Neural Computing and Applications
https://doi.org/10.1007/s00521-024-10354-4

ORIGINAL ARTICLE

Hybrid-Mode tracker with online SA-LSTM updater

Hongsheng Zheng[1] · Yun Gao[1] · Yaqing Hu[1] · Xuejie Zhang[1]

Received: 28 February 2023 / Accepted: 29 July 2024

Abstract

The backbone network and target template are pivotal factors influencing the performance of Siamese trackers. However, traditional approaches encounter challenges in eliminating local redundancy and establishing global dependencies when learning visual data representations. While convolutional neural networks (CNNs) and vision transformers (ViTs) are commonly employed as backbones in Siamese-based trackers, each primarily addresses only one of these challenges. Furthermore, tracking is a dynamic process. Nonetheless, in many Siamese trackers, solely a fixed initial template is employed to facilitate target state matching. This approach often proves inadequate for effectively handling scenes characterized by target deformation, occlusion, and fast motion. In this paper, we propose a Hybrid-Mode Siamese tracker featuring an online SA-LSTM updater. Distinct learning operators are tailored to exploit characteristics at different depth levels of the backbone, integrating convolution and transformers to form a Hybrid-Mode backbone. This backbone efficiently learns global dependencies among input tokens while minimizing redundant computations in local domains, enhancing feature richness for target tracking. The online SA-LSTM updater comprehensively integrates spatial–temporal context during tracking, producing dynamic template features with enhanced representations of target appearance. Extensive experiments across multiple benchmark datasets, including GOT-10K, LaSOT, TrackingNet, OTB-100, UAV123, and NFS, demonstrate that the proposed method achieves outstanding performance, running at 35 FPS on a single GPU.

Keywords Object tracking · Siamese network · Hybrid-Mode · Transformer · Online update

1 Introduction

Visual object tracking [1–3], a subfield of computer vision, holds significant importance across various real-world applications, thus garnering substantial attention and research efforts. In recent years, owing to advancements in downstream tasks like object recognition and detection, the field of object tracking has witnessed notable progress. Nonetheless, despite decades of development, it continues to encounter numerous challenges that necessitate resolution. These challenges include occlusion, blur, background clutter, illumination changes, scale variations, target deformation, and fast motion.

Trackers based on the Siamese architecture (SiamFC [4], SiamFC++ [5], SiamOA [6], ESiamFC [7], etc.) are prevailing methods to solve the visual tracking problem recently. Typically, the Siamese tracker comprises two primary branches: the template branch and the search branch. The template branch initializes the target state, followed by the selection of the candidate target with the highest similarity to this initial state via similarity learning in the search branch. This similarity-matching process relies on target-related features extracted from the backbone architecture employed for both input branches. Most of the current backbones use convolutional neural networks such as ResNet50 [8], VGGNet [9], and GoogLeNet [10]. In these methodologies, the convolution operation assumes a critical role in extracting feature representations for both

✉ Yun Gao
gaoyun@ynu.edu.cn

Hongsheng Zheng
hsz181811@163.com

Yaqing Hu
hahuyaqing@163.com

Xuejie Zhang
xjzhang@ynu.edu.cn

1 School of Information Science and Engineering, Yunnan University, Kunming 650504, Yunnan, China

branches through the convolution kernel. However, the convolution operation encounters limitations stemming from the size of its receptive field during the learning process, restricting its capacity to employ a fixed kernel view for localized modeling of the input image. To establish global dependencies within the feature representations, continual deepening of the depth becomes imperative, albeit this augmentation poses challenges in terms of training complexity.

Recently, transformer [11] has exhibited significant potential in computer vision (ViTs [12], Swin-T [13], and PVT [14]), overcoming the limitations of traditional convolutions by employing an attention mechanism to capture global dependencies. The transformer backbone initiates from shallow layers to model the global representation of input image patches using attention with a broader view field compared to convolutions. However, it tends to be more computationally intensive ($O(N^2C)$) than CNNs, cannot handle multiscale features [15], and demands more training data to achieve desired outcomes. References [16, 17] have highlighted that visual data exhibit certain local domain similarities in the shallow layers of the network. At this stage, utilizing attention for global feature learning from shallow layers may result in excessive redundant similarity comparisons.

On the flip side, achieving precise tracking necessitates correlating target information across different patches to model prolonged global dependencies effectively. Therefore, in pursuit of more discriminative target appearance features, we advocate for the design of distinct feature learning operators tailored to the characteristics of various depth levels within the backbone network. Specifically, CNNs and transformer exhibit complementary attributes and can be amalgamated into a novel backbone, enabling the network to concurrently acquire local representations between inputs and establish global dependencies more efficiently. This integration can be seamlessly realized through a cascading approach, forming a Hybrid-Mode network. In the shallow layers of the backbone, detailed target information is predominantly learned in the local domain solely through convolution, thereby extracting local dependencies while minimizing unnecessary inefficiencies and redundant comparisons to the fullest extent possible. As the network depth increases and the image resolution diminishes, leveraging the self-attention mechanism inherent in transformers becomes instrumental for acquiring deeper global semantic information.

On the other hand, during the tracking process, using the fixed initial template of the first frame may not accurately provide reliable information about the current target due to the deformation, occlusion, fast motion, similar interference, and other factors of the appearance of the object. In this case, it is particularly important to obtain more status information about the target through template updating. However, most current Siamese-based trackers either do not perform template updates [18–23] or use simple replacement updates [24–26] or linear updates [27].

The strategy of not updating the template can lead to tracking failures as it fails to adapt to target changes. The replacement update method, directly substituting the template with an additional frame, overlooks the learning of target features in intermediate states and disregards timing information during tracking. While the linear cumulative update approach preserves target features from previous frames to some extent, it also accumulates error information, leading to an overemphasis on recent frame content and eventual loss of access to the initial template, exacerbating tracking drift.

To address these limitations, we propose the online SA-LSTM updater, a template update method utilizing a self-attention LSTM network to temporally and spatially model historical feature sequences. This method generates an online template with a broader range of target appearances, operating independently and with minimal impact on tracking timeliness.

We integrate these advancements into a unified tracker and conduct extensive experimental validations across multiple benchmark datasets. The results underscore the superiority of our proposed method in both accuracy and speed, achieving real-time performance on a single GPU device. Figure 1 provides a qualitative comparison of our tracking method with four other prominent trackers using sequences from the OTB-100 dataset (Brid2, Diving, MotorRolling, and Skating2). Our approach consistently outperforms competing methods, delivering more precise results and effectively managing challenges such as fast motion, deformation, out-of-plane rotation, and scale changes.

The main contributions of this paper are summarized as follows:

- We introduce a novel Hybrid-Mode backbone designed for feature extraction in Siamese-based trackers (section 3.2). This backbone seamlessly integrates CNNs and transformer, offering the capability to learn global dependencies from inputs and mitigate local redundancy, all while minimizing computational costs.
- We propose an innovative template update method termed the online SA-LSTM updater (section 3.3). It employs SA-LSTM to model the spatiotemporal dynamics features over time, thereby generating an online template that more closely resembles the target when updates are necessary. The entire update process operates independently online, ensuring minimal disruption to the timeliness of tracking.

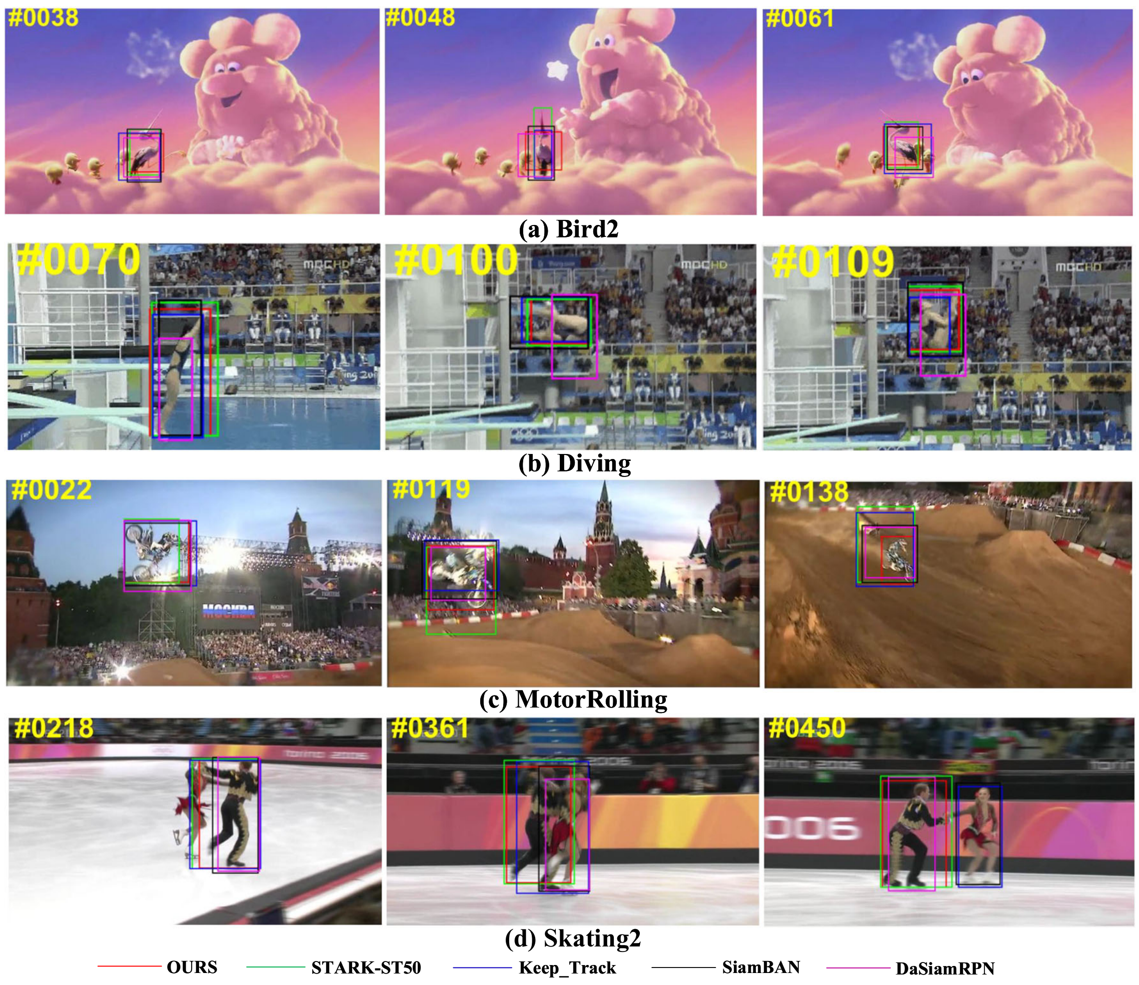

Fig. 1 Qualitative results analysis of the proposed method and four other popular trackers (STARK-ST50, Keep_Track, SiamBAN, and DaSiamRPN) on OTB-100 dataset

- Extensive experiments demonstrate that the proposed tracker can achieve competitive results on multiple benchmark datasets, including GOT-10K [28], LaSOT [29], TrackingNet [30], OTB-100 [31], UAV123 [32], and NFS [33], at a speed of 35 *FPS* on a single GPU (Sect. 4).

2 Related works

2.1 Siamese tracker

Recently, Siamese-based trackers have become increasingly popular in the tracking field due to their excellent performance. SINT [34] is one of the pioneering works focused on the Siamese structure in the tracking task. It regards the target information in the first frame as a template and learns a matching function for multiple candidate target information and templates in each subsequent frame. Then, it selects the highest score areas as the target by a matching calculation. SiamFC [4] treats tracking as a similarity learning problem. It calculates the similarity between all candidate positions in the search and template images by a full convolution and chooses the candidate with the greatest similarity to the initial target. SiamRPN [35] adds a region proposal network to Siamese trackers, which can effectively solve the situation that SiamFC cannot adapt to the change of target scale. SiamRPN++ [36] utilizes a position-balanced sampling strategy and stacks multiple region proposal network (RPN) layers to address the challenges of the target translation invariance.

SiamFC++ [5] proposes four tracking guidelines based on SiamFC. It argues that tracking should be regarded as classification and position estimation tasks. DaSimRPN [37] adds an interferer sensing system based on SiamRPN, which can effectively solve similar interferences in the tracking process. SNL-RPN [38] takes full advantage of the language description about the target and adds it to SiamRPN++ using natural language processing. ATOM

[39] introduces IoU-Net in object detection into the tracking field. ODTrack [40] presents a video tracking pipeline that employs contextual relationships among video frames in an online token propagation manner, enabling it to handle video streams of varying lengths. EVPTrack [41] leverages spatiotemporal markers for propagating information between consecutive frames, diverging from template updating. This strategy mitigates challenges regarding when to update and circumvents hyperparameters related to update policies.

2.2 Transformer tracker

Reference [11] first introduces the transformer to process the natural language. Recently, ViTs [12] and others have verified that it also has impressive capabilities in processing visual data. TransT [22] designs ego-context augment (ECA) and cross-feature augment (CFA) modules with an attention mechanism. ECA and CFA can fuse features between templates and search regions, and replace the traditional operation using cross-correlation fusion. STARK [24] proposes a transformer architecture for visual tracking. It captures the global dependencies of temporal and spatial information in video sequences by replacing the original cross-correlation fusion network with an encoder–decoder and adding a template update mechanism. TrSiam and TrDiMP [42] divide the traditional transformer into two parallel branches (encoder and decoder) that act on the Siamese tracker and DCF tracker separately. The encoder branch facilitates target templates through attention-based feature augmentation, which facilitates the generation of high-quality tracking models, while the decoder branch simplifies the target search process by propagating tracking cues from previous templates to the current frame.

UTT [43] develops a tracking framework using the transformer for feature fusion, which unifies single-object tracking and multi-object tracking tasks. Reference [44] proposes a global transformer to encode object features in all frames and use track query to group these encodings into trajectories for tracking. Although these methods proved the feature fusion processing capability of the transformer in tracking tasks, they did not explore the advantages it could bring to tracking in terms of feature extraction. CiteTracker [45] enriches target modeling and reasoning in tracking by establishing connections between image and text descriptions. Its text generation module converts image patches into descriptions that encompass their respective categories and attributes. ARTrackV2 [46] seamlessly integrates both localization and target appearance analysis into the tracking process. This method can concurrently simulate the trajectory of target motion while capturing continuous changes in target appearance.

2.3 Tracker updating

Template updating can effectively tackle the challenges of target deformation, occlusion, and rapid movement during the tracking process. UpdateNet [27] introduces a convolutional neural network to update the template. A new template is obtained for prediction in the next frame by performing convolutional modeling on the initial template, the intermediate accumulated template, and the current frame template. Reference [25] proposes using the LSTM network to update the template. In this method, the LSTM only sends control signals to the reading and writing of the template pool and does not directly operate on the features. LTMU [47] designs an offline-trained meta-learner to decide when to update the model according to the changes in the appearance, location, and classification confidence of the target. AFAT [48] uses a quality prediction network, which combines convolution and LSTM, to judge whether the current needs to be updated. The starting point is similar to LTMU and addresses the problem of when to update rather than how to update. Reference [26] uses reinforcement learning to determine whether to update the template based on the quality of the predicted results, but it only utilizes the simple replacement template. STARK [24] provides more details of the target through dynamic frames. During template updating execution, regions of the current search frame surpassing a predetermined threshold are delimited, and this delimited area substitutes the original dynamic template.

3 The proposed method

3.1 Framework of proposed approach

In this section, we present the details of our proposed tracker. As shown in Fig. 2, our tracker consists of four components, Hybrid-Mode backbone, transformer fusion module, prediction head, and online SA-LSTM updater. To garner finer-grained intermediate state information amidst target motion, we employ random sampling to obtain additional dynamic frames. Both the initial frame and additional dynamic frame constitute the shared input for the template branch, while the search area is directed into the search branch. The dynamic frame, initial frame, and search region are concurrently fed into the Hybrid-Mode backbone for feature extraction. This is described in Sect. 3.2. The features extracted from the backbone undergo concatenation before being inputted into a conventional transformer [11] fusion module, facilitating the learning of similarity between the search region and target templates. Subsequently, a prediction head is employed for

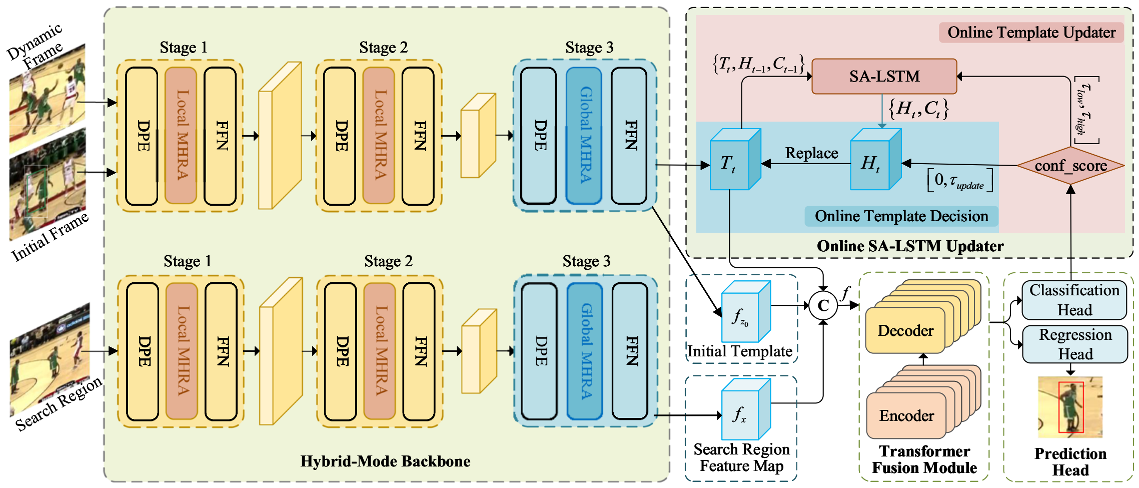

Fig. 2 Overview of the proposed framework encompasses several key components: Hybrid-Mode backbone, transformer fusion module, prediction head, and online SA-LSTM updater. In the Hybrid-Mode backbone, DPE represents dynamic position embedding, and FFN represents the feed-forward network. The local MHRA of the shallow layers is implemented by pointwise convolution (PWConv) and depthwise convolution (DWConv). The Global MHRA in the deep layer is implemented through spatiotemporal self-attention

object classification and regression. Notably, the confidence score produced by the classification head can serve as a guiding metric for both the generation and refinement processes of the online SA-LSTM template. For details of the online template updater, see Sect. 3.3.

3.2 Hybrid-Mode backbone

The feature maps extracted by the backbone are the foundation of tracking tasks. Although convolution-based neural networks (ResNet [8], AlexNet [49], etc.) have always excelled in feature extraction, it is difficult for them to obtain global dependencies in image information due to the limitation of fixed kernel fields. Most methods address this constraint by continuously stacking layers, which greatly increases the training difficulty and parameters of the network. Recently, transformer [11] has shown great potential in visual tasks. The integration of an attention mechanism facilitates the acquisition of enhanced feature representations and more profound semantic insights. Nevertheless, the visual data often exhibit localized similarities within the shallow layers of the network, thereby predisposing the system to an influx of redundant computations if employing the full attention mechanism. Concurrently, it is necessary to dynamically correlate targets across diverse regions to establish long-term dependencies for effective tracking purposes.

Convolutional operations inherently aggregate contextual information within local, small neighborhoods, thereby circumventing redundant global computations. Conversely, self-attention mechanisms naturally establish associations between distant targets by assessing global similarities. Consequently, tailored feature learning operators ought to be devised in alignment with the distinctive characteristics of various levels of the backbone network. The convolutional and self-attention can synergistically complement each other. Specifically, convolutional layers excel at capturing local similarities among input tokens, particularly within the shallower backbone layers. However, as the depth of the backbone increases, convolutional layers may struggle to capture broader similarities and global dependencies across tokens. At this time, we employ the transformer with an attention mechanism to learn the similarities among all inputs, fostering the establishment of long-range dependencies within the deeper layers. Leveraging convolution in the shallow layers effectively mitigates numerous redundant computations and reduces the output feature resolution. Consequently, the computational overhead utilizing the transformer in the deeper layers is also diminished. Hence, this well-designed architecture achieves a commendable equilibrium between accuracy and computational efficiency.

Hybrid architecture backbone networks (e.g., UniFormer [17]) have achieved good performance in other downstream tasks such as image classification, object detection, and instance segmentation.

Nevertheless, within these networks, the incorporation of numerous transformer stages escalates computational complexity, resulting in a significant reduction in the size of the output feature map. However, the smaller feature size is unsuitable for the accurate prediction of target positions in tracking tasks. In response to these challenges and the imperative to balance computational demands with feature size requirements for tracking predictions within the Siamese tracking framework, we devise a Hybrid-Mode backbone network, drawing inspiration from [17].

This backbone consists of three stages, two convolutional stages and only one transformer stage. As shown in Fig. 3, there are three modules in each stage, including dynamic position embedding (DPE), multi-head relation aggregator (MHRA), and feed-forward network (FFN). In MHRA, representation learning is performed on different tokens by designing Local MHRA and Global MHRA. Specifically, in the initial two stages, Local MHRA employs pointwise convolution (PWConv) and depthwise convolution (DWConv) to perform the representation learning of the local target while concurrently mitigating redundant comparisons. Subsequently, in the third stage, Global MHRA employs spatiotemporal self-attention to establish long-term dependencies across the input sequence.

The image pairs are input into the backbone in a triplet manner, which includes the initial frame: $z_0 \in \mathbb{R}^{3\times H_z\times W_z}$, the dynamic frame: $z_t \in \mathbb{R}^{3\times H_z\times W_z}$, and the search region: $x \in \mathbb{R}^{3\times H_x\times W_x}$, where H_z, W_z H_x, and W_x represent the height and width of the input image, respectively, and are set to 256 simultaneously. We observe that adding a random dynamic frame can provide more intermediate target status features for online template update operation. The feature maps output by the backbone are, respectively, expressed as $f_{z_0} \in \mathbb{R}^{C_1\times\frac{H_z}{16}\times\frac{W_z}{16}}$, $T_t \in \mathbb{R}^{C_1\times\frac{H_z}{16}\times\frac{W_z}{16}}$ and $f_x \in \mathbb{R}^{C_1\times\frac{H_x}{16}\times\frac{W_x}{16}}$, where C_1 is the channel dimension. As shown in formula (1), the features of the backbone output are firstly concated and flattened in the spatial dimension to obtain a new feature map, a 1×1 convolution is used to reduce its channel dimension to C_2, and the feature $f \in \mathbb{R}^{C_2\times(2\times\frac{H_z}{16}\times\frac{W_z}{16}+\frac{H_x}{16}\times\frac{W_x}{16})}$ is obtained. It can then be fed into the fusion network for subsequent processing.

$$f = Conv(cat(f_{z_0}, T_t, f_x)) \tag{1}$$

3.3 Online SA-LSTM updater

To address the problem that the initial template cannot accurately express the current target state due to the rapid movement, deformation, occlusion, and other factors. We propose a novel template update method, named online SA-LSTM updater. SA-LSTM is an LSTM network with a self-attention mechanism. The time-series characteristic of LSTM provides the probability for our updater to work in an online style. Each dynamic frame can provide more target intermediate state features for our updater. The self-attention module aggregates and enhances features extracted from the dynamic frames in the spatial dimension, augmenting the representation of T_t. These features are then incrementally learned via the LSTM network, forming an online template. This mechanism facilitates the real-time update of the object template in the temporal dimension, effectively leveraging temporal information during tracking. In the update process, the *conf_score* generated by the classification head serves as a crucial metric, indicating the reliability of the current dynamic template. In summary, the online SA-LSTM updater encompasses two key functionalities: the online template updater and the online template decision.

One is the online template updater. To make the online template reflect the real appearance of the current target, new information about the change in the target's appearance should be introduced dynamically. Therefore, we introduce dynamic frames to provide more timing information, and each dynamic frame can be randomly selected in a set stationary sampling interval. The dynamic frame is sent to the backbone for extracting features, and the feature map can be output, which is denoted as $T_t \in \mathbb{R}^{C_1\times\frac{H_z}{16}\times\frac{W_z}{16}}$.

We propose a template update criterion based on the *conf_score*, where judgments are made every n frames within the sampling interval using predefined thresholds τ. If $conf\_score > \tau_{high}$, it indicates a high degree of alignment between the current frame's tracking result and the target's initial template, suggesting no immediate need for updates. If $\tau_{low} \le conf\_score \le \tau_{high}$, it indicates that while the current results generally align with the initial template, some noticeable changes have begun, potentially leading to template offset if not promptly addressed. In such cases, T_t is forwarded to the SA-LSTM network for online template updating.

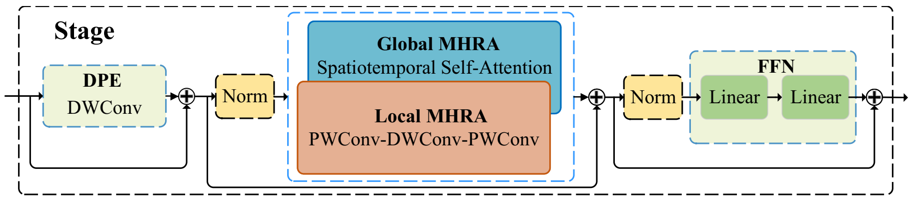

Fig. 3 Details of the stages in Hybrid-Mode backbone. There are three modules in each stage: DPE, MHRA, and FFN, which follow the stage of [17]. In the first and second stages, Local MHRA (brown) which is implemented by point-by-point convolution (PWConv) and deep convolution (DWConv) is used to learn the local representations of the target and eliminate redundant comparisons. In the third stage, Global MHRA (blue) which is implemented by spatiotemporal self-attention is used to build long-term dependencies

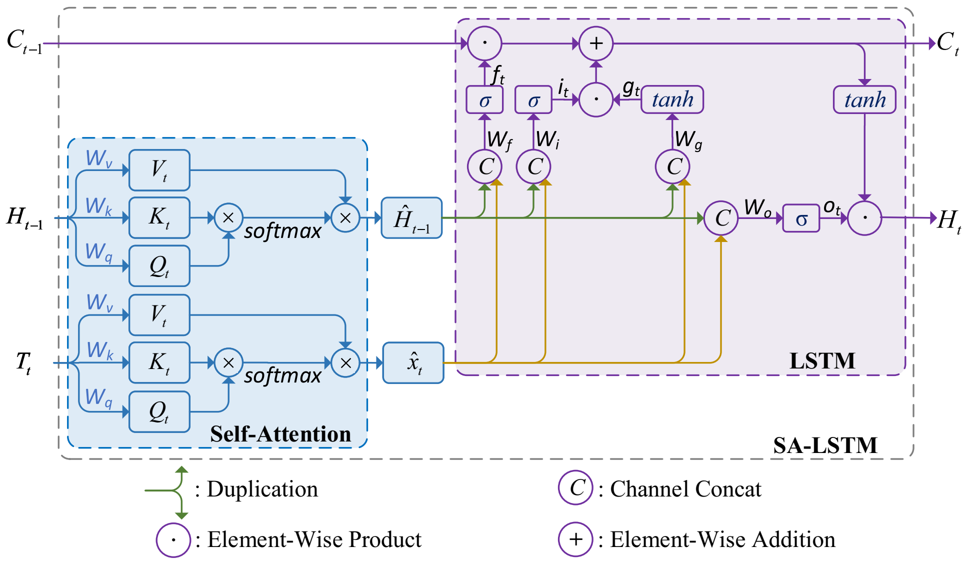

Fig. 4 Architecture of SA-LSTM, consisting of self-attention module and LSTM in a cascaded manner. Self-attention can enhance the aggregation of feature sequences in space and then hand it over to LSTM to process the temporal information of features

As depicted in Fig. 4, the SA-LSTM module takes three inputs, T_t, H_{t-1}, C_{t-1}, and produces two outputs, H_t, C_t. Here H_{t-1} and C_{t-1} denote the output and cell state of the module at the previous time step, while H_t and C_t represent the output and cell state at the current time step, respectively. Further elucidation is provided below. H_t signifies the template refined and augmented by SA-LSTM, encapsulating reliable cues regarding the target's historical temporal and spatial dynamics.

The SA-LSTM network is designed to capture long-distance dependencies across a broader spatial and temporal range. We integrate the self-attention module into the original LSTM network in a cascaded manner to construct the SA-LSTM network. At each time step, the self-attention module selectively aggregates input features from all locations by computing weighted sums and outputs the features $\hat{x}_t$ and $\hat{H}_{t-1}$. These features are then fed into the LSTM network for temporal information processing. The LSTM network depicted in Fig. 4 comprises three gating mechanisms: the forget gate f_t, input gate i_t, and output gate o_t, regulating the cell state and output of the network unit.

Firstly, the forget gate f_t, activated by a sigmoid layer, determines which information in $\hat{x}_t$ and $\hat{H}_{t-1}$ is retained, effectively filtering out unimportant features. Subsequently, the input gate i_t, also activated by a sigmoid layer, identifies the feature information requiring updates. A new candidate value g_t, generated by the tanh layer, is introduced to the current network as a potential addition to the cell state. The old cell state C_{t-1} is then updated by multiplying it with f_t to discard obsolete information and adding $i_t \circ g_t$ as the new candidate value, adapting according to the decision to update each state.

Finally, the model's output is determined. The output gate o_t, processed through a sigmoid layer, provides the initial output, which is then scaled between -1 and 1 using a tanh layer. A dot product operation is performed between the scaled C_t and the initial output, yielding the historical accumulated feature H_t. Given the structural attributes of LSTM, only the output H_t, C_t needs to be stored at each time step. These operations are represented in formulas (2)-(8), where SA denotes the self-attention module, *cat* denotes channel concatenation, and b_f, b_i, b_c, and b_o represent biases.

$$\hat{x}_t = SA(T_t), \hat{H}_{t-1} = SA(H_{t-1}) \tag{2}$$

$$f_t = \sigma(W_f * cat(\hat{x}_t + \hat{H}_{t-1}) + b_f) \tag{3}$$

$$i_t = \sigma(W_i * cat(\hat{x}_t + \hat{H}_{t-1}) + b_i) \tag{4}$$

$$g_t = tanh(W_g * cat(\hat{x}_t + \hat{H}_{t-1}) + b_c) \tag{5}$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot g_t \tag{6}$$

$$o_t = \sigma(W_o * cat(\hat{x}_t + \hat{H}_{t-1}) + b_o) \tag{7}$$

$$H_t = o_t \cdot tanh(C_t) \tag{8}$$

The other is the online template decision. T_t and H_t together make up the online template. In a sampling interval, if $conf\_scoregt\tau_{update}$, then dynamic frame features can reflect the current state of the target, and T_t is used as the online template.

If consecutive N frames exhibit $conf\_score < \tau_{update}$, it signifies that neither the initial template nor the dynamic frame feature adequately represents the current state of the target. Consequently, the feature H_t, accumulated from historical information, is utilized as the online template in such instances.

3.4 Training loss

In the first stage of training, the regression head loss function combines L_1 and *DIoU* [50], and the classification head uses binary cross entropy *(BCE)* as the loss function. The update module training uses the linear weighting of L_1 and L_2 as the loss function. This is shown in formulas (9)–(11):

$$L_{diou} = 1 - IOU + \frac{\rho^2(b_i, b_i{'})}{c^2} \tag{9}$$

$$L_{reg} = \lambda_{diou} L_{diou}(b_i, b_i{'}) + \lambda_{L_1} L_1(b_i, b_i{'}) \tag{10}$$

$$L_{update} = \lambda_{L_1} L_1 + \lambda_{L_2} L_2 \tag{11}$$

where λ_{diou}, λ_{L_1}, and λ_{L_2} are hyperparameters, IOU is the intersection over union, and b_i and b_i' represent the ground-truth bounding box and the predicted box, respectively. $\rho(\cdot)$ is the Euclidean distance, and c represents the diagonal length of the minimum outer matrix of b_i,b_i'. Compared with GIoU [51], DIoU considers both the overlapping area and the center distance between b_i and b_i', and the convergence faster in the case where b_i waps b_i'.

4 Experiments

4.1 Implementation details

Model Settings We train the Hybrid-Mode backbone based on the pre-trained result of UniFormer-small [17], the channel dimension C_1 varies from 64 to 128 to 320, and C_2 is set to 256. The MHA of each encoder and decoder layer in the transformer fusion module uses 8 heads, and the channel dimension is set to 256, while the hidden dimension of the FFN is equal to 2048. The regression head is a fully convolutional network composed of five stacked Conv-BN-ReLU layers, and the classification head is a three-layer perceptron with 256 hidden units in each layer. To keep the output of the SA-LSTM module and the original input features in the same size, the attention hidden layer dimension, LSTM hidden layer dimension, and output layer dimension are all set to 320. The SA-LSTM is organized into a three-layer architecture with the addition of LayerNorm [52].

Training details Our tracker is trained on the server with a single NVIDIA-RTX3090 GPU with Python 3.8. A total of 600 epochs are trained in two steps. The former step requires 500 epochs to train the first three components. The latter only trains the online SA-LSTM updater and freezes the parameters of the first three parts. Before the image pair is input to the backbone, the size of the search region and the template branch is processed into [320×320] and [128×128], respectively. In the first step, AdamW is used as the optimizer, and the weight decay rate and learning rate are both set to 10^{-4}, the weights λ_{L_1} and λ_{diou} of the loss function are 5 and 2, respectively, and the mini-batch is 16. The learning rate drops by a factor of 10 after the 400th epoch. ADAM is used as the optimizer in the second step, the learning rate is set to 10^{-3}, λ_{L_1} and λ_{L_2} is both 1, and the mini-batch is 32.

Testing details In the tracking stage, the sampling interval of dynamic frames is set to 200, and n of the judgment interval update is set to 20. Meanwhile, the thresholds N and τ_{update} for judging whether to update are set to 3 and 0.3, and the thresholds τ_{low} and τ_{high} for controlling the addition of online template generation and update are 0.5 and 0.7, respectively.

4.2 Results and comparisons

To verify the effectiveness of the proposed method, we evaluate our tracker on multiple benchmark datasets GOT-10K [28], LaSOT [29], TrackingNet [30], OTB-100 [31], NFS [33], and UAV123 [32], and compared with current state-of-the-art results.

GOT-10K. GOT-10K [28] is a large-scale dataset for short-term object tracking, covering common real-world target moving scenarios. It provides over 10,000 video sequences for training and 180 sequences for testing. We evaluate the performance according to the protocol proposed in [28]. The results are shown in Table 1, where our tracker achieves an AO of 69.0%, $SR_{0.5}$ of 78.3%, and $SR_{0.75}$ of 64.3%. Compared to STARK-ST50 using ResNet50 as the backbone and using updating with a simple replacement strategy, they increased by 1.0%, 0.6%, and 2.0%, respectively.

LaSOT. LaSOT [29] is a large-scale high-quality single-object tracking benchmark dataset. It contains 1400 video sequences, of which 1120 are used for training. The average number of frames per video sequence is over 2500 frames, which makes it more reflective of the actual performance of the tracker. We test the tracker on its 280 test sequences and obtain 67.3% AUC, 77.1% N.P, and 72.3% Pre. Compared with DyTrack-Medi [53] and STARK-ST50 [24], gain effects of 0.8% and 0.8% are obtained in AUC, respectively. The results are shown in Table 2 and Fig. 5.

The challenge attribute-based evaluation results of the state-of-the-art six trackers on LaSOT are compared in Fig. 6. Our method outperforms the others on several properties, especially in partial occlusion, deformation, motion blur, rotation, scale variation, and out-of-view.

Table 1 Comparison of results on GOT-10K

	Ours	STARK-ST50 [24]	TransT [22]	TrDiMP [42]	DyTrack-Medi [53]	TrSiam [42]	Siam-RCNN [54]	DCFST [55]	KYS [56]	PrDiMP [57]	Ocean [58]
AO (%)	**69.0**	*68.0*	***67.1***	67.1	66.2	66.0	64.9	63.8	63.6	63.4	61.1
$SR_{0.5}$(%)	**78.3**	*77.7*	***76.8***	77.7	75.2	76.6	72.8	75.3	75.1	73.8	72.1
$SR_{0.75}$(%)	**64.3**	*62.3*	***60.9***	58.3	59.4	57.1	59.7	49.8	51.5	54.3	47.3

(Bold: Top, Italic: Second, and Bold italic: Third)

Table 2 Comparison of results on LaSOT

	Ours	DyTrack-Medi [53]	STARK-ST50 [24]	TransT [22]	Siam-RCNN [54]	TrDiMP [42]	TrSiam [42]	PrDiMP [57]	Auto-Match [59]	DiMP [60]	Ocean [58]
AUC (%)	**67.3**	*66.5*	***66.4***	64.9	64.8	63.9	62.9	59.9	58.2	57.9	52.6
N.Prec. (%)	**77.1**	***75.5***	*76.3*	73.8	72.2	–	–	68.8	–	65.0	–
Prec. (%)	**72.3**	***70.4***	*71.2*	69.0	68.4	66.2	65.0	60.8	59.9	57.7	52.6

(Bold: Top, Italic: Second, and Bold italic: Third)

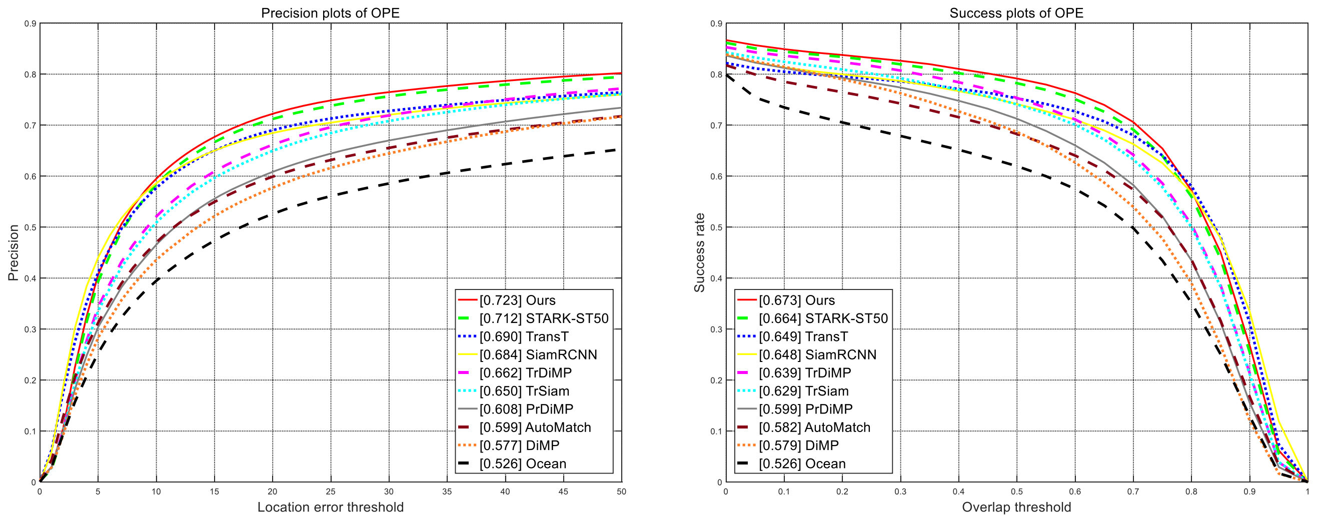

Fig. 5 Comparison of precision and success plots on LaSOT

TrackingNet. TrackingNet [30] is a large-scale dataset specifically designed for object tracking tasks that contain more than 30,000 videos and more than 14 million annotated boxes. We evaluate our tracker on 511 test sequences provided by TrackingNet, and the results are listed in Table 3. From the table, we can see that our tracker achieves better performance than the others, with AUC achieving 81.9% state-of-the-art results, which are 0.6% and 0.5% higher than those of STARK-ST50 and TransT, respectively.

OTB-100. OTB-100 [31] is a popular tracking benchmark consisting of 100 video sequences. We compare the tracking results with 10 current state-of-the-art trackers: KYS [56], DiMP [60], ATOM [39], SiamR-CNN [54], SiamBAN [19], SiamRPN++ [36], PrDiMP [57], STARK-ST50 [24], SiamFC++ [5], and CIResNet22 [61].

Figure 7 shows the precision and success plots of these methods on the OTB-100. It can be seen that ours achieves very competitive results, where the AUC achieves the best result of 71.0%. When compared with STARK-ST50 and SiamR-CNN, 2.5% and 1% gains were obtained,

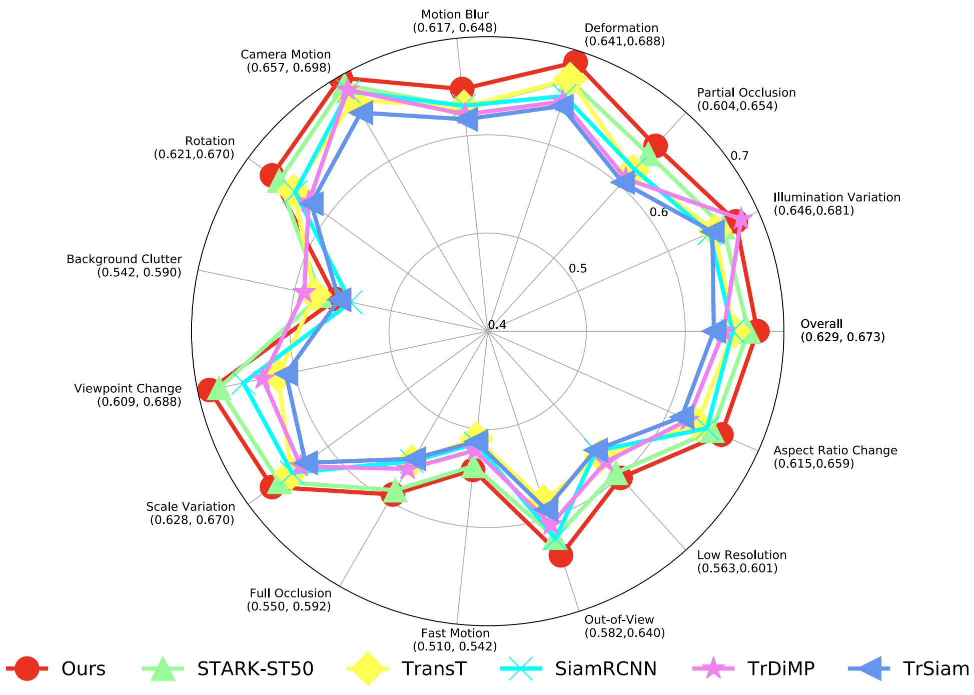

Fig. 6 AUC scores of all attributes on LaSOT dataset

Table 3 Comparison of results on TrackingNet

	Ours	TransT [22]	STARK-ST50 [24]	Siam-RCNN [54]	DyTrack-Medi [53]	TrDiMP [42]	TrSiam [42]	PrDiMP [57]	MAML [62]	SiamFC++ [5]	DCFST [55]
AUC (%)	**81.9**	*81.4*	***81.3***	81.2	80.9	78.4	78.1	75.8	75.7	75.4	75.2
N.Prec. (%)	**86.8**	*86.7*	***86.1***	85.4	85.5	83.3	82.9	81.6	82.2	80.0	80.0
Prec. (%)	*80.0*	**80.3**	–	*80.0*	***77.8***	73.1	72.7	70.4	72.5	70.5	70.0

(Bold: Top, Italic: Second, and Bold italic: Third)

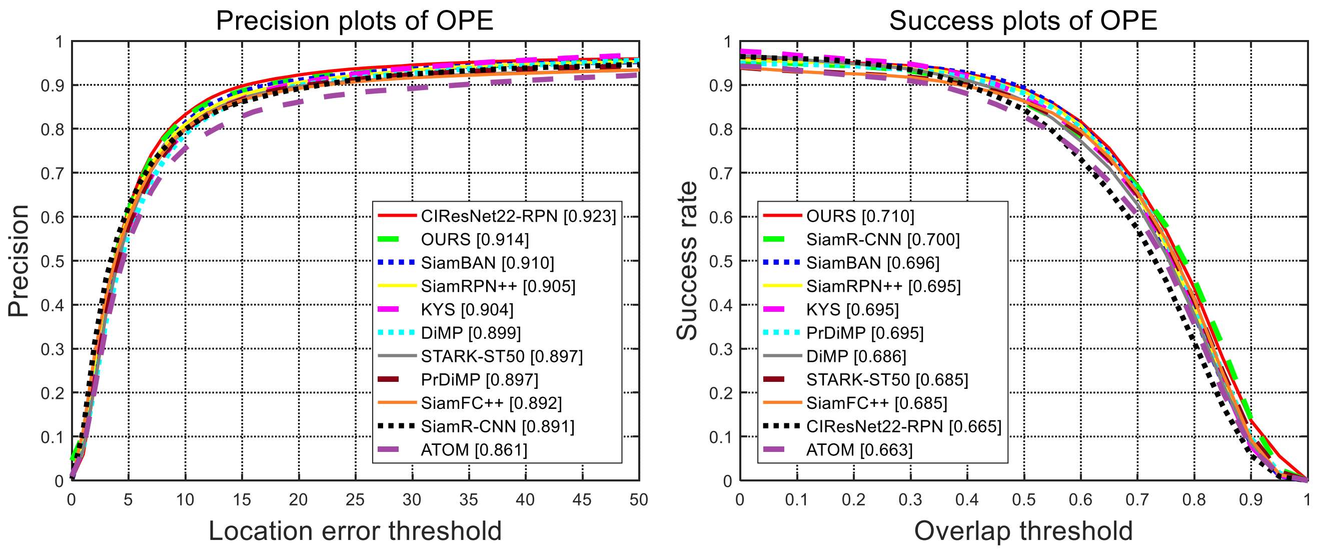

Fig. 7 Comparison of precision and success results on OTB-100

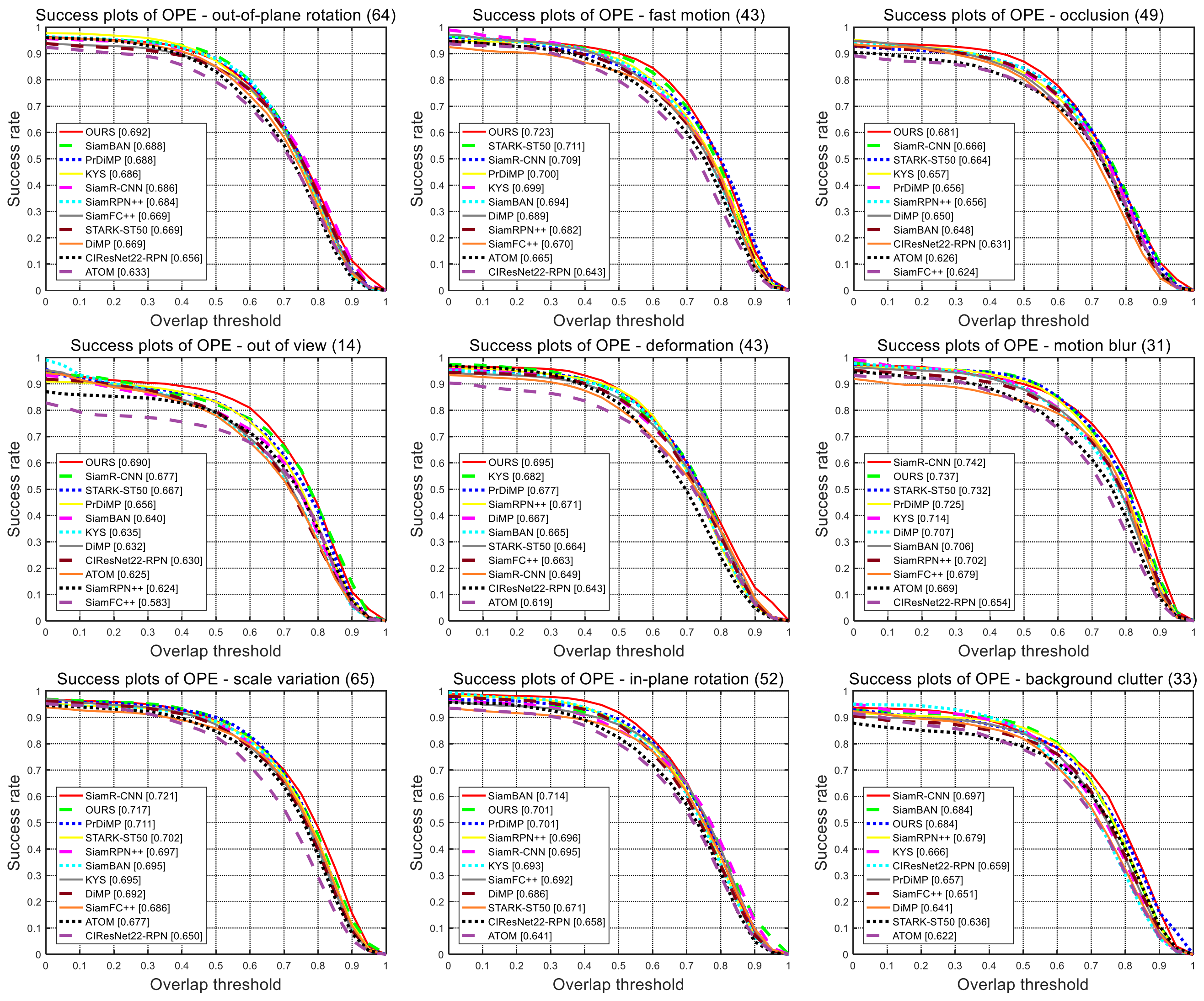

Fig. 8 Success plots for nine main challenge attributes on OTB-100 dataset

Table 4 Comparison of AUC (%) scores on NFS and UAV123 datasets

	Ours	TrDiMP [42]	Keep-Track [63]	TrSiam [42]	TransT [22]	Siam-RCNN [54]	SiamRPN++ [36]	PrDiMP [57]	SiamBAN [19]	KYS [56]
NFS	**66.7**	*66.5*	***66.4***	65.8	65.7	63.9	50.2	63.5	59.4	63.5
UAV123	*69.6*	67.5	**69.7**	67.4	***69.1***	64.9	61.3	68.0	63.1	–

(Bold: Top, Italic: Second, and Bold italic: Third)

respectively. In terms of precision, our tracker also outperforms most others reaching 91.4%.

To further explore the ability of the tracker to deal with various challenging scenarios, we also compared the performance of the above trackers in different challenging scenarios, and the results are shown in Fig. 8. The figure shows that our tracker achieves the best performance under multiple challenges, including out-of-plane rotation, fast motion, occlusion, out-of-view, and deformation, and it is equally competitive in other challenges.

NFS. NFS [33] dataset also has 100 video sequences, and each sequence contains fast-moving objects, which better reflects the importance of template updating. From Table 4, we can see that our tracker outperforms other

Table 5 Ablation experiments with different types of backbones on OTB-100 and GOT-10K

Backbone	OTB-100		GOT-10K			Params(M↓)
	AUC	Pre	AO	$SR_{0.5}$	$SR_{0.75}$	
ResNet50	***68.3***	88.9	***67.2***	76.1	61.2	**23.3**
Swin-T	*68.5*	89.4	*67.4*	76.9	62.4	***35.6***
Hybrid-Mode	**69.0**	90.3	**68.0**	77.2	62.5	*26.9*

(Bold: Top, Italic: Second, and Bold italic: Third)

current methods, achieving a state-of-the-art result of 66.7% in AUC.

UVA123. UAV123 [32] is a video dataset collected by drones, with a total of 123 sequences. The results of each method are compared in Table 4. Our tracker achieves the current second level with an AUC value of 69.6%.

4.3 Ablation study

In this section, we undertake a detailed exploration of the distinct roles of each proposed component, accompanied by an ablation analysis conducted on OTB-100 [31] and GOT-10K [28].

First, we conduct a comparative analysis involving three distinct backbone architectures for object tracking: ResNet50 [8] with pure convolution, Swin Transformer [13] with pure transformer, and our Hybrid-Mode network. In this experiment, we choose STARK-S with ResNet50 as the baseline, the 1st row in Table 5. Notably, solely the backbone is adjusted across all trackers to ensure experimental rigor and reliability; no template update mechanism is added.

It can be seen that when replacing the convolutional backbone with Swin-T in the 2nd row, the performance is improved to some extent on both datasets, where AUC increased by 0.2% on OTB-100 and AO also increased by 0.2% on GOT-10K. The attention mechanism facilitates

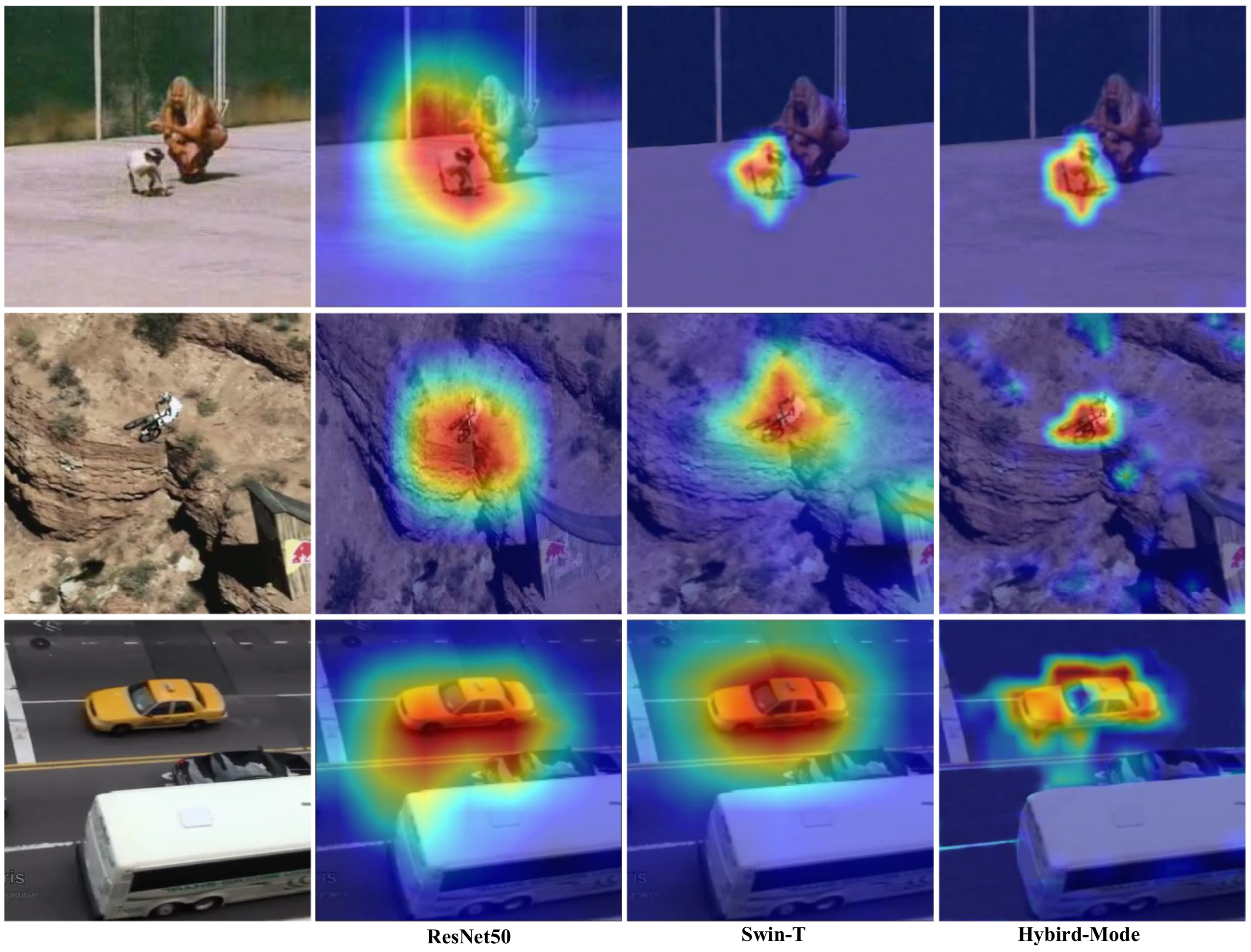

Fig. 9 Attention visualization of different backbones

Table 6 Ablation experiments of different update modules on OTB-100 and GOT-10K

Tracker	OTB-100		GOT-10K		
	AUC	Pre	AO	$SR_{0.5}$	$SR_{0.75}$
Hybrid-Mode + no updater	69.0	90.3	68.0	77.2	62.5
Hybrid-Mode + replace updater	***69.4***	90.7	***68.2***	77.5	63.2
Hybrid-Mode + UpdateNet	*70.1*	91.1	*68.5*	77.9	63.7
Hybrid-Mode + SA-LSTM	**71.0**	91.4	**69.0**	78.3	64.3

(Bold: Top, Italic: Second, and Bold italic: Third)

the generation of more compact feature representations and deeper semantic information during feature extraction. However, the inherent mechanism of Swin-T entails attention computation across the global range of input tokens, leading to a proliferation of redundant comparisons and consequently a substantial increase in model parameters (+12.3M). This phenomenon inevitably impacts tracker speed adversely.

We observe a further enhancement in the overall network performance for our proposed Hybrid-Mode backbone, in the 3^{rd} row. Compared to ResNet50, where AUC on OTB-100 improved by 0.7% and AO on GOT-10K improved by 0.8%, the increase in the number of model parameters was only marginal (+3.6M). Additionally, compared to the Swin-T backbone, AUC on OTB-100 improved by 0.5% and AO on GOT-10K improved by 0.6%, while the model parameters were significantly reduced (− 8.7M).

This is mainly because the shallow layers of the backbone learn to model the input tokens only in the local domain by convolution, avoiding a large number of inefficient computations when acquiring primary features about the target.

On the other hand, in the deeper layer of the backbone, spatiotemporal self-attention overcomes the limitation of traditional self-attention, which tends to focus solely on inter-tokens queries within the neighboring range. Spatiotemporal self-attention exhibits the capability to establish long-term relationships across the feature space, enabling it to learn more intricate semantic information and produce more discriminative feature representations. Furthermore, since the entire backbone can accommodate variations in input size, it enables the attainment of a favorable balance between accuracy and computational efficiency. We visualize the feature attention maps extracted from three backbones in Fig. 9, where the Hybrid-Mode approach demonstrates superior focusing ability and better localization of the target position.

In the second part of our analysis, we delve deeper into the performance of the proposed online SA-LSTM updater.

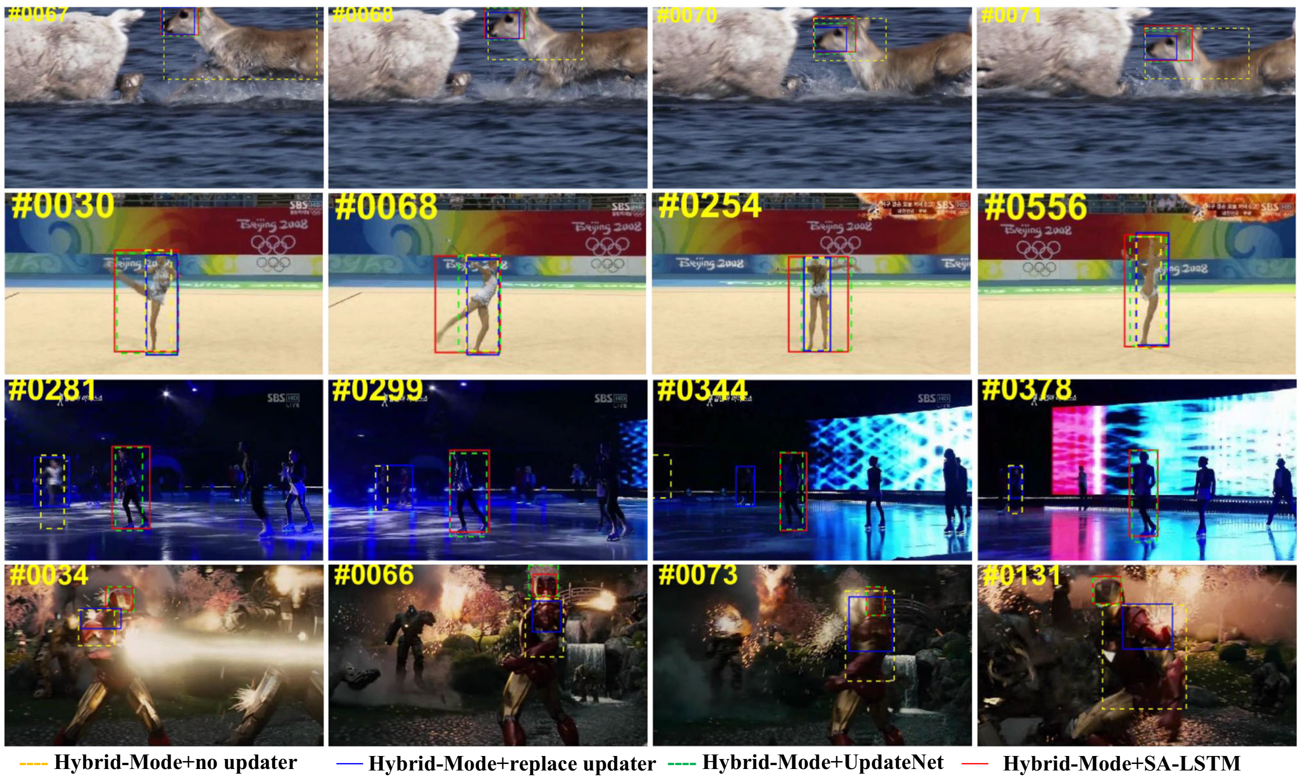

Fig. 10 Visualization of different update strategies in fast motion, deformation, and other scenarios

We compare four template update mechanisms based on the Hybrid-Mode backbone: no updater, replace updater, UpdateNet [27], and SA-LSTM. The results are summarized in Table 6. We establish a baseline using the Hybrid-Mode backbone without any template update mechanism, denoted in the 1st row. Subsequently, we augment the baseline with the replace updater mechanism, as shown in the 2nd row. A slight improvement over the baseline is observed, with gains of 0.4% and 0.2% on AUC for OTB-100 and AO for GOT-10K, respectively. To further evaluate the efficacy of the proposed online SA-LSTM updater, we integrate UpdateNet into the baseline for comparison, as presented in the 3rd row. Compared to the baseline, we observe an increase in AUC by 1.1% on OTB-100 and an increase in AO by 0.5% on GOT-10K. Compared to the replacement updater, AUC increased by 0.7% and AO increased by 0.3%.

The performance enhancement is achieved when incorporating the proposed Hybrid-Mode backbone and online SA-LSTM updater, as depicted in the 4th row. Here, we observe increases of 2.0% and 1.0% in AUC for OTB-100 and AO for GOT-10K, respectively, compared to the baseline. Compared to the replace updater, gains of 1.6% and 0.8% are obtained. Moreover, compared to UpdateNet, improvements of 0.9% and 0.5% are observed, respectively. These results indicate that the SA-LSTM Updater can effectively handle historical information in the tracking process, modeling the historical sequence in spatiotemporal aspects to capture apparent changes in the target, thus enhancing robustness.

For a more comprehensive understanding of the performance of different update strategies, Fig. 10 visualizes their qualitative analysis results. Our SA-LSTM updater outperforms other update methods, demonstrating superior predictive ability, especially in scenarios involving fast motion and deformation. When combined with the Hybrid-Mode backbone, the SA-LSTM updater yields more accurate predictions of target location.

5 Conclusion

In this paper, we propose an innovative Siamese tracking framework. Initially, we argue that distinct feature learning operators should be designed for distinct hierarchical properties of the backbone. Thus, we develop a Hybrid-Mode backbone, which organically integrates convolutional and transformer components. This backbone is thereby capable of extracting highly discriminative target appearance features while concurrently minimizing computational expenditure. Furthermore, conventional Siamese-based trackers often rely on either a fixed template or a simplistic linear update template throughout the tracking process to characterize the state of the target object. To address this limitation, we present an extensible online SA-LSTM updater. This updater effectively leverages historical information to model target features both spatially and temporally. Extensive experimental evaluations demonstrate the superior performance of the proposed method across multiple tracking benchmark datasets. Notably, our approach excels in mitigating challenges such as out-of-plane rotation, rapid motion, occlusion, out-of-view scenarios, and deformation, etc.

Acknowledgements This work was supported by the National Natural Science Foundation of China (No. 61802337 and No. 62266051).

Data availability The OTB-100 dataset analyzed during the current study is available in http://cvlab.hanyang.ac.kr/tracker benchmark; the GOT-10K dataset is available in http://got-10k.aitestunion.com/; the LaSOT dataset is available in https://cis.temple.edu/lasot/; the TrackingNet dataset is available in https://tracking-net.org/; the UAV123 dataset is available in https://cemse.kaust.edu.sa/ivul/uav123; and the NFS dataset is available in http://ci2cv.net/nfs/index.html.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Javed S, Danelljan M, Khan FS, Khan MH, Felsberg M, Matas J (2023) Visual object tracking with discriminative filters and siamese networks: a survey and outlook. IEEE Trans Pattern Anal Mach Intell 45(5):6552–6574. https://doi.org/10.1109/TPAMI.2022.3212594
2. Zhang L, Fan H (2023) Visual object tracking: progress, challenge, and future. Innovation. https://doi.org/10.1016/j.xinn.2023.100402
3. Kugarajeevan J, Kokul T, Ramanan A, Fernando S (2023) Transformers in single object tracking: An experimental survey. IEEE Access
4. Bertinetto L, Valmadre J, Henriques JF, Vedaldi A, Torr PHS (2016) Fully-convolutional siamese networks for object tracking **9914 LNCS**, 850–865. arXiv:1606.09549. https://doi.org/10.1007/978-3-319-48881-3_56
5. Xu Y, Wang Z, Li Z, Yuan Y, Yu G (2020) Siamfc++: towards robust and accurate visual tracking with target estimation guidelines. Proc. AAAI Conf Artif Intell 34:12549–12556
6. Zhang J, Xie X, Zheng Z, Kuang L-D, Zhang Y (2022) Siamoa: siamese offset-aware object tracking. Neural Comput Appl 34(24):22223–22239
7. Huang H, Liu G, Zhang Y, Xiong R, Zhang S (2022) Ensemble siamese networks for object tracking. Neural Comput Appl 34(10):8173–8191
8. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 770–778

9. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556
10. Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1–9
11. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez A.N, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Advances in Neural Information Processing Systems **30**
12. Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, Dehghani M, Minderer M, Heigold G, Gelly S et al (2020) An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929
13. Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, Lin S, Guo B (2021) Swin transformer: Hierarchical vision transformer using shifted windows. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 10012–10022
14. Wang W, Xie E, Li X, Fan D.-P, Song K, Liang D, Lu T, Luo P, Shao L (2021) Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 568–578
15. Guo J, Han K, Wu H, Tang Y, Chen X, Wang Y, Xu C (2022) Cmt: Convolutional neural networks meet vision transformers. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 12175–12185
16. Raghu M, Unterthiner T, Kornblith S, Zhang C, Dosovitskiy A (2021) Do vision transformers see like convolutional neural networks? Adv Neural Inf Process Syst 34:12116–12128
17. Li K, Wang Y, Zhang J, Gao P, Song G, Liu Y, Li H, Qiao Y (2022) Uniformer: Unifying convolution and self-attention for visual recognition. arXiv preprint arXiv:2201.09450
18. Chen Z, Zhong B, Li G, Zhang S, Ji R, Tang Z, Li X (2022) Siamban: Target-aware tracking with siamese box adaptive network. IEEE Trans. Pattern Anal. Mach. Intell. 1–17. https://doi.org/10.1109/TPAMI.2022.3195759
19. Chen Z, Zhong B, Li G, Zhang S, Ji R (2020) Siamese box adaptive network for visual tracking. In: Proceedings of the ieee conference on computer vision and pattern recognition, pp 6668–6677
20. Guo D, Wang J, Cui Y, Wang Z, Chen S (2020) Siamcar: Siamese fully convolutional classification and regression for visual tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6269–6277 (2020)
21. Lin L, Fan H, Xu Y, Ling H (2021) Swintrack: A simple and strong baseline for transformer tracking. arXiv preprint arXiv:2112.00995
22. Chen X, Yan B, Zhu J, Wang D, Yang X, Lu H (2021) Transformer tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8126–8135
23. Peng J, Jiang Z, Gu Y, Wu Y, Wang Y, Tai Y, Wang C, Lin W (2021) Siamrcr: Reciprocal classification and regression for visual object tracking. arXiv preprint arXiv:2105.11237
24. Yan B, Peng H, Fu J, Wang D, Lu H (2021) Learning spatio-temporal transformer for visual tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 10448–10457
25. Yang T, Chan AB (2018) Learning dynamic memory networks for object tracking. In: Proceedings of the European conference on computer vision (ECCV), pp 152–167
26. Sun M, Xiao J, Lim E.G, Zhang B, Zhao Y (2020) Fast template matching and update for video object tracking and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 10791–10799
27. Zhang L, Gonzalez-Garcia A, Weijer Jvd, Danelljan M, Khan FS (2019) Learning the model update for siamese trackers. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4010–4019
28. Huang L, Zhao X, Huang K (2019) Got-10k: A large high-diversity benchmark for generic object tracking in the wild. IEEE Trans Pattern Anal Mach Intell 43(5):1562–1577
29. Fan H, Lin L, Yang F, Chu P, Deng G, Yu S, Bai H, Xu Y, Liao C, Ling H (2019) Lasot: A high-quality benchmark for large-scale single object tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5374–5383
30. Muller M, Bibi A, Giancola S, Alsubaihi S, Ghanem B (2018) Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In: Proceedings of the European conference on computer vision (ECCV), pp 300–317
31. Wu Y, Lim J, Yang M-H (2013) Online object tracking: A benchmark. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2411–2418
32. Mueller M, Smith N, Ghanem B (2016) A benchmark and simulator for uav tracking. In: Proceedings of the European conference on computer vision (ECCV), pp 445–461. Springer
33. Kiani Galoogahi H, Fagg A, Huang C, Ramanan D, Lucey S (2017) Need for speed: A benchmark for higher frame rate object tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1125–1134
34. Tao R, Gavves E, Smeulders AW (2016) Siamese instance search for tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1420–1429
35. Li B, Yan J, Wu W, Zhu Z, Hu X (2018) High performance visual tracking with siamese region proposal network. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8971–8980
36. Li B, Wu W, Wang Q, Zhang F, Xing J, Yan J (2019) SIAMRPN++: Evolution of siamese visual tracking with very deep networks. Proceedings of the IEEE conference on computer vision and pattern recognition 2019-June, 4277–4286. arXiv:1812.11703. https://doi.org/10.1109/CVPR.2019.00441
37. Zhu Z, Wang Q, Li B, Wu W, Yan J, Hu W (2018) Distractor-aware siamese networks for visual object tracking. In: Proceedings of the European conference on computer vision (ECCV), pp 101–117
38. Feng Q, Ablavsky V, Bai Q, Sclaroff S (2021) Siamese natural language tracker: Tracking by natural language descriptions with siamese trackers. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5851–5860
39. Danelljan M, Bhat G, Khan FS, Felsberg M (2019) Atom: Accurate tracking by overlap maximization. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4660–4669
40. Zheng Y, Zhong B, Liang Q, Mo Z, Zhang S, Li X (2024) Odtrack: Online dense temporal token learning for visual tracking. In: AAAI
41. Shi L, Zhong B, Liang Q, Li N, Zhang S, Li X (2024) Explicit visual prompts for visual object tracking. In: AAAI
42. Wang N, Zhou W, Wang J, Li H (2021) Transformer meets tracker: Exploiting temporal context for robust visual tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 1571–1580
43. Ma F, Shou MZ, Zhu L, Fan H, Xu Y, Yang Y, Yan Z (2022) Unified transformer tracker for object tracking. In: Proceedings of the IEEE conference on computer vision and pattern Recognition, pp 8781–8790
44. Zhou X, Yin T, Koltun V, Krähenbühl P (2022) Global tracking transformers. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8771–8780

45. Li X, Huang Y, He Z, Wang Y, Lu H, Yang, M-H (2023) Citetracker: Correlating image and text for visual tracking. In: Proceedings of the IEEE/CVF international conference on computer vision (ICCV), pp 9974–9983
46. Bai Y, Zhao Z, Gong Y, Wei X (2024) Artrackv2: Prompting autoregressive tracker where to look and how to describe. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)
47. Dai K, Zhang Y, Wang D, Li J, Lu H, Yang X (2020) High-performance long-term tracking with meta-updater. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6298–6307
48. Xu T, Feng Z-H, Wu X-J, Kittler J (2020) Afat: adaptive failure-aware tracker for robust visual object tracking. arXiv preprint arXiv:2005.13708
49. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. Commun ACM 60(6):84–90
50. Zheng Z, Wang P, Liu W, Li J, Ye R, Ren D (2020) Distance-iou loss: Faster and better learning for bounding box regression. Proc. AAAI Conf Artif Intell 34:12993–13000
51. Rezatofighi H, Tsoi N, Gwak J, Sadeghian A, Reid I, Savarese S (2019) Generalized intersection over union: a metric and a loss for bounding box regression. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 658–666
52. Ba JL, Kiros JR, Hinton GE (2016) Layer normalization. arXiv preprint arXiv:1607.06450
53. Zhu J, Chen X, Diao H, Li S, He J-Y, Li C, Luo B, Wang D, Lu H (2024) Exploring dynamic transformer for efficient object tracking. arXiv preprint arXiv:2403.17651
54. Voigtlaender P, Luiten J, Torr PH, Leibe B (2020) Siam r-cnn: Visual tracking by re-detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6578–6588
55. Zheng L, Tang M, Chen Y, Wang J, Lu H (2020) Learning feature embeddings for discriminant model based tracking. In: Proceedings of the European conference on computer vision (ECCV), pp 759–775. Springer
56. Bhat G, Danelljan M, Gool LV, Timofte R (2020) Know your surroundings: Exploiting scene information for object tracking. In: Proceedings of the European conference on computer vision (ECCV), pp 205–221. Springer
57. Danelljan M, Van Gool L, Timofte R (2020) Probabilistic regression for visual tracking. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., 7181–7190. arXiv:2003.12565. https://doi.org/10.1109/CVPR42600.2020.00721
58. Zhang Z, Peng H, Fu J, Li B, Hu W (2020) Ocean: Object-aware anchor-free tracking. In: Proceedings of the European conference on computer vision (ECCV), pp 771–787. Springer
59. Zhang Z, Liu Y, Wang X, Li B, Hu W (2021) Learn to match: Automatic matching network design for visual tracking. In: Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), pp 13339–13348
60. Bhat G, Danelljan M, Gool LV, Timofte R (2019) Learning discriminative model prediction for tracking. In: Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), pp 6182–6191
61. Zhang Z, Peng H (2019) Deeper and wider siamese networks for real-time visual tracking. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4591–4600
62. Wang G, Luo C, Sun X, Xiong Z, Zeng W (2020) Tracking by instance detection: A meta-learning approach. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 6288–6297
63. Mayer C, Danelljan M, Paudel DP, Van Gool L (2021) Learning target candidate association to keep track of what not to track. In: Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV), pp 13444–13454