Article

Not peer-reviewed version

Enhancing the Scale Adaptation of Global Tracker for Infrared UAV Tracking

Zicheng Feng , Wenlong Zhang * , Erting Pan , Donghui Liu , Qifeng Yu

Posted Date: 11 June 2025

doi: 10.20944/preprints202506.0960.v1

Keywords: infrared target tracking; scale adaptation; global tracker; UAV tracking; deep learning

Article

Enhancing the Scale Adaptation of Global Tracker for Infrared UAV Tracking

Zicheng Feng [1,2], Wenlong Zhang [1,2,*], Erting Pan [1,2], Donghui Liu [1,2] and Qifeng Yu [1,2]

[1] College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China

[2] Hunan Provincial Key Laboratory of Image Measurement and Vision Navigation, Changsha 410073, China

* Correspondence: wenlong@nudt.edu.cn

Abstract: Tracking an unmanned aerial vehicle (UAV) in the infrared video is an essential technology for the anti-UAV task. Given the frequent UAV target disappearance caused by occlusion or moving out of view, the global tracker, which has the unique advantage of recapturing the target, is widely used in infrared UAV tracking. However, the global tracker performs poorly when dealing with large target scale variation, because it cannot maintain the approximate consistency between the target sizes in the template and the search region. To enhance the scale adaptation of global trackers, we propose a plug-and-play scale adaptation enhancement module (SAEM). It can generate a scale adaptation enhancement kernel according to the target size in the previous frame, and then perform implicit scale adjustment on the extracted target template features. To optimize the training, we introduce an auxiliary branch to supervise the learning of SAEM, and add Gaussian noise to the input size to improve the robustness of SAEM. In addition, we propose a one-stage anchor-free global tracker (OSGT), which has a more concise structure than other global trackers to meet the real-time requirement. Extensive experiments on three Anti-UAV Challenge datasets and the Anti-UAV410 dataset demonstrate the superior performance of our method, and verify that our proposed SAEM can effectively enhance the scale adaptation of existing global trackers.

Keywords: infrared target tracking; scale adaptation; global tracker; UAV tracking; deep learning

1. Introduction

Infrared UAV tracking is to continuously locate a specific UAV in an infrared video and estimate its scale [1]. Benefiting from the excellent imaging ability of infrared cameras under all-weather and low-light conditions, this technology is widely applied to the anti-UAV task to protect individual privacy and public safety [2,3]. In essence, infrared UAV tracking belongs to the research field of single object tracking [4]. However, different from tracking generic objects such as pedestrians and vehicles, tracking high-speed moving UAVs in infrared videos needs to deal with more challenges:

(1) The infrared UAV tracking is susceptible to occlusion, thermal crossover and interference in complex scenarios such as trees, buildings, heavy clouds, and strong clutter.

(2) Due to the rapid movement of the UAV target or the instability of the infrared camera platform, the position of the UAV target will change drastically between two adja-

cent frames or even move out of view.

(3) The target scale variation is dramatic when the camera adjusts its focal length or the target moves rapidly closer or farther away, especially in the UAV-to-UAV task [5].

According to the size of the search region, single object trackers can be categorized into local trackers [6,7] and global trackers [8–10]. **The local tracker** crops a small patch from the current frame (Frame T) as the search region according to the target position and size in the previous frame, as shown in Figure 1(a). This is beneficial to reduce the computation and the background clutter. However, the local tracker relies on tracking a target stably. When the target disappears, it is liable to

 doi:10.20944/preprints202506.0960.v1

select the wrong local search region, which leads to the failure of subsequent tracking. **The global tracker** uses the whole current frame as the search region, as shown in Figure 1(b), so it has the advantage of recapturing the disappeared target. For the above challenge (1) and (2), where targets frequently disappear due to occlusion or moving out of view, global trackers demonstrate greater robustness than local trackers, making them the preferred choice for the infrared UAV tracking. Table 1 shows the comparison of the two categories of trackers.

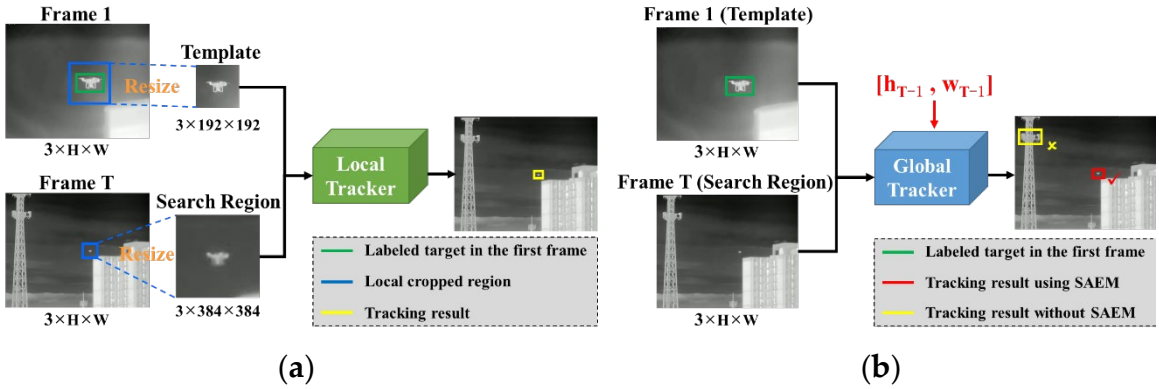


Figure 1. Comparison of the local tracker and the global tracker in handling target scale variation. (**a**) The local tracker can use the resizing operation in the preprocessing to cope with target scale variation; (**b**) The global tracker performs poorly when dealing with target scale variation. The red arrow indicates that our method uses the previous target size to implicitly resize the template to improve the scale adaptation.

Table 1. Comparison of local and global trackers in the infrared UAV tracking task.

Scheme	Search region	Efficiency	Scale adaptation	Occlusion or moving out of view	Fast target or camera motion
Local tracker	Local patch	High	✓	×	×
Global tracker	Whole frame	Low	×	✓	✓

For challenge (3), the key to achieving scale adaptation is to maintain the approximate consistency between the sizes of the target in the template and the search region, which is beneficial to match the features of both for robust tracking. Traditional local trackers [11,12] employ pyramid sampling to obtain multi-scale search regions to deal with the target scale variation. Local trackers [13,14] based on deep learning adjust the template and the search region to the fixed sizes by resizing operation, as shown in Figure 1(a). This operation makes the target size in the search region approximately consistent with the target size in the template, which avoids the undesirable effect of target scale variation on tracking. However, global trackers cannot follow the above methods. On the one hand, if the whole current frame is resized according to the target scale change factor, the size of the image input to the model will increase dramatically, which leads to a sharp increase in the computation. On the other hand, if the target template is directly resized, the features of the resized template must be extracted every frame during online tracking, which also leads to a serious reduction in the running efficiency. Therefore, the existing global trackers have poor scale adaptation and cannot handle the target scale variation well, especially when the scale changes drastically. As shown in Figure 1(b), the global tracker incorrectly identifies the tower structure as the target, because the labeled target in the template is more approximate in size to the tower structure rather than the ground truth in the search region.

Inspired by the scale-arbitrary image super-resolution [15,16], which can adaptively adjust model parameters to generate image features based on the required scaling factor, we propose an implicit scale adjustment method for the target template using the target size in the previous frame as a guide, as shown in Figure 1(b). Specifically, we propose a scale adaptation enhance module

 doi:10.20944/preprints202506.0960.v1

(SAEM), which can generate a scale adaptation enhance kernel according to the previous target size, and then directly process the scale information of the extracted target template features. This method has less computation than the explicit method, which need to first resize the target template and then extract its features every frame. During training, we set up an auxiliary branch that adopts the above explicit method to supervise the learning of SAEM, and we also add Gaussian noise to the input size to improve the robustness of SAEM to handle the inaccurate input size in complex scenarios. During the online tracking, an adaptive threshold is proposed to more accurately judge whether the target is disappearing. This is used to prevent SAEM from receiving the incorrect size input caused by target disappearance. Moreover, SAEM is a plug-and-play module, which can be embedded into other global trackers to enhance their scale adaptation, especially when the typical scale change is more than 10× in infrared UAV tracking.

In addition, we propose a concise one-stage anchor-free global tracker (OSGT) to meet the real-time requirement of the infrared UAV tracking task. It is superior to the complex and inefficient two-stage anchor-based structure commonly used in existing global trackers [10,17]. In detail, it combines the feature fusion module of GlobalTrack [8] improved by using hierarchical cross-correlation and the output head of FCOS [18] without the centerness branch. OSGT can run in real time at 30.9 fps in the infrared video with a resolution of 512×640. After SAEM is embedded, it can still run at 27.3 fps.

In summary, the main contributions of the work in this paper are:

(1) We propose a plug-and-play scale adaptation enhancement module, which can implicitly resize the target template to enhance the scale adaptation of existing global trackers for the infrared UAV tracking task.

(2) During the training, we design an auxiliary branch to supervise the learning of SAEM and add Gaussian noise to the input size to enhance its robustness. During the online tracking, an adaptive threshold is proposed to accurately judge target disappearance and avoid SAEM being affected by the incorrect input size.

(3) We propose a one-stage anchor-free global tracker with a simpler structure, which can track UAVs in real time.

The rest of this paper is organized as follows. In Section 2, we briefly review the related works. In Section 3, we describe the proposed method in detail. In Section 4, experimental results are presented and analyzed. Finally, we conclude this paper in Section 5.

2. Related Work

This section focuses on approaches closely related to our work, including single object tracking, global tracker, infrared UAV tracking and scale-arbitrary image super-resolution.

2.1. Single Object Tracking

Single object tracking requires a target labeled in the first frame to be continuously located and scale-estimated in subsequent video frames. The target types are not limited, and the types of videos are usually RGB and infrared. Traditional target tracking algorithms use particle filtering [19], structured SVM [20] and correlation filter [21]. Among them, the correlation filter tackles the visual tracking by solving the ridge regression in Fourier domain, which has been widely developed because of its high efficiency. Benefiting from the development of deep learning, researchers have designed many high-performance deep trackers. These trackers usually adopt the Siamese network [22,23] to extract the features of the target template and the search region respectively, and then fuse them to search the target. According to the feature fusion methods, deep trackers can be roughly categorized into CNN-based and Transformer-based. The former mostly uses cross-correlation [22,24] and discriminative filtering [25,26], which has a simpler network structure and faster running speed. The latter utilizes the attention mechanism of Transformer to fuse the features [27,28], which exhibits attractive performance. However, these trackers need to use multiple Transformer modules in series, which leads to more computation and memory occupation. OSTrack [7] and MixFormer

 doi:10.20944/preprints202506.0960.v1

[29] design Transformer-based one-stream frameworks, which combine feature extraction and feature fusion into one stage to deeply explore more suitable visual features for tracking. Moreover, ARTrack [30] and AQATrack [31] have used the autoregressive method to introduce the temporal motion information into the tracking model and achieve excellent performance.

2.2. Global Tracker

Different from the local tracker which uses a small image patch as the search region, the global tracker searches for the target over the entire current frame [8,32]. Therefore, the global tracker has the ability to recapture the target, which can effectively deal with the rapid target movement and the temporary target disappearance caused by occlusion and moving out of view. This advantage makes it perform well when tracking UAVs in complex infrared scenarios [9,10,17,33,34]. In contrast, the local tracker needs to attach a global detection module to recapture the reappearing target [35,36], which makes the model structure so complex that it needs to be trained in modules or stages. In fact, the global tracker is also often used as the global detection module of the local tracker, for example, LTMU [36] uses GlobalTrack to recapture the reappearing target. Nevertheless, the existing global trackers have poor scale adaptation, because they cannot keep the approximate consistency between the target sizes in the template and the search region, which makes them unable to cope well with the drastic target scale variation.

2.3. Infrared UAV Tracking

Compared with RGB cameras, infrared cameras are more suitable for the anti-UAV task because they can work well under all-weather and low-light conditions. However, tracking UAVs in infrared video is still challenging due to the complex and various application scenarios. In order to accurately track small UAV targets, TransIST [37] combines multi-scale attention and side window filter into a Transformer-based tracking model. Considering the camera motion, Zhao et al. [38] extracts feature points and calculates the homography matrix to compensate for the motion between frames. For the problem of background interference, Fang et al. [9] adopts Kalman filtering to estimate target motion, while SiamDT [34] detects and memorizes interference objects in the scene to exclude wrong candidate targets. In the common case where the target disappears during UAV tracking, Wu et al. [39] searches for the target by expanding the search region, Zhao et al. [38] and Yu et al. [40] use the global detection module to recapture the target, as well as SiamSTA [17], CAMTracker [10] and SiamDT adopt the global tracking scheme. When the target scale changes, our work solves the problem of poor scale adaptation of global trackers to track UAVs more robustly.

2.4. Scale-Arbitrary Image Super-Resolution

Our proposed SAEM refers to some research results on image super-resolution, which are briefly introduced here. Image super-resolution aims to reconstruct a high-resolution image from a low-resolution one, which is widely used in remote sensing, medical, surveillance, etc. Recently, researchers have proposed some deep super-resolution models such as EDSR [41] and SwinIR [42]. However, these methods usually need to set a fixed scale factor, such as ×2 or ×4, which lacks flexibility. To address this problem, MetaSR [15] generates convolution kernels based on the desired factor and pixel coordinates for achieving image super-resolution with arbitrary magnification. ArbRCAN [16] combines the experts learned during training based on the desired factor and then generates a convolution kernel to achieve scale-arbitrary super-resolution. Inspired by these methods, we improve the scale-aware convolutional layer of ArbRCAN to implicitly resize the target template of the global tracker.

3. Methodology

 doi:10.20944/preprints202506.0960.v1

We propose an efficient one-stage anchor-free global tracker (OSGT), and introduce a scale adaptation enhancement module (SAEM) to implicitly resize the target template to solve the problem that global trackers do not cope well with target scale variation. As shown in Figure 2, our proposed OSGT adopts Siamese network: firstly, the multi-level features of the first frame (Frame 1, template) and the current frame (Frame T, search region) are extracted by parameter-shared ResNet50 and FPN (feature pyramid network); secondly, the feature fusion part fuses the features of the target template and the search region; finally, the output heads generate the multi-level score maps and regression maps, and the bounding box corresponding to the position with the largest score is taken as the tracking result of the current frame. In addition, our proposed SAEM is embedded into the feature fusion part, and directly processes the extracted target template features with the input of the initial target size and the target size in the previous frame (Frame T-1). During the training, an auxiliary branch is designed to supervise the learning of SAEM, and Gaussian noise is added to the input size to enhance its robustness. During the online tracking, an adaptive threshold is proposed to more accurately judge target disappearance and effectively avoid SAEM being affected by the incorrect input size.

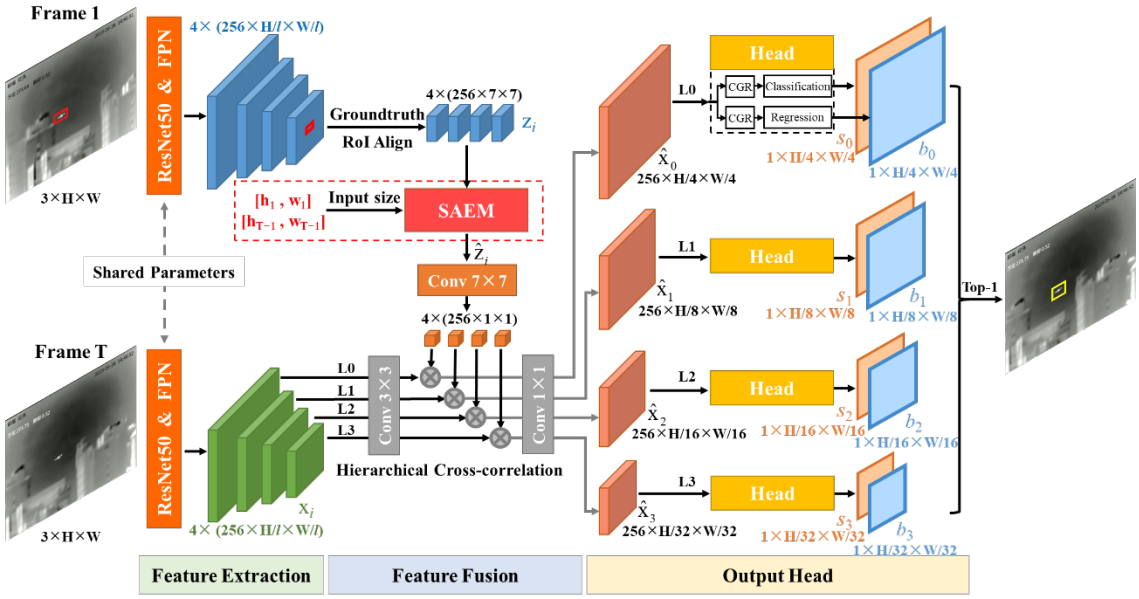


Figure 2. The overall framework of OSGT consists of feature extraction, feature fusion and output head. In addition, our proposed SAEM is embedded into the feature fusion part, as shown in the red dashed box.

3.1. One-stage Anchor-free Global Tracker

GlobalTrack, SiamSTA and SiamDT, based on RCNN [43], adopt the two-stage anchor-based framework, including the region proposal stage and the target regression stage. These methods have complex structures and require two rounds of feature fusion between the target template and the search region, making them unable to run in real time. In the field of object detection, one-stage anchor-free detection algorithms such as FCOS and CenterNet [44] use simpler structures that can run faster and perform well. Therefore, we combine the feature fusion module of GlobalTrack and the output head of FCOS to propose a concise and efficient one-stage anchor-free global tracker.

3.1.1. Feature Extraction

ResNet50 is used to extract the features of the first and current frame for obtaining four levels of features with different resolutions, and their sizes are $[H/l, W/l]$, where $l \in \{4,8,16,32\}$, and then FPN is used for multi-scale feature enhancement. Note that the features of the first frame are extracted only once as the template during the online tracking.

 doi:10.20944/preprints202506.0960.v1

Our OSGT is trained to use feature maps with different resolutions to track UAV targets of corresponding sizes, as shown in Figure 2, where the feature map of level L_i ($i = 0,1,2,3$ correspond to l) processes the targets with sizes belonging to $[0,10)$, $[10,20)$, $[20,40)$ and $[40,+\infty)$, respectively. In this way, small targets with fewer appearance features can be tracked using low-level texture features, while large targets with rich appearance features can be tracked using high-level semantic features.

3.1.2. Feature Fusion

The feature fusion part of GlobalTrack uses RoI Align [45] (region of interest) based on the ground truth to extract a level of the target features, and the level index is determined by the target size. Instead, we extract all four levels of the target features $z_i \in \mathbb{R}^{256\times7\times7}$, and then fuse z_i and the features of the current frame $x_i \in \mathbb{R}^{256\times H/l\times W/l}$ using hierarchical cross-correlation to obtain the fusion feature $\hat{x}_i \in \mathbb{R}^{256\times H/l\times W/l}$. When SAEM is not embedded, the above process is calculated as follows:

$$\hat{x}_i = C_i(z_i, x_i) = \mathcal{F}_i^{1\times1}\left(\mathcal{F}_i^{7\times7}(z_i) \otimes x_i\right), \quad i = 0,1,2,3 \qquad (1)$$

where $\mathcal{F}_i^{7\times7}$ is a 7×7 convolutional layer with padding 0 for converting z_i to a 256×1×1 convolutional kernel, $\otimes$ is the convolutional operation and $\mathcal{F}_i^{1\times1}$ is a 1×1 convolutional layer for transforming the number of channels. The hierarchical cross-correlation fuses the features of the target template and the search region separately according to the feature depth, which avoids the semantic confusion caused by correlating features from different levels. In addition, extracting all levels of the target features can provides our proposed SAEM with rich multi-scale information.

3.1.3. Output Head

Our proposed OSGT adopts the head of FCOS, but removes the centerness branch and some convolutional layers. In detail, the fusion feature $\hat{x}_i$ is processed with two CGRs (Conv+GroupNorm+ReLU) instead of the four CGRs used in FCOS, which is sufficient for the infrared UAV tracking task and conducive to improving the running speed. Then, a classification branch (consisting of a 3×3 convolutional layer) is set up to obtain the score map $s_i \in \mathbb{R}^{1\times H/l\times W/l}$, and a regression branch (consisting of a 3×3 convolutional layer) is set up to obtain the bounding box $b_i \in \mathbb{R}^{4\times H/l\times W/l}$, where the first dimension denotes the upper left and lower right corner points of the target (x_0, y_0, x_1, y_1). Finally, we adopt the Top-1 prediction to take the bounding box corresponding to the position with the highest score as the tracking result of the current frame.

3.2. Enhancing the Scale Adaptation of Global Tracker

Figure 3 shows the changes in four levels of the target features z_i after 2× upsampling and 2× downsampling. It indicates that the target features not only contain appearance information such as shape, texture and brightness, but also are closely related to the target size. Therefore, the global tracker tends to search for the object in the current frame that is close in size to the target template. However, when the target scale changes greatly, existing global trackers cannot adjust the target template based on the scale change, resulting in tracking failure in case of being disturbed by other objects of similar size.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 11 June 2025 doi:10.20944/preprints202506.0960.v1

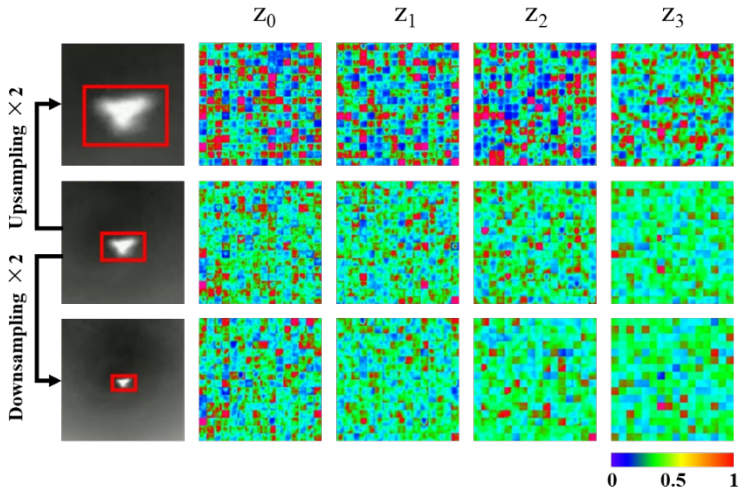


Figure 3. Visualization feature maps of the target template with different scales. For the convenience of display, we arrange the 256×7×7 tensor into 16×16 small squares with the size 7×7.

In order to enhance the scale adaptation of global trackers, the target templates need to be dynamically adjusted to keep approximately the same size as the target in the search region. Inspired by the scale-arbitrary image super-resolution, we propose a plug-and-play scale adaptation enhancement module that can be embedded in the feature fusion part, as shown in Figure 2. It can generate the scale adaptation enhancement kernel according to the target size of the previous frame, and then directly process the extracted features of the target template to resize it implicitly, as shown in Figure 4.

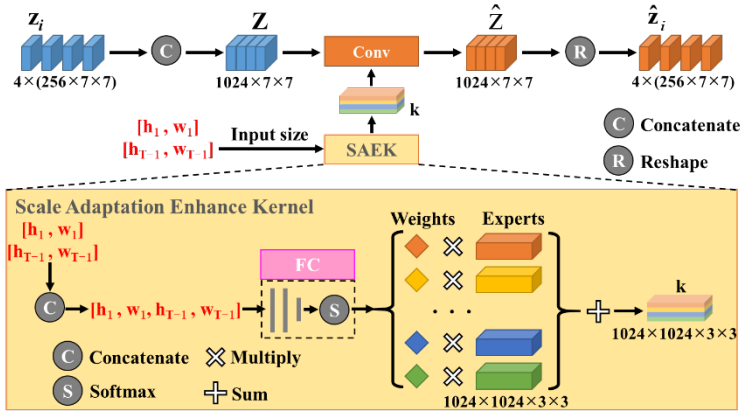


Figure 4. Our proposed scale adaptation enhancement module.

3.2.1. Scale Adaptation Enhancement Module

SAEM aims to directly adjust the scale information of the target template features based on the target size of the previous frame $[h_{T-1}, w_{T-1}]$, as shown in the upper part of Figure 4. Firstly, the four levels of the target template features z_i are concatenated into $Z \in \mathbb{R}^{1024\times7\times7}$. Next, Z is convolved with a scale adaptation enhancement kernel (SAEK) to obtain $\hat{Z}$. Finally, we divide $\hat{Z}$ into $\hat{z}_i \in \mathbb{R}^{256\times7\times7}$. The concatenation operation can ensure that each $\hat{z}_i$ can obtain information from all four levels of z_i. After SAEM is embedded, the feature fusion is calculated as follows:

$$\hat{x}_i = C_i(\hat{z}_i, x_i) = \mathcal{F}_i^{1\times1}\left(\mathcal{F}_i^{7\times7}(\hat{z}_i) \otimes x_i\right), \quad i = 0,1,2,3. \tag{2}$$

SAEK is obtained based on the initial size of the target template $[h_1, w_1]$ and the target size of the previous frame $[h_{T-1}, w_{T-1}]$. Referring to ArbRCAN for scale-arbitrary super-resolution, SAEK is designed to contain M (the default value is 12) experts, and each expert is a learnable tensor with a

 doi:10.20944/preprints202506.0960.v1

dimension 1024×1024×3×3, as shown in the lower part of Figure 4. During training, the experts learn and save knowledge that can be used to adjust the scale information of target template features. In addition, $[h_1, w_1]$ and $[h_{T-1}, w_{T-1}]$ are input to the fully connected network to obtain the expert weight, and then the experts are weighted and summed to obtain SAEK. Therefore, the parameters of SAEK can change dynamically according to the input target size.

As mentioned above, we concatenate $[h_1, w_1]$ and $[h_{T-1}, w_{T-1}]$ as the input, which is different from the ratio input $[h_{T-1}/h_1, w_{T-1}/w_1]$ in ArbRCAN. The advantage of our method is that it can provide a base scale for the model to avoid confusion. For example, a target with a size $[8,10]$ is upsampled ×2 to $[16,20]$, and the other target with a size $[32,40]$ is also upsampled ×2 to $[64,80]$. Although these two operations have the same magnification factor, different experts (learned knowledge) should be used and their SAEKs should also be different.

3.2.2. Supervision and Gaussian Noise

The essence of SAEM is to map the template features with the initial size $[h_1, w_1]$ to a new template features with the size $[h_{T-1}, w_{T-1}]$. The latter can also be obtained through an explicit method that involves: (1) resizing the template according to the target size in the previous frame, and (2) extracting its target features. These extracted features can serve as the ground truth for training the SAEM. In this way, we set up an auxiliary branch to supervise the learning of SAEM, as shown by the red flowline in Figure 5. Firstly, *Template* is resized by a factor of $[h_{T-1}/h_1, w_{T-1}/w_1]$ to obtain *Template**; then, the target features in *Template** are extracted to obtain z_i^*; finally, we take z_i^* as the ground truth and use the smooth *L1* loss to calculate the loss L_{sup} between $\hat{z}_i$ and z_i^*. For training our model, the loss is calculated as follows:

$$L_{\text{total}} = \frac{1}{4}\sum_{i=0}^{3}\left[L_{\text{cls}}(s_i, s_i^*) + L_{\text{reg}}(b_i, b_i^*) + L_{\text{sup}}(\hat{z}_i, z_i^*)\right], \tag{3}$$

where $L_{\text{cls}}(\bullet)$ is the focal loss for classification, $L_{\text{reg}}(\bullet)$ is the giou loss for regression, $s_i^* \in \mathbb{R}^{1\times H/l\times W/l}$ and $b_i^* \in \mathbb{R}^{4\times H/l\times W/l}$ denote the ground truth of score and bounding box at the i-th layer, respectively. The total loss L_{total} is the average of four-layer losses.

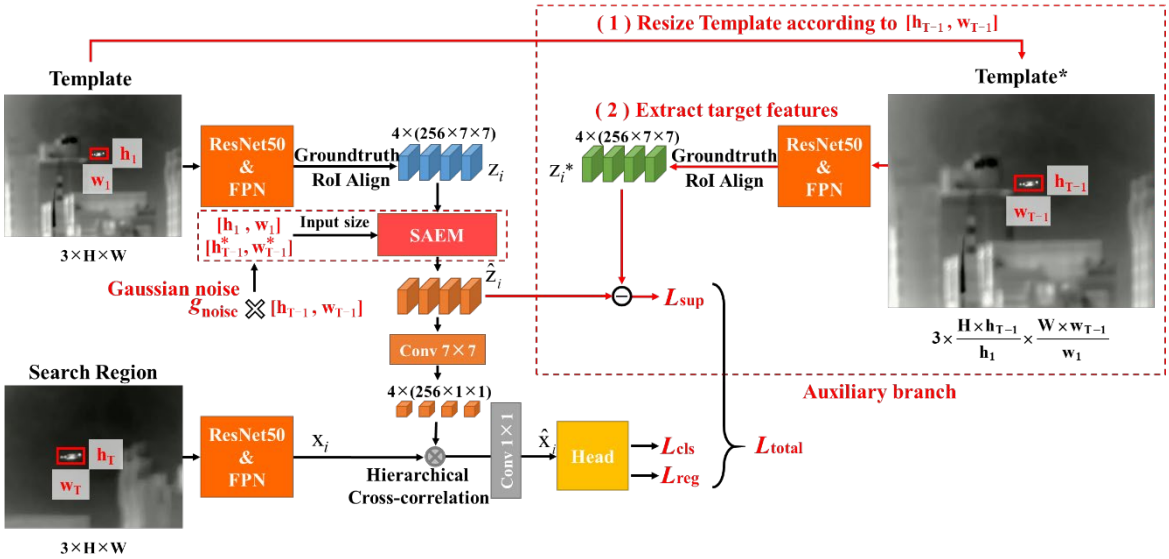


Figure 5. The auxiliary branch for supervising the learning of SAEM and Gaussian noise is added to the previous target size to enhance the robustness of SAEM.

During online tracking, the previous target size $[h_{T-1}, w_{T-1}]$ input to SAEM may be inaccurate due to occlusion and background clutter. To enhance the robustness of SAEM, $[h_{T-1}, w_{T-1}]$ is multiplied by a random value to simulate the size bias during the training. Considering that some

 doi:10.20944/preprints202506.0960.v1

trackers [7,27] employing the Gaussian corner heatmap as imprecise annotations of target bounding boxes, we use the random value that follow a Gaussian distribution with mean $\mu = 1.0$ and standard deviation $\sigma = 0.1$. And we perform a clamp operation to limit its range within $[\mu - 3\sigma, \mu + 3\sigma]$ to avoid extreme values with small probability, as shown below:

$$[h^*_{\text{T-1}}, w^*_{\text{T-1}}] = [h_{\text{T-1}}, w_{\text{T-1}}] \times g_{\text{noise}}, \quad g_{\text{noise}} \in [\mu - 3\sigma, \mu + 3\sigma], \tag{4}$$

where $[h^*_{\text{T-1}}, w^*_{\text{T-1}}]$ is the previous target size after adding Gaussian noise, as shown in Figure 5. The above process can be regarded as adding noise to the exact $[h_{\text{T-1}}, w_{\text{T-1}}]$ to achieve a data augmentation, which can effectively reduce the sensitivity of SAEM to the input size.

3.2.3. An Adaptive Threshold for Judging Target Disappearance

During the online tracking, the previous target size $[h_{\text{T-1}}, w_{\text{T-1}}]$ entered into SAEM is initialized with the target annotation in the first frame, and it is updated frame by frame in the subsequent tracking. However, $[h_{\text{T-1}}, w_{\text{T-1}}]$ should stop updating when the target disappears due to occlusion or moving out of view. Otherwise, if the wrong $[h_{\text{T-1}}, w_{\text{T-1}}]$ is input to SAEM, it will lead to tracking failure and incorrect recapture. Therefore, it is important to accurately judge whether the target exists for online tracking. A common way is to set a specific threshold for the maximum score $s^{\max}$, for example, if $s^{\max} > 0.5$, the target is present, and vice versa. Nevertheless, as shown in Figure 6, the smaller the target size, the harder it is to be tracked, and the lower the maximum score $s^{\max}$. If 0.5 is used as the threshold, targets with the size less than 10 will be misjudged as disappearance; if a small threshold is used, such as 0.3, there will be more false positive cases when judging larger targets. Therefore, a fixed threshold is not appropriate. To address this problem, we propose an adaptive method to judge whether a target exists or not, as follows:

$$e = \begin{cases} 1, & s^{\max} > 0.5 s^{\max}_{\text{T=1}} \\ 0, & otherwise \end{cases} \tag{5}$$

where '1' means the target is present, '0' means the target is absent, and $s^{\max}_{\text{T=1}}$ represents the target score obtained by inputting the first frame into the tracker, which is typically the maximum value. $s^{\max}_{\text{T=1}}$ provides a base value for the threshold, which can avoid misjudging the existence of small and dim targets compared with directly using 0.5 as the threshold.

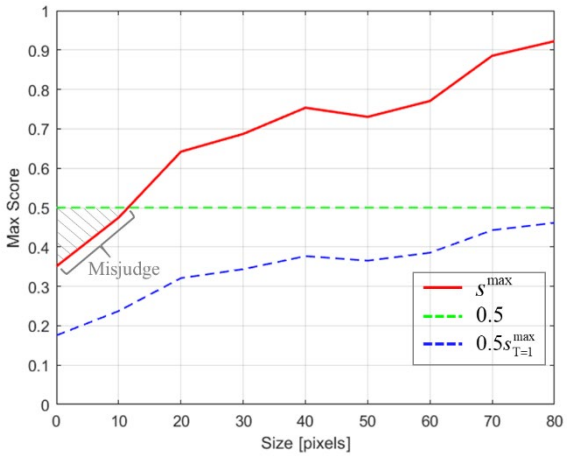


Figure 6. The red line shows the relationship between the target size and the maximum score $s^{\max}$, the green dashed line represents the threshold of 0.5, the blue dashed line represents our proposed adaptive threshold, and the gray shaded area shows the misjudgment when using 0.5 as the threshold.

4. Experiment

4.1. Datasets and Evaluation Metrics

 doi:10.20944/preprints202506.0960.v1

We use the public datasets of the Anti-UAV Challenge and the Anti-UAV410 dataset for training and testing. These datasets are closely related to the infrared UAV tracking task.

Our proposed OSGT is trained on the most comprehensive 3rd Anti-UAV training dataset, which contains 150 infrared videos, and tested on the 1st [1], 2nd Anti-UAV test-dev [46] and 3rd Anti-UAV validation datasets. These three test datasets contain 100, 140 and 50 infrared videos, respectively, with increasing scene complexity and tracking difficulty. Each video contains 1000 to 1500 frames, and the resolution of each frame is 512×640. These datasets cover UAVs at different scales and in various scenarios such as cloudy skies, cities and mountains.

The Anti-UAV410 dataset [34] is augmented with a large number of diverse and complex scenarios based on the 1st and 2nd Anti-UAV datasets. These scenarios included a wide range of backgrounds such as buildings, mountains, forests, urban areas, clouds, water surfaces, and others. The Anti-UAV410 dataset is divided into three sets: the train set, which consists of 200 videos; the validation set, which consists of 90 videos; and the test set, which consists of 120 videos. These videos contain 1069 frames on average and their resolution is 512×640. There are some differences in the data distribution between the Anti-UAV410 dataset and the 3rd Anti-UAV dataset. The latter has a higher proportion of difficult scenarios, which increases the difficulty of tracking UAVs.

On three Anti-UAV Challenge test datasets, two commonly used evaluation metrics, namely precision plot and success plot, are utilized to assess the performance of location and scale estimation of all trackers through One-Pass Evaluation (OPE). Specifically, the precision plot calculates the percentage of frames in which the estimated target location falls within a given distance threshold from the ground truth. Precision (P, with a threshold of 20 pixels) and normal precision (P_{Norm}) can be used to measure the accuracy of locating the target. The success plot measures the fraction of successful frames where the Intersection over Union (IoU) between the predicted bounding box and the ground truth is greater than a threshold ranging from 0 to 1. Area Under Curve (AUC) of the success plot and 50% overlap precision (OP50) can be used to evaluate the accuracy of estimating the target scale. The above metrics are widely used in tracking benchmarks.

According to [34], on the Anti-UAV410 dataset, we use the state accuracy (SA) to comprehensively evaluate the performance of all trackers, which is calculated as follows:

$$SA = \sum_{t} \frac{IoU_t \times v_t + (1 - e_t) \times (1 - v_t)}{T}, \quad (6)$$

where v_t and e_t are the ground truth and predicted value of whether the target exists in the t-th frame, respectively. This metric not only requires the tracker to accurately predict the position and scale of the target, but also to determine the state of the target's presence in the current frame, and make a judgment when the target disappears from the field of view.

4.2. Implementation Details

We use the SGD optimizer to train our proposed model for 24 epochs, and each epoch contains about 1200 iterations. The initial learning rate is 0.005, and is divided by 10 at the 8th and 16th epochs. The batch size is set to 12. We use 4 NVIDIA RTX 2080Ti GPUs for training and use one of them for testing.

During training, we use common data augmentation techniques such as horizontal flipping and photometric distortion, and also use random scaling to increase the data proportion of target scale variation. Our proposed auxiliary branch supervision and Gaussian noise are only used to train SAEM, while they are removed during testing.

4.3. Quantitative Evaluation

Our method is tested against 17 high-performance deep trackers, including 11 local trackers, ATOM [25], DiMP [26], PrDiMP [47], KYS [48], STARK [27], AiATrack [49], TOMP [28], MixFormer [29], OSTrack [7], SeqTrack[50], AQATrack [31], and 6 trackers with target redetection capability DaSiamRPN [24], SiamSTA [17], LTMU [36], GlobalTrack [8], CAMTracker [10], SiamDT [34]. Among

 doi:10.20944/preprints202506.0960.v1

them, SiamSTA, GlobalTrack and SiamDT are global trackers; DaSiamRPN is able to expand its search region when the target disappears; LTMU is attached with a global search module; SiamSTA, CAMTracker and SiamDT are specifically designed for the infrared UAV tracking task. For a fair comparison, these trackers are retrained on the 3rd Anti-UAV training dataset and Anti-UAV410 training dataset, respectively.

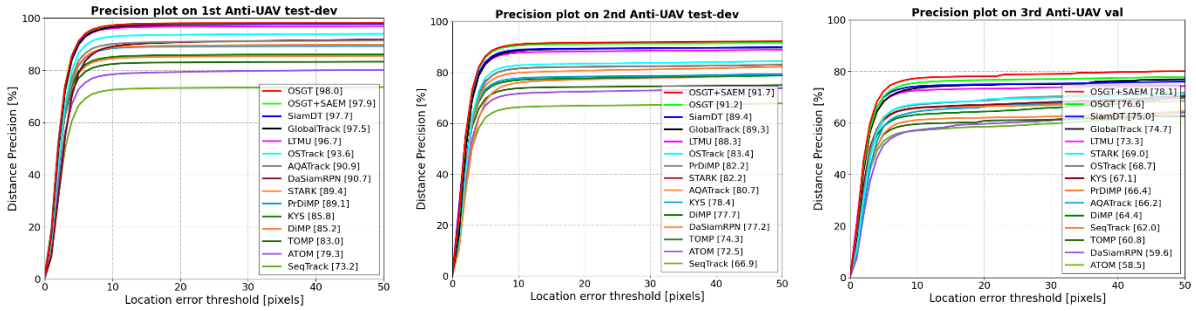
4.3.1. Comparison Results on Anti-UAV Challenge Datasets

Table 2 shows the quantitative comparison results between our proposed method and 14 deep trackers. Figure 7 shows the corresponding precision plots (ranked based on P) and success plots (ranked based on AUC). Overall, the trackers with target recapture capability perform better than local trackers, and it proves that global search is essential for handling challenging scenarios in the infrared UAV tracking task. In addition, OSTrack is the best local tracker; DaSiamRPN expands the search region to recapture the target, and its performance is lower than that of other trackers that directly redetect the target over the whole frame. Our proposed OSGT outperforms state-of-the-art SiamDT in almost all four evaluation metrics on three test datasets. Specifically, P is improved by 1.8% on 2nd Anti-UAV test-dev; AUC is improved by 1.9% and P is improved by 1.6% on the most difficult 3rd Anti-UAV val. After SAEM is embedded, the performance of OSGT is further improved, and it reaches the best place in most of the metrics on the three test datasets. In particular, AUC, OP50, P and P_{Norm} are improved by 1.3%, 1.5%, 1.5% and 1.2% on 3rd Anti-UAV val, respectively. It indicates that the proposed SAEM is beneficial to improve the performance of OSGT.

Table 2. Quantitative comparison results of our method and 14 deep trackers on three Anti-UAV challenge test datasets.

Method	Publication	1st Anti-UAV test-dev				2nd Anti-UAV test-dev				3rd Anti-UAV val			
		AUC	OP50	P	P_{Norm}	AUC	OP50	P	P_{Norm}	AUC	OP50	P	P_{Norm}
ATOM [25]	CVPR 2019	61.6	77.9	79.3	78.9	54.1	68.8	72.5	69.5	43.1	54.7	58.5	57.6
DiMP [26]	ICCV 2019	66.8	84.0	85.2	84.9	59.1	74.6	77.7	75.3	47.4	58.8	64.4	62.1
PrDiMP [47]	CVPR 2020	69.2	87.7	89.1	88.7	61.3	78.1	82.2	79.0	49.0	62.1	66.4	64.2
KYS [48]	ECCV 2020	67.3	84.5	85.8	85.5	59.6	75.3	78.4	76.0	49.0	60.9	67.1	63.5
STARK [27]	ICCV 2021	69.5	87.4	89.4	88.5	62.0	78.3	82.2	79.1	48.8	62.1	69.0	64.0
TOMP [28]	CVPR 2022	65.8	82.0	83.0	82.8	57.8	72.1	74.3	72.9	43.8	55.2	60.8	57.8
OSTrack [7]	ECCV 2022	72.4	91.3	93.6	92.7	62.7	79.5	83.4	79.9	51.9	64.8	68.7	67.2
SeqTrack [50]	CVPR 2023	55.3	71.4	73.2	72.9	50.1	63.7	66.9	65.2	43.5	55.3	62.0	57.9
AQATrack [31]	CVPR 2024	70.3	88.9	90.9	89.9	60.9	77.0	80.7	78.0	47.5	59.6	66.2	62.3
DaSiamRPN [24]	ECCV 2018	68.7	88.1	90.7	87.9	57.7	74.5	77.2	74.8	42.0	53.0	59.6	55.7
GlobalTrack [8]	AAAI 2020	75.6	95.5	97.5	96.4	65.5	83.1	89.3	85.2	53.0	66.3	74.7	70.5
LTMU [36]	CVPR 2020	75.8	95.3	96.7	96.2	68.6	86.4	88.3	88.1	55.4	69.2	73.3	72.3
SiamSTA # [17]	ICCVW 2021	72.6	—	96.9	—	65.5	—	88.8	—	—	—	—	—
SiamDT [34]	PAMI 2024	76.4	96.2	97.7	97.2	68.5	87.1	89.4	89.1	53.3	67.1	75.0	70.3
OSGT	—	76.2	96.6	98.0	97.3	68.6	88.3	91.2	89.8	55.2	70.5	76.6	75.2
OSGT+SAEM	—	76.4	96.2	97.9	97.3	69.4	88.9	91.7	90.5	56.5	72.0	78.1	76.4

* The best three results are shown in red, blue, and green fonts. The gray background indicates the results of trackers with target redetection capability. # means the values are taken from their publications.



 doi:10.20944/preprints202506.0960.v1

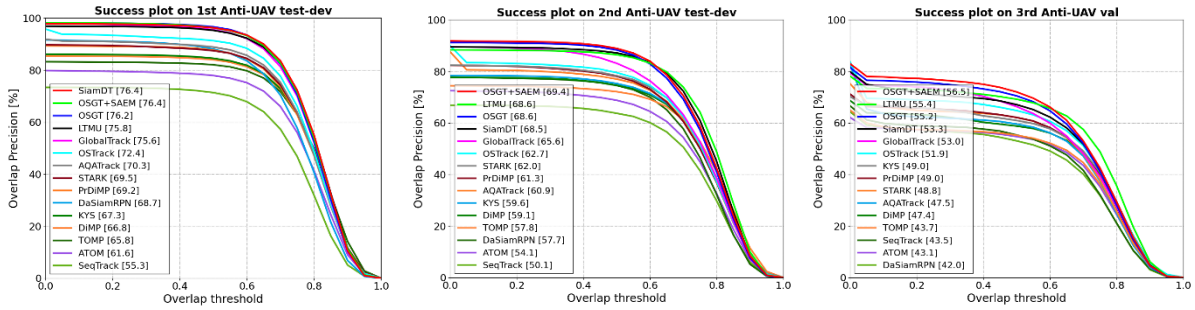


Figure 7. Precision plots and success plots of our method and 13 deep trackers on three Anti-UAV Challenge test datasets.

It is noted that OSGT has a slight performance degradation on the 1st test-dev dataset after embedding SAEM. This is because this dataset consists mostly of simple scenarios without obvious target scale variation, but the template features still change slightly through SAEM, which affects the precision of location and scale estimation to a certain extent.

4.3.2. Comparison Results on Anti-UAV410 Dataset

Table 3 shows the quantitative comparison results between our proposed method and 8 deep trackers. OSGT outperforms the baseline GlobalTrack. After embedding SAEM, OSGT achieves the state-of-the-art performance, outperforming SiamDT with improving SA by 0.79%. OSGT and SiamDT respectively focus on solving different problems of global trackers in the infrared UAV tracking task. SiamDT attempts to enhance the anti-interference ability of global trackers in complex backgrounds. To be specific, it detects and memorizes interference objects in the scene to exclude wrong candidate targets. In contrast, OSGT attempts to improve the scale adaptation of global trackers by implicitly resizing the template when the target scale changes. These two algorithms effectively improve the performance of global trackers from different aspects.

Table 3. Quantitative comparison results of our method and 8 deep trackers on the Anti-UAV410 test dataset.

Method	PrDiMP [47]	STARK [27]	AiATrack [49]	OSTrack [7]	MixFormer [29]	GlobalTrack [8]	CAMTracker # [10]	SiamDT # [34]	OSGT	OSGT +SAEM
Publication	CVPR 2020	ICCV 2021	ECCV 2022	ECCV 2022	CVPR 2023	AAAI 2020	RS 2024	PAMI 2024	—	—
SA	54.69	57.15	59.56	60.15	59.65	66.45	67.10	68.19	67.03	68.98

4.3.3. Inference Performance Comparison

Table 4 shows the inference performance of our proposed OSGT compared to four trackers with target redetection capability on an NVIDIA RTX 2080Ti GPU. These statistics include the preprocessing time (about 10ms) for image loading, normalization, etc. The running speed of OSGT reaches 30.9 fps, which meets the real-time requirement, and far exceeds SiamDT and LTMU. After SAEM is embedded, OSGT can still run in real time at 27.3 fps. It verifies that OSGT and SAEM are efficient.

Table 4. Inference performance of our method compared to 4 trackers with target redetection capability on an NVIDIA RTX 2080Ti GPU.

Method	DaSiamRPN	GlobalTrack	LTMU	SiamDT	OSGT	OSGT+SAEM
Speed (fps)	22.7	22.3	1.5	9.1	**30.9**	27.3

 doi:10.20944/preprints202506.0960.v1

4.4. Qualitative Evaluation

Figure 8 shows the visualization results of our method, OSTrack, GlobalTrack, LTMU and SiamDT in four different scenarios. The first row shows that OSGT can robustly track targets in complex backgrounds, especially it can estimate the size of the target more accurately. The second row shows that OSGT can track small targets well: in detail, the first two images show the case where the target position changes rapidly caused by camera movement, and the last two images show the case where the target temporarily disappears due to moving out of view, both of which are better handled by the global tracker than the local tracker by utilizing global search mechanism. The target scales in the third and fourth rows change from small to large and from large to small, respectively. GlobalTrack, SiamDT and OSGT without SAEM track wrong targets such as flying birds and towers that are similar in size and appearance to the target template. OSTrack and LTMU have good scale adaptation by resizing the search region, but they still fail to track targets when the target scale changes drastically, as shown in the fourth row. After SAEM is embedded, OSGT can timely adjust the scale information of the target template features according to the target size change, which improves the scale adaptation of OSGT. In addition, it can also be regarded as a specific scale constraint, which allows OSGT to effectively avoid being distracted by other objects with undesired scales.

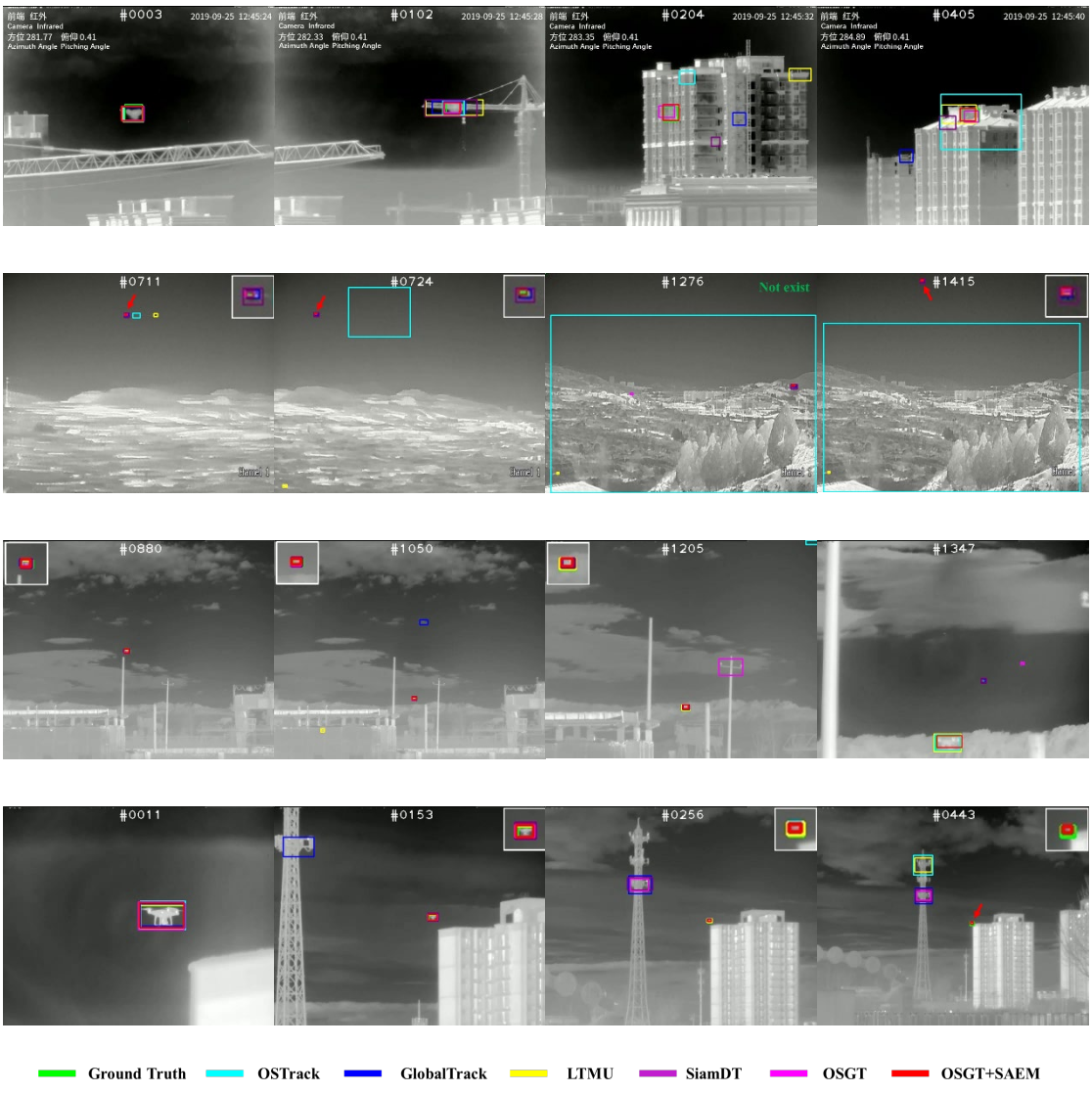


Figure 8. Visualization results of our method, OSTrack, GlobalTrack, LTMU and SiamDT.

4.5. Model Analysis

 doi:10.20944/preprints202506.0960.v1

4.5.1. Ablation Study

Table 5 shows the effect of SAEM, supervision and Gaussian noise on OSGT's performance, and the latter two are used during training SAEM. The role of Gaussian noise is to simulate the inaccurate input to SAEM. Obviously, training without Gaussian noise causes AUC and OP50 to severely decrease by 3.2% and 2.2%, respectively. This is because occlusion and background clutter lead to imprecise estimation of the target size in previous frames during online tracking, as shown in Figure 9. If SAEM processes the target template features according to the imprecise size, it will increase the error of the estimated bounding box. More seriously, if the auxiliary branch is used to supervise module training, SAEM will be more dependent on the exact input size, and the reduction in AUC and OP50 will be larger and reach 4.6% and 3.9%, respectively. This proves the importance of Gaussian noise to improve the robustness of estimating bounding boxes.

Under the premise of introducing Gaussian noise, using the auxiliary branch to supervise SAEM training can further improve the performance of OSGT. To be specific, AUC, OP50, P and P_{Norm} are improved by 0.6%, 0.5%, 0.9% and 0.8%, respectively. Compared with SAEM which implicitly adjusts the scale information of the extracted target template features, the auxiliary branch adopts a more intuitive approach of first resizing the target template and then extracting its features. It can provide a ground truth for the output of SAEM during training, which significantly reduces the difficulty of module training. In summary, SAEM can better improve the scale adaptation of the global tracker only when both the supervision and Gaussian noise are used during training.

Table 5. The influence of SAEM, supervision, and Gaussian noise on model performance. The model is tested on 3rd Anti-UAV val.

OSGT	SAEM	Supervision	Gaussian noise	AUC	OP50	P	P_{Norm}
✓				55.2	70.5	76.6	75.2
✓	✓			52.7	69.3	76.8	75.3
✓	✓	✓		51.9	68.1	76.8	74.5
✓	✓		✓	55.9	71.5	77.0	75.6
✓	✓	✓	✓	**56.5**	**72.0**	**78.1**	**76.4**

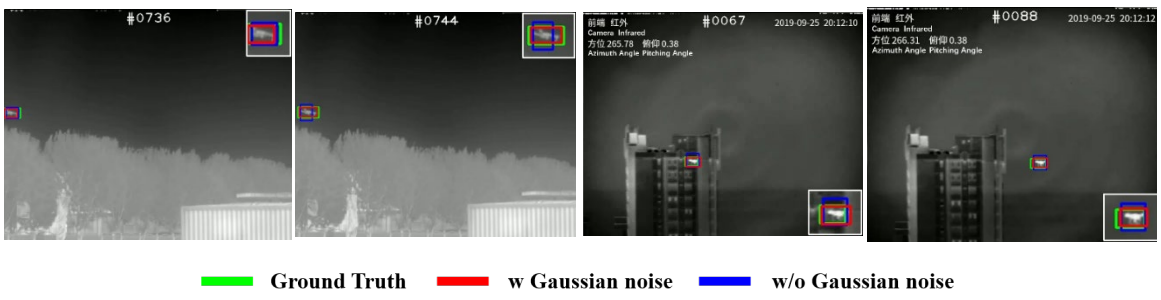


Figure 9. Using Gaussian noise during training enhances the performance of our model to accurately estimate the target size.

4.5.2. Effectiveness of SAEM

The purpose of SAEM is to implicitly adjust the scale information of the extracted template features according to the target scale change. There is an explicit way to achieve this goal is to first resize the template and then extract its target features, which is also how the auxiliary branch supervision works for training SAEM. Figure 10 visualizes the template features obtained using the two methods. The results of both are approximately the same, which proves that SAEM achieves the expected effect.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 11 June 2025 doi:10.20944/preprints202506.0960.v1

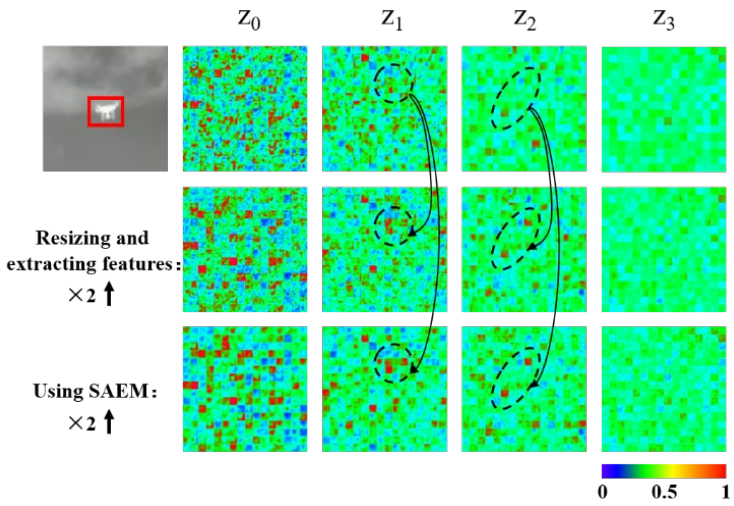


Figure 10. The template feature maps are obtained by using SAEM and the explicit method, respectively. Some feature variations are shown in the black dashed circle.

4.5.3. Compatibility of SAEM

Our proposed SAEM is a plug-and-play module, which can be compatible with other global trackers to improve their scale adaptation. Table 6 shows the impact of embedding SAEM into accessible global trackers, such as GlobalTrack and SiamDT. After SAEM is embedded, all metrics of GlobalTrack and SiamDT are significantly improved. To show the increment of scale adaptation, we test 3 global trackers using SAEM in the scale variation scenario, as shown in Figure 11. To be specific, P and AUC of GlobalTrack are improved by 2.1% and 4.1%, respectively; P and AUC of SiamDT are improved by 3.7% and 2.7%, respectively; P and AUC of OSGT are improved by 5.9% and 2.7%, respectively. It not only verifies the effectiveness of SAEM in improving the scale adaptation of OSGT, but also proves that SAEM can be embedded into other global trackers to effectively improve their ability to cope with the target scale variation.

Table 6. The impact of embedding SAEM into different global trackers.

Metrics	GlobalTrack	GlobalTrack +SAEM	SiamDT	SiamDT +SAEM	OSGT	OSGT +SAEM
AUC	53.0	54.4 (1.4↑)	53.3	55.3 (2.0↑)	55.2	56.5 (1.3↑)
OP50	66.3	68.2 (1.9↑)	67.1	69.5 (2.4↑)	70.5	72.0 (1.5↑)
P	74.7	76.1 (1.4↑)	75.0	76.1 (1.1↑)	76.6	78.1 (1.5↑)
P_{Norm}	70.5	73.1 (2.6↑)	70.3	72.8 (2.5↑)	75.2	76.4 (1.2↑)

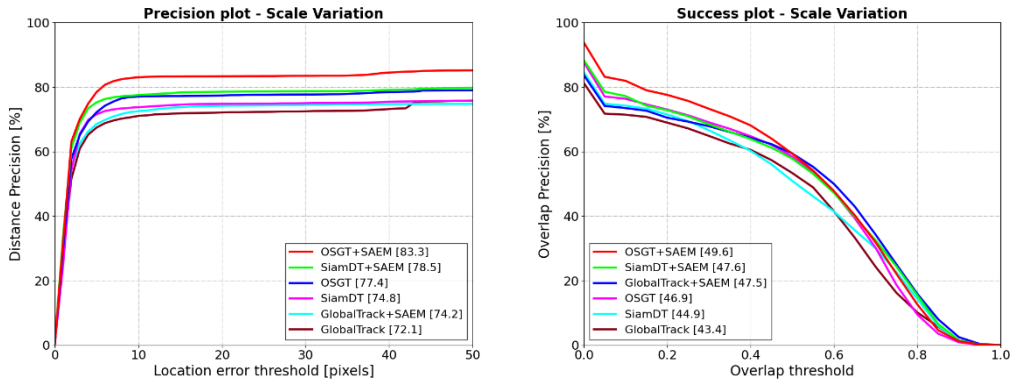


Figure 11. The enhanced scale adaptation of OSGT, GlobalTrack and SiamDT after embedding SAEM.

 doi:10.20944/preprints202506.0960.v1

4.5.4. Number of Experts in SAEM

In of Section 3.2.1, the expert in SAEK is the knowledge learned during training to adjust the scale information of the extracted target template features, and its specific form is the learnable model parameters. The impact of the number of experts on the model and its performance is shown in Table 7. When four more experts are added to SAEK, the number of model parameters increases by 0.13K, the calculation amount increases by 0.28K, the time consumption increases by about 0.8ms, and the running speed decreases by about 0.6 fps. The more experts, the higher the tracking performance of the model, but too many experts degrade the performance due to model overfitting and increased training difficulty. The tracker performs best when the number of experts is 12. Overall, SAEM requires very little computation and time consumption.

Table 7. The impact of the number of experts on the model and its performance.

Experts	Params.	FLOPs	Time	AUC	OP50	P	P_{Norm}	Speed
4	1.35K	2.58K	1.72ms	55.6	70.6	76.9	74.6	**28.5**
8	1.48K	2.86K	2.53ms	55.8	71.4	76.9	76.0	27.9
12	1.61K	3.14K	3.31ms	**56.5**	**72.0**	**78.1**	**76.4**	27.3
16	1.74K	3.42K	4.11ms	56.2	**72.0**	77.5	75.2	26.6

4.5.5. Different Input Forms in SAEM

SAEM requires the change information of the target scale as input. There are usually two input forms, one is the ratio form $[h_{T\text{-}1}/h_1, w_{T\text{-}1}/w_1]$ in ArbRCAN, and the other is our proposed concatenation form $[h_1, w_1, h_{T\text{-}1}, w_{T\text{-}1}]$. Table 8 shows the influence of these two input forms on the model performance. Compared with the ratio form, the concatenation form can provide the model with a base scale to avoid confusion and get better results.

Table 8. The influence of different input forms in SAEM on model performance.

Input forms	AUC	OP50	P	P_{Norm}
Ratio form	56.2	**72.0**	77.4	75.6
Concatenation form	**56.5**	**72.0**	**78.1**	**76.4**

4.5.6. Using Different Thresholds to Judge the Target Disappearance

In of Section 3.2.3, considering the target is occluded or out of view, the algorithm needs to accurately judge the disappearance of the target and stop updating the previous target size input to SAEM. Otherwise, when the target reappears, it cannot be recaptured correctly due to inputting the wrong size to SAEM. Table 9 shows the effect of using different thresholds to judge the target disappearance on the model performance. When the threshold is 0, it means that the algorithm does not judge whether the target exists and updates the input of SAEM every frame. In this case, the performance of the tracker decreases drastically. As shown in the left of Figure 12, the previous target size is inaccurate due to the target disappearance, and if the size input of SAEM is updated, the tracker cannot track the target correctly when the target reappears. Our proposed adaptive threshold takes the maximum score of the first frame $s_{T=1}^{\max}$ as the base value, which can more accurately judge whether the small and dim target exists. Compared with using 0.5 as the threshold, our method improves AUC, OP50 and P_{Norm} by 0.5%, 0.9% and 0.2%, respectively. As shown in the right of Figure 12, the target size is small and its maximum score is 0.3531. If 0.5 is used as the threshold, the tracker will misjudge that the target has disappeared and stop updating the input of SAEM, and it causes that the target size cannot be correctly estimated in the subsequent tracking. In contrast, our method can adjust the threshold to 0.1766 and correctly determine the existence of small targets.

Table 9. The effect of using different thresholds to judge the target disappearance on model performance.

 doi:10.20944/preprints202506.0960.v1

Threshold	AUC	OP50	P	P_{Norm}
0.0	55.4	70.5	77.0	74.3
0.5	56.0	71.1	**78.4**	76.2
$0.5s_{T=1}^{max}$	**56.5**	**72.0**	78.1	**76.4**

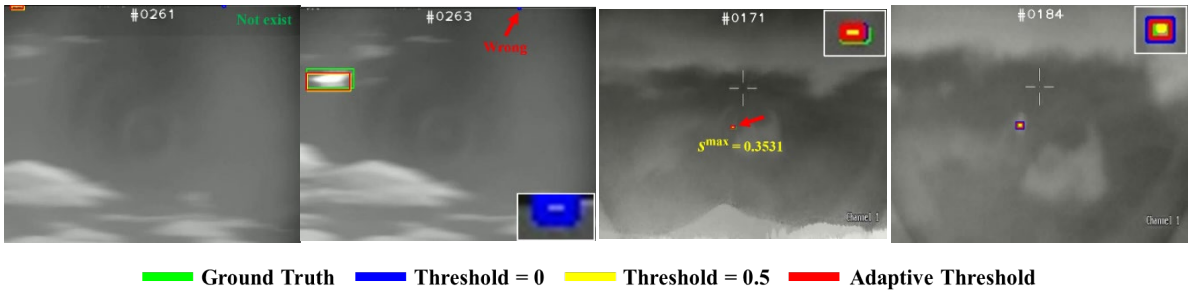


Figure 12. The influence of using different thresholds to judge the target disappearance on tracking results.

4.5.7. Comparison Results of Different Template Update Methods

Template update is a commonly used trick to deal with target appearance variations during online tracking. It is typically a temporal appearance update method, where the tracker replaces the original template with the tracking result every N frames [27]. Different from it, we only update the scale information of the target template feature, including an explicit update method (first resize the target template and then extract its features) and an implicit update method (using our proposed SAEM).

Table 10 shows the effect of different template update methods on the OSGT's tracking results. The temporal appearance update (with N=100) causes the degradation of the tracking performance. This is because most infrared UAV tracking scenarios are highly complex and prone to disturbing the tracking result, which makes the updated template unreliable, leading to subsequent tracking failure. In contrast, both scale update methods significantly improve the OSGT's performance by enhancing it scale adaptation. However, compared with our proposed implicit scale update, the explicit scale update exhibits lower efficiency, because it need to extract features from the resized template every frame.

Table 10. Comparison results of different template update methods.

Template update methods	AUC	OP50	P	P_{Norm}	Speed
None	55.2	70.5	76.6	75.2	**30.9**
Temporal appearance update	54.9	70.2	76.4	74.7	29.9
Explicit scale update	56.2	71.4	77.7	76.0	19.3
Implicit scale update (SAEM)	**56.5**	**72.0**	**78.1**	**76.4**	27.3

4.5.8. Tracking Failure Cases

The global tracker inherently possesses the ability to recapture targets due to taking the whole frame as its search region. This advantage enables it to effectively handle target disappearance caused by occlusion and moving out of view. However, a large search region introduces more background interference, especially in extremely complex scenarios, leading to tracking failures, as shown in Figure 13. Therefore, in addition to the poor scale adaptation, anti-interference is another critical research issue for global trackers.

 doi:10.20944/preprints202506.0960.v1

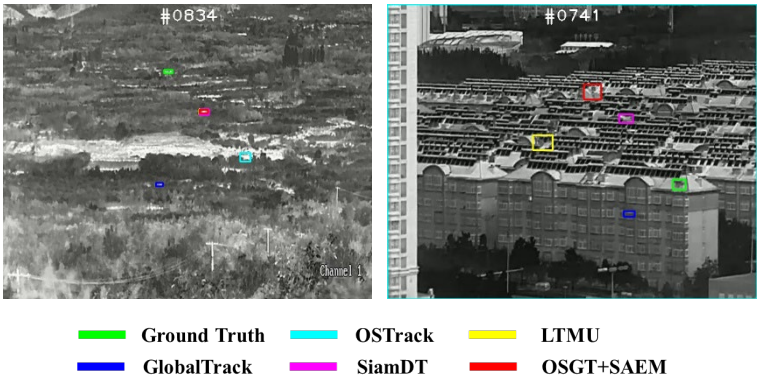


Figure 13. Tracking failure cases due to background interference.

5. Conclusions

This paper focuses on solving the problem of poor scale adaptation of global trackers in the infrared UAV tracking task. To address this problem, we propose a plug-and-play scale adaptation enhancement module, which can implicitly resize the target template according to the target size in the previous frame. Moreover, we optimize the learning of SAEM by setting the auxiliary branch supervision and Gaussian noise. During the online tracking, an adaptive threshold is proposed to judge target disappearance and avoid SAEM being affected by the wrong input size. In addition, we propose a one-stage anchor-free global tracker, which has a concise structure to track UAVs in real time in infrared videos. On three Anti-UAV Challenge datasets and the Anti-UAV410 dataset, our proposed tracker is compared with 17 deep trackers and shows excellent performance, especially when dealing with the dramatic target scale variation.

Our future work will explore combining the anti-interference and scale adaptation of the global tracker to deal with more complex tracking scenarios.

Author Contributions: Methodology, Z.F.; software, Z.F. and D.L.; validation, W.Z. and D.L.; investigation, W.Z. and Q.Y.; writing, E.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by National Natural Science Foundation of China, grant number 12202485.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

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

References

1. Jiang, N.; Wang, K.; Peng, X.; Yu, X.; Wang, Q.; Xing, J.; Li, G.; Guo, G.; Ye, Q.; Jiao, J.; Zhao, J.; Han, Z. Anti-UAV: A Large-scale Benchmark for Vision-based UAV Tracking. *IEEE Trans. Multimedia.* **2023**, *25*, 486–500. [CrossRef]
2. Yang, H.; Liang, B.; Feng, S.; Jiang, J.; Fang, A.; Li, C. Lightweight UAV Detection Method Based on IASL-YOLO. *Drones* **2025**, *9*, 325. [CrossRef]
3. Ye, Z.; You, J.; Gu, J.; Kou, H.; Li, G. Modeling and Simulation of Urban Laser Countermeasures Against Low-Slow-Small UAVs. *Drones* **2025**, *9*, 419. [CrossRef]
4. Javed, S.; Danelljan, M.; Khan, F.; Khan, M.; Felsberg, M.; Matas, J. Visual Object Tracking with Discriminative Filters and Siamese Networks: A Survey and Outlook. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 6552–6574. [CrossRef]
5. Li, J.; Ye, D. H.; Kolsch, M.; Wachs, J. P.; Bouman, C. A. Fast and Robust UAV to UAV Detection and Tracking from Video. *IEEE Trans. Emerg. Topics Comput.* **2021**, *10*, 1519–1531. [CrossRef]

6. Danelljan, M.; Bhat, G.; Khan, F.S.; Felsberg, M. ECO: Efficient Convolution Operators for Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 6931–6939.
7. Ye, B.; Chang, H.; Ma, B.; Shan, S.; Chen, X. Joint Feature Learning and Relation Modeling for Tracking: A One-Stream Framework. In Proceedings of the European Conference on Computer Vision (ECCV), Tel Aviv, Israel, 23–27 October 2022; pp. 341–357.
8. Huang, L.; Zhao, X.; Huang, K. Globaltrack: A Simple and Strong Baseline for Long-term Tracking. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; Volume 34, pp. 11037–11044.
9. Fang, H.; Wang, X.; Liao, Z.; Chang, Y.; Yan, L. A Real-Time Anti-Distractor Infrared UAV Tracker with Channel Feature Refinement Module. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Montreal, BC, Canada, 11–17 October 2021; pp. 1240–1248.
10. Wang, Z.; Hu, Y.; Yang, J.; Zhou, G.; Liu, F.; Liu, Y. A Contrastive-Augmented Memory Network for Anti-UAV Tracking in TIR Videos. *Remote Sens.* **2024**, *16*, 4775. [CrossRef]
11. Li, Y.; Zhu, J. A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration. In Proceedings of the European Conference on Computer Vision (ECCV), Zurich, Switzerland, 6–12 September 2015; pp. 254–265.
12. Danelljan, M.; Hager, G.; Khan, F.S.; Felsberg, M. Discriminative Scale Space Tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1561–1575. [CrossRef]
13. Zhang, Z.; Peng, H.; Fu, J.; Li, B.; Hu, W. Ocean: Object-Aware Anchor-Free Tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Glasgow, UK, 23–28 August 2020; pp. 771–787.
14. Chen, X.; Yan, B.; Zhu, J.; Wang, D.; Yang, X.; Lu, H. Transformer Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 19–25 June 2021; pp. 8122–8131.
15. Hu, X.; Mu, H.; Zhang, X.; Wang, Z.; Tan, T.; Sun, J. Meta-SR: A Magnification-Arbitrary Network for Super-Resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019; pp. 1575–1584.
16. Wang, L.; Wang, Y.; Lin, Z.; Yang, J.; An, W.; Guo, Y. Learning a Single Network for Scale-Arbitrary Super-Resolution. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, BC, Canada, 11–17 October 2021; pp. 4781–4790. [CrossRef]
17. Huang, B.; Chen, J.; Xu, T.; Wang, Y.; Jiang, S.; Wang, Y.; Wang, L.; Li, J. SiamSTA: Spatio-temporal Attention based Siamese Tracker for Tracking UAVs. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Montreal, BC, Canada, 11–17 October 2021; pp. 1204–1212.
18. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully Convolutional One-Stage Object Detection. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 9626–9635.
19. Li, S.; Zhao, S.; Cheng, B.; Zhao, E.; Chen, J. Robust Visual Tracking via Hierarchical Particle Filter and Ensemble Deep Features. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *30*, 179–191. [CrossRef]
20. Hare, S.; Golodetz, S.; Saffari, A.; Vineet, V.; Cheng, M.-M.; Hicks, S.L.; Torr, P.H.S. Struck: Structured Output Tracking with Kernels. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *38*, 2096–2109. [CrossRef]
21. 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 (ECCV), Florence, Italy, 7–13 October 2012; pp. 702–715.
22. 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 (CVPR), Long Beach, CA, USA, 15–21 June 2019; pp. 4277–4286.
23. Xie, X.; Xi, J.; Yang, X.; Lu, R.; Xia, W. STFTrack: Spatio-Temporal-Focused Siamese Network for Infrared UAV Tracking. *Drones* **2023**, *7*, 296. [CrossRef]
24. 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. 103–119.

25. Danelljan, M.; Bhat, G.; Khan, F.S.; Felsberg, M. ATOM: Accurate Tracking by Overlap Maximization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–21 June 2019; pp. 4655–4664.
26. Bhat, G.; Danelljan, M.; Van Gool, L.; Timofte, R. Learning Discriminative Model Prediction for Tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 6181–6190. [CrossRef]
27. Yan, B.; Peng, H.; Fu, J.; Wang, D.; Lu, H. Learning Spatio-Temporal Transformer for Visual Tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Virtual, 11–17 October 2021; pp. 10428–10437.
28. Mayer, C.; Danelljan, M.; Bhat, G.; Paul, M.; Paudel, D.P.; Yu, F.; Van Gool, L. Transforming model prediction for tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 19–24 June 2022; pp. 8731–8740.
29. Cui, Y.; Song, T.; Wu, G.; Wang, L. MixFormerV2: Efficient Fully Transformer Tracking. *arXiv* **2023**, arXiv:2305.15896.
30. Wei, X.; Bai, Y.; Zheng, Y.; Shi, D.; Gong, Y. Autoregressive Visual Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 9697–9706.
31. Xie, J.; Zhong, B.; Mo, Z.; Zhang, S.; Shi, L.; Song, S.; Ji, R. Autoregressive Queries for Adaptive Tracking with Spatio-Temporal Transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, United States, 17-21 June 2024; pp. 19300–19309.
32. Voigtlaender, P.; Luiten, J.; Torr, P.H.S.; Leibe, B. Siam R-CNN: Visual Tracking by Re-Detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14-19 June 2020; pp. 6577–6587. [CrossRef]
33. Huang, B.; Dou, Z.; Chen, J.; Li, J.; Shen, N.; Wang, Y.; Xu, T. Searching Region-Free and Template-Free Siamese Network for Tracking Drones in TIR Videos. *IEEE Trans. Geosci. Remote Sens.* **2023**, *62*, 5000315. [CrossRef]
34. Huang, B.; Li, J.; Chen, J.; Wang, G.; Zhao, J.; Xu, T. Anti-UAV410: A Thermal Infrared Benchmark and Customized Scheme for Tracking Drones in the Wild. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 2852–2865. [CrossRef]
35. Yan, B.; Zhao, H.; Wang, D.; Lu, H.; Yang, X. 'Skimming-Perusal' Tracking: A Framework for Real-Time and Robust Long-Term Tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019; pp. 2385–2393.
36. Dai, K.; Zhang, Y.; Wang, D.; Li, J.; Lu, H.; Yang, X. High-Performance Long-Term Tracking with Meta-Updater. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 6297–6306. [CrossRef]
37. Qian, K.; Zhu, D.; Wu, Y.; Shen, J.; Zhang, S. TransIST: Transformer Based Infrared Small Target Tracking Using Multi-Scale Feature and Exponential Moving Average Learning. *Infrared Phys. Technol.* **2025**, *145*.
38. Zhao, J.; Zhang, X.; Zhang, P. A Unified Approach for Tracking UAVs in Infrared. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Virtual, 11–17 October 2021; pp. 1213–1222.
39. Wu, H.; Li, W.; Li, W.; Liu, G. A Real-Time Robust Approach for Tracking UAVs in Infrared Videos. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Virtual, 14–19 June 2020; pp. 4448–4455. [CrossRef]
40. Yu, Q.; Ma, Y.; He, J.; Yang, D.; Zhang, T. A Unified Transformer Based Tracker for Anti-UAV Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Vancouver, BC, Canada, 18–22 June 2023; pp. 3035–3045.
41. Lim, B.; Son, S.; Kim, H.; Nah, S.; Lee, K.M. Enhanced Deep Residual Networks for Single Image Super-Resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017; pp. 1132–1140.

 doi:10.20944/preprints202506.0960.v1

42. Liang, J.; Cao, J.; Sun, G.; Zhang, K.; Van Gool, L.; Timofte, R. SwinIR: Image Restoration Using Swin Transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), Virtual, 11–17 October 2021; pp. 1833–1844. [CrossRef]
43. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014; pp. 580–587. [CrossRef]
44. Zhou, X.; Wang, D.; Krähenbühl, P. Objects as Points. *arXiv* **2019**, arXiv:1904.07850.
45. 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]
46. Zhao, J.; Wang, G.; Li, J.; Jin, L.; Fan, N.; Wang, M.; Wang, X.; Yong, T.; Deng, Y.; Guo, Y.; Ge, S.; Guo, G. The 2nd Anti-UAV Workshop & Challenge: Methods and Results. *arXiv* **2021**, arXiv:2108.09909.
47. Danelljan, M.; Van Gool, L.; Timofte, R. Probabilistic Regression for Visual Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Virtual, 14–19 June 2020; pp. 7181–7190. [CrossRef]
48. Bhat, G.; Danelljan, M.; Van Gool, L.; Timofte, R. Know Your Surroundings: Exploiting Scene Information for Object Tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Virtual, 23–28 August 2020; pp. 205–221.
49. Gao, S.; Zhou, C.; Ma, C.; Wang, X.; Yuan, J. AiATrack: Attention in Attention for Transformer Visual Tracking. In Proceedings of the European Conference on Computer Vision (ECCV), Tel Aviv, Israel, 23–27 October 2022; pp. 146–164.
50. Chen, X.; Peng, H.; Wang, D.; Lu, H.; Hu, H. SeqTrack: Sequence to Sequence Learning for Visual Object Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 14572–14581. [CrossRef]