Robust Object Modeling for Visual Tracking

Yidong Cai   Jie Liu*   Jie Tang   Gangshan Wu
State Key Laboratory for Novel Software Technology, Nanjing University, China
yidong_cai@smail.nju.edu.cn; {liujie, tangjie, gswu}@nju.edu.cn

Abstract

Object modeling has become a core part of recent tracking frameworks. Current popular tackers use Transformer attention to extract the template feature separately or interactively with the search region. However, separate template learning lacks communication between the template and search regions, which brings difficulty in extracting discriminative target-oriented features. On the other hand, interactive template learning produces hybrid template features, which may introduce potential distractors to the template via the cluttered search regions. To enjoy the merits of both methods, we propose a robust object modeling framework for visual tracking (ROMTrack), which simultaneously models the inherent template and the hybrid template features. As a result, harmful distractors can be suppressed by combining the inherent features of target objects with search regions' guidance. Target-related features can also be extracted using the hybrid template, thus resulting in a more robust object modeling framework. To further enhance robustness, we present novel variation tokens to depict the ever-changing appearance of target objects. Variation tokens are adaptable to object deformation and appearance variations, which can boost overall performance with negligible computation. Experiments show that our ROMTrack sets a new state-of-the-art on multiple benchmarks.

1. Introduction

Visual object tracking (VOT) [1, 4, 8, 12, 22, 25, 35, 39, 41, 47, 57, 60] is a fundamental task in computer vision, which aims at localizing an arbitrary target in video sequences given its initial status. The occlusion, scale variation, object deformation, and co-occurrence of distractor objects pose a challenge to acquiring an effective tracker in real-world scenarios. Current dominating trackers typically address these problems with a Transformer-based [42] architecture.

*Corresponding author.

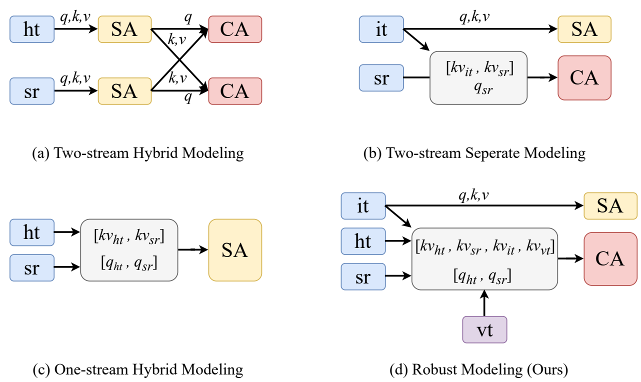


Figure 1: Three typical object modeling methods for template-search feature learning, together with our Robust Modeling design. ht, it, and vt represent hybrid template, inherent template, and variation tokens, respectively. sr represents the search region. SA and CA denote self-attention and cross-attention, respectively.

The core components in a typical Transformer tracking framework are the object modeling blocks. As demonstrated in Figure 1(a), the two-stream hybrid modeling methods [6, 50] learn the template feature interactively with the search region via two cross-attention (CA) operations. Instead of cross-attention, the one-stream hybrid modeling methods [5, 54] in Figure 1(c) jointly learn the hybrid template feature and search region feature with one self-attention (SA) operation. Different from hybrid template modeling, the two-stream separate modeling [10, 23] in Figure 1(b) keeps an inherent template stream to ensure the purity of template features. Separate template learning can keep the inherent features of target templates, which prevents interference from search regions. Though suffering from potential distractors, hybrid template learning conducts extensive feature matching between the template and search region, thus allowing mutual guidance for target-oriented feature extraction.

In order to enjoy the merits of separate and hybrid template modeling simultaneously, we propose a robust object modeling framework for visual tracking (named ROMTrack). As shown in Figure 1(d), our robust modeling scheme involves two kinds of templates, the inherent template it and the hybrid template ht. Meanwhile, our scheme also designs the novel variation tokens vt. The inherent template applies self-attention separately to enhance its learned feature. Besides, it accepts queries from the hybrid template and the search region features to provide inherent information for discriminative target-oriented feature learning. The bottom part of Figure 1(d) is a hybrid attention that adopts a standard cross-attention operation to enhance the template and search region features with mutual guidance. Furthermore, it is well-recognized that tracking is a task suffering from object deformation and appearance variations [10, 57]. We tackle this problem by introducing novel variation tokens to improve robustness. It is observed that the target's motion during a short period is usually smooth but may be accompanied by large changes in appearance [6, 58]. The tracker can easily handle smooth motion, but appearance changes are hard to distinguish. Therefore, we generate variation tokens from hybrid template features to leverage appearance information during tracking. Despite the simplicity, our variation tokens perform well with negligible computation.

The main contributions of this work are three-fold:

- We propose a robust object modeling framework for visual tracking (ROMTrack). It can keep the inherent information of the target template and enables mutual feature matching between the target and the search region simultaneously.

- We present a neat and effective variation-token design that embeds the object's appearance context during tracking into the attention calculation of hybrid target-search features.

- The proposed ROMTrack sets a new state-of-the-art performance on six challenging benchmarks, including GOT-10k [26], LaSOT [18], TrackingNet [38], $\text{LaSOT}_{\textbf{ext}}$ [17], OTB100 [49], and NFS30 [21].

2. Related Work

In this section, we briefly review different visual object tracking methods and the Transformer attention mechanism in general vision tasks.

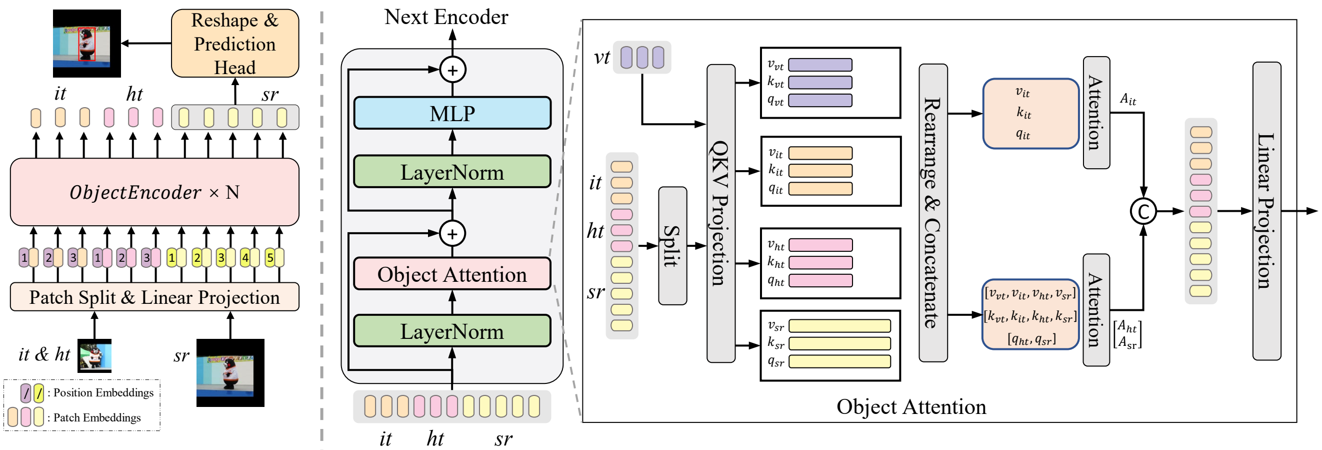
Visual Object Tracking. Early Siamese-based trackers [2, 7, 16, 19, 29, 30, 45, 51, 56, 58] first extract the template and search region features separately by a CNN (Convolutional Neural Network) backbone with shared structure and parameters. Then, a correlation-based network is responsible for computing the similarity between the template and the search region. Correlation modeling plays a critical role in tracking networks. However, conventional correlation-based networks do not fully use the global context. Therefore, recent dominating trackers [6, 9, 10, 20, 23, 31, 44, 53, 54, 55] introduce stacked Transformer layers for better relation modeling.

The pioneering Transformer tracking method TransT [6] adopts a similar pipeline as Siamese-based trackers, where the lightweight relation modeling network is replaced with relatively heavy Transformer layers. The two-stream attention in TransT enables bi-directional information interaction. Unlike TransT, MixFormer [10] utilizes the flexibility of attention operations for simultaneous feature extraction and relation modeling. MixFormer also adopts a two-stream attention pipeline but prunes the cross-attention from the target's query to the search area, eliminating potential negative influence from distractors. AiATrack [23] employs a similar asymmetric scheme, where the search region conducts queries on the target feature while the target only enhances its feature with self-attention blocks. In order to bridge a free information flow between the template and search region, OSTrack [54] adopts a one-stream attention scheme. It concatenates the flattened template and search region and feeds them into stacked self-attention layers for joint feature learning and relation modeling. However, the extensive feature fusion of self-attention layers may bring interfering information to the target feature due to potential distractors. Instead, we propose a robust object modeling scheme that contains an inherent template stream, a variation-token stream, and a bi-directional template-search stream, leading to a more accurate transformer tracker.

Transformer Attention. The attention mechanism [42] has played an increasingly important role in computer vision in the past few years. And recently, in most vision tasks, attention architectures represented by Transformer have obtained impressive performances. To be more specific, Transformer attention is usually helpful for modeling spatial features and temporal relations. For example, the Vision Transformer (ViT) [15] and other following works, including PVT [46], CVT [48], and Swin-Transformer [33] have shown their capacity to aggregate spatial information and benefit many downstream tasks. Transformer attention has also been utilized in visual tracking but has yet to be fully exploited. Most of them focus on designing complex structures. Instead, in this work, we try to explain the potential defects of previous trackers and seek an approach for robust object modeling. We follow the pure Transformer architecture to further explore attention mechanism for visual tracking.

3. Method

We propose ROMTrack, a robust object modeling network for tracking, in Figure 2. We first give an overview of the proposed ROMTrack architecture and then elaborate on the proposed object encoder. Finally, we give a discussion



(a) ROMTrack (b) Object Encoder in ROMTrack

Figure 2: (a) Overview of the proposed ROMTrack framework. The template and search region images are split into patches, and then linearly projected, concatenated, and fed into stacked encoder layers for robust object modeling. it, ht, and sr denote the inherent template, the hybrid template, and the search region, respectively. (b) Architecture of the object encoder layer. vt denotes variation tokens.

on our modeling method.

3.1. Overall Architecture

Backbone. As shown in Figure 2(a), we adopt the vanilla ViT [15] as the backbone. More concretely, we replace the conventional ViT encoder with the proposed object encoder and add a prediction head on the output tokens of the last encoder. The input of ROMTrack is a triplet of images containing a template image pair $(it_{img}, ht_{img}) \in \mathbb{R}^{3\times H_t \times W_t}$ and one search region image $sr_{img} \in \mathbb{R}^{3\times H_{sr}\times W_{sr}}$. The it_{img} is responsible for learning inherent template features and ht_{img} is accountable for learning hybrid template features. Following ViT [15], we split all input images and flatten them into sequences of patches: $it_p \in \mathbb{R}^{N_t\times 3\cdot P^2}$, $ht_p \in \mathbb{R}^{N_t\times 3\cdot P^2}$, and $sr_p \in \mathbb{R}^{N_{sr}\times 3\cdot P^2}$, where $P \times P$ is the resolution of each patch, $N_t = H_t W_t/P^2$ and $N_{sr} = H_{sr} W_{sr}/P^2$ are the number of patches of the templates and the search region, respectively. Then we generate D-dimensional patch embeddings with a linear projection layer. After adding position embeddings, the resulting token sequences are ready for N stacked object encoders. The encoder layer employs robust object modeling to learn discriminative feature representations, which will be elaborated in Section 3.2.

Prediction Head. As pointed out in previous work [10], corner-based [28] localization heads may have a bad effect on the modeling capacity of deeper transformer encoders. Consequently, We adopt a fully convolutional center-based [59] localization head to estimate the bounding box of tracked objects, which consists of L stacked Conv-BN-ReLU layers. Specifically, the target classification score map $C \in [0,1]^{\frac{H_{sr}}{P}\times\frac{W_{sr}}{P}}$, the local offset map $O \in [0,1]^{2\times\frac{H_{sr}}{P}\times\frac{W_{sr}}{P}}$, and the normalized bounding box size map $S \in [0,1]^{2\times\frac{H_{sr}}{P}\times\frac{W_{sr}}{P}}$ are generated by the center head. Finally, the position with the highest classification score in C is considered the target position and the target bounding box can be calculated using O and S.

The classification branch is supervised using Gaussian weighted focal loss [28] during training. Specifically, given a ground truth target center $\hat{c}$ and the corresponding position $\tilde{c} = [\tilde{c}_x, \tilde{c}_y]$ in feature map, the ground truth heatmap can be formulated as $\hat{C}_{xy} = e^{-\frac{(x-\tilde{c}_x)^2+(y-\tilde{c}_y)^2}{2\sigma^2}}$, where σ is a standard deviation adaptive to object size. So the Gaussian weighted focal loss is employed as follows:

$$
\begin{aligned}
L_{cls} = -\sum_{xy}[&\mathbb{I}(\hat{C}_{xy}=1)(1-C_{xy})^{\alpha}\log(C_{xy}) \\
&+ (1-\hat{C}_{xy})^{\beta}(C_{xy})^{\alpha}\log(1-C_{xy})],
\end{aligned} \tag{1}
$$

where $\mathbb{I}(\cdot)$ is the indicator function, α and β are hyper-parameters, and we set them to 2 and 4 following [28, 54].

As for the bounding box regression branch, L_1 loss and $GIoU$ loss are adopted. Generally, We set different weights for different losses: $\lambda_{L_1} = 5$, $\lambda_{giou} = 2$ and $\lambda_{cls} = 1$. And both training stages share the same loss function as follows:

$$
L_{total} = \lambda_{L_1} L_1 + \lambda_{giou} L_{giou} + \lambda_{cls} L_{cls}. \tag{2}
$$

3.2. Object Encoder

The proposed object encoder in Figure 2(b) contains two critical components, *i.e.*, variation tokens and robust object modeling. Before describing the principles of robust object modeling, we first explain our design of variation tokens.

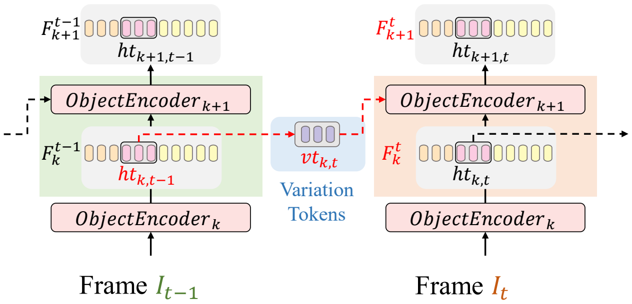


Figure 3: Schema of the proposed variation-token design.

Variation Tokens. Variation tokens are the embedding of contextual appearance changes of the target object, which helps to tackle the problem of object deformation and appearance variations. As shown in Figure 3, the variation tokens vt are generated after each object encoder and encode the variation of appearance context from the search region, which will be further demonstrated later. The generation and usage of variation tokens can be formulated as follows:

$$vt_{k,t} = ht_{k,t-1}, \tag{3}$$

$$F_{k+1}^t = ObjectEncoder_{k+1}(\text{Concat}(vt_{k,t}, F_k^t)), \tag{4}$$

where F represents the output features, k is the encoder index and t denotes the t-th frame. So F_k^t is the output features of k-th encoder in frame I_t, and $ht_{k,t}$ is the hybrid template part of F_k^t, which incorporates appearance information from the search region.

Equation 3 indicates that we reserve hybrid template tokens of frame I_{t-1} as the input to frame I_t, because appearance variations of the target object in I_t relative to I_1 are encoded at the feature level of these tokens. Furthermore, Equation 4 aims to embed the variation of object appearance into the network. Specifically, it feeds the variation tokens $vt_{k,t}$ together with the output features F_k^t to the $(k+1)$-th encoder when tracking the t-th frame. The MACs are negligible because the construction of variation tokens only includes embedding assignments (Equation 3). Meanwhile, the employment of variation tokens is just a combination of token concatenation and a series of lightweight cross-attention operations related to $vt_{k,t}$ (Equation 4).

Robust Object Modeling. One-stream hybrid modeling enables extensive bi-directional information flows between the template-search image pairs, and discriminative target-oriented features can be dynamically extracted by mutual guidance. However, excessive communications may suffer from tracking failures and background clutters. Two-stream separate modeling can keep a separate template stream to avoid negative influences from potential distractors, but its extracted template features are inadequate to object deformations and appearance changes.

To address these problems, we propose a robust object modeling method. As shown in Figure 2(b), the input of object attention consists of four parts, *i.e.*, the inherent template $it \in \mathbb{R}^{N_t \times D}$, the hybrid template $ht \in \mathbb{R}^{N_t \times D}$, the search region $sr \in \mathbb{R}^{N_{sr} \times D}$, and the variation tokens $vt \in \mathbb{R}^{N_t \times D}$. Following the conventional attention block, we use a linear projection layer to produce the d-dimentional $(query, key, value)$ triplet. For example, the triplet for it is (q_{it}, k_{it}, v_{it}). Then we conduct self-attention on it to learn pure template features:

$$A_{it} = \text{Softmax}(\frac{q_{it}k_{it}^T}{\sqrt{d}})v_{it}, \tag{5}$$

where A_{it} represents the output of the self-attention operation. The inherent template feature is enhanced through self-attention, eliminating interference from the search region. Meanwhile, the hybrid template feature and the search region feature are learned via a cross-attention operation. Let (q_z, k_z, v_z) denote the $(query, key, value)$ triplet of the cross-attention, where q_z, k_z, and v_z are defined as follows:

$$q_z = [q_{ht}, q_{sr}], \tag{6}$$

$$k_z = [k_{vt}, k_{it}, k_{ht}, k_{sr}], \tag{7}$$

$$v_z = [v_{vt}, v_{it}, k_{ht}, v_{sr}]. \tag{8}$$

Equation 6 - 8 show that the inputs of cross-attention are a rearrangement and concatenation (denoted by $[\dots]$) of the search region features (indicated by subscript sr), the two types of template features (indicated by subscript ht and it), and the variation token features (indicated by subscript vt). The output of cross-attention operation A_z can be obtained via:

$$A_z = \text{Softmax}(\frac{q_z k_z^T}{\sqrt{d}})v_z. \tag{9}$$

The hybrid template feature and search region feature inside A_z get enhanced by fusing informative features from the inherent template and variation tokens. As a result, the network is able to obtain the information of both the original target (*i.e.*, object in the first frame) and the ever-changing target (*i.e.*, object in the $(t-1)$-th frame) when tracking the t-th frame.

For further explanation, we conduct a more in-depth analysis below. Let M_z be the correlation map calculated in the cross-attention, then M_z can be written as:

$$\begin{aligned} M_z =& \text{Softmax}(\frac{q_z k_z^T}{\sqrt{d}}) \\ =& \text{Softmax}(\frac{\begin{bmatrix} q_{ht}k_{vt}^T & q_{ht}k_{it}^T & q_{ht}k_{ht}^T & q_{ht}k_{sr}^T \\ q_{sr}k_{vt}^T & q_{sr}k_{it}^T & q_{sr}k_{ht}^T & q_{sr}k_{sr}^T \end{bmatrix}}{\sqrt{d}}) \\ \triangleq& \begin{bmatrix} M_{ht,vt} & M_{ht,it} & M_{ht,ht} & M_{ht,sr} \\ M_{sr,vt} & M_{sr,it} & M_{sr,ht} & M_{sr,sr} \end{bmatrix}, \end{aligned} \tag{10}$$

where $M_{a,b}$ is a measure of similarity between a and b, *e.g.*, $M_{ht,sr}$ refers to the similarity between the hybrid template and search region. Based on Equation 10, the attention output A_z can be rewritten as:

$$\begin{aligned}
A_z =& M_z v_z \\
=& \text{Softmax}(\frac{q_z k_z^T}{\sqrt{d}}) v_z \\
=& \begin{bmatrix} M_{ht,vt} & M_{ht,it} & M_{ht,ht} & M_{ht,sr} \\ M_{sr,vt} & M_{sr,it} & M_{sr,ht} & M_{sr,sr} \end{bmatrix} \begin{bmatrix} v_{vt} \\ v_{it} \\ v_{ht} \\ v_{sr} \end{bmatrix} \\
=& \begin{bmatrix} M_{ht,vt}v_{vt} + M_{ht,it}v_{it} + M_{ht,ht}v_{ht} + M_{ht,sr}v_{sr} \\ M_{sr,vt}v_{vt} + M_{sr,it}v_{it} + M_{sr,ht}v_{ht} + M_{sr,sr}v_{sr} \end{bmatrix} \\
\triangleq& \begin{bmatrix} A_{ht} \\ A_{sr} \end{bmatrix},
\end{aligned} \tag{11}$$

where A_{ht} and A_{sr} denote the generated hybrid template and search region features, respectively. It is easy to figure out that both A_{ht} and A_{sr} aggregate information from the inherent template (*e.g.*, $M_{ht,it}v_{it}$ and $M_{sr,it}v_{it}$) and variation tokens (*e.g.*, $M_{ht,vt}v_{vt}$ and $M_{sr,vt}v_{vt}$) to enhance their features.

Recall that in Figure 3, we use the hybrid template $ht_{k,t-1}$ to generate the variation tokens $vt_{k,t}$ to provide variation information of contextual appearance for the next frame. This is reasonable because the $M_{ht,sr}v_{sr}$ term in A_{ht} has incorporated the feature of search region into the hybrid template, helping the output hybrid template tokens to capture current information of the search region. In other words, the appearance information of the target object in both the first frame and the current frame is incorporated into the hybrid template tokens, making them sensitive to contextual appearance changes.

Therefore, we can cache the hybrid template as variation tokens in the next frame to leverage appearance information during tracking. Overall, with the variation-token design, the feature extraction and information integration process are unified in the proposed object modeling framework.

3.3. Discussions

Necessity of Hybrid Template. The hybrid template serves two primary purposes. The first is to conduct extensive feature matching between the template and search region, thus allowing mutual guidance for target-oriented feature extraction. The second is to encode the variation of appearance context by interacting with the search region, which helps variation tokens model appearance changes of objects between adjacent frames. Further analysis is conducted in Section 4.3.

Training and Inference. The training process contains two stages. In the first stage, we follow the standard training recipe of mainstream trackers [6, 10, 53] to train our ROMrack without variation tokens, *i.e.*, only with the inherent and hybrid templates. In the second stage, we add variation tokens into training by sampling two search regions in consecutive frames of the same sequence to model the appearance variations between them. For inference, only the initial template and the cropped search region are fed into the ROMTrack pipeline to produce the target bounding box. The initial template serves as the input for both inherent and hybrid templates. During the tracking procedure, the variation tokens are obtained per frame and employed for subsequent tracking.

4. Experiments

4.1. Implementation Details

Our trackers are implemented using Python 3.6.13 and PyTorch 1.7.1. The models are trained on 8 Tesla V100 GPUs, and we test the inference speed on a single NVIDIA1080Ti GPU.

Model. We adopt the vanilla ViT-Base [15] model pretrained with MAE [24] on ImageNet [14] as the backbone of our ROMTrack. All the input images are split into 16×16 patches. As for the prediction head, we adopt a lightweight FCN consisting of 4 stacked Conv-BN-ReLU layers for each output. To build an efficient tracker, we adopt a smaller image resolution than other trackers [10, 23, 53]. Namely, the sizes of the template and search images are 128×128 pixels and 256×256 pixels, respectively. Furthermore, to verify the scalability of our proposed ROMTrack, we also provide an implementation with a higher resolution called ROMTrack-384, and the sizes of the template and search images are 192×192 pixels and 384×384 pixels.

Training. The training splits of COCO [32], GOT-10k [26], LaSOT [18], and TrackingNet [38] are used for training. While for the GOT-10k test, we follow the one-shot protocol by only using the GOT-10k train split for training. The training process of ROMTrack consists of two stages: the first 300 epochs are for the backbone and head, and the extra 100 are to merge the variation tokens into our architecture. For data augmentations, horizontal flip and brightness jittering are used following the convention [10, 53, 54]. We train the ROMTrack using AdamW [34] with weight decay set to 10^{-4}. For the first stage, the learning rate is initialized as 4×10^{-4} and decreased to 4×10^{-5} at the epoch of 240. For the second stage, the learning rate is initialized as 4×10^{-5} and decreased to 4×10^{-6} at the epoch of 80.

Inference. We adopt the Hanning window penalty to utilize positional prior in tracking following the common practice [6, 54, 58]. To be more specific, the classification map C is multiplied by the Hanning window with the same size to generate confidence scores, and we simply select the prediction box with the highest confidence score as result.

Method	Source	GOT-10k*			LaSOT			TrackingNet			$\text{LaSOT}_{\text{ext}}$		
		$AO(\%)$	$SR_{0.5}(\%)$	$SR_{0.75}(\%)$	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$
ROMTrack	Ours	**72.9**	**82.9**	**70.2**	**69.3**	78.8	**75.6**	**83.6**	**88.4**	**82.7**	**48.9**	**59.3**	**55.0**
SwinTrack-T-224 [31]	NIPS22	**71.3**	**81.9**	64.5	67.2	-	70.8	81.1	-	78.4	47.6	-	53.9
OSTrack-256 [54]	ECCV22	71.0	80.4	**68.2**	69.1	78.7	**75.2**	83.1	87.8	82.0	47.4	57.3	53.3
OSTrack-256(w/o CE) [54]	ECCV22	71.0	80.3	68.2	68.7	78.1	74.6	82.9	87.5	81.6	-	-	-
AiATrack [23]	ECCV22	69.6	80.0	63.2	69.0	**79.4**	73.8	82.7	87.8	80.4	46.8	54.4	**54.2**
SimTrack-B/16 [5]	ECCV22	68.6	78.9	62.4	**69.3**	78.5	74.0	82.3	86.5	-	-	-	-
Unicorn [52]	ECCV22	-	-	-	68.5	76.6	74.1	83.0	86.4	**82.2**	-	-	-
MixFormer-22k [10]	CVPR22	70.7	80.0	67.8	69.2	78.7	74.7	**83.1**	**88.1**	81.6	-	-	-
MixFormer-1k [10]	CVPR22	71.2	79.9	65.8	67.9	77.3	73.9	82.6	87.7	81.2	-	-	-
ToMP50 [36]	CVPR22	-	-	-	67.6	78.0	72.2	81.2	86.2	78.6	45.4	57.6	-
ToMP101 [36]	CVPR22	-	-	-	68.5	**79.2**	73.5	81.5	86.4	78.9	45.9	**58.1**	-
SBT-large [50]	CVPR22	70.4	80.8	64.7	66.7	-	71.1	-	-	-	-	-	-
KeepTrack [37]	ICCV21	-	-	-	67.1	77.2	70.2	-	-	-	**48.2**	58.0	-
STARK [53]	ICCV21	68.8	78.1	64.1	67.1	77.0	-	82.0	86.9	-	-	-	-
DTT [55]	ICCV21	63.4	74.9	51.4	60.1	-	-	79.6	85.0	78.9	-	-	-
TransT [6]	CVPR21	67.1	76.8	60.9	64.9	73.8	69.0	81.4	86.7	80.3	45.1	51.3	51.2
TrDiMP [44]	CVPR21	67.1	77.7	58.3	63.9	-	61.4	78.4	83.3	73.1	-	-	-
LTMU [11]	CVPR20	-	-	-	57.2	-	57.2	-	-	-	41.4	49.9	47.3
SiamR-CNN [43]	CVPR20	64.9	72.8	59.7	64.8	72.2	-	81.2	85.4	80.0	-	-	-
Ocean [58]	ECCV20	61.1	72.1	47.3	56.0	65.1	56.6	-	-	-	-	-	-
DiMP [3]	ICCV19	61.1	71.7	49.2	56.9	65.0	56.7	74.0	80.1	68.7	39.2	47.6	45.1
SiamRPN++ [29]	CVPR19	51.7	61.6	32.5	49.6	56.9	49.1	73.3	80.0	69.4	34.0	41.6	39.6
MDNet [39]	CVPR16	29.9	30.3	9.9	39.7	46.0	37.3	60.6	70.5	56.5	27.9	34.9	31.8
SiamFC [2]	ECCV16	34.8	35.3	9.8	33.6	42.0	33.9	57.1	66.3	53.3	23.0	31.1	26.9
Trackers with Higher Resolution or Larger Model													
ROMTrack-384	Ours	**74.2**	**84.3**	**72.4**	**71.4**	**81.4**	**78.2**	**84.1**	**89.0**	**83.7**	**51.3**	**62.4**	**58.6**
SwinTrack-B-384 [31]	NIPS22	72.4	80.5	67.8	**71.3**	-	76.5	**84.0**	-	82.8	49.1	-	55.6
OSTrack-384 [54]	ECCV22	**73.7**	**83.2**	**70.8**	71.1	**81.1**	**77.6**	83.9	88.5	**83.2**	**50.5**	**61.3**	**57.6**
SimTrack-L/14 [5]	ECCV22	69.8	78.8	66.0	70.5	79.7	76.2	83.4	87.4	-	-	-	-
MixFormer-L [10]	CVPR22	-	-	-	70.1	79.9	76.3	83.9	**88.9**	83.1	-	-	-

Table 1: Comparison with state-of-the-art on four large-scale benchmarks: GOT-10k, LaSOT, TrackingNet, $\text{LaSOT}_{\text{ext}}$. The best two results are shown in **red** and **blue** fonts. * denotes the model trained with only GOT-10k train split.

	SiamRPN++ [29]	PrDiMP [13]	SuperDiMP [27]	TransT [6]	STARK [53]	KeepTrack [37]	RTS [40]	ToMP [36]	MixFormer-L [10]	OSTrack-384 [54]	AiATrack [23]	ROMTrack (Ours)	ROMTrack-384 (Ours)
OTB100	69.6	69.6	70.1	69.4	68.5	70.9	-	70.1	70.4	-	69.6	**71.4**	**70.9**
NFS30	50.3	63.5	64.8	65.7	65.2	66.4	65.4	66.7	-	66.5	67.9	**68.0**	**68.8**

Table 2: Comparison with state-of-the-art trackers on two small-scale benchmarks: OTB100 and NFS30. Results are compared in terms of AUC(%) score. The best two results are shown in **red** and **blue** fonts.

4.2. Comparison with State-of-the-art Trackers

We compare our ROMTrack with state-of-the-art (SOTA) trackers on six different benchmarks, including four well-known large-scale benchmarks and two commonly used small-scale benchmarks. Results on other datasets are available in Appendix.

GOT-10k. GOT-10k [26] is a large-scale dataset containing more than 10000 video segments of real-world moving objects. The object classes between train and test sets are zero-overlapped. We follow the one-shot protocol to only train our model on the GOT-10k training split and evaluate the results through the evaluation server. As presented in Table 1, ROMTrack improves all metrics by a large margin, *e.g.*, 1.6% in AO compared with SwinTrack-T-224 and 2% in $\text{SR}_{0.75}$ compared with OSTrack-256, which indicates the capability in accurate discrimination and localization of objects. Furthermore, our higher resolution model ROMTrack-384 sets a new SOTA on the GOT-10k test split, demonstrating that our method has excellent potential to track objects of unseen classes by robust object modeling.

LaSOT. LaSOT [18] is a large-scale, long-term tracking benchmark containing 1400 video sequences: 1120 for training and 280 for testing. We evaluate our ROMTrack on the test set to compare with previous SOTA trackers. As reported in Table 1, our ROMTrack shows more accurate and balanced performance, surpassing both OSTrack and MixFormer in all three metrics. Specifically, our higher resolution model ROMTrack-384 establishes a new state-of-the-art on AUC of 71.4%. The result demonstrates that our approach benefits the long-term tracking scenarios. More analysis of the performance improvements on the LaSOT dataset can be found in Appendix.

TrackingNet. TrackingNet [38] is a large-scale short-term tracking benchmark that provides more than 30000 video sequences with over 14 million boxes. The test split of TrackingNet contains 511 sequences without publicly available ground truth and covers diverse target classes and scenes. We submit the tracking results to the official eval-

Method	Speed (FPS)	MACs (G)	Params (M)	LaSOT AUC(%)	GOT-10k* AO(%)
OSTrack-256 (w/o CE) [54]	65	29.0	92.1	68.7	71.0
MixFormer-22k [10]	25	23.0	35.6	69.2	70.7
ROMTrack	62	34.5	92.1	**69.3**	**72.9**
OSTrack-384 (w/o CE)	29	65.3	92.1	71.0	73.7
MixFormer-L	18	127.8	183.9	70.1	-
ROMTrack-384	28	77.7	92.1	**71.4**	**74.2**

Table 3: Comparison of inference speed, MACs, and Params. We include the results of OSTrack without candidate elimination (w/o CE) here for a fair speed comparison. * denotes the model trained with only GOT-10k train split.

uation server and make comparisons with previous SOTA trackers in Table 1. The results show that our ROMTrack obtains 83.6% in AUC and 88.4% in NP, outperforming previous SOTA MixFormer-22k appreciably. It demonstrates that our approach also benefits short-term visual tracking.

LaSOT$_{\text{ext}}$. LaSOT$_{\text{ext}}$ [17] is an extension of the LaSOT dataset, which contains 150 videos of 15 new object classes. These recently proposed sequences are pretty challenging due to the existence of many similar distractors in the video. The results in Table 1 show that our ROMTrack surpasses all other trackers with a large margin and achieves the top-ranked performance on NP of 59.3%, surpassing ToMP by 1.2%. Our higher resolution model ROMTrack-384 also outperforms previous trackers by a large margin in all three metrics, setting a new state-of-the-art on LaSOT$_{\text{ext}}$, which indicates that our tracker not only has a remarkable generalization ability of unseen classes but also has a robust discrimination ability of similar distractors.

NFS30 and OTB100. Finally, we report our results on two additional small-scale benchmarks: NFS30 [21] and OTB100 [49]. As shown in Table 2, our ROMTrack and ROMTrack-384 have good performances on both benchmarks, establishing SOTA performances. It further indicates the generality of our method.

Speed, MACs and Params. We compare the inference speed, MACs, and Params with state-of-the-art trackers in Table 3. We include the results of OSTrack [54] without candidate elimination (w/o CE) for a fair speed comparison. Our ROMTrack can run in real-time at more than 60 FPS, which is on par with OSTrack-256 (w/o CE). Besides, ROMTrack is 2.5× faster than MixFormer [10], demonstrating the effectiveness of our robust object modeling. For a larger input resolution, our ROMTrack-384 can also achieve comparable speed with OSTrack-384 (w/o CE) and is much faster than MixFormer-L. Moreover, our ROMTrack-384 outperforms MixFormer-L with only 61% and 50% of its MACs and Params, respectively.

Discussions. Since OSTrack [54] also adopts MAE pretrain, we would like to compare with it. OSTrack employs the one-stream hybrid modeling strategy mentioned in Figure 1(c), together with a Candidate Elimination (CE) Module, which brings a considerable performance boost (shown in Table 1). Differently, our ROMTrack does not employ the CE strategy, and it focuses on the robust object modeling mentioned in Figure 1(d). As Table 1 shown, our ROMTrack outperforms OSTrack-256 (w/o CE) by a large margin even on the challenging LaSOT dataset (+0.6% AUC), indicating that the proposed robust object modeling is preferable for more accurate discrimination and localization of objects.

4.3. Ablation Study and Analysis

We perform a detailed ablation study and an extensive analysis to verify the effectiveness of our method.

Study on Inherent Template and Hybrid Template. The most important part of our ROMTrack is the object encoder which unifies the object modeling of template features and search region features. To further verify the effectiveness of the object encoder, we conduct experiments to analyze the performance of the inherent template and the hybrid template designed in our object encoder. Specifically, we remove the variation tokens from our object encoder to form a model called ROMTrack (w/o vt), which remains the inherent template and the hybrid template. And then, we compare it with two other commonly used approaches: separate template modeling (STM) and hybrid template modeling (HTM). Note that HTM is a reproduction of OSTrack-256 (w/o CE) to make a fair comparison. We also include the results of OSTrack-256 (w/o CE) for more comprehensive comparisons. The architecture comparison can be found in Table 4(a). For fairness, we implement STM, HTM, and ROMTrack (w/o vt) under the same framework and experimental settings. As demonstrated in Table 4(b), our ROMTrack (w/o vt) performs best in all metrics on benchmark datasets, showing the superiority of our robust object modeling. Moreover, ROMTrack (w/o vt) achieves impressive performances on LaSOT$_{\text{ext}}$ dataset, surpassing HTM and STM by 2.1% and 1.5% in AUC separately. In addition, we provide visualizations of attention maps and features for different modeling methods in the Appendix.

Study on Variation Tokens. To verify the effectiveness of variation tokens, we replace the variation tokens with template features extracted using the last prediction result and form a model named ROMTrack-lpr. Specifically, sup-

Method	Inherent Template (it)	Hybrid Template (ht)
OSTrack-256 (w/o CE)	✗	✓
STM	✓	✗
HTM	✗	✓
ROMTrack (w/o vt)	✓	✓

(a)

Method	LaSOT			$\text{LaSOT}_{\textbf{ext}}$			GOT-10k*		
	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$	$AO(\%)$	$SR_{0.5}(\%)$	$SR_{0.75}(\%)$
OSTrack-256 (w/o CE)	68.7	78.1	74.6	-	-	-	71.0	80.3	68.2
STM	68.7	78.1	74.4	46.7	56.4	52.7	72.0	81.5	69.2
HTM	68.7	78.2	74.5	46.1	55.7	51.5	72.1	81.2	69.1
ROMTrack (w/o vt)	**68.8**	**78.4**	**75.0**	**48.2**	**58.4**	**54.1**	**72.5**	**82.4**	**69.8**

(b)

Table 4: (a) Architectures of OSTrack-256 (w/o CE), STM, HTM and our ROMTrack (w/o vt). (b) Ablation study on inherent template and hybrid template. The best results are in **bold** fonts. * denotes the model trained with only GOT-10k train split.

Method	LaSOT			$\text{LaSOT}_{\textbf{ext}}$		
	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$
ROMTrack (w/o vt)	68.8	78.4	75.0	48.2	58.4	54.1
ROMTrack-lpr	65.0	72.7	69.6	44.2	53.1	48.2
ROMTrack	**69.3**	**78.8**	**75.6**	**48.9**	**59.3**	**55.0**
HTM	68.7	78.2	74.5	46.1	55.7	51.5
HTM-vt	69.1	78.6	75.2	48.6	58.8	54.5
HTM-vt-online	**69.6**	**79.0**	**76.0**	**49.0**	**59.4**	**55.0**

Table 5: Ablation study on variation tokens and exploration of template updating. The best results are in **bold** fonts.

Method	LaSOT			$\text{LaSOT}_{\textbf{ext}}$		
	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$	$AUC(\%)$	$P_{Norm}(\%)$	$P(\%)$
HTM	68.7	78.4	74.7	46.1	55.7	51.5
HTM-400	68.8	78.3	74.6	46.3	56.0	52.0
ROMTrack	**69.3**	**78.8**	**75.6**	**48.9**	**59.3**	**55.0**
ROMTrack-RS	69.0	78.4	75.1	48.4	58.6	54.3
ROMTrack-CS	**69.3**	**78.8**	**75.6**	**48.9**	**59.3**	**55.0**

Table 6: Ablation study on aligned comparison and sampling strategy. RS and CS denote random sampling and consecutive sampling. The best results are in **bold** fonts.

pose we get a prediction result B_t of the target object in Frame I_t. Then in Frame I_{t+1}, we directly extract template features using B_t and replace variation tokens with these feature tokens to see whether the network can automatically model appearance changes. As shown in the first three rows of Table 5, ROMTrack-lpr performs even worse than ROMTrack (w/o vt) because direct employment of temporal template features tends to motivate the network to conduct more unguided feature fusion and fails to recognize appearance changes of the target object, which harms tracking performance greatly. On the contrary, our ROMTrack outperforms ROMTrack (w/o vt) by 0.5% and 0.7% in AUC score on LaSOT and $\text{LaSOT}_{\textbf{ext}}$ respectively, suggesting that our variation-token design does have the capability of modeling contextual appearance changes by prompting the network to adjust appearance modeling for target objects.

Exploration of Template Updating. Our method does not employ template updating, but we have also conducted some exploration studies to combine template updating with our variation-token design. Firstly, we add variation tokens to HTM (mentioned in Table 4) to form a model named HTM-vt. Secondly, based on HTM-vt, we add the online template and a simple score prediction branch borrowed from MixFormer [10] to form a model named HTM-vt-online. As depicted in the last three rows of Table 5, variation tokens bring significant improvement to HTM, *e.g.*, HTM-vt outperforms HTM by 0.4% and 2.5% in AUC score on LaSOT and $\text{LaSOT}_{\textbf{ext}}$. In addition, with the online template updating strategy, HTM-vt-online further improves the AUC score by 0.5% and 0.4% separately. Compared to MixFormer, we take the same online strategy but achieve better performance, proving the superiority of our method. It is concluded that our variation-token design is vital to improve the overall performance of trackers. With the assistance of a template updating strategy, the performance can be further boosted, indicating that our variation tokens work complementary with the template update strategy.

Study on Aligned Comparison. The training of our tracker consists of two stages, adding up to 400 epochs in total. To prove that the outstanding performance of our tracker is not due to a longer training process, we select the HTM approach and train it for 400 epochs, denoted as HTM-400. The results are shown in the first two rows of Table 6. Our method still outperforms HTM-400 by a large margin (*e.g.*, +0.5% AUC on LaSOT and +2.6% AUC on $\text{LaSOT}_{\textbf{ext}}$), which proves that a simple extension of the training process is not the critical factor for excellent performance. Therefore, the robust object modeling approach is undoubtedly helpful for feature extraction and relation modeling.

Study on Sampling Strategy. During the second training stage, we employ a particular sampling strategy called consecutive sampling (CS). Different from random sampling (RS), two consecutive frames instead of two random frames in the same video sequence are sampled as the search region. The training process is described in Section 3.3. The results in Table 6 show that the consecutive sampling strategy obtains better performance because it helps the model to learn more about the temporal variation of object appearance. In addition, the model trained with random sampling also performs better than HTM, indicating the effectiveness of our robust object modeling.

Visualizations. To explore how the robust object modeling method works in our framework, we also visualize some

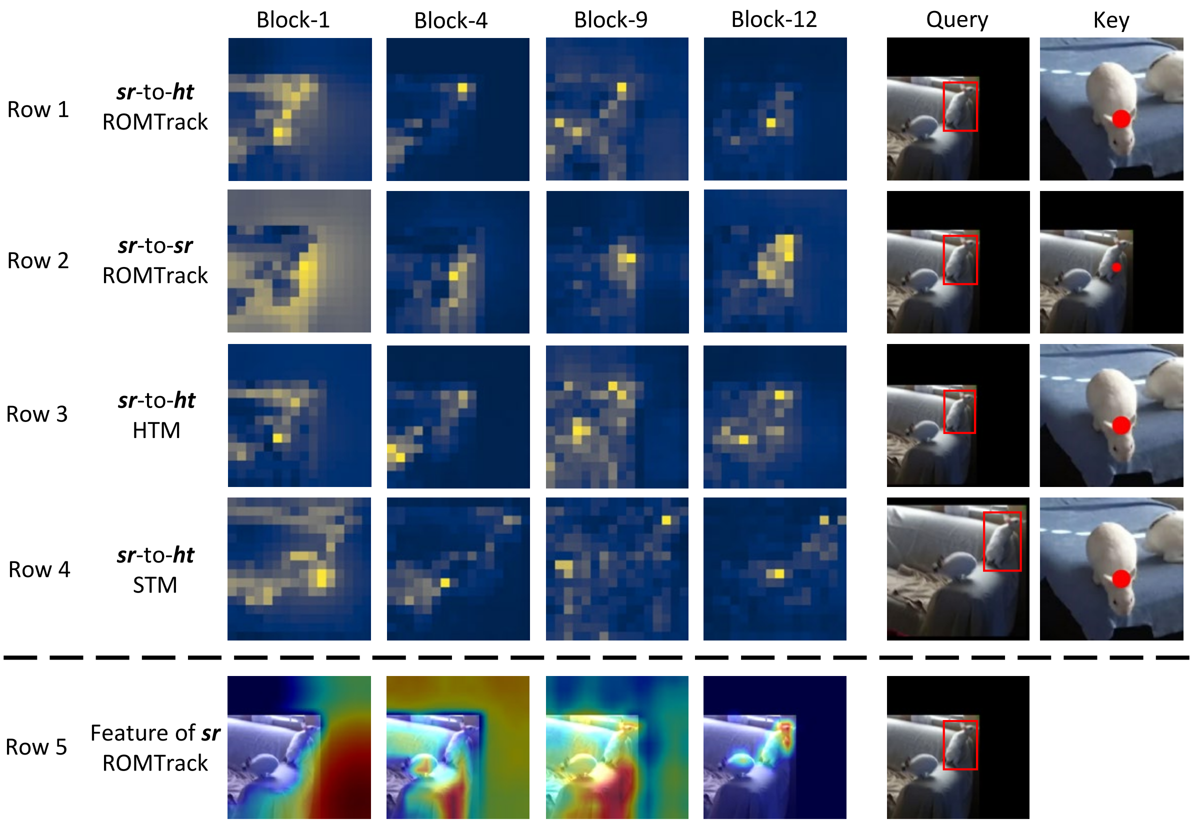


Figure 4: Visualizations of attention maps and features after different blocks. The red rectangle indicates the target object to be tracked. **sr**-to-**ht** refers to search-to-template attention and **sr**-to-**sr** refers to search-to-search attention. The red circle stands for the part chosen as the key, and the attention maps are generated with the query and key pair. STM, HTM, and our ROMTrack are mentioned in Section 4.3. It can be observed that our method effectively concentrates on the target object rather than the distractor.

attention maps and search region features in Figure 4. We observe that:

- The object in the search region is enhanced layer by layer through interaction with the two template streams and the variation tokens.
- Possible distractors in the background get suppressed with our ROMTrack (Row 1, Row 2, and Row 5), suggesting the robustness of our method.
- Both HTM (Row 3) and STM (Row 4) have difficulty in distinguishing distractors with target objects while our ROMTrack locates objects more accurately.

As a result, our ROMTrack shows excellent tracking performance.

5. Conclusions

This work proposes a robust object modeling framework for visual tracking (ROMTrack). The proposed ROMTrack utilizes two template streams to learn robust and discriminative feature representations. The inherent template keeps original features of target objects, and the hybrid template learns mixed template-search features. The hybrid template can extract helpful information from the inherent template to form target-oriented features. Besides, the variation tokens are introduced to embed appearance context, thus adaptable to object deformation and appearance variations. The variation-token design is subtly integrated into attention computation, leading to a neat and effective tracker. Extensive experiments show that ROMTrack performs better than previous methods on multiple benchmarks.

Acknowledgements. This work is supported by National Key R&D Program of China (No. 2022ZD0160900), National Natural Science Foundation of China (No. 62076119, No. 61921006, No. 62072232), Fundamental Research Funds for the Central Universities (No. 020214380091, No. 020214380099), and Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

[1] Luca Bertinetto, Jack Valmadre, Stuart Golodetz, Ondrej Miksik, and Philip H. S. Torr. Staple: Complementary learners for real-time tracking. In *CVPR*, pages 1401–1409. IEEE Computer Society, 2016. 1

[2] Luca Bertinetto, Jack Valmadre, João F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. Fully-convolutional siamese networks for object tracking. In *ECCV Workshops (2)*, volume 9914 of *Lecture Notes in Computer Science*, pages 850–865, 2016. 2, 6

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, pages 6181–6190. IEEE, 2019. 6

[4] Goutam Bhat, Joakim Johnander, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Unveiling the power of deep tracking. In *ECCV (2)*, volume 11206 of *Lecture Notes in Computer Science*, pages 493–509. Springer, 2018. 1

[5] Boyu Chen, Peixia Li, Lei Bai, Lei Qiao, Qiuhong Shen, Bo Li, Weihao Gan, Wei Wu, and Wanli Ouyang. Backbone is all your need: A simplified architecture for visual object tracking. In *ECCV (22)*, volume 13682 of *Lecture Notes in Computer Science*, pages 375–392. Springer, 2022. 1, 6

[6] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, pages 8126–8135. Computer Vision Foundation / IEEE, 2021. 1, 2, 5, 6

[7] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. In *CVPR*, pages 6667–6676. Computer Vision Foundation / IEEE, 2020. 2

[8] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Fully convolutional online tracking. *CoRR*, abs/2004.07109, 2020. 1

[9] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Target transformed regression for accurate tracking. *CoRR*, abs/2104.00403, 2021. 2

[10] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *CVPR*, pages 13598–13608. IEEE, 2022. 1, 2, 3, 5, 6, 7, 8

[11] Kenan Dai, Yunhua Zhang, Dong Wang, Jianhua Li, Huchuan Lu, and Xiaoyun Yang. High-performance long-term tracking with meta-updater. In *CVPR*, pages 6297–6306. Computer Vision Foundation / IEEE, 2020. 6

[12] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ATOM: accurate tracking by overlap maximization. In *CVPR*, pages 4660–4669. Computer Vision Foundation / IEEE, 2019. 1

[13] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *CVPR*, pages 7181–7190. Computer Vision Foundation / IEEE, 2020. 6

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, pages 248–255. IEEE Computer Society, 2009. 5

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*. OpenReview.net, 2021. 2, 3, 5

[16] Fei Du, Peng Liu, Wei Zhao, and Xianglong Tang. Correlation-guided attention for corner detection based visual tracking. In *CVPR*, pages 6835–6844. Computer Vision Foundation / IEEE, 2020. 2

[17] Heng Fan, Hexin Bai, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Harshit, Mingzhen Huang, Juehuan Liu, Yong Xu, Chunyuan Liao, Lin Yuan, and Haibin Ling. Lasot: A high-quality large-scale single object tracking benchmark. *Int. J. Comput. Vis.*, 129(2):439–461, 2021. 2, 7

[18] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *CVPR*, pages 5374–5383. Computer Vision Foundation / IEEE, 2019. 2, 5, 6

[19] Heng Fan and Haibin Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *CVPR*, pages 7952–7961. Computer Vision Foundation / IEEE, 2019. 2

[20] Zhihong Fu, Qingjie Liu, Zehua Fu, and Yunhong Wang. Stmtrack: Template-free visual tracking with space-time memory networks. In *CVPR*, pages 13774–13783. Computer Vision Foundation / IEEE, 2021. 2

[21] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, pages 1134–1143. IEEE Computer Society, 2017. 2, 7

[22] Hamed Kiani Galoogahi, Ashton Fagg, and Simon Lucey. Learning background-aware correlation filters for visual tracking. In *ICCV*, pages 1144–1152. IEEE Computer Society, 2017. 1

[23] Shenyuan Gao, Chunluan Zhou, Chao Ma, Xinggang Wang, and Junsong Yuan. Aiatrack: Attention in attention for transformer visual tracking. In *ECCV (22)*, volume 13682 of *Lecture Notes in Computer Science*, pages 146–164. Springer, 2022. 1, 2, 5, 6

[24] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, pages 15979–15988. IEEE, 2022. 5

[25] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(3):583–596, 2015. 1

[26] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(5):1562–1577, 2021. 2, 5, 6

[27] Matej Kristan, Ales Leonardis, et al. The sixth visual object tracking VOT2018 challenge results. In *ECCV Workshops (1)*, volume 11129 of *Lecture Notes in Computer Science*, pages 3–53. Springer, 2018. 6

[28] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. *Int. J. Comput. Vis.*, 128(3):642–656, 2020. 3

[29] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In *CVPR*, pages 4282–4291. Computer Vision Foundation / IEEE, 2019. 2, 6

[30] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, pages 8971–8980. Computer Vision Foundation / IEEE Computer Society, 2018. 2

[31] Liting Lin, Heng Fan, Yong Xu, and Haibin Ling. Swintrack: A simple and strong baseline for transformer tracking. *CoRR*, abs/2112.00995, 2021. 2, 6

[32] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *ECCV (5)*, volume 8693 of *Lecture Notes in Computer Science*, pages 740–755. Springer, 2014. 5

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 9992–10002. IEEE, 2021. 2

[34] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR (Poster)*. OpenReview.net, 2019. 5

[35] Alan Lukezic, Tomas Vojir, Luka Cehovin Zajc, Jiri Matas, and Matej Kristan. Discriminative correlation filter with channel and spatial reliability. In *CVPR*, pages 4847–4856. IEEE Computer Society, 2017. 1

[36] Christoph Mayer, Martin Danelljan, Goutam Bhat, Matthieu Paul, Danda Pani Paudel, Fisher Yu, and Luc Van Gool. Transforming model prediction for tracking. In *CVPR*, pages 8721–8730. IEEE, 2022. 6

[37] Christoph Mayer, Martin Danelljan, Danda Pani Paudel, and Luc Van Gool. Learning target candidate association to keep track of what not to track. In *ICCV*, pages 13424–13434. IEEE, 2021. 6

[38] Matthias Müller, Adel Bibi, Silvio Giancola, Salman Al-Subaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV (1)*, volume 11205 of *Lecture Notes in Computer Science*, pages 310–327. Springer, 2018. 2, 5, 6

[39] Hyeonseob Nam and Bohyung Han. Learning multi-domain convolutional neural networks for visual tracking. In *CVPR*, pages 4293–4302. IEEE Computer Society, 2016. 1, 6

[40] Matthieu Paul, Martin Danelljan, Christoph Mayer, and Luc Van Gool. Robust visual tracking by segmentation. In *ECCV (22)*, volume 13682 of *Lecture Notes in Computer Science*, pages 571–588. Springer, 2022. 6

[41] Yibing Song, Chao Ma, Xiaohe Wu, Lijun Gong, Linchao Bao, Wangmeng Zuo, Chunhua Shen, Rynson W. H. Lau, and Ming-Hsuan Yang. VITAL: visual tracking via adversarial learning. In *CVPR*, pages 8990–8999. Computer Vision Foundation / IEEE Computer Society, 2018. 1

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, pages 5998–6008, 2017. 1, 2

[43] Paul Voigtlaender, Jonathon Luiten, Philip H. S. Torr, and Bastian Leibe. Siam R-CNN: visual tracking by re-detection. In *CVPR*, pages 6577–6587. Computer Vision Foundation / IEEE, 2020. 6

[44] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, pages 1571–1580. Computer Vision Foundation / IEEE, 2021. 2, 6

[45] Qiang Wang, Zhu Teng, Junliang Xing, Jin Gao, Weiming Hu, and Stephen J. Maybank. Learning attentions: Residual attentional siamese network for high performance online visual tracking. In *CVPR*, pages 4854–4863. Computer Vision Foundation / IEEE Computer Society, 2018. 2

[46] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, pages 548–558. IEEE, 2021. 2

[47] Zhongdao Wang, Hengshuang Zhao, Ya-Li Li, Shengjin Wang, Philip H. S. Torr, and Luca Bertinetto. Do different tracking tasks require different appearance models? In *NeurIPS*, pages 726–738, 2021. 1

[48] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. Cvt: Introducing convolutions to vision transformers. In *ICCV*, pages 22–31. IEEE, 2021. 2

[49] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.*, 37(9):1834–1848, 2015. 2, 7

[50] Fei Xie, Chunyu Wang, Guangting Wang, Yue Cao, Wankou Yang, and Wenjun Zeng. Correlation-aware deep tracking. In *CVPR*, pages 8741–8750. IEEE, 2022. 1, 6

[51] Yinda Xu, Zeyu Wang, Zuoxin Li, Yuan Ye, and Gang Yu. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In *AAAI*, pages 12549–12556. AAAI Press, 2020. 2

[52] Bin Yan, Yi Jiang, Peize Sun, Dong Wang, Zehuan Yuan, Ping Luo, and Huchuan Lu. Towards grand unification of object tracking. In *ECCV (21)*, volume 13681 of *Lecture Notes in Computer Science*, pages 733–751. Springer, 2022. 6

[53] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *ICCV*, pages 10428–10437. IEEE, 2021. 2, 5, 6

[54] Botao Ye, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Joint feature learning and relation modeling for tracking: A one-stream framework. In *ECCV (22)*, volume 13682 of *Lecture Notes in Computer Science*, pages 341–357. Springer, 2022. 1, 2, 3, 5, 6, 7

[55] Bin Yu, Ming Tang, Linyu Zheng, Guibo Zhu, Jinqiao Wang, Hao Feng, Xuetao Feng, and Hanqing Lu. High-performance discriminative tracking with transformers. In *ICCV*, pages 9836–9845. IEEE, 2021. 2, 6

[56] Yuechen Yu, Yilei Xiong, Weilin Huang, and Matthew R. Scott. Deformable siamese attention networks for visual object tracking. In *CVPR*, pages 6727–6736. Computer Vision Foundation / IEEE, 2020. 2

[57] Lichao Zhang, Abel Gonzalez-Garcia, Joost van de Weijer, Martin Danelljan, and Fahad Shahbaz Khan. Learning the model update for siamese trackers. In *ICCV*, pages 4009–4018. IEEE, 2019. 1, 2

[58] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *ECCV (21)*, volume 12366 of *Lecture Notes in Computer Science*, pages 771–787. Springer, 2020. 2, 5, 6

[59] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. *CoRR*, abs/1904.07850, 2019. 3

[60] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *ECCV (9)*, volume 11213 of *Lecture Notes in Computer Science*, pages 103–119. Springer, 2018. 1