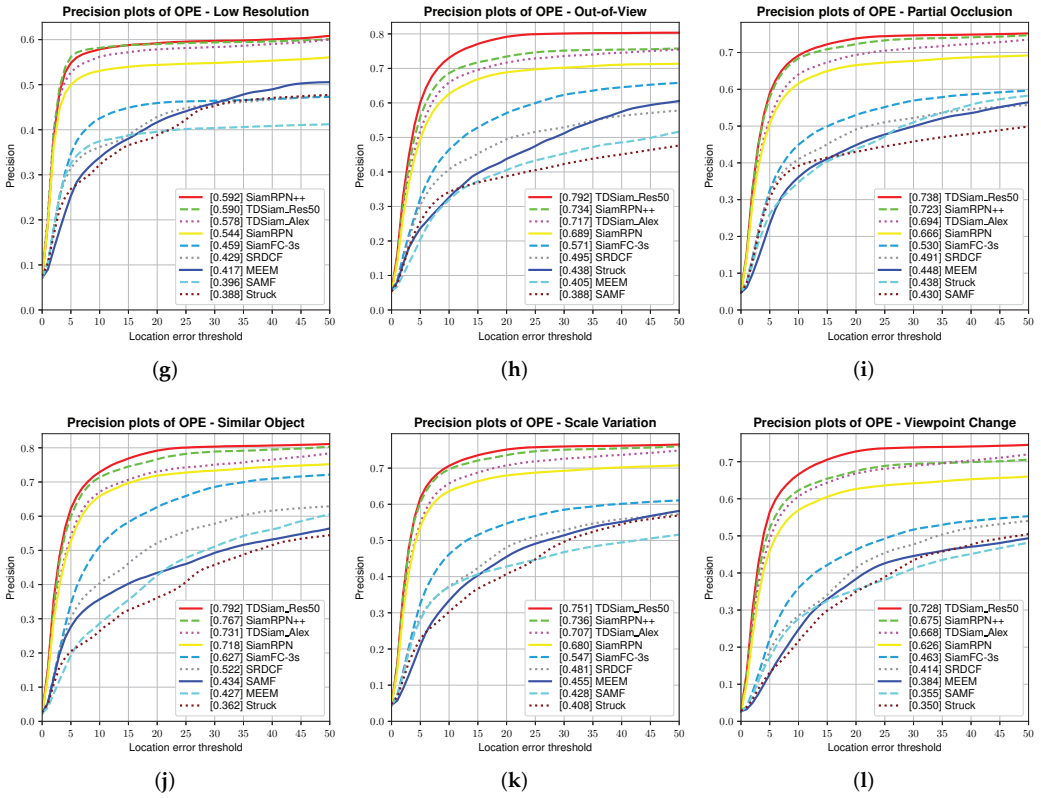


Figure 10. Precision plots corresponding to some different scenarios. The UAV20L dataset was used for the evaluation. (**a**) Aspect Ratio Change; (**b**) Background Clutter; (**c**) Camera Motion; (**d**) Fast Motion; (**e**) Full Occlusion; (**f**) Illumination Variation; (**g**) Low Resolution; (**h**) Out of View; (**i**) Partial Occlusion; (**j**) Similar Object; (**k**) Scale Variation; (**l**) Viewpoint Change.

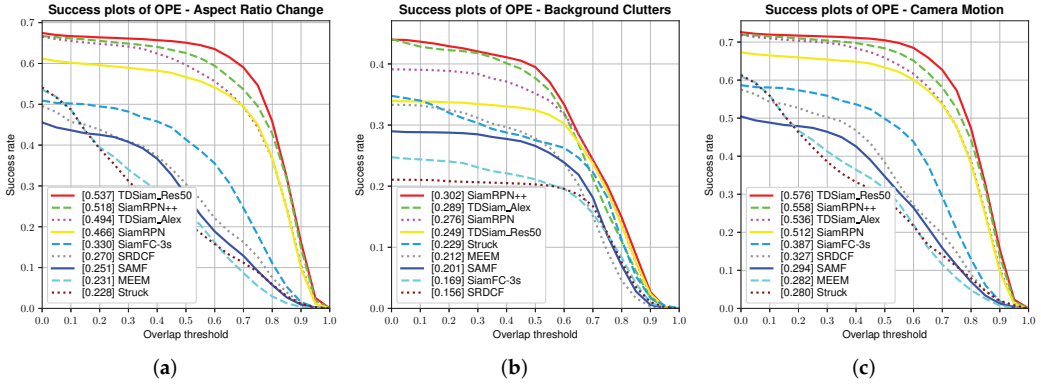


Figure 11. *Cont.*

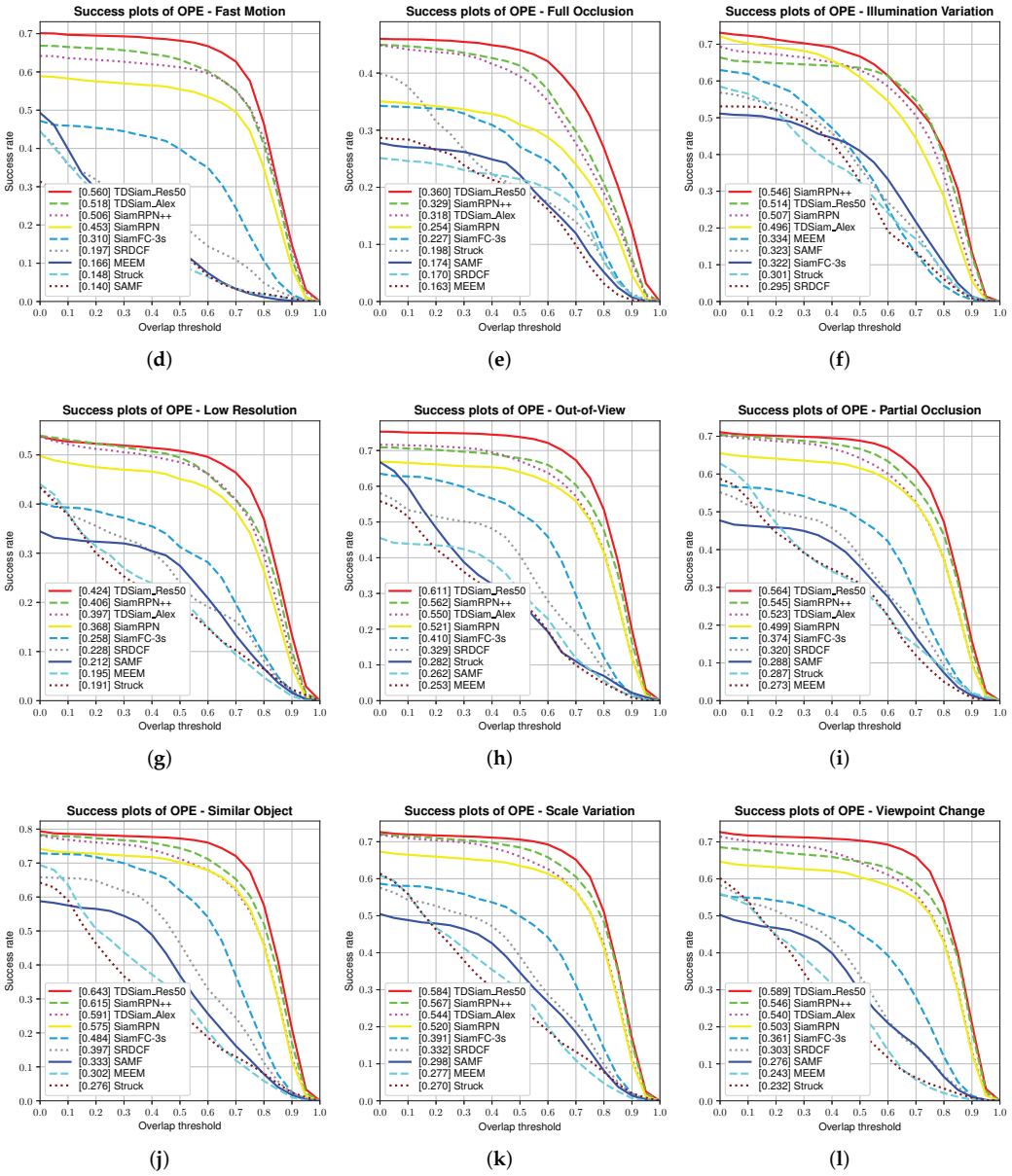


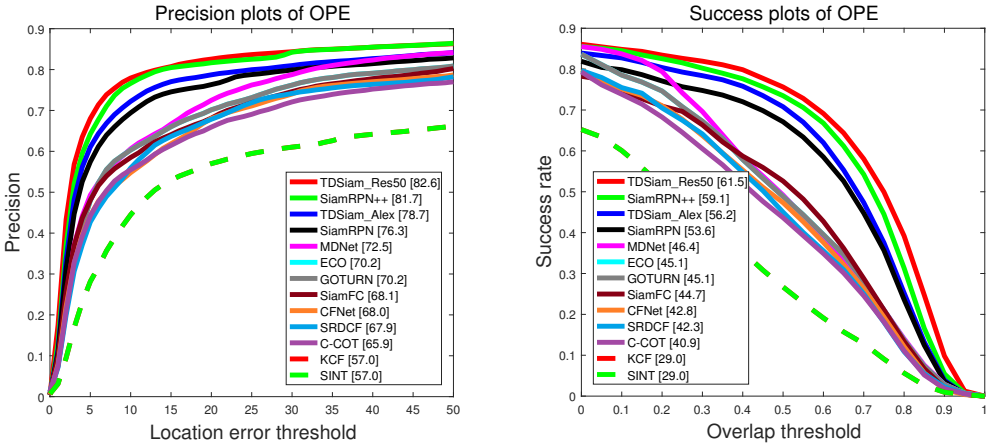
Figure 11. Success plots corresponding to some different scenarios. The UAV20L dataset was used for the evaluation. (**a**) Aspect Ratio Change; (**b**) Background Clutter; (**c**) Camera Motion; (**d**) Fast Motion; (**e**) Full Occlusion; (**f**) Illumination Variation; (**g**) Low Resolution; (**h**) Out of View; (**i**) Partial Occlusion; (**j**) Similar Object; (**k**) Scale Variation; (**l**) Viewpoint Change.

4.4. Experiments on the UAVDT Benchmark

In the experiment, the algorithms involved in the comparisons are SiamRPN [14], SiamRPN++ [28], SiamFC [12], and some methods provided by the UAVDT benchmark, including MDNet [13], ECO, GOTURN[20], CFNet [21], SRDCF [46], C-COT [47], KCF and SINT [19].

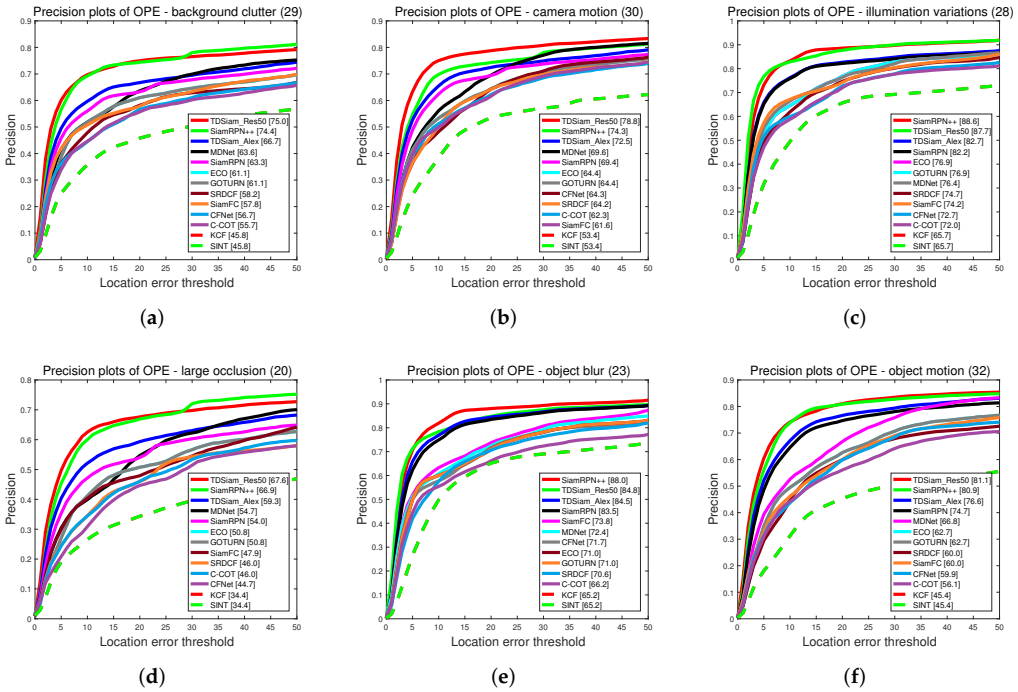
Overall Evaluation: The precision plots and Success plots are shown in Figure 12. Our proposed method TDSiam achieved better performance under both the AlexNet and ResNet backbones. The TDSiam_Alex achieved a success rate of 0.562 and a precision score of 0.787. Compared with the SiamRPN, the success rate is improved by 4.9% and the precision score is improved by 3.1%. In addition, TDSiam_Res50 also achieved the best results. The success rate and precision score exceeded SiamRPN++, reaching 0.615 and 0.826, respectively.

Attribute-Based Evaluation: The performance is evaluated under eight attributes and the comparison results are shown in Figures 13 and 14. In terms of success rate, the TDSiam_Res50 achieved the best performance under all the attributes, which were BC (0.548), CM (0.587), IV (0.66), LO (0.521), OB (0.627), OM (0.600), SMO (0.610), SV (0.616). In particular, compared with SiamRPN++, the success rate under CM, SMO and IV was increased by 9.3%, 6.3% and 5.3%, respectively. In addition, compared with the baseline tracker SiamRPN, the TDSiam_Alex also achieved better performance under all the attributes. As for precision score, the TDSiam_Res50 achieved the best performance under six attributes, which were BC (0.750), CM (0.788), LO (0.676), OM (0.811), SMO (0.879), SV (0.808), respectively. In addition, compared with the baseline tracker SiamRPN, TDSiam_Alex achieved better performance under all the attributes.



(**a**) (**b**)

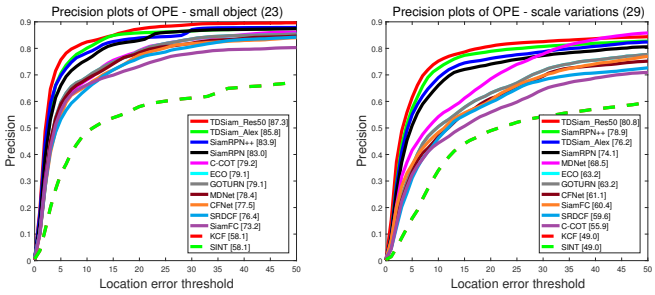
Figure 12. Experimental results of using different methods on the UAVDT dataset. (**a**) Precision plots; (**b**) Success plots.



(**a**) (**b**) (**c**)

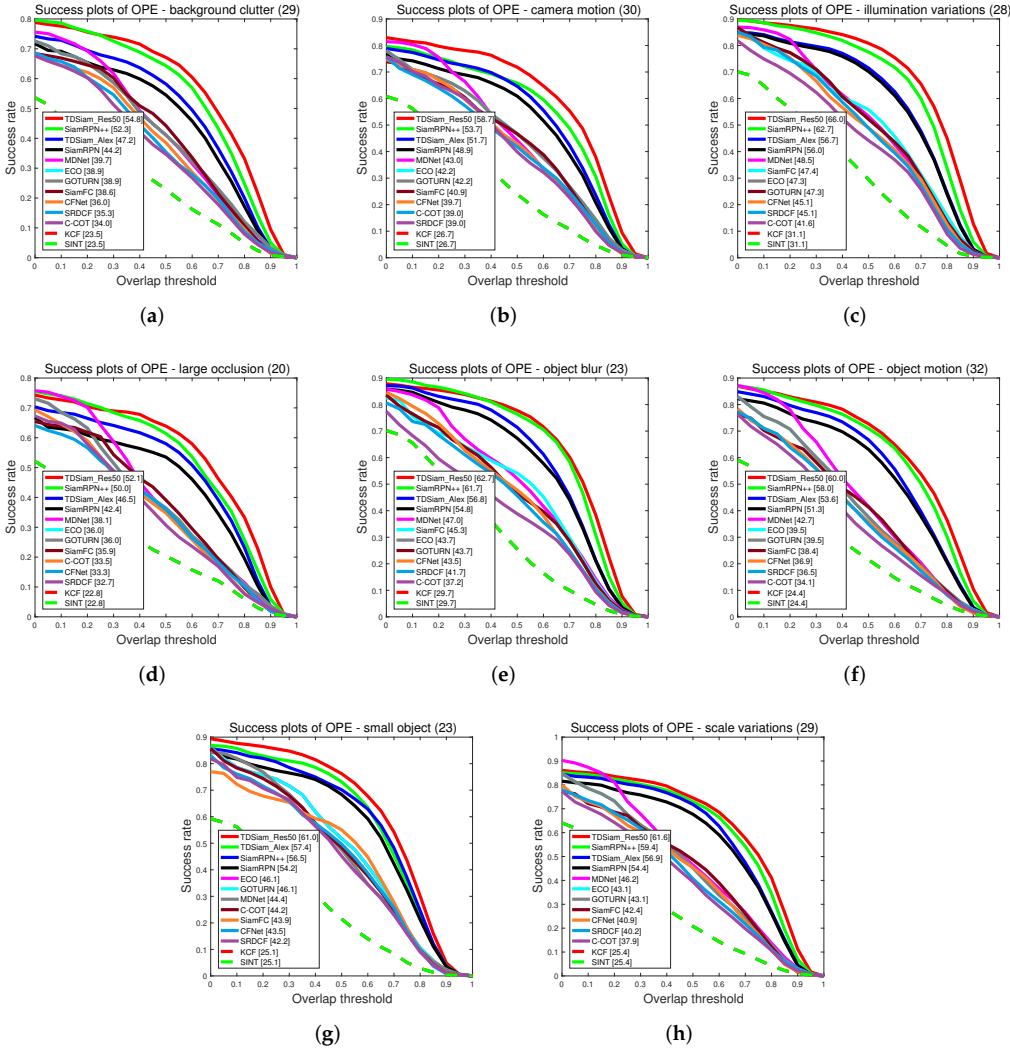
(**d**) (**e**) (**f**)

Figure 13. *Cont.*



(**g**) (**h**)

Figure 13. Precision plots corresponding to some different scenarios. The UAVDT dataset was used for the evaluation. (**a**) Background Clutter; (**b**) Camera Motion; (**c**) Illumination Variations; (**d**) Large Occlusion; (**e**) Object Blur; (**f**) Object Motion; (**g**) Small Object; (**h**) Scale Variations.



(**a**) (**b**) (**c**)

(**d**) (**e**) (**f**)

(**g**) (**h**)

Figure 14. Success plots corresponding to some different scenarios. The UAVDT dataset was used for the evaluation. (**a**) Background Clutter; (**b**) Camera Motion; (**c**) Illumination Variations; (**d**) Large Occlusion; (**e**) Object Blur; (**f**) Object Motion; (**g**) Small Object; (**h**) Scale Variations.

4.5. Qualitative Evaluation

In order to present the tracking performance more intuitively, we carried out qualitative experiments with the other algorithms, including SiamRPN++ [28], DaSiamRPN [23],

SiamRPN [14] and SiamFC-3s [12], in the UAV123 dataset. We selected five typical video sequences in UAV123 for analysis, as shown in Figure 15, including VC, CM, IV, SV and other common attributes.

boat3: The boat3 video sequence contains a typical view-changing scene. At the 114th frame, the performance gap of all tracking algorithms is not obvious. From the 331st to the 871st frame, the appearance and the scale of the target change slowly, our proposed method TDSiam_Res50 can still track the target stably, while the tracking performance of other methods has declined.

car2_s: This video sequence is a process of continuous rotation and rolling of a vehicle, and the appearance of the target changes dramatically due to the change of illumination in the scene. In the first 48 frames, the appearance of the target has not changed significantly, and all methods can track continuously. However, in the 176th and 275th frames, other algorithms have failed to trace completely due to rotation and illumination changes, but our method can still locate the target effectively.

car4: The car4 sequence shows the whole process of a car being blocked and then reappearing. In the 404th frame, all trackers lost their target due to occlusion and the DaSiamRPN even drifted to another car. In the rest of the video sequence, only our method TDSiam_Res50 can find the target again and track it continuously.

car12: In this video sequence, a car passes through a straight road, which is blocked by many trees. In the whole video sequence, only our method can continuously complete the tracking, and other algorithms cannot track after losing the target. In the 174th frame, the DaSiamRPN finds the target again, but the scale estimation is not accurate.

truck2: The scale of the target in the truck2 sequence is small, and the scene also contains many similar buildings. From the 112nd frame, the tracking performance of other algorithms is no longer stable, especially after the occlusion of the 189th frame, the other algorithms even lose the target. Fortunately, our method TDSiam_Res50 can recover the tracking performance more quickly.

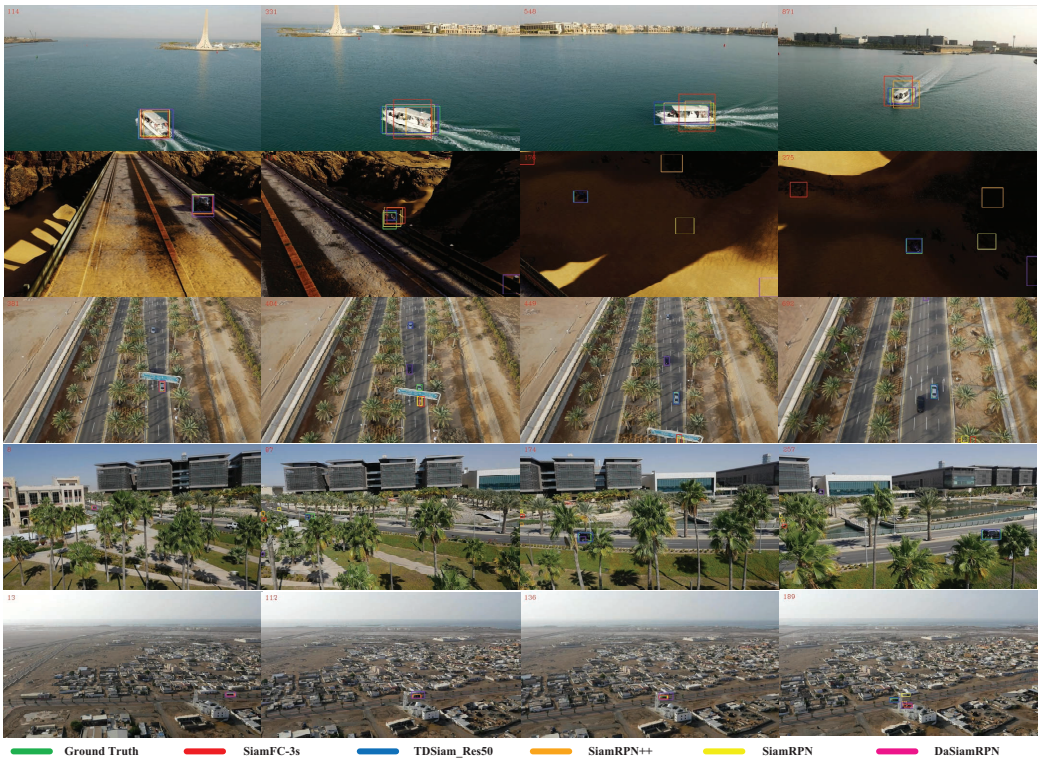


Figure 15. Qualitative evaluation of the proposed TDSiam_Res50 and other state-of-art trackers on the UAV123 [18] dataset. From left to right and top to down are the tracking results on the videos of boat3, car2_s, car4, car12 and truck2.

4.6. Ablation Study

To further verify the effectiveness of the each module, we conducted an ablation study using the UAV123 dataset. It aims to show which module contributes to the overall performance of the tracker. As shown in Table 2, the symbol ✓represents that a particular module has been used, whereas the symbol × indicates that the module has not been used.

Table 2. Ablation study of the proposed tracker on UAV123. We compared the performance of partial module deactivation, ✓and × represent module in use or not, respectively.

ID	TL	APCE	FA	Success	Precision
①	×	×	×	0.618	0.808
②	✓	×	×	0.598	0.802
③	✓	✓	×	0.630	0.815
④	✓	✓	✓	0.643	0.828

As shown in the second row of Table 2, the performance of the proposed template-driven Siamese network is worse than that of baseline because the interference is easy to introduce once the target is occluded. In other words, the template-driven Siamese network using the polluted template may harm the tracking performance. To solve this problem, we propose an effective template updating strategy in this paper. By adding the APCE index, the template-driven Siamese network can recognize whether the target is occluded or not. When the target is not occluded, the template library can bring performance gains to the tracker by updating the template. This strategy is indispensable for the template-driven Siamese network and guarantees that the introduced template is not polluted. On the contrary, the tracking performance will be worse than the baseline if the template library does not use the template updating strategy with the APCE index.

APCE boosts the performance significantly with a score of 0.630 in terms of success rate, exceeding baseline by 1.3%. As shown in the last row of the table, on incorporating the FA module, our proposed method achieves a success score of 0.643 and a precision score of 0.828, which leads to a performance improvement of 4% and 2.5%, respectively. Experimental results show that template library (TL), feature aligned module (FA) and average peak-to-correlation energy (APCE) all play an important role in the algorithm.

4.7. Speed Performance

Figure 16 shows the speed performance of different methods, including SiamRPN++ [28], DaSiamRPN [23], SiamRPN [14], SiamFC-3s [12], SRDCF [17], MEEM [43] and SAMF [44]. Due to the addition of template library, the tracking speed of our proposed method decreases slightly compared with the SiamRPN++ and SiamRPN. Although our method TDSiam_Res50 is not the fastest, it can still meet the real-time requirements and reach 35 fps.

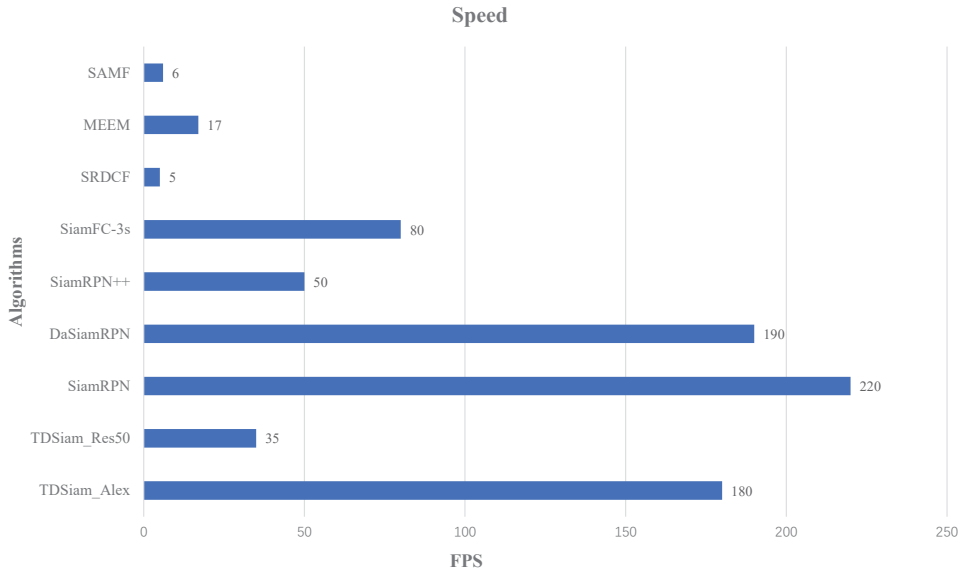


Figure 16. Speed performance of the proposed method and other state-of-art trackers on the UAV123 dataset.

5. Conclusions

To improve the performance of the visual object tracking in unmanned aerial vehicle videos, this paper has proposed a template-driven Siamese network (TDSiam), in which a template library has been integrated into the Siamese network for solving the problem that the appearance of the target in UAV videos often changes. A feature alignment module is proposed to fuse the features from the template library more efficiently. Moreover, we have designed a template updating strategy to guarantee the effectiveness of templates in the inference stage. Performance comparisons on three challenging UAV video benchmarks, including UAV123, UAV20L and UAVDT, have demonstrated that the proposed approach can bring significant performance improvement. It also provides help for the application of UAVs in practical tracking scenarios. However, there are some problems, such as the long-term tracking performance degradation, which still need to be solved. To summarize, the proposed approach has room for development and improvement, and future research should focus on exploring the trajectory prediction and redetection strategy for UAV tracking.

Author Contributions: All authors participated in devising the tracking approach and made significant contributions to this work. L.S. and Z.Y. devised the approach and performed the experiments; J.Z. and Z.F. provided advice for the preparation and revision of the work; and Z.H. helped with the experiments. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Thirteen-Five Equipment Pre-Research Foundation of China (No. 61403120207), the Aeronautical Science Foundation of China (No. 20185142003), Natural Science Foundation of Henan Province, China (No. 222300420433), the Science and Technology Innovative Talents in Universities of Henan Province (No. 21HASTIT030) and Young Backbone Teachers in Universities of Henan Province (No. 2020GGJS073).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Fu, C.; Lin, F.; Li, Y.; Chen, G. Correlation filter-based visual tracking for UAV with online multi-feature learning. *Remote Sens.* **2019**, *11*, 549. [CrossRef]
2. Zhang, S.; Zhuo, L.; Zhang, H.; Li, J. Object tracking in unmanned aerial vehicle videos via multifeature discrimination and instance-aware attention network. *Remote Sens.* **2020**, *12*, 2646. [CrossRef]
3. Zhuo, L.; Liu, B.; Zhang, H.; Zhang, S.; Li, J. MultiRPN-DIDNet: Multiple RPNs and Distance-IoU Discriminative Network for Real-Time UAV Target Tracking. *Remote Sens.* **2021**, *13*, 2772. [CrossRef]
4. Xue, X.; Li, Y.; Dong, H.; Shen, Q. Robust correlation tracking for UAV videos via feature fusion and saliency proposals. *Remote Sens.* **2018**, *10*, 1644. [CrossRef]
5. Wu, Y.; Lim, J.; Yang, M.H. Online object tracking: A benchmark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, 23–28 June 2013; pp. 2411–2418.
6. Zhu, Z.; Wu, W.; Zou, W.; Yan, J. End-to-end flow correlation tracking with spatial-temporal attention. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 548–557.
7. Zhang, K.; Zhang, L.; Yang, M.H. Fast compressive tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *36*, 2002–2015. [CrossRef] [PubMed]
8. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550.
9. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. Exploiting the circulant structure of tracking-by-detection with kernels. In Proceedings of the European Conference on Computer Vision, Proceedings of the Computer Vision–ECCV 2012, 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; Springer: Berlin/Heidelberg, Germany, 2012; pp. 702–715.
10. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *37*, 583–596. [CrossRef] [PubMed]
11. Wang, N.; Yeung, D.Y. Learning a deep compact image representation for visual tracking. In Proceedings of the Advances in Neural Information Processing Systems, Harrahs, NV, USA, 5–10 December 2013; pp. 809–817.
12. Wang, L.; Ouyang, W.; Wang, X.; Lu, H. Visual tracking with fully convolutional networks. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 3119–3127.
13. Nam, H.; Han, B. Learning multi-domain convolutional neural networks for visual tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 4293–4302.
14. Li, B.; Yan, J.; Wu, W.; Zhu, Z.; Hu, X. High performance visual tracking with siamese region proposal network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8971–8980.
15. Danelljan, M.; Bhat, G.; Shahbaz Khan, F.; Felsberg, M. Eco: Efficient convolution operators for tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6638–6646.
16. Bertinetto, L.; Valmadre, J.; Henriques, J.F.; Vedaldi, A.; Torr, P.H. Fully-convolutional siamese networks for object tracking. In Proceedings of the European Conference on Computer Vision, Computer Vision–ECCV 2016 Workshops, Amsterdam, The Netherlands, 8–10 and 15–16 October 2016; Springer: Cham, Switzerland, 2016; pp. 850–865.
17. Danelljan, M.; Hager, G.; Shahbaz Khan, F.; Felsberg, M. Convolutional features for correlation filter based visual tracking. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 58–66.
18. Mueller, M.; Smith, N.; Ghanem, B. A benchmark and simulator for uav tracking. In Proceedings of the European Conference on Computer Vision, Computer Vision–ECCV 2016, 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 445–461.
19. Tao, R.; Gavves, E.; Smeulders, A.W. Siamese instance search for tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 1420–1429.
20. Held, D.; Thrun, S.; Savarese, S. Learning to track at 100 fps with deep regression networks. In Proceedings of the European Conference on Computer Vision, Computer Vision–ECCV 2016, 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 749–765.
21. Valmadre, J.; Bertinetto, L.; Henriques, J.; Vedaldi, A.; Torr, P.H. End-to-end representation learning for correlation filter based tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2805–2813.
22. Wang, Q.; Teng, Z.; Xing, J.; Gao, J.; Hu, W.; Maybank, S. Learning attentions: residual attentional siamese network for high performance online visual tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4854–4863.
23. Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; Hu, W. Distractor-aware siamese networks for visual object tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 101–117.
24. Wang, M.; Liu, Y.; Huang, Z. Large margin object tracking with circulant feature maps. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4021–4029.
25. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In Proceedings of the Advances in Neural Information Processing Systems, Harrahs, NV, USA, 3–8 December 2012; pp. 1097–1105.

26. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
27. Zhang, Z.; Peng, H. Deeper and wider siamese networks for real-time visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4591–4600.
28. Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; Yan, J. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4282–4291.
29. Yu, Y.; Xiong, Y.; Huang, W.; Scott, M.R. Deformable siamese attention networks for visual object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 6728–6737.
30. Zhang, L.; Gonzalez-Garcia, A.; Weijer, J.v.d.; Danelljan, M.; Khan, F.S. Learning the model update for siamese trackers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4010–4019.
31. Xu, Y.; Wang, Z.; Li, Z.; Yuan, Y.; Yu, G. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 12549–12556.
32. Guo, D.; Wang, J.; Cui, Y.; Wang, Z.; Chen, S. SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 6269–6277.
33. Chen, Z.; Zhong, B.; Li, G.; Zhang, S.; Ji, R. Siamese box adaptive network for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 6668–6677.
34. Huang, Z.; Fu, C.; Li, Y.; Lin, F.; Lu, P. Learning aberrance repressed correlation filters for real-time uav tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Seoul, Korea, 27–28 October 2019; pp. 2891–2900.
35. Yao, S.; Han, X.; Zhang, H.; Wang, X.; Cao, X. Learning Deep Lucas-Kanade Siamese Network for Visual Tracking. *IEEE Trans. Image Process.* **2021**, *30*, 4814–4827. [CrossRef] [PubMed]
36. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
37. Lin, T.Y.; Dollár, P.; Girshick, R.; He, K.; Hariharan, B.; Belongie, S. Feature pyramid networks for object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 2117–2125.
38. Du, D.; Qi, Y.; Yu, H.; Yang, Y.; Duan, K.; Li, G.; Zhang, W.; Huang, Q.; Tian, Q. The unmanned aerial vehicle benchmark: Object detection and tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 370–386.
39. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
40. Huang, L.; Zhao, X.; Huang, K. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1562–1577. [CrossRef] [PubMed]
41. Real, E.; Shlens, J.; Mazzocchi, S.; Pan, X.; Vanhoucke, V. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 5296–5305.
42. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
43. Zhang, J.; Ma, S.; Sclaroff, S. MEEM: robust tracking via multiple experts using entropy minimization. In Proceedings of the European Conference on Computer Vision, Computer Vision–ECCV 2014, 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 188–203.
44. Li, Y.; Zhu, J. A scale adaptive kernel correlation filter tracker with feature integration. In Proceedings of the European Conference on Computer Vision, Computer Vision–ECCV 2014, 13th European Conference, Zurich, Switzerland, 6–12 September 2014; Springer: Cham, Switzerland, 2014; pp. 254–265.
45. Hare, S.; Golodetz, S.; Saffari, A.; Vineet, V.; Cheng, M.M.; Hicks, S.L.; Torr, P.H. Struck: Structured output tracking with kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *38*, 2096–2109. [CrossRef] [PubMed]
46. Danelljan, M.; Hager, G.; Shahbaz Khan, F.; Felsberg, M. Learning spatially regularized correlation filters for visual tracking. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4310–4318.
47. Danelljan, M.; Robinson, A.; Khan, F.S.; Felsberg, M. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In Proceedings of the European Conference on Computer Vision, Computer Vision–ECCV 2016, 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; Springer: Cham, Switzerland, 2016; pp. 472–488.