Received 28 June 2024, accepted 19 July 2024, date of publication 23 July 2024, date of current version 31 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3432782

RESEARCH ARTICLE

SiamTAR: Enhancing Object Tracking With Joint Template Updating and Relocation Mechanisms

JIA WEN AND KEJUN REN
School of Information Science and Engineering, Yanshan University, Qinhuangdao 066004, China
The Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Qinhuangdao 066004, China

Corresponding author: Kejun Ren (1961711972@qq.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 62273293; in part by Shandong Provincial Natural Science Foundation, China, under Grant ZR2022LZH002; and in part by the Innovation Capability Improvement Plan Project of Hebei Province under Grant 22567626H.

ABSTRACT Siamese network-based trackers have demonstrated competitive performance in the domain of single object tracking. However, their effectiveness is significantly hindered when the target undergoes challenges such as deformation and illumination changes, due to the fixed template features from the first frame. To address this issue, we propose a novel tracking framework called SiamTAR. This framework adaptively updates the current frame's template by fusing template features from the first frame with updated template features and tracking box features from the previous frame, thereby effectively improving tracking accuracy. Additionally, to reduce the tracker's attention to redundant information such as similar shapes, colors, and textures near the target, we designed a feature refinement module. This module integrates three attention mechanisms through two parallel branches to capture critical target information, allowing the tracker to ignore some redundant information. To tackle issues of tracking box drift and inaccurate scale estimation during online tracking, we introduce a relocation mechanism. This mechanism corrects the tracking box position by merging the output tracking box features with the template features. Extensive experiments on multiple datasets validate the superior tracking performance of SiamTAR. Specifically, on the GOT-10K dataset, SiamTAR surpasses the current leading Siamese tracker, SiamPW-RBO, by 1.5% in AO and 7.5% in $SR_{0.75}$ metrics, achieving a tracking speed of 26.23 FPS. Source code is available at https://github.com/rkj12345/SiamTAR.

INDEX TERMS Object tracking, Siamese network, template update, feature refinement, relocation.

I. INTRODUCTION

Object tracking refers to the prediction of the target's position in each subsequent frame of a video, given the initial position of the target. It is a highly active research area within the field of computer vision, with broad applications in domains such as autonomous driving [1], intelligent surveillance [2], satellite video [3] and drone technology [4]. However, existing tracking algorithms face numerous challenges, particularly in real-world applications, where the tracking process is often affected by significant changes in lighting, variations in appearance, occlusions, and motion blur. Currently, tracking models face a dilemma where they are unable to balance speed and accuracy. Regression-based models are fast but not precise, while high-accuracy models rely on classification and are slow, usually processing only a few frames per second. These models dramatically limit their application. We chose to significantly improve the accuracy at the expense of a small amount of speed. Nonetheless, the model achieves real-time processing and meets the tracking speed required by most applications.

There are three main types of algorithms have been developed for target tracking so far. They are generative algorithms, correlation filtering algorithms, and deep learning algorithms. Among them, deep learning algorithms have become the mainstream trend today. Essentially, deep learning algorithms are tracking algorithms based on discriminative models. Since the deep features extracted by deep

The associate editor coordinating the review of this manuscript and approving it for publication was Gongbo Zhou.

learning networks have stronger information representation capability than traditional manual features, more and more researchers have applied deep learning networks to the field of target tracking, and have obtained superior performance.

The most popular approach in deep learning is the Siamese tracking model, which simplifies visual tracking into an object matching problem by learning a generic similarity map between the target template and the search region to determine the object's location. For instance, Bertinetto et al. [5] introduced Siamese networks to the field of single object tracking in 2016. SiamRPN [6] was the first to integrate the Region Proposal Network [7] from object detection into Siamese tracking algorithms. Subsequently, visual tracking models based on Siamese networks and object detection frameworks have thrived. Key advancements in improving the performance of deep learning trackers include DaSiamRPN [8] and SiamRPN++ [9]. In 2020, researchers observed that the introduction of RPN [7] resulted in an abundance of redundant and fixed-sized anchor boxes, making the trackers sensitive to parameters such as anchor box quantity, size, and aspect ratio, with reduced robustness. Consequently, a plethora of anchor-free Siamese tracking algorithms were proposed, including SiamCAR [10], SiamBAN [11], SiamFC++ [12], and Ocean [13]. These trackers have once again elevated Siamese tracking models to new heights.

Since most Siamese trackers are trained offline using large datasets, they can not update templates online. This results in the template features remaining fixed and unchanged once the first frame is initialized during tracking. Consequently, it becomes challenging to accurately track objects undergoing significant deformations, rapid movements, or occlusions, significantly increasing the risk of tracking drift or frame loss during the process. ResNet50 [14] and AlexNet [15] are widely employed feature extraction networks in numerous Siamese trackers. While ResNet50 [14] is more complex and capable of extracting richer feature information compared to AlexNet [15], it is slower. ResNet 50 [14] without fully connected layers strikes a balance between speed and rich feature extraction, making it the preferred choice for the backbone network. Therefore, the backbone network can already extract sufficient feature information for tracking. However, it has been observed that the features extracted by the Siamese network's two branches are not effectively utilized, leading to excessive redundancy in response maps when dealing with similar interferences or background noise. To address this issue, we propose a Feature Refinement Module that operates on the Siamese network. This module greatly improves the situation, resulting in a more robust tracker.

Based on the above development status and problems, we propose a novel tracking framework, SiamTAR, which is an improvement on SiamCAR [10]. Its main contributions are as follows:

- We introduce the Updatenet [16] network, enabling the model to generate updating templates in a learnable manner. This network, through multi-stage training with different learning rates, progressively attain a tracker with higher tracking precision and enhanced robustness.
- We propose a Feature Refinement Module (FRM) to facilitate the fusion between template features and search features. In comparison to traditional methods, this approach significantly enhances the matching of salient information and effectively conveys target information to the search region, further enhancing the accuracy of fusion. Importantly, this module can be easily extended to other scenarios where enhancing feature saliency is required.
- To solve the random drift of tracking box and scale mismatch of the target in complex scenes, we introduce a relocation mechanism in the tracker's output stage, which can mitigate this situation well.
- SiamTAR has achieved leading performance on multiple official datasets, including GOT-10K [17], UAV123 [18], OTB100 [19], and DTB70 [20]. Moreover, the tracking speed exceeds 25 frames per second (26.23 fps), reaching real-time levels.

II. RELATED WORK

A. TRACKING FRAMEWORK

In the aspect of tracking frameworks, siamese-based trackers have achieved an optimal balance between accuracy and efficiency. As one of the pioneering works, SiamFC [5] established a fully convolutional Siamese network to train trackers. Subsequently, numerous researchers have focused on this work and proposed updated models. Inspired by the Region Proposal Network [7] from object detection, SiamRPN [6] performed region proposal extraction after the Siamese network, yielding effective results through training of classification and regression branches. However, it struggled with handling interferences that share a resemblance with the target appearance. DaSiamRPN [8] introduced hard negative samples during training and alleviated the foreground-background imbalance issue through data augmentation. To address the issue of limited performance improvement with deeper networks in AlexNet [15], SiamRPN++ [9] enhanced the model architecture by using ResNet50 [14] as the backbone network. However, the use of anchor-based networks is highly sensitive to hyperparameters. In 2020, a substantial number of anchor-free trackers were proposed, including SiamCAR [10], Ocean [13], SiamBAN [11], SiamFC++ [12]. These trackers address both classification and regression problems, utilizing one or multiple prediction heads to regress bounding boxes pixel-wise from response maps. Tang et al. [21] introduced a ranking-based optimization algorithm to explore the relationship among different proposals. This algorithm is compatible with most Siamese trackers and does not yield additional speculation.

In recent times, target tracking algorithms utilizing the Transformer framework have gained increasing popularity. Chen et al. [22] employed the Transformer structure to fuse features from Siamese networks for improvement.

TABLE 1. Comparison of transformer tracker and siamese tracker.

Tracker	Strengths	Weaknesses
Siamese Tracker	Fast speed, low training data and computational requirements.	Poor tracking accuracy, struggles in challenging scenarios.
Transformer Tracker	High tracking accuracy, performs well in challenging scenarios.	High training cost, large training data volume, slow speed.

Wang et al. [23] introduced a Transformer algorithm that leverages temporal features. Lin et al. [24] proposed a novel motion token embedding historical target trajectory information, enhancing tracking by providing temporal cues. To address interference from complex backgrounds during tracking, Yang et al. [25] introduced a novel Foreground-Background Distribution Modeling Transformer.

While these trackers achieve outstanding performance, their training cost and computational expenses cannot be overlooked. For a clearer comparison of the strengths and weaknesses between Siamese and Transformer trackers, we've condensed the findings into the following Table 1.

B. TEMPLATE UPDATING

In terms of template updating, the majority of trackers either employ simple linear interpolation [26], [27], [28] to update the template for each frame or lack template updating capability altogether [29], [30]. Trackers with template updating are few and far between. Such updating mechanisms are far from sufficient when faced with appearance changes due to deformations, fast motion, or occlusions. To address this issue, Danelljan et al. [27], [28] suggested using a subset of historical frames as training samples when computing the convolutional kernel for each frame. While this approach yields better results, it comes at the cost of significant computation and memory consumption, greatly reducing tracking speed. Yang et al. [29] introduced Long Short-Term Memory (LSTM) to estimate the current template by storing previous templates in memory during online tracking. However, this incurs substantial computational expenses. Yao et al. [30] proposed offline SGD learning for updating coefficients of CF trackers, which remain fixed and do not require updates during tracking. Most models with template updating are based on correlation filtering, and traditional Siamese trackers have yet to deliver satisfactory performance due to limited representation capabilities and the lack of suitable template updating strategies. Therefore, we introduce Updatenet [16] into the model to enable online template updating.

C. ATTENTIONAL MECHANISM

Regarding attention mechanisms have found widespread application in various tasks. Hu et al. [31] introduced an SENet, which enhances network representation capabilities by focusing on relationships between different channels. Fu et al. [32] introduced a self-attention mechanism to acquire contextual information for semantic segmentation. Jiang et al. [33] introduced wavelet attention to eliminate rain streak disturbances and restore clean backgrounds from rainy images. By proposing the DAWN network, they achieved significant performance improvements in both image deraining and object detection tasks. Additionally, Jiang et al. [34] enhanced the representation of rain layer information through an improved non-local attention mechanism, significantly improving visual results. Xiao et al. [35] applied selective self-attention to the super-resolution task for remote sensing images to achieve progressive feature representation. Their proposed TTST model outperformed state-of-the-art super-resolution methods in PSNR on average. Chen et al. [36] proposed an EDS module, similar to attention, to map ship positions in consecutive maritime images, thereby improving multi-object tracking accuracy. Furthermore, Chen et al. [37], [38], [39]investigated a network analysis process that enhances the representation capability of complex relationships.

Wang et al. [26] proposed RASNet by incorporating an attention mechanism into the Siamese tracker, but it solely utilizes template information, which may limit its representational power. To better explore the feature maps extracted from the Siamese network, we propose the Feature Refinement Module (FRM), utilizing both template and test image information to enhance the discriminative representation of the target.

III. PROPOSED METHODS

In this section, we present a comprehensive description of the proposed Siamese tracker with template update and relocation mechanism, named SiamTAR. It consists of the following components: (1) Feature extraction, (2) Template update, (3) Feature refinement, (4) Target location head, and (5) Relocation and output. The overall framework of the SiamTAR tracker is illustrated in Fig.1. The general workflow of this framework can be summarized as follows: (1) Initially, the Siamese network performs feature extraction on the first frame image and subsequent frame images, obtaining template features and search features, respectively. (2) During the template update phase, the template features are updated based on the template features of the first frame, the output tracking box features from the previous frame, and the updated template features from the previous frame. Specifically, these three types of prior features are concatenated along the channel dimension and fed into the updatenet network for adaptive learning. (3) In the feature refinement phase, the updated template features and search image features are inputted into the FRM module to filter redundant

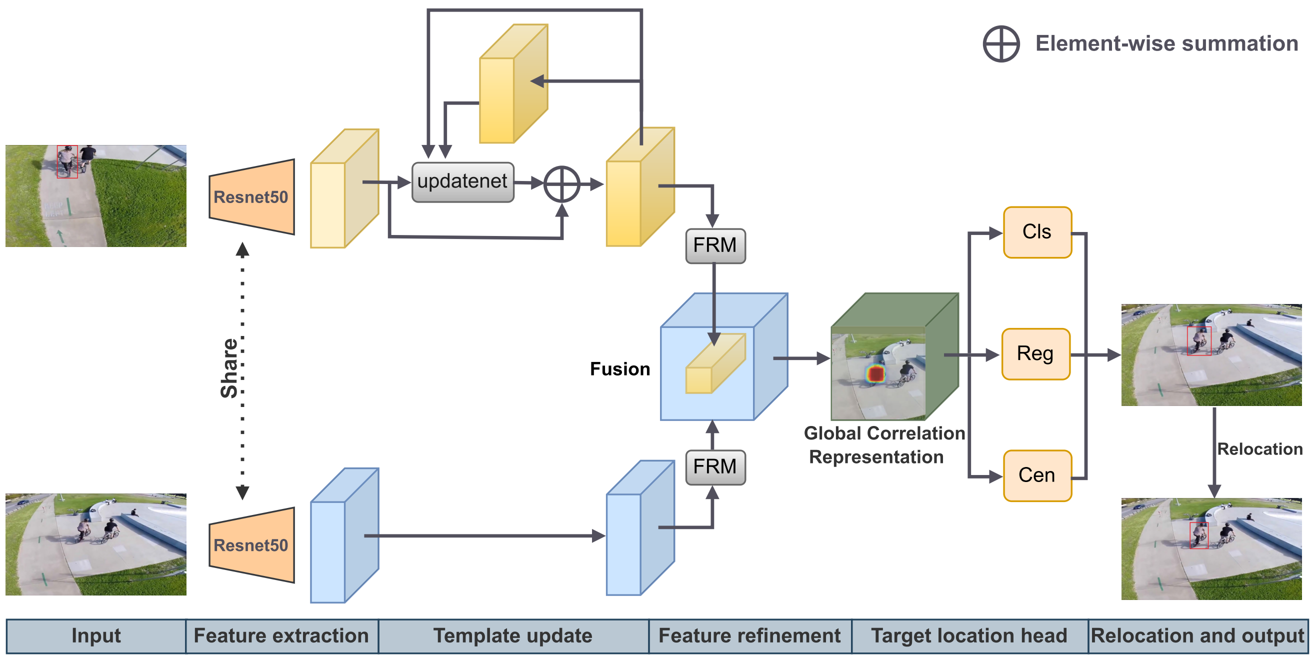

FIGURE 1. Overview for the proposed tracking framework.

features and preserve more foreground information. This module integrates three attention mechanisms to optimize features comprehensively. Subsequently, the refined template features and search features are cross-correlated to obtain the global correlation representation. (4) In the target location head phase, the global correlation representation is inputted into three branches: the classification branch, regression branch, and center branch, to obtain the first-stage tracking box. (5) Lastly, in the relocation output phase, we expand the region of the first-stage tracking box to twice the size of the concentric area for secondary feature extraction. Subsequently, the extracted features are cross-correlated with the template features to obtain the final tracking box output.

A. FEATURE EXTRACTION

Given a template image and a search image, deep learning features are first extracted using a Siamese network, represented as $F_T \in \mathbb{R}^{c\times h\times w}$ and $F_S \in \mathbb{R}^{C\times H\times W}$. h, w, and c denote the height, width, and number of channels of the template features, while H, W, and C respectively refer to the dimensions of the search features. Here, we employ the widely used ResNet50 [14] pretrained on the Imagenet [40] dataset as the feature extraction network, given its excellent feature learning capabilities. It's worth noting that, at this stage, we adopted the same operation as the baseline [10], which is to concatenate the features of the last three layers of the backbone network along the channel dimension for subsequent processing. This is because the last three layers contain more semantic information, and target tracking can be regarded as a foreground-background separation task, making the understanding of semantic information crucial. Using multi-level information helps preserve more semantic and detail information, contributing to more accurate localization. Subsequently, to selectively extract key-level information, we pass the concatenated features through a 1 × 1 convolutional layer to reduce the number of channels to 256, aiming to decrease computational complexity and memory requirements.

To optimize the feature extraction performance of the template and search images before inputting them into the Siamese network, we employed various preprocessing steps. For the template images, we applied enhancement operations such as translation, scaling, blurring, flipping, and color adjustments. Similarly, for the search images, we utilized similar preprocessing operations. Through these preprocessing steps, we significantly improved the effectiveness of the feature extraction process, thereby enhancing the overall performance of the Siamese network. Data augmentation operations enhance the robustness of the model, enabling it to better adapt to image variations in various real-world applications.

B. TEMPLATE UPDATE

In the actual tracking process, since the target's is a state of constant motion, as well as affected by some external factors (e.g.illumination, semi-obscuration), the shape and color of the target are often subject to change. Therefore it is not desirable to utilize only the first frame image of the video sequence as a fixed template feature, some trackers consider using a simple average weighting strategy, such as (1) for template updating. Although this method brings some improvements, this simple method also limits the effectiveness of

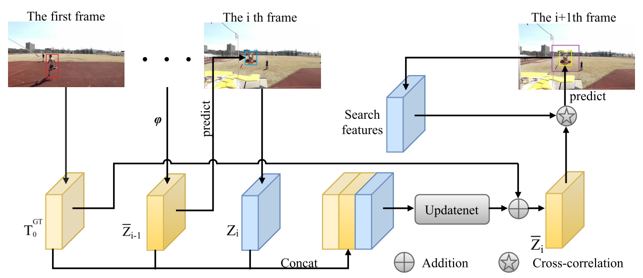

FIGURE 2. Overview for the template update module.

template updating.

$$X_i = \alpha T_i + (1 - \alpha) X_{i-1} \tag{1}$$

Here, i represents the i th frame of the video sequence, T_i denotes the template feature extracted from the predicted box of the i th frame, X_{i-1} represents the template feature used during the position prediction of the i th frame, and X_i denotes the template feature used for predicting the i+1 frame image. Assuming the object's appearance varies consistently and smoothly over successive frames, the update rate is typically set to a fixed ratio value (e.g. $\alpha = 0.01$). But this simple update strategy suffers from several problems:

- It updates the templates at a constant rate for each video sequence, but the template updates required for the same or different video sequences may change dynamically over time.
- It does not make use of the information in the first frame. During tracking, the information from the first frame is the most credible. When there is a target drift or multiple similar interfering objects, this update strategy can be persistently wrong and the prediction box is difficult to locate the correct target.
- This simple linear combination strategy severely limits it. This strategy often fails to meet the situation where the complex appearance of the target changes.

To address the aforementioned issue, we propose a template update module, as illustrated in the Fig. 2. This module primarily updates the template using Updatenet [16], which is a learnable adaptive updating strategy represented by $\varphi(*)$.

$$\overline{Z}_i = \varphi(T_0^{GT}, \overline{Z}_{i-1}, Z_i) \tag{2}$$

Here, T_0^{GT} represents the initial template of the first frame of the video sequence; $\overline{Z}_{i-1}$ denotes the template feature used in predicting the i th frame image; Z_i denotes the feature information obtained from the prediction result of the i th frame image; $\overline{Z}_i$ denotes the updated template features used in the i+1 th frame prediction.

Specifically, Updatenet consists of two convolutional layers. The first convolutional layer has 2304 input channels and 192 output channels, utilizing the ReLU activation function. The second convolutional layer has 192 input channels and 768 output channels. This design allows the network to dynamically adjust feature responses based on the current input, adapting to varying input conditions. During forward propagation, we concatenate T_0^{GT}, $\overline{Z}_{i-1}$, and Z_i along the channel dimension and feed them into the Updatenet network to learn the updated template features. Additionally, we incorporate residual learning by using the template features from the first frame as the residual. The final output is obtained by adding the updated template features to the residual. This design retains the original template feature information while effectively utilizing the updated template features, thereby enhancing the model's accuracy and robustness.

Each update incorporates the initial frame's template information as prior knowledge, thereby preventing large template errors resulting from sudden target changes and subsequent drifting of the tracking box. It is worth noting that due to the simple structure of Updatenet, its tracking speed on the same device only decreases by 5.85 FPS compared to the linear weighting strategy. To avoid overfitting of Updatenet, we selected 20 video sequences from the Lasot [41] dataset for separate training of the template update module. These video sequences cover most tracking challenge attributes, including illumination changes, scale variations, occlusions, and rapid motions. This helps ensure that the SiamTAR model can better adapt to these challenging scenarios when updating the template. For specific training details, please refer to III-E2.

C. FEATURE REFINEMENT MODULE

To refine the response region of target features and filter redundant features, thus better assisting in tracking, we propose a feature refinement module, as illustrated in Fig. 3 (a). This module integrates three types of attention structures: Channel-Attention Block, Spatial-Attention Block, and Self-Attention Block, as shown in Fig. 3 (b), (c), and (d), respectively.

Specifically, the module consists of two parallel branches. One branch concatenates the channel attention module and the spatial attention module, fully utilizing their advantages. Firstly, the channel attention filters and enhances important features by weighting the importance of different feature channels, thereby highlighting critical features and enhancing the model's ability to distinguish between targets and backgrounds. Then, the spatial attention further focuses on the target location by enhancing fine-grained spatial information in the feature map, which is helpful for precise target localization. The other branch consists of a separate self-attention module. Compared to the former two, it captures dependencies between features in the global scope of the tracking image, suitable for handling long-range dependencies and complex backgrounds, beneficial for addressing the rapid motion challenges in tracking. However, the self-attention mechanism involves higher computational costs. Therefore, by treating it as a separate parallel branch, a minimal loss in processing speed can be incurred. This module achieves comprehensive feature extraction and optimization, thereby improving the overall performance of single-target tracking. Next, we will elaborate on these three attention block.

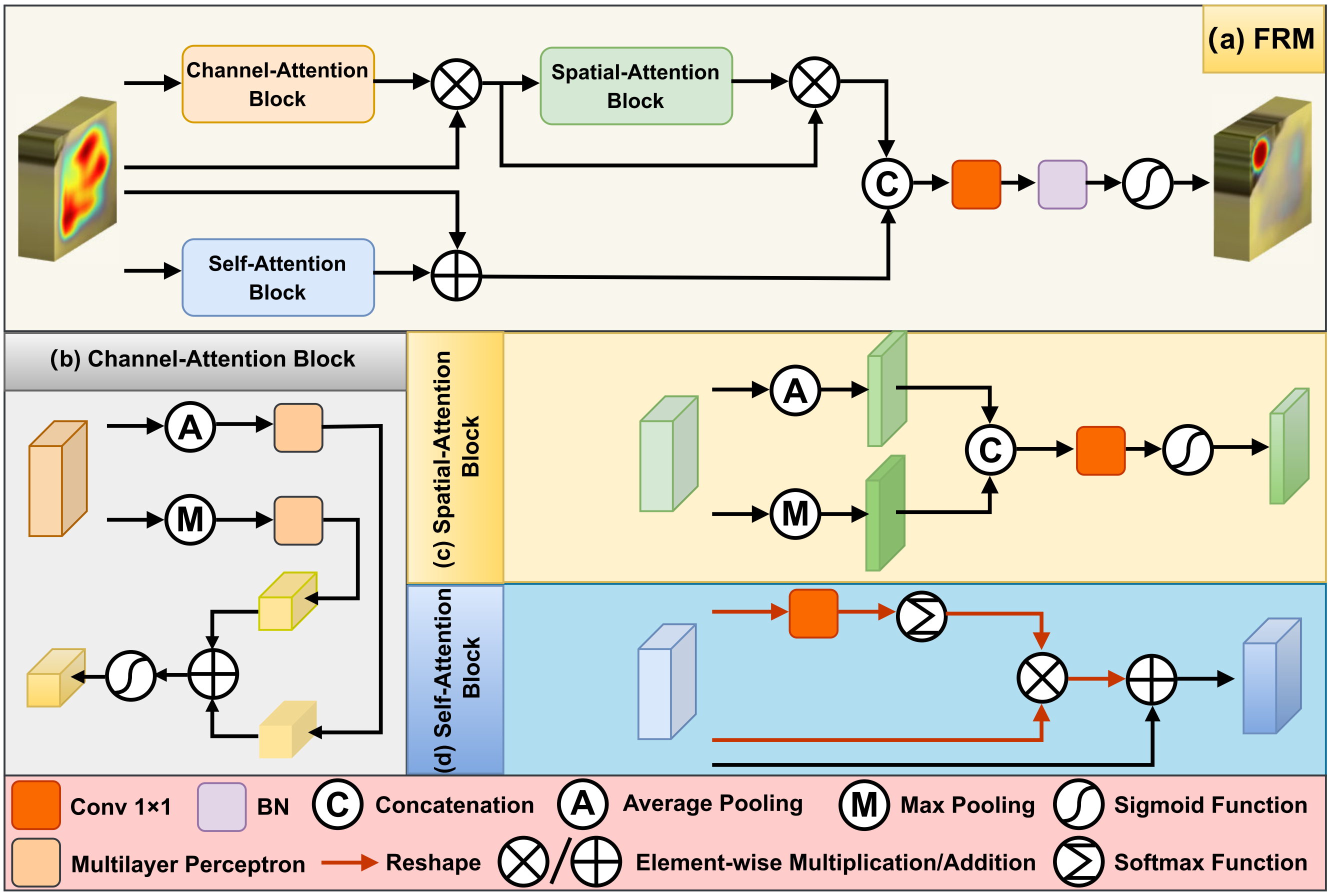

FIGURE 3. Detailed architectures of the feature refinement module. (a) Overview of feature refinement module, (b) Channel-Attention block, (c) Spatial-Attention block, (d) Self-Attention block.

1) CHANNEL ATTENTION BLOCK

This module primarily focuses on the relationships between different channels of the input features. Unlike predefined categories in object detection and classification tasks, object tracking lacks prior information regarding object categories, and the category remains constant throughout the tracking process. Various channel features of deep convolutional network features typically respond to specific object categories; therefore, treating each channel feature equally would hinder its representational capabilities. Specifically, assuming X $\in \mathbb{R}^{C\times H\times W}$ as the input to this module. Without altering its channel size, we first allow X to pass through average-pooling and max-pooling layers to obtain corresponding pooled features. Subsequently, to emphasize the weights between channels, we pass the pooled features through two layers of perceptrons to obtain $X^A \in \mathbb{R}^{C\times 1\times 1}$ and $X^M \in \mathbb{R}^{C\times 1\times 1}$. Then, we add the two to obtain the merged channel attention weights $A^C \in \mathbb{R}^{C\times 1\times 1}$.

$$\mathbf{A}^{\mathbf{C}} = MLP(AvgPooling(\mathbf{X})) + MLP(MaxPooling(\mathbf{X})) \quad (3)$$

At last, the channel attention weights are element-wise multiplied with the input X to obtain the output of this module, $X^C \in \mathbb{R}^{C\times H\times W}$.

2) SPATIAL ATTENTION BLOCK

When features from two branches of the feature extraction network are correlated with each other, the features computed from each spatial location on the search feature can only be obtained from a local region due to the limitation of the size of the sensory field of the template features, thus ignoring the importance of understanding the global context from the whole image. We therefore use the output of the channel attention module as the input to this module, and the two modules are connected in series. Given the input feature $X \in \mathbb{R}^{C\times H\times W}$. Without altering the feature spatial dimensions, we first pass X through two pooling layers. The resulting features are then concatenated along the channel dimension to obtain $X^{A+M} \in \mathbb{R}^{2\times H\times W}$. Subsequently, we utilize a 1 × 1 convolutional layer to reduce the number of channels while maintaining the spatial features, resulting in the generation of spatial attention weights $A^S \in \mathbb{R}^{1\times H\times W}$.

$$\mathbf{A}^{\mathbf{S}} = conv2d(concat(AvgPooling(\mathbf{X}), MaxPooling(\mathbf{X}))) \quad (4)$$

Similarly, the final step involves element-wise multiplication of the spatial attention weights with the input features,

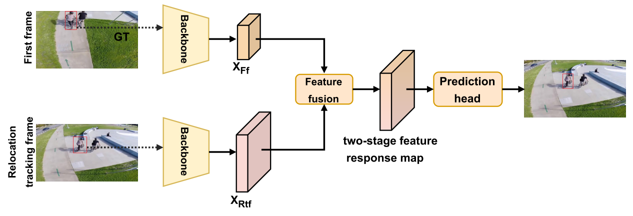

FIGURE 4. **Detailed architectures of the relocation mechanism.**

yielding the output $X^S \in \mathbb{R}^{C\times H\times W}$.

$$\mathbf{X^S} = Sigmoid(\mathbf{A^S})\mathbf{X} \tag{5}$$

3) SELF ATTENTION BLOCK

In real-world tracking, it is common for target occlusion, and similar target interference to occur at the same time, so single-branch feature refinement tends to lose some effective features. To alleviate this situation, we choose to separate this module as a parallel branch so that the features can focus on their own information at the same time. Suppose the input feature is X $\in \mathbb{R}^{C\times H\times W}$. In one branch, X is reshaped into $X^1 \in \mathbb{R}^{1\times C\times N}$, where N = H × W. In the other branch, X is processed using a 1 × 1 convolutional layer with reshape operation to generate $X^2 \in \mathbb{R}^{1\times N\times 1}$, where N = H × W. Then, X^1 and X^2 are multiplied together to obtain $A^K \in \mathbb{R}^{1\times C\times 1}$.

$$\mathbf{A^K} = \mathbf{X^1} softmax(\mathbf{X^2}) \tag{6}$$

Following this step, the reshaped self-attention matrix $A^K \in \mathbb{R}^{1\times C\times 1}$ is simply element-wise added to X, resulting in the self-attention feature $X^{SF} \in \mathbb{R}^{C\times H\times W}$.

The Feature refinement module ultimately concatenates the spatial attention feature X^S and the self-attention feature X^{SF} along the channel dimension to obtain the joint feature $X^{SFS} \in \mathbb{R}^{2C\times H\times W}$. To ensure that the input and output dimensions of the feature refinement module are consistent while preserving salient information, X^{SFS} is processed through a 1 × 1 convolutional layer with channel compression. Subsequently, X^{SFS} is processed by batch normalization and the Sigmoid function, and then element-wise added to the input X to obtain the output of the Feature refinement module, $X^{FRM} \in \mathbb{R}^{C\times H\times W}$.

$$\mathbf{X^{FRM}} = Sigmoid(BN(\mathbf{X^{SFS}})) + \mathbf{X} \tag{7}$$

It is important to note that the FRM module is designed as a plug-and-play tool that can seamlessly integrate with various Siamese network tracking algorithms. Its primary function is to optimize features based on the outputs of the Siamese network, providing more accurate inputs for subsequent cross-correlation operations. This effectively enhances the overall performance of the tracking system.

D. RELOCATION MECHANISM

In practical tracking applications, when the target faces common challenging scenarios such as occlusion, rapid movement, and deformation, SiamTAR mainly relies on the template update module and the feature refinement module to handle these situations. However, after the first stage tracking box is output, if extreme conditions occur, such as temporary loss of the target, interference from similar objects leading to tracking box drift, and mismatched tracking box size, the template update module and feature refinement module alone are insufficient.

To address these issues, we introduce a relocation mechanism [42] into the tracking framework, with details shown in Fig. 4. The main idea of this mechanism is to expand the first stage tracking box region to twice its size concentrically, and then perform secondary feature extraction. It is important to note that this search area is approximately twice the size of the target, which is much smaller than the search area of the initial tracker. Feature extraction uses a lightweight backbone network, EfficientNet [43], retaining only the third, fourth, and fifth layer features. The newly extracted search area features are then cross-correlated with the most reliable first frame template features to obtain the second stage feature response map. This second stage feature response map is more focused than the first stage because the smaller search area suppresses the confusing background regions, allowing the model to concentrate on the foreground region, thus aiding in more precise localization. Finally, the feature response map is fed into the prediction head to produce the second stage tracking box, which is the final output of the SiamTAR model.

This mechanism improves the tracker's performance while introducing minimal latency, primarily because the smaller search area and lightweight backbone network do not significantly increase computational costs. The relocation mechanism is designed as a plug-and-play method, typically applied after the tracker outputs the tracking box. By implementing this mechanism, the tracker's performance and robustness are further enhanced.

E. TRAINING THE OVERALL TRACKING FRAMEWORK

The training of the model consists of two parts: offline training and online training. First, the baseline and feature refinement modules are trained offline, and the parameters with the best performance are selected through offline training. This part of training does not include template update. The next online training unfreezes the template update part while loading the optimal parameters obtained from the offline training, and trains only the template update part. In order to obtain a more robust tracker, the online training uses a multi-learning rate stepwise optimization strategy. It is worth noting that the relocalization part is trained separately.

1) OFFLINE TRAINING

The model is trained end-to-end offline. As shown in Fig.1, and its target location head comprises three components: the classification branch(Cls), the regression branch(Reg), and the center branch(Cen). The classification branch produces the similarity scores between targets and non-targets at each location of the response map, denoted as $A^{cls} \in \mathbb{R}^{w\times h\times 2}$. The regression branch produces the vertical distances from each location to the four edges of the target bounding box. Noted as $A^{reg} \in \mathbb{R}^{w\times h\times 4}$. The center branch is a weight map which is mainly used to remove outliers. Noted as $A^{cen} \in \mathbb{R}^{w\times h\times 1}$. This is because locations far from the center of the target tend to produce low-quality predicted bounding boxes.

Assuming that (x0, y0) and (x1, y1) represent the top-left and bottom-right corners of the ground truth bounding box, and (x, y) represents the position in the original image corresponding to each point (i, j) on the response map, the four-dimensional coordinates of A^{reg} at point (i, j) can be expressed as:

$$\begin{aligned} T^0_{(i,j)} &= \tilde{l} = x - x0, \quad T^1_{(i,j)} = \tilde{t} = y - y0 \\ T^2_{(i,j)} &= \tilde{r} = x1 - x, \quad T^3_{(i,j)} = \tilde{b} = y1 - y \end{aligned} \tag{8}$$

We then calculate the regression loss function by using the following formula:

$$L_{reg} = \frac{1}{\sum \vartheta(T_{(i,j)})} \sum_{i,j} \vartheta(T_{(i,j)}) L_{IOU}(A^{reg}(i,j,:), T_{(x,y)}) \tag{9}$$

L_{IOU} denotes the Intersection over Union loss function between the truth box and the prediction box. The $\vartheta(*)$function can be defined as:

$$\vartheta(T_{(i,j)}) = \begin{cases} 1, & if\ T^k_{(i,j)} > 0, k = 0,1,2,3 \\ 0, & otherwise \end{cases} \tag{10}$$

Each element $C_{(i,j)}$ in the center branch $\mathbf{A^{cen}} \in \mathbb{R}^{w\times h\times 1}$ is defined as:

$$C_{(i,j)} = \vartheta(T_{(i,j)}) \times \sqrt{\frac{min(\tilde{l},\tilde{r})}{max(\tilde{l},\tilde{r})} \times \frac{min(\tilde{t},\tilde{b})}{max(\tilde{t},\tilde{b})}} \tag{11}$$

The center branch loss L_{CEN} is defined as:

$$\begin{aligned} L_{CEN} = \frac{-1}{\sum \vartheta(T_{(i,j)})} \sum_{\vartheta(T_{(i,j)})==1} & C_{(i,j)} * log A^{cen}_{(i,j)} \\ & + (1 - C_{(i,j)}) * log(1 - A^{cen}_{(i,j)}) \end{aligned} \tag{12}$$

The total loss function of SiamTAR is defined as:

$$L_{total} = L_{cls} + \mu_1 L_{cen} + \mu_2 L_{reg} \tag{13}$$

where L_{cls} denotes the cross-entropy loss of the classification branch, μ_1, μ_2 represent the weight parameters of the center branch loss and the regression branch loss, respectively, and during the model training period, we set $\mu_1 = 1$ and $\mu_2 = 3$, respectively.

For the relocation mechanism, we choose two random frames X_{Ff} and X_{Rtf} with video sequence interval less than 50 frames as input pairs for training, where X_{Ff} is cropped by expanding the region twice as much as the real bounding box, and X_{Rtf} is cropped by adding random jitter magnitude on top of X_{Ff}, The details are given in [42].

2) ONLINE TRAINING

After the offline training, we have to load the optimal parameters and unfreeze the template update part for the two-stage online training. The purpose of online training is to make the template features generated by the template update part more effective, and the whole training process is achieved by minimizing the Euclidean distance between the updated template features and the truth box features of the current frame.

$$L_2 = \|\varphi(T^{GT}_0, \overline{Z}_{i-1}, Z_i) - T^{GT}_{i+1}\|_2 \tag{14}$$

where the template features of the initial frame T^{GT}_0 and the current frame T^{GT}_{i+1} can be obtained by extracting features from the truth box in the corresponding frame. In addition to this, the template updating part adopts a multi-stage training strategy. The first stage uses standard linear updating on the training dataset to generate cumulative templates and actual predicted positions for each frame.

$$\bar{Z}^0_i = (1-\gamma)\bar{Z}^0_{i-1} + \gamma \bar{Z}^0_i \tag{15}$$

γ is set to 0.0102. The IOU values of the generated prediction box and the truth box are used to determine whether the target is lost to follow, preventing training through a large number of lost frames, which would move the training effect in a negative direction. The next two phases use a template update structure to obtain cumulative templates and actual position predictions for training. Each stage uses the best performing model from the previous stage to generate cumulative templates and actual position predictions. The training dataset was taken from 20 of the 280 sequences in the lasot dataset. In each of these stages we train twice. For the first training we chose learning rates of 10^{-5} to 10^{-6}, 10^{-6} to 10^{-7}, and 10^{-7} to 10^{-8}. the second training loads the best model from the first training, with learning rates set to 10^{-8} to 10^{-9}, 10^{-9} to 10^{-10}, 10^{-10} to 10^{-11}, respectively.

IV. EXPERIMENTS

A. IMPLEMENTATION DETAILS

The proposed SiamTAR is implemented on a 3090 graphics card using Pytorch 1.11 and cuda 11.1. We employ ResNet-50 [14] as the backbone network, which has been pre-trained on ImageNet [40]. During training, the batch size is set to 16, and 20 epochs are trained using Stochastic gradient descent (SGD) with an original learning rate of 0.001. In the first 10 epochs, the backbone network is frozen. In the last 10 epochs, the last 3 layers of the backbone network are unfrozen to train together. We trained SiamTAR on five datasets: GOT-10K [17], COCO [44], ImageNet DET, ImageNet VID [45], YouTube-BB [46], and tested it on five tracking benchmarks. It should be noted that the tests conducted on GOT-10K [17] and DTB70 [20] were only trained with the GOT-10K [17] dataset.

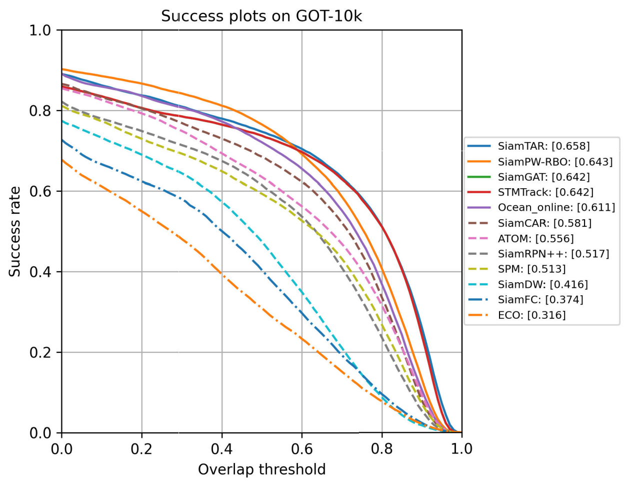

FIGURE 5. Success plots for GOT-10K test set. Best viewed on high-resolution display.

TABLE 2. Quantitative comparison on the GOT-10K test set.

Tracker	AO	$SR_{0.5}$	$SR_{0.75}$
ECO [47]	0.316	0.309	0.111
SiamFC [5]	0.374	0.404	0.144
SiamDW [48]	0.416	0.475	0.144
SPM [49]	0.513	0.593	0.359
SiamRPN++ [9]	0.517	0.615	0.329
ATOM [50]	0.556	0.634	0.402
SiamCAR [10]	0.581	0.685	0.444
Ocean-online [13]	0.611	0.721	0.473
STMTrack [51]	0.642	0.737	0.579
SiamGAT [52]	0.642	0.737	0.579
SiamPW-RBO [21]	0.643	0.765	0.508
SiamTAR(ours)	0.658	0.747	0.583

B. PUBLIC DATASET EVALUATION

1) EVALUATION ON GOT-10K DATASET

GOT-10K [17] is a large, high-level general purpose target tracking dataset. It contains videos of over 10000 real-world objects in motion. The categories of the training dataset and the test dataset do not overlap. The authors need train their model using the provided training dataset and evaluate it using the 180 provided video sequences. To get a more authoritative assessment result, we need to upload the generated prediction target box file to the official website. The assessment metrics provided on the website include success rate, average overlap, and speed. AO represents the average of all predicted bounding boxes overlapped with annotated truth boxes. $SR_{0.5}$ represents the proportion of frames with more than 50% overlap, while $SR_{0.75}$ represents the proportion of frames with more than 75% overlap. To assess the generality of SiamTAR, we tested it on the GOT-10K [17] dataset and compared it to state-of-the-art trackers, including SiamGAT [52], STMTrack [51], SiamCAR [10], SiamPW-RBO [21] and some classical baselines. As shown in Fig. 5. SiamTAR performs the most prominently on these 12 trackers. Table 2 presents the details of the comparison of the different metrics, and SiamTAR ranks first in two metrics. Compared with SiamPW-RBO [21], our tracker scores 7.5% higher on $SR_{0.75}$ than SiamPW-RBO [21], even though it scores 1.8% lower on $SR_{0.5}$.

SiamTAR is compared with four well-known trackers (SiamGAT [52] and SiamCAR [10] and SiamRPN++ [9] and STMTrack [51]) for visualization of 6 video sequences tracked through the GOT-10K [17] dataset, as shown in Fig.6. SiamTAR can draw more accurate and closer to the target tracking box in the presence of similar objects, fast motion, scale change and full occlusion. Taking the first video sequence tracking the ship as an example, the remaining four trackers are no longer able to track the target accurately compared to the SiamTAR due to the large deformation produced by the ship's rapid motion at sea level and the similarity of some of the ship's detailed information to the background. This is mainly benefited from SiamTAR's template update as well as feature refinement module. Even if the tracked object has deformation, the template update module can still accurately predict the next frame template features from the previous information, which enhances the robustness of the trackers. The feature refinement module helps to accurately find the saliency information of the target features and thus locate the position of the target object on the search image. In the second video sequence, benefiting from the relocalization mechanism in SiamTAR, its tracking box is closer to the target size than the remaining four trackers. Since the trackers follow the GOT-10K [17] protocol and the labeled frames of the test dataset are invisible to us and the trackers, the tracking results on GOT-10K [17] are more plausible than those on the other datasets.

2) EVALUATION ON OTB100 DATASET

OTB100 [19] is a classical test dataset for visual object tracking, which provides a fair robustness testbed. It has 100 video sequences labeled with 11 interference properties, These properties are occlusion (OCC), background clutter (BC), in-plane rotation (IPR), out-of-view (OV), scale variation (SV), fast motion (FM), out-of-plane rotation (OPR), motion blur (MB), deformation (DEF), low resolution (LR) and illumination variation (IV). The OTB100 benchmark includes two metrics: precision and success. The precision score is the percentage of frames where the Euclidean distance between the center of the predicted box and the center of the ground-truth box is under 20 pixels. The success score indicates the proportion of frames correctly tracked to the target at different thresholds.

SiamTAR is contrasted with 9 trackers including SiamPW-RBO [21], Ocean-online [13], SiamRPN++ [9], SiamBAN [11] etc. Fig. 7 shows that SiamTAR performs first among all trackers regarding precision and success rate for OTB100 [19]. As shown in Fig. 8 shows that SiamTAR achieves a competitive performance among all the trackers in the OTB100, and it is worth noting that our tracker is 0.1%

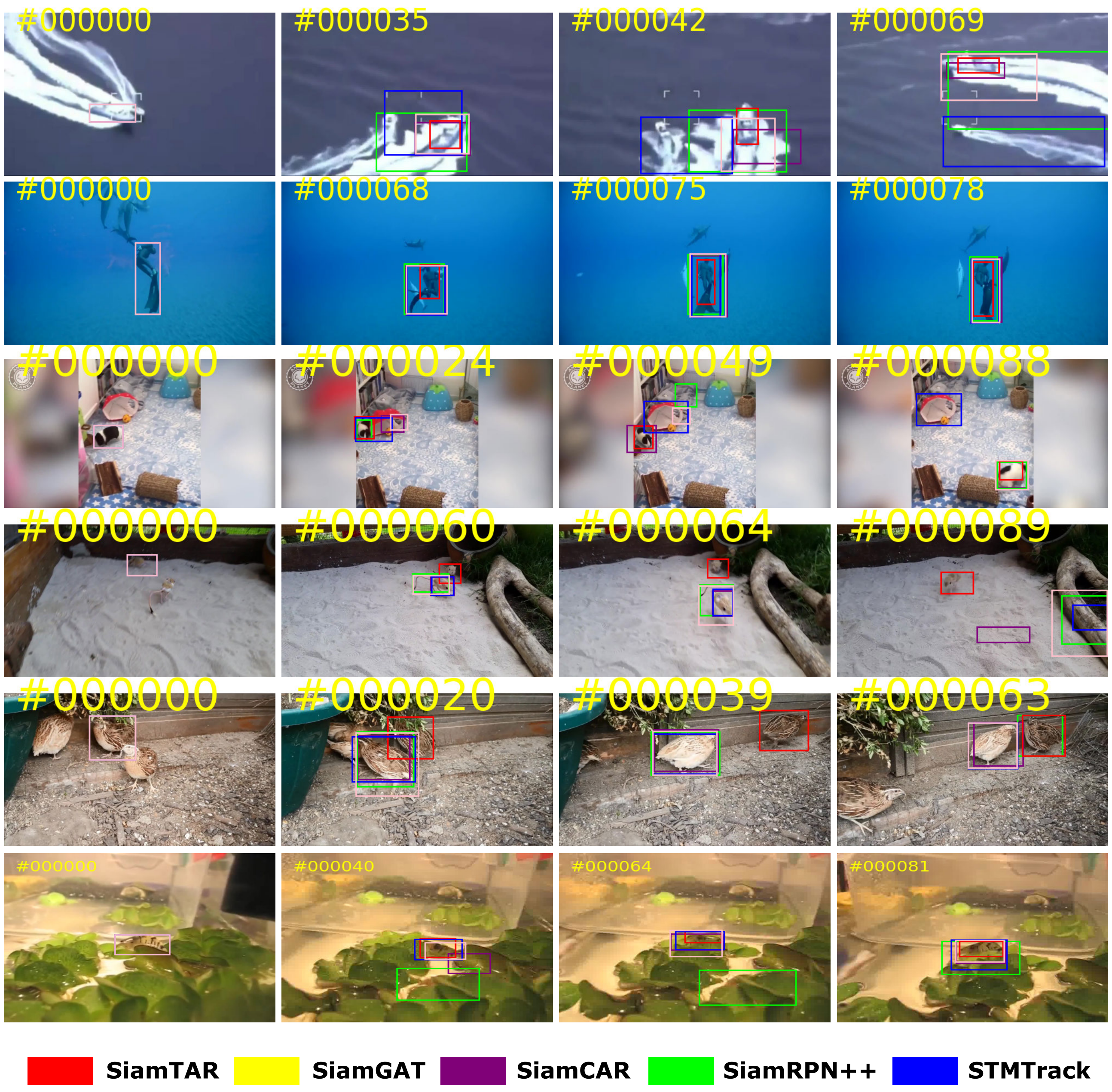

FIGURE 6. Qualitative results on the GOT-10K test set. Tracking results are for sequences GOT-10k_Test_000004, GOT-10k_Test_000008, GOT-10k_Test_000096, GOT-10k_Test_000100, GOT-10k_Test_000146, and GOT-10k_Test_000175. Best viewed on a high resolution display.

points lower than Ocean-online in terms of accuracy, but 3.8% points higher in terms of success rate, which is a solid first place in the performance among these trackers. Fig. 8 presents success plots as well as accuracy plots for different attributes in the OTB100 dataset. SiamTAR is able to achieve high performance metrics on all challenging attributes, with significant performance gains over the baseline tracker on attributes such as background clutter, deformation, low resolution, and out-of-view. These comparisons show that SiamTAR is better able to handle challenging interference and large posture changes.

3) EVALUATION ON UAV123 DATASET

UAV123 [18] is a collection of 123 sequences recorded by low-altitude UAVs., including over 110K frames. Vertical bounding boxes are used to annotate every sequence thoroughly. Different from other datasets, the view of UAV123 is aerial and tracking objects are typically quite tiny. The dataset's objects are mostly affected by fast motion, large scale variations, large light variations, and occlusions, which make the tracking process diffcult.

SiamTAR is compared to 9 superior trackers, including Gift [53], STMTrack [51], SiamCAR [10], and SiamRPN++

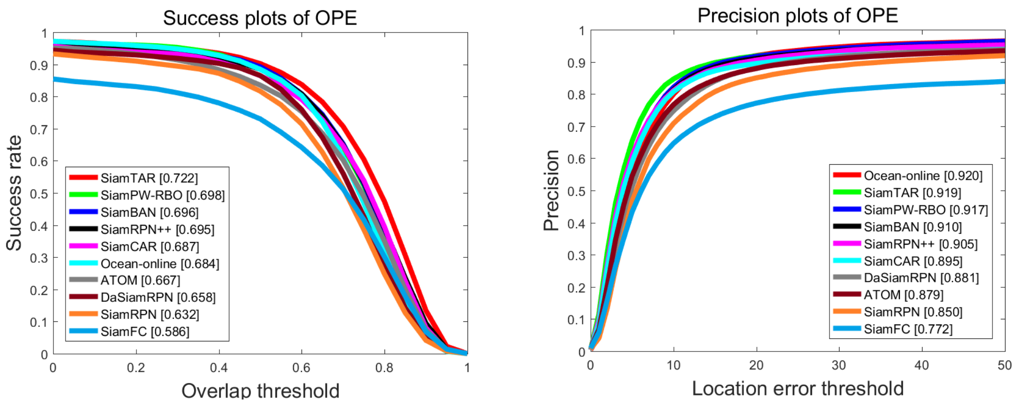

FIGURE 7. Success and precision plots for OTB100 test set. Best viewed on high-resolution display.

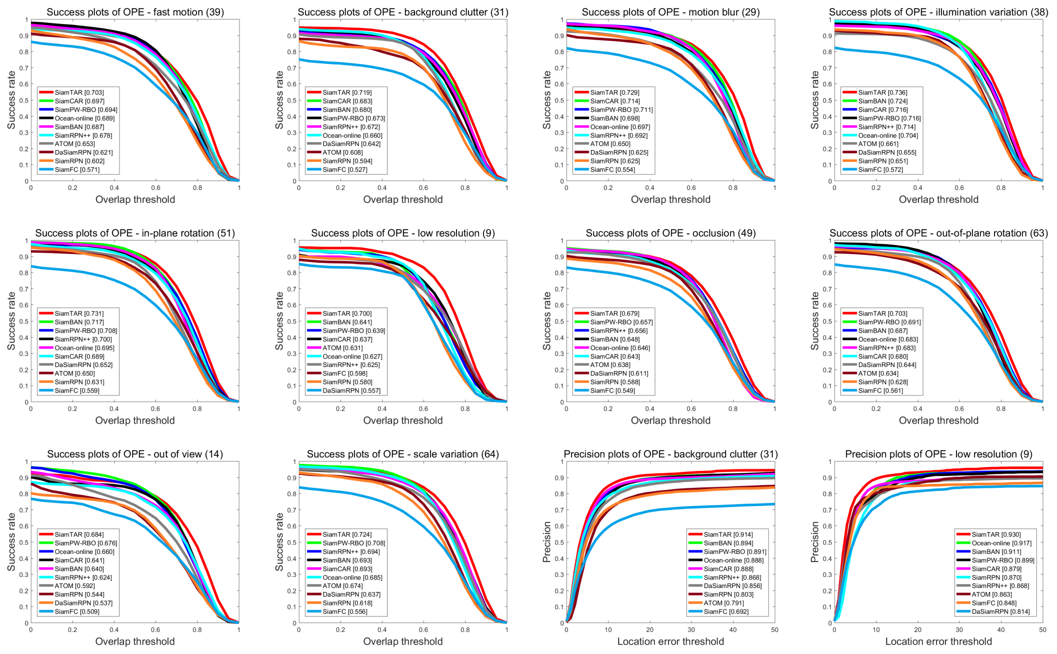

FIGURE 8. Success plots for 11 challenge attributes of OTB100 test set. Best viewed on high resolution display.

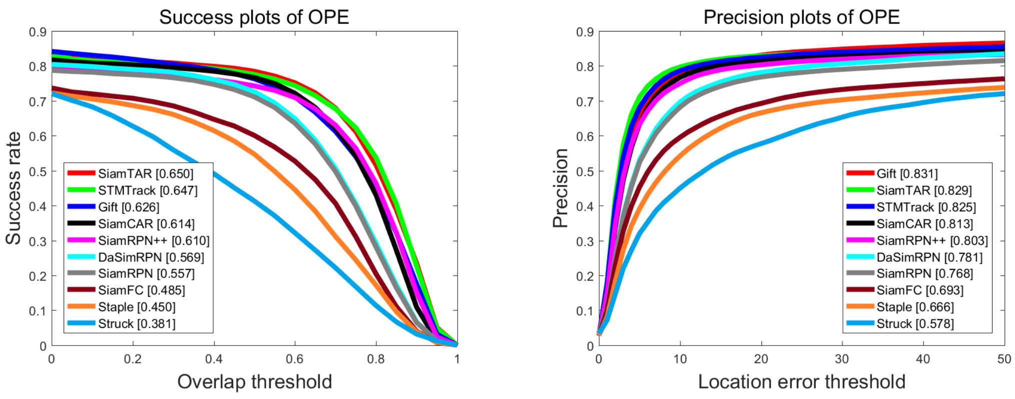

FIGURE 9. Success and precision plots for UAV123 test set. Best viewed on high-resolution display.

[9]. Success rate and precision are used as metrics to assess the model's good performance. As seen in Fig. 9, SiamTAR performs exceptionally well on both of these metrics, especially surpassing SiamCAR [10] by 3.6% in success rate and

TABLE 3. Ablation study on compression ratio (cr) of the FRM module.

Method	Success	Precision
Baseline+FRM(cr=2)	0.608	0.785
Baseline+FRM(cr=4)	0.610	0.790
Baseline+FRM(cr=8)	0.611	0.801
Baseline+FRM(cr=16)	0.620	0.809
Baseline+FRM(cr=32)	0.614	0.805
Baseline+FRM(cr=64)	0.606	0.793
Baseline+FRM(cr=128)	0.601	0.780

TABLE 4. Ablation study on GOT-10K dataset.

Method	AO	$SR_{0.5}$	$SR_{0.75}$
Baseline	0.581	0.685	0.444
Baseline+TUM	0.596	0.705	0.455
Baseline+FRM	0.595	0.701	0.460
Baseline+RM	0.627	0.704	0.556
Baseline+TUM+FRM	0.611	0.724	0.477
Baseline+TUM+RM	0.634	0.713	0.573
Baseline+FRM+RM	0.640	0.723	0.577
Full Model (SiamTAR)	0.658	0.747	0.583

1.6% in precision rate. While our model falls short of Gift [53] by 0.2% in precision, it outperforms it by 2.4% in success.

C. ABLATION EXPERIMENT

In the subsequent experiment, we investigated the influence of individual components in SiamTAR and conducted ablation studies on the DTB70 [20] dataset and the GOT-10K [17] dataset. SiamCAR [10] is considered the baseline model.

1) ABLATION EXPERIMENT OF FRM ON DTB70 DATASET

We conducted an ablation study on the DTB70 dataset to determine the optimal channel compression ratio for the channel attention module in the FRM module. The channel compression ratios were set to 2, 4, 8, 16, 32, 64, and 128. The FRM modules with different compression ratios were integrated into the Baseline network to evaluate performance. The detailed results are shown in Table 3. When the compression ratio is set to 16, the tracking performance is optimal, with Success of 0.620 and Precision of 0.809.

2) ABLATION EXPERIMENT OF SiamTAR ON GOT-10k DATASET

As shown in Table 4, in the ablation study evaluating the impact of three modules on the GOT-10K [17] dataset, we observed that the TUM (Template Update Module) outperforms the FRM module in $SR_{0.5}$ and AO metrics but does not excel in $SR_{0.75}$. This suggests that the TUM module enhances the tracker's stability by generating adaptive template features, reducing the likelihood of predicting low-quality bounding boxes during tracking. However, it does not excel in predicting high-quality bounding boxes. The FRM module is integrated into the Siamese network features before fusion, enhancing crucial features and suppressing secondary features, generating more accurate feature response maps, and thus aiding in the generation of higher-quality prediction boxes. For RM (Relocation Mechanism), the highest improvement was observed with individual components, confirming the compatibility of this mechanism with our model framework. Additionally, we conducted ablation experiments by combining each component in pairs, and it is evident that they all yielded different positive effects. Compared to the baseline, SiamTAR achieved improvements of 7.7%, 6.2%, and 13.9% in AO, $SR_{0.5}$, and $SR_{0.75}$, respectively.

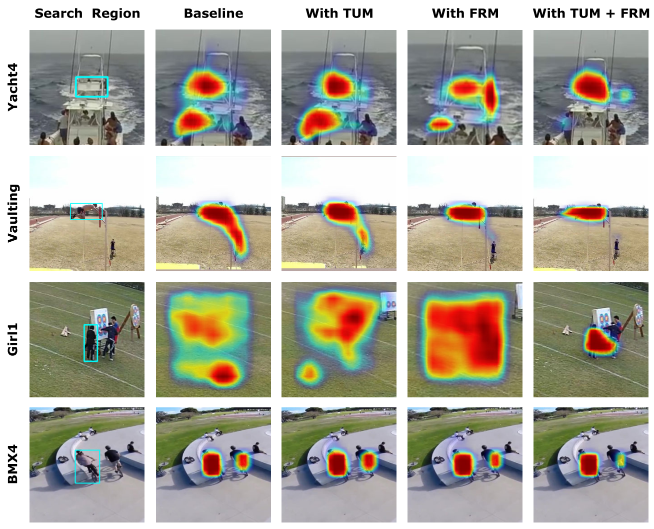

FIGURE 10. Visualization of heatmaps using without adding any modules (second column), the proposed TUM module (third column), FRM module (fourth column), and TUM+FRM module (fifth column) for four sequences plotted in DTB70: Yacht4, Vaulting, Girl1, and BMX4.

TABLE 5. Ablation study on DTB70 dataset.

Method	Success	Precision
Baseline	0.595	0.781
Baseline+TUM	0.617	0.796
Baseline+FRM	0.620	0.809
Baseline+TUM+FRM	0.642	0.842

3) ABLATION EXPERIMENT OF SiamTAR ON DTB70 DATASET

In our ablative study on DTB70 [20], we conducted heat map visualization analysis of the feature outputs for the TUM and FRM modules, as shown in Fig. 10. Table 5 corresponds to the ablation study of these two modules. The first column represents the search regions of video frames, the second column displays the visualization of baseline feature outputs (without any modules), the third column shows the feature visualization with only the TUM module, the fourth column presents the feature visualization with only the FRM module, and the last column illustrates the feature visualization with both modules. From Fig. 10, it is evident that the tracker combining both modules can better identify the target's location, with response regions more suited to the target's appearance, effectively suppressing background noise and interference from similar objects better than other trackers. This demonstrates that the feature refinement and template update modules in SiamTAR can optimize the tracker in different aspects, and their collaboration yields positive effects.

V. CONCLUSION

In this paper, we proposed a tracking framework SiamTAR that can adaptively update templates online, which consists of a Template Update Module and a Feature Refinement Module. The Template Update Module fuses the initial frame, the historical frames, and the current frame's prior information to obtain new template feature. This module significantly enhances the feature representation of the target in the face of illumination changes, scale changes, object deformation, and low-resolution challenges, and improves the robustness of the tracker. The Feature Refinement Module, which refines and efficiently combines the target features in three dimensions. It highlights critical features and weakens the influence of secondary features. By integrating this module in Siamese network leads to a closer contextual relationship between the template features and the search features, which in turn generates a response map that is more conducive to target localization. In addition, we introduce a relocalization mechanism, which significantly improves the accuracy of the tracking frame. Extensive experimental results show that the proposed SiamTAR can achieve competitive performance and real-time speed on four mainstream tracking benchmarks.

In the future, the FRM module will be applied to more Siamese network trackers (e.g., SiamBAN [11] and Ocean [13]), and further research will be conducted on the lightweight design of the template update module and the relocation mechanism.

REFERENCES

[1] D.-H. Lee, K.-L. Chen, K.-H. Liou, C.-L. Liu, and J.-L. Liu, "Deep learning and control algorithms of direct perception for autonomous driving," *Int. J. Speech Technol.*, vol. 51, no. 1, pp. 237–247, Jan. 2021.

[2] J. Liu, W. Song, C. Chen, and F. Liu, "Cross-modality person re-identification via channel-based partition network," *Int. J. Speech Technol.*, vol. 52, no. 3, pp. 2423–2435, Feb. 2022.

[3] Y. Xiao, Q. Yuan, J. He, Q. Zhang, J. Sun, X. Su, J. Wu, and L. Zhang, "Space-time super-resolution for satellite video: A joint framework based on multi-scale spatial–temporal transformer," *Int. J. Appl. Earth Observ. Geoinf.*, vol. 108, Apr. 2022, Art. no. 102731.

[4] Z. Cao, Z. Huang, L. Pan, S. Zhang, Z. Liu, and C. Fu, "TCTrack: Temporal contexts for aerial tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 14798–14808.

[5] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional Siamese networks for object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, Amsterdam, The Netherlands. Cham, Switzerland: Springer, Oct. 2016, pp. 850–865.

[6] B. Li, J. Yan, W. Wu, Z. Zhu, and X. Hu, "High performance visual tracking with Siamese region proposal network," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 8971–8980.

[7] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015.

[8] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, and W. Hu, "Distractor-aware Siamese networks for visual object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 101–117.

[9] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, and J. Yan, "SiamRPN++: Evolution of Siamese visual tracking with very deep networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4277–4286.

[10] D. Guo, J. Wang, Y. Cui, Z. Wang, and S. Chen, "SiamCAR: Siamese fully convolutional classification and regression for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6268–6276.

[11] Z. Chen, B. Zhong, G. Li, S. Zhang, and R. Ji, "Siamese box adaptive network for visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 6667–6676.

[12] Y. Xu, Z. Wang, Z. Li, Y. Yuan, and G. Yu, "SiamFC++: Towards robust and accurate visual tracking with target estimation guidelines," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 7, Apr. 2020, pp. 12549–12556.

[13] Z. Zhang, H. Peng, J. Fu, B. Li, and W. Hu, "Ocean: Object-aware anchor-free tracking," in *Proc. 16th Eur. Conf. Comput. Vis.*, Glasgow, U.K. Cham, Switzerland: Springer, Aug. 2020, pp. 771–787.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 25, 2012.

[16] L. Zhang, A. Gonzalez-Garcia, J. V. D. Weijer, M. Danelljan, and F. S. Khan, "Learning the model update for Siamese trackers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 4009–4018.

[17] L. Huang, X. Zhao, and K. Huang, "GOT-10k: A large high-diversity benchmark for generic object tracking in the wild," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 43, no. 5, pp. 1562–1577, May 2021.

[18] M. Mueller, N. G. Smith, and B. Ghanem, "A benchmark and simulator for UAV tracking," in *Proc. 14th Eur. Conf. Comput. Vis. (ECCV)*, Amsterdam, The Netherlands. Cham, Switzerland: Springer, Oct. 2016, pp. 445–461.

[19] Y. Wu, J. Lim, and M.-H. Yang, "Online object tracking: A benchmark," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 2411–2418.

[20] S. Li and D.-Y. Yeung, "Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models," in *Proc. AAAI Conf. Artif. Intell.*, vol. 31, 2017.

[21] F. Tang and Q. Ling, "Ranking-based Siamese visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 8731–8740.

[22] X. Chen, B. Yan, J. Zhu, D. Wang, X. Yang, and H. Lu, "Transformer tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 8122–8131.

[23] N. Wang, W. Zhou, J. Wang, and H. Li, "Transformer meets tracker: Exploiting temporal context for robust visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 1571–1580.

[24] L. Huang, X. Zhao, and K. Huang, "GlobalTrack: A simple and strong baseline for long-term tracking," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2020, vol. 34, no. 7, pp. 11037–11044.

[25] D. Yang, J. He, Y. Ma, Q. Yu, and T. Zhang, "Foreground-background distribution modeling transformer for visual object tracking," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 10117–10127.

[26] Q. Wang, Z. Teng, J. Xing, J. Gao, W. Hu, and S. Maybank, "Learning attentions: Residual attentional Siamese network for high performance online visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4854–4863.

[27] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg, "Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1430–1438.

[28] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proc. 14th Eur. Conf. Comput. Vis.*, Amsterdam, The Netherlands. Cham, Switzerland: Springer, Oct. 2016, pp. 472–488.

[29] T. Yang and A. B. Chan, "Learning dynamic memory networks for object tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 152–167.

[30] Y. Yao, X. Wu, L. Zhang, S. Shan, and W. Zuo, "Joint representation and truncated inference learning for correlation filter based tracking," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 552–567.

[31] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 7132–7141.

[32] J. Fu, J. Liu, H. Tian, Y. Li, Y. Bao, Z. Fang, and H. Lu, "Dual attention network for scene segmentation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3141–3149.

[33] K. Jiang, W. Liu, Z. Wang, X. Zhong, J. Jiang, and C.-W. Lin, "DAWN: Direction-aware attention wavelet network for image deraining," in *Proc. 31st ACM Int. Conf. Multimedia*, Oct. 2023, pp. 7065–7074.

[34] K. Jiang, Z. Wang, P. Yi, C. Chen, Z. Han, T. Lu, B. Huang, and J. Jiang, "Decomposition makes better rain removal: An improved attention-guided deraining network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3981–3995, Oct. 2021.

[35] Y. Xiao, Q. Yuan, K. Jiang, J. He, C.-W. Lin, and L. Zhang, "TTST: A top-*k* token selective transformer for remote sensing image super-resolution," *IEEE Trans. Image Process.*, vol. 33, pp. 738–752, 2024.

[36] X. Chen, M. Wang, J. Ling, H. Wu, B. Wu, and C. Li, "Ship imaging trajectory extraction via an aggregated you only look once (YOLO) model," *Eng. Appl. Artif. Intell.*, vol. 130, Apr. 2024, Art. no. 107742.

[37] X. Chen, S. Liu, R. W. Liu, H. Wu, B. Han, and J. Zhao, "Quantifying Arctic oil spilling event risk by integrating an analytic network process and a fuzzy comprehensive evaluation model," *Ocean Coastal Manage.*, vol. 228, Sep. 2022, Art. no. 106326.

[38] X. Chen, S. Liu, J. Zhao, H. Wu, J. Xian, and J. Montewka, "Autonomous port management based AGV path planning and optimization via an ensemble reinforcement learning framework," *Ocean Coastal Manage.*, vol. 251, May 2024, Art. no. 107087.

[39] X. Chen, S. Lv, W.-L. Shang, H. Wu, J. Xian, and C. Song, "Ship energy consumption analysis and carbon emission exploitation via spatial–temporal maritime data," *Appl. Energy*, vol. 360, Apr. 2024, Art. no. 122886.

[40] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[41] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, and H. Ling, "LaSOT: A high-quality benchmark for large-scale single object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 5369–5378.

[42] B. Yan, X. Zhang, D. Wang, H. Lu, and X. Yang, "Alpha-refine: Boosting tracking performance by precise bounding box estimation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 5285–5294.

[43] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.

[44] T. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft COCO: Common objects in context," in *Proc. 13th Eur. Conf. Comput. Vis.*, Zurich, Switzerland. Cham, Switzerland: Springer, Sep. 2014, pp. 740–755.

[45] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Dec. 2015.

[46] E. Real, J. Shlens, S. Mazzocchi, X. Pan, and V. Vanhoucke, "YouTube-BoundingBoxes: A large high-precision human-annotated data set for object detection in video," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7464–7473.

[47] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ECO: Efficient convolution operators for tracking," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6931–6939.

[48] Z. Zhang and H. Peng, "Deeper and wider Siamese networks for real-time visual tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4586–4595.

[49] G. Wang, C. Luo, Z. Xiong, and W. Zeng, "SPM-tracker: Series-parallel matching for real-time visual object tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 3638–3647.

[50] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "ATOM: Accurate tracking by overlap maximization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 4655–4664.

[51] Z. Fu, Q. Liu, Z. Fu, and Y. Wang, "STMTrack: Template-free visual tracking with space-time memory networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 13769–13778.

[52] D. Guo, Y. Shao, Y. Cui, Z. Wang, L. Zhang, and C. Shen, "Graph attention tracking," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9538–9547.

[53] L. Wei, Z. Xi, Z. Hu, and H. Sun, "Graph attention information fusion for Siamese adaptive attention tracking," *Int. J. Speech Technol.*, vol. 53, no. 2, pp. 2068–2087, Jan. 2023.

JIA WEN received the Ph.D. degree in computer application from Beihang University, China, in 2011. She is currently an Associate Professor with Yanshan University. Her recent research interests include intelligent surveillance systems and recognizing and understanding activities in video through machine learning.

KEJUN REN received the bachelor's degree in software engineering from Shangqiu Normal University. He is currently pursuing the master's degree in computer technology with Yanshan University, Qinhuangdao, China. His research interests include single target tracking and deep learning.