Correlation-Aware Deep Tracking

Fei Xie †*, Chunyu Wang ‡, Guangting Wang ‡, Yue Cao ‡, Wankou Yang †,Wenjun Zeng ‡
† Southeast University, China
‡ Microsoft Research Asia

jaffe0319@gmail.com, chnuwa@microsoft.com, flylight@mail.ustc.edu.cn
yuecao@microsoft.com, wkyang@seu.edu.cn, wezeng@microsoft.com

Abstract

Robustness and discrimination power are two fundamental requirements in visual object tracking. In most tracking paradigms, we find that the features extracted by the popular Siamese-like networks cannot fully discriminatively model the tracked targets and distractor objects, hindering them from simultaneously meeting these two requirements. While most methods focus on designing robust correlation operations, we propose a novel target-dependent feature network inspired by the self-/cross-attention scheme. In contrast to the Siamese-like feature extraction, our network deeply embeds cross-image feature correlation in multiple layers of the feature network. By extensively matching the features of the two images through multiple layers, it is able to suppress non-target features, resulting in instance-varying feature extraction. The output features of the search image can be directly used for predicting target locations without extra correlation step. Moreover, our model can be flexibly pre-trained on abundant unpaired images, leading to notably faster convergence than the existing methods. Extensive experiments show our method achieves the state-of-the-art results while running at real-time. Our feature networks also can be applied to existing tracking pipelines seamlessly to raise the tracking performance.

1. Introduction

Visual object tracking (VOT) is a long-standing topic in computer vision. There are two fundamental yet competing goals in VOT: on one hand, it needs to recognize the target undergoing large appearance variations; on the other hand, it needs to filter out the distractors in the background which may be very similar to the target.

Most appearance-based approaches address this challenge in two perspectives: the first is to learn a more expressive feature embedding space by Siamese-like extraction network [22,58]; the second is to develop a more robust correlation operation, such as Siamese cropping [23,58], online filter learning [3,18] and Transformer-based fusion [5,50]. Since the modern backbones [17,34] become the mainstream choice in deep era, most trackers devote to the correlation operation, hoping to discriminate targets from distractors given their features. Despite their great success, few of these tracking paradigms notice that the two competing goals may put the feature network into a target-distractor dilemma, bringing much difficulties to the correlation step. The underlying reasons are three folds: 1) The Siamese encoding process is unaware of the template and search images, which weakens the instance-level discrimination of learned embeddings. 2) There is no explicit modelling for the backbone to learn the decision boundary that separates the two competing goals, leading to a sub-optimal embedding space. 3) Each training video only annotates one single object while arbitrary objects including distractors can be tracked during inference. This gap is further enlarged by 2). Our key insight is that feature extraction should have dynamic instance-varying behaviors to generate "appropriate"

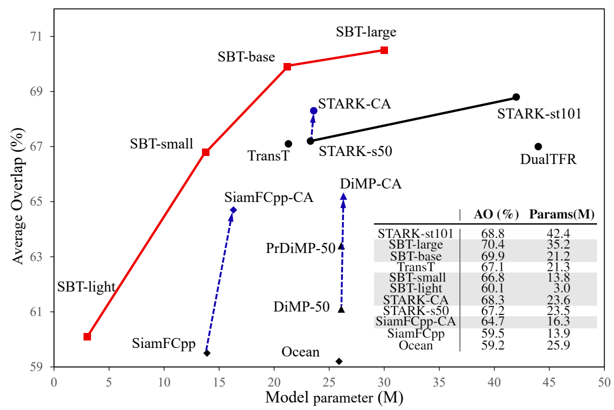


Figure 1. Comparison with the state-of-the-arts on GOT-10k [19]. We visualize the AO performance with respect to the model size. All reported trackers follow the official GOT-10k test protocol. Our SBT tracker achieves superior results while multiple trackers (with suffix "CA") can benefit from our correlation-aware features.

*Work performed when Fei Xie was an intern of MSRA

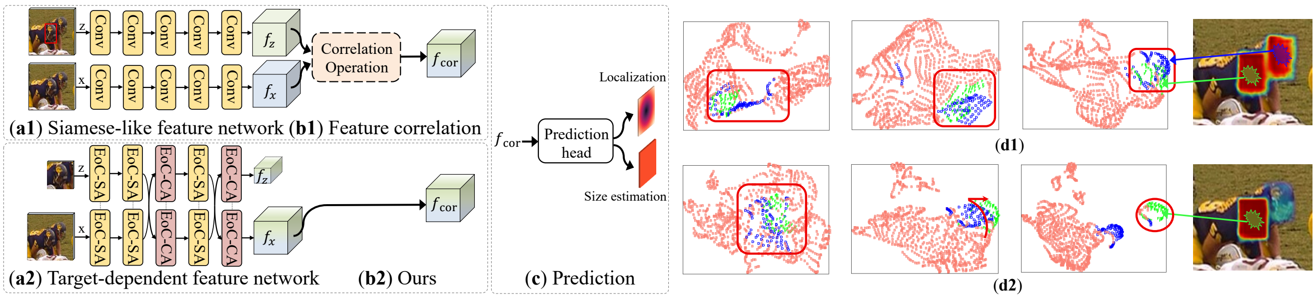


Figure 2. **(a1)** standard Siamese-like feature extraction; **(a2)** our target-dependent feature extraction; **(b1)** correlation step, such as Siamese cropping correlation [23], DCF [11] and Transformer-based correlation [5] ; **(b2)** our pipeline removes separated correlation step; **(c)** prediction stage; **(d1)/(d2)** are the TSNE [36] visualizations of search features in **(a1)/(a2)** when feature networks go deeper.

embeddings for VOT to ease the dilemma. In more details, it needs to generate *coherent* features for the same object in all frames of a video in spite of the variations; on the other hand, it needs to generate *contrasting* features for the target and distractors with similar appearance.

To this end, we present a novel dynamic feature network on top of the attention scheme [37]. As shown in Fig.2 (a2), our Single Branch Transformer (SBT) network allows the features of the two images to deeply interact with each other at the stage of feature extraction. Intuitively, the cross-attention weights gradually filter out target-irrelevant features layer by layer while the self-attention weights enrich the feature representations for better matching. Thus, the feature extraction process is target-dependent and asymmetrical for image pair, allowing the network to achieve a win-win scenario: it differentiates the target from similar distractors while preserving the coherent characteristics among dissimilar targets. The effectiveness of features from SBT is validated in Fig. 2 (d2). The features belonging to the target (green) become more and more separated from the background (pink) and distractors (blue) while the search features from Siamese extraction are totally target-unaware.

The overall framework of SBT is shown in Fig. 3. It has three model stages on top of Extract-or-Correlation (EoC) blocks. The patch embedding produces embeddings for the template and search images. Then the embeddings are fed to the stacked EoC blocks. There are two variants of EoC, *i.e*. EoC-SA and EoC-CA, which use Self-Attention (SA) and Cross-Attention (CA) as its core operator, respectively. The EoC-SA block fuses features within the same image while the EoC-CA block mixes features across images. The output features of the search image are directly fed to the prediction heads to obtain a spatial score map and a size embedding map. Our key technical innovation is introducing one single stream for template and search image pair processing that jointly extract or correlate through homogeneous attention-based blocks. Thus, SBT can be pre-trained on abundant unpaired images such as ImageNet [33], leading to a fast convergence in the fine-tune on tracking.

Extensive experiments are conducted to compare different SBT network designs. Based on the insights, we summarize a number of general principals. Our method achieves superior performance and improves Siamese, DCF and Transformer-based trackers as can be seen in Fig. 1. The main contributions of this work are as follows:

- We present a novel tracking framework which allows the features of the search and template image to be deeply fused for tracking. It further improves existing popular tracking pipelines. To our best, we are the first to propose a specialized target-dependent feature network for VOT.
- We conduct a systematic study on SBT tracking both experimentally and theoretically, and summarize several general principles for following works.

The rest of the paper is organized as follows. We discuss related work in Sec. 2. The SBT framework is presented in Sec. 3. Then, we conduct empirical studies and theoretical analysis on SBT in Sec. 4 and Sec. 5, respectively. Finally, we provide extensive experimental results in Sec. 6 and conclude the paper in Sec. 7.

2. Related Work

Visual Tracking. The Siamese network [2] based trackers have drawn great attention in recent years. By introducing the powerful backbones [22, 58] and elaborated prediction networks [16, 23, 49], Siamese trackers obtain superior performance. However, the offline target matching with a shallow correlation structure [2] lacks of discriminative power towards distractors. Then, the dedicated modifications rise, including attention mechanism [15, 41, 54], online module [59, 61], cascaded frameworks [7, 14, 39], update mechanism [55] and target-aware model fine-tuning [24, 38]. Despite the improvements, most of them bring much complexity to the Siamese tracking pipeline. Instead, our target-dependent feature network can upgrade the original network seamlessly. Moreover, our feature network formulates a novel and conceptually simple tracking pipeline by removing the separated correlation step in Siamese trackers.

Discriminative Correlation Filter (DCF) tracker [18] learns a target model by solving least-squares based regression online. It is further improved by fast gradient algorithm [11], end-to-end learning [3, 60] and CNN-based size

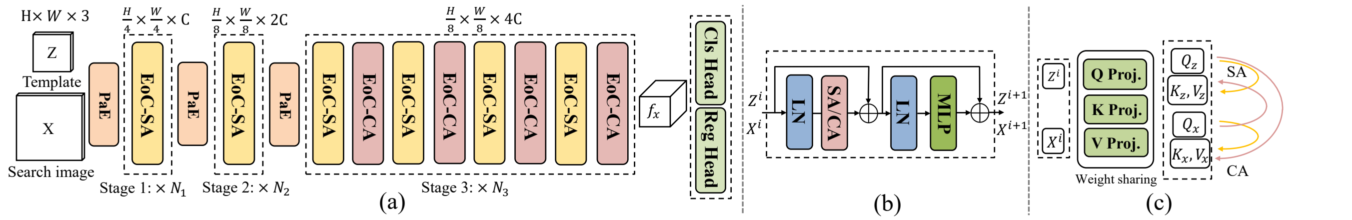


Figure 3. (a) architecture of our proposed Single Branch Transformer for tracking. Different from Siamese, DCF and Transformer-based methods, it does not have a standalone module for computing correlation. Instead, it embeds correlation in all Cross-Attention layers which exist at different levels of the networks. The fully fused features of the search image are directly fed to Classification Head (Cls Head) and Regression Head (Reg Head) to obtain localization and size embedding maps. (b) shows the structure of a Extract-or-Correlation (EoC) block. (c) shows the difference of EoC-SA and EoC-CA. PaE denotes patch embedding. LN denotes layer normalization.

estimation [1, 52]. However, DCF is highly sensitive to the complex handcrafted optimization, as well as the quality of features which may lack of instance-level discrimination under challenging scenarios. To improve this, our discriminative target-dependent features can greatly lighten the burden for the online DCF.

Recent rising Transformer-based methods [5, 40, 46, 50, 53] exploit the long-range modelling of Transformer to effectively fuse the features. Thus, they can track robustly without online learning. However, the Transformer [37] mainly designed for language processing domain is difficult to be initialized properly for vision tasks during training, resulting in enormous costs. Instead of using Transformer as fusion module [5, 50, 52], we leverage the attention scheme to dynamically generate customized features which establish the hierarchical fine-grained correspondence between target and search area.

Vision Backbone. Modern CNNs [17, 34] generally serve as the backbone network in vision tasks. Recently, Vision Transformer (ViT) [12, 26, 43], guided by the principles from CNN, achieves impressive results as vision backbone. Deeper and more effective architectures are the two pillars of powerful backbones, which boost numerous downstream tasks. Similarly, the improvements brought by powerful backbone in VOT mainly attribute to the more expressive feature embedding [22, 58], which has subtle differences to other tasks, *e.g*. object detection. However, the dynamic nature of VOT actually requires asymmetrical encoding for template and search image, which has not been given sufficient attention in most prior works. By considering that, we propose a dynamic instance-varying backbone for VOT, beyond only pursuing an expressive embedding.

3. Architecture

This section introduces the overall architecture of our Single Branch Transformer (SBT) (Fig. 3) as well as its main building block (EoC block). Then, in the next section, we evaluate a number of instantiations of the architecture followed by a summary of favorable design principals.

3.1. Patch Embedding

Our model takes two images as input, comprising a template image $z \in \mathbb{R}^{3\times H_z \times W_z}$ and a larger search image $x \in \mathbb{R}^{3\times H_x \times W_x}$. In general, z is centered on the target object while x represents a larger region in the subsequent frame which contains the target. In the Patch Embedding (Pa.E) stage, the two images are fed to a convolutional layer φ_p^0 with kernel size 7×7 and stride 4, followed by a layer normalization (LN) layer. It embeds the images into feature maps of f_z^0 and f_x^0, respectively.

$$f_z^0, f_x^0 = \mathrm{LN}(\varphi_p^0(z)), \mathrm{LN}(\varphi_p^0(x)), \tag{1}$$

where $f_z^0 \in \mathbb{R}^{C_0 \times \frac{H_z}{4} \times \frac{W_z}{4}}, f_x^0 \in \mathbb{R}^{C_0 \times \frac{H_x}{4} \times \frac{W_x}{4}}$ and C_0 is the number of channels.

3.2. Extract-or-Correlation Block

EoC block which can simultaneously implement Self-Attention (SA) and Cross-Attention (CA) is the main building block. Intuitively, they gradually fuse features from the same and different images, respectively. It is known that computing attention globally among all tokens leads to quadratic complexity [26]. To address this, there are a number of works which attempt to reduce the computation cost. We present a general formulation for different efficient attention methods. On top of the formulation, we describe our SA and CA operations.

Let $\chi_{(\cdot)}$ denote a function that reshapes/arranges a feature maps into the desired form. The function varies for different methods. We compute the q, k, v features as:

$$\begin{aligned} q_i &= [\chi_q(f_i)]^\mathsf{T}\omega_q, \quad i \in \{z, x\}, \\ k_i &= [\chi_k(f_i)]^\mathsf{T}\omega_k, \quad i \in \{z, x\}, \\ v_i &= [\chi_v(f_i)]^\mathsf{T}\omega_v, \quad i \in \{z, x\}, \end{aligned} \tag{2}$$

where $\{\omega_q, \omega_k, \omega_v\}$ represent linear projections.

The Vanilla Global attention (VG) [12] computes attention among all tokens. So $\{\chi_q, \chi_k, \chi_v\}$ represent identity mapping. The Spatial-Reduction Global attention (SRG) [43, 56] uses a convolution with a stride larger than one (*i.e*. $\{\chi_k, \chi_v\}$) to reduce the spatial resolution of the

Table 1. The left part compares different factors of SBT including attention computation methods (ATTN), position encoding methods (PE), patch embedding methods (PaE), number of model parameters and flops. The right part compares the rest of factors based on A_5 (described in the left part) such as the feature dimensions (DIM) and the number of blocks (BLK), as well as the stride of the feature maps in each stage. All models unless explained follow the same setting: training from scratch, interleaved EoC-SA/EoC-CA block in the third stage, 128×128 for template image and 256×256 search image.

Setting	A_1[1]	A_2[2]	A_3	A_4	A_5	A_6	A_7	Setting	B_1	B_2	B_3	B_4	B_5	B_6	B_7	B_8
Refer to	[12]	[26]	[56]	[43]	[43]	[43]	[8]	**DIM(1,2)**	[64, 128]	[64, 128]	[64, 128]	[64, 128]	[64, 128]	[64, 128]	[64, 128]	[32, 64]
ATTN	VG	SL	SRG	SRG	SRG	SRG	VL/SRG	**DIM(3,4)**	[320]	[320,512]	[320,512]	[512]	[320]	[320,512]	[320]	[320]
PE	Abs	Rel	Cond	Cond	Cond	Rel	Cond	**BLK**	[3,4,10]	[4,2,6,1]	[2,2,6,2]	[2,2,4]	[3,4,10]	[2,4,6,1]	[3,4,12]	[3,4,10]
PaE	H_1[3]	H_2[3]	Conv	H_2[3]	Conv	Conv	Conv	**STR**	[4,2,1]	[4,2,1,1]	[4,2,1,1]	[4,2,1]	[4,1,2]	[4,2,1,1]	[4,2,2][4]	[4,2,1]
Param.(M)	22.5	40.2	23.9	20.1	21.3	21.0	19.6	**Param.(M)**	21.3	18.6	21.1	20.5	20.8	19.3	20.8	15.1
Flops(G)	35.1	36.5	20.2	18.9	19.6	19.3	17.5	**Flops(G)**	19.6	19.3	22.5	19.2	24.4	24.7	12.1	14.5
AO	47.5	56.4	63.7	61.7	63.5	63.1	60.1	**AO**	63.5	57.4	60.9	56.7	63.3	60.6	52.2	56.2

[1] A_1 does not have hierarchical structure, so we adopt 4 downsampling ratio at the beginning and drops the classification token.
[2] For A_2, we set the same image size (224×224) for template and search image for simplicity.
[3] H_1 denotes the A_1 splits an input image into non-overlapping patches (4×4). H_2 denotes a linear layer to change dimensions after patch split.
[4] For model settings with total network stride 16, we increase the search image size to 320×320 for a fair comparison.

key and value features. The resolution of the query features is not changed. Then it computes global attention as VG. The method largely reduces the computational overhead. The Vanilla Local window attention (VL) [8] splits feature tokens in groups based on their spatial locations and only computes attention within each group. Swin Transformer [26] further adds a Shift window mechanism to vanilla Local attention (SL) for global modelling.

Since the target object may appear anywhere in the search image, it is not practical to use local attention methods for CA. In our work, we use SRG to implement SA and CA. More discussions are in Sec. 4. The following equation shows how we compute SA or CA:

$$\tilde{f}_{ij} = \mathrm{Softmax}(\frac{q_i k_j^\top}{\sqrt{d_h}}) v_j, \quad i, j \in \{z, x\}, \tag{3}$$

In SA, i and j are from the same source (either z or x) and the resulting feature update is:

$$f_z := f_z + \tilde{f}_{zz}, \quad f_x := f_x + \tilde{f}_{xx}, \tag{4}$$

In CA, it mixes the features from different sources:

$$f_z := f_z + \tilde{f}_{zx}, \quad f_x := f_x + \tilde{f}_{xz}. \tag{5}$$

We can see that the correlation between the two images is deeply embedded in feature extraction seamlessly. EoC block also consists of two LN layers and a 2-layer MLP as shown in Fig 3 (b).

3.3. Position Encoding

For majority methods [4, 12, 26], the encoding is generated by the sinusoidal functions with Absolute coordinates (Abs) or Relative distances (Rel) between tokens. Being much simpler, Conditional positional encoding [9, 43, 56] (Cond) generates dynamic encoding by convolutional layers. In our model, we add a 3×3 depth-wise convolutional layer φ_{pe} to MLP before GELU as conditional PE.

3.4. Direct Prediction

Different from the existing tracking methods, we directly add a classification head Φ_{cls} and regression head Φ_{reg} on top of the search feature $\hat{f}_x$ from SBT Ω without additional correlation operations:

$$\hat{f}_x = \Omega(z, x), \quad y_{reg} = \Phi_{reg}(\hat{f}_x), \quad y_{cls} = \Phi_{cls}(\hat{f}_x), \tag{6}$$

where y_{reg}, y_{cls} denote the target regression and classification results to estimate the location and shape of the target.

We implement Φ_{reg} and Φ_{cls} by stacking multiple Mix-MLP Blocks (MMB) which can jointly model the dependency between the spatial and channel dimensions of the input features $\hat{f}^{i-1}$ in the i^{th} MMB:

$$\hat{f}^i = \varphi_{sp}(\mathrm{RS}(\varphi_{cn}(\mathrm{RS}(\hat{f}^{i-1})))), \tag{7}$$

where φ_{sp} and φ_{cn} consist of a linear layer followed by RELU activation. RS represents reshape. φ_{cn} is applied to features along the channel dimension, and the weights are shared for all spatial locations. In contrast, the operator φ_{sp} is shared for all channels.

4. Empirical Study of SBT Instantiations

In this section[1], we conduct empirical studies on SBT variants by raising a number of questions.

As efficient attention computing is vital to the SBT, we firstly ablate other network factors including hierarchical structure, position encoding and patch embedding. As shown in Tab. 1, it is obvious that *hierarchical structure performs much better than singe stage because of multi-scale representation* (A_1 Vs. A_2 to A_7). Conditional PE only surpasses the relative PE by 0.4 points (A_5 Vs. A_6). The difference between PE methods is rather small, indicating that *PE does not have key impacts on performance*. We also

[1] All the experiments follow the official GOT-10k [19] test protocol.

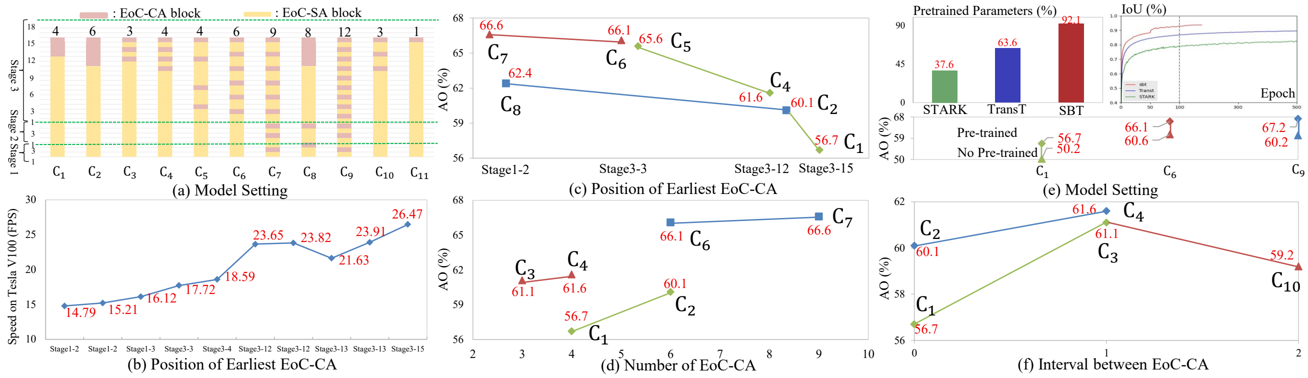


Figure 4. Studies on the number/position of EoC-CA block. (a): different model settings, (b) speed Vs. different model settings,(c): tracking performance Vs. position of earliest EoC-CA block, (d): tracking performance Vs. number of EoC-CA block, (e): tracking performance Vs. pre-trained or not, (f): tracking performance Vs. intervals between EoC-CA block.

find that *convolutional PaE is more practical and expressive than hand-crafted patch merging* (A_4 Vs. A_5).

Which attention computation is better for SBT tracker? The main difference between attention computation lies in the operation to reduce complexity (global/local attention). We find that local attention (VL/SL) block cannot directly perform Cross-Attention as the inequality of local windows in template and search image. Thus, for SBT constructed by pure local attention blocks, we adopt same image size (224×224) for template/search image (A_2) to avoid tedious hand-crafted cross strategies. Comparing to the settings with global attention block (VG/SRG) (A_3 to A_7) which adopt 128×128 as template size, the performance of pure local attention (A_2) drop at least 3.6 points in *AO* with more parameters and flops. *This is mainly due to the negative impacts of over background information in template which may confuse the search branch.* We also investigate the mix setting of SRG and VL block (A_7). To be specific, the VL block is for Self-Attention while SRG block is for Cross-Attention. We observe that the pure SRG block design achieves the better performance (A_5 Vs. A_7). This illustrates that *SBT benefits from unified block choice.* A_3 also validates the effectiveness of pure SRG blocks with 63.7% in *AO*. We conclude that *pure SRG block is more practical and efficient for SBT tracker.*

Do earlier and more EoC-CA blocks help to tracking better? With a baseline designed from above principles, it strikes to us that SBT may benefit from earlier and more cross correlation. We ablate different position/number of EoC-CA block in Fig. 4. As shown in Fig. 4 (d), when the number of EoC-CA blocks increases, the performance of model rises consistently with the same EoC-SA/EoC-CA position pattern (C_3 vs. C_4, C_1 vs. C_2, C_6 vs. C_9). It proves that *SBT tracker benefits from more comprehensive Cross-Attention between template and search branch.* In Fig. 4 (d), when the number of EoC-CA block is the same, earlier cross design has significant positive impacts (C_4 surpasses C_1 by 4.9 points, C_6 surpasses C_2 by 6.5 points). The underlying reason is that *early-cross generates target-dependent features which help tracker to see better.*

Is tracking performance related to the placement pattern of EoC-CA blocks? As the position and number of EoC-CA block have significant impacts on performance, it comes to us which pattern of placement is the optimal choice. So we make attempts to place EoC-SA/EoC-CA block differently. In Fig. 4 (f), we surprisingly find that interleaved EoC-SA/EoC-CA pipeline performs better than the separation pattern even with less Cross-Attention and latter earliest cross position (C_3 vs. C_1). The potential cause is that *EoC-SA block can refine the template/search feature after the correlation, resulting in a more expressive feature space for matching.* In Fig. 4 (f), model (C_9) achieves the best performance 67.2% when the interval is 1. When the interval increases to 2, the performances drops from 61.1% to 59.2% (C_3 vs. C_{10}). Thus, *we prefer an interleaved EoC-SA/EoC-CA block design for SBT tracker.*

What is the optimal network variants for tracking model? Then, it comes to us a long-standing problem for designing a deep tracker. We ablate different network stride, model stage and model size. As shown in Tab. 1, *over parameters and flops in shallow level (stage 1 and 2) is harmful.* It is mainly because the low dimension cannot formulate informative representations (57.4 of B_2 Vs. 60.6 of B_6). We also observe that increasing the head number slightly improves the performance but decreases the speed. *With the same total network stride, three-stage model performs better than four-stage model* (63.5 of B_1 Vs. 57.4 of B_2) with comparable parameters and flops. Though setting the network stride to 16 can reduce the flops, the performance drops 11.3 points (B_1 Vs. B_7), indicating that *SBT tracker prefers larger spatial size of features.* As the channel dimensions influence model size a lot, *it is vital to achieve a balance between block numbers and channel dimensions* (56.7 of B_4 Vs. 63.3 of B_5).

Does flexible design of EoC-SA/EoC-CA bring negative/positive effects? We examine the potential negative/-

positive effects in SBT. From Fig. 4 (b) and Fig. 4 (c), we observe that early-cross in shallow-level (stage 1 and 2) does not bring much improvements (C_2 vs. C_8, C_6 vs. C_7) but lowers the inference speed. It is because the early-cross destroys the one-shot inference. The shallow-level EoC-SA blocks perform as buffer. *So the trade-off between early-cross and speed should be well-considered.* In Fig. 4 (e), *SBT tracker benefits from more pre-trained weights and converges much faster than Transformer-based trackers,* such as TransT [5] and STARK [50].

	Light	Small	Base	Large
PaE	$\mathrm{Conv}(7, 32, 4)$	$\mathrm{Conv}(7, 64, 4)$	$\mathrm{Conv}(7, 64, 4)$	$\mathrm{Conv}(7, 64, 4)$
Stage1	$\begin{bmatrix}\text{EoCA\_1\_8}\\\text{MLP\_32}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_1\_8}\\\text{MLP\_64}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_1\_8}\\\text{MLP\_64}\end{bmatrix} \times 3$	$\begin{bmatrix}\text{EoCA\_1\_8}\\\text{MLP\_64}\end{bmatrix} \times 3$
PaE	$\mathrm{Conv}(3, 64, 2)$	$\mathrm{Conv}(3, 128, 2)$	$\mathrm{Conv}(3, 128, 2)$	$\mathrm{Conv}(3, 128, 2)$
Stage2	$\begin{bmatrix}\text{EoCA\_2\_4}\\\text{MLP\_64}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_2\_4}\\\text{MLP\_128}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_2\_4}\\\text{MLP\_128}\end{bmatrix} \times 4$	$\begin{bmatrix}\text{EoCA\_2\_4}\\\text{MLP\_128}\end{bmatrix} \times 4$
PaE	$\mathrm{Conv}(3, 160, 1)$	$\mathrm{Conv}(3, 320, 1)$	$\mathrm{Conv}(3, 320, 1)$	$\mathrm{Conv}(3, 320, 1)$
Stage3	$\begin{bmatrix}\text{EoCA\_5\_2}\\\text{MLP\_160}\end{bmatrix} \times 6$	$\begin{bmatrix}\text{EoCA\_5\_2}\\\text{MLP\_320}\end{bmatrix} \times 6$	$\begin{bmatrix}\text{EoCA\_5\_2}\\\text{MLP\_320}\end{bmatrix} \times 10$	$\begin{bmatrix}\text{EoCA\_5\_2}\\\text{MLP\_320}\end{bmatrix} \times 18$
PaE	$\mathrm{Conv}(3, 256, 2)$	$\mathrm{Conv}(3, 512, 2)$	$\mathrm{Conv}(3, 512, 2)$	$\mathrm{Conv}(3, 512, 2)$
stage4	$\begin{bmatrix}\text{EoCA\_8\_1}\\\text{MLP\_256}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_8\_1}\\\text{MLP\_512}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_8\_1}\\\text{MLP\_512}\end{bmatrix} \times 2$	$\begin{bmatrix}\text{EoCA\_8\_1}\\\text{MLP\_512}\end{bmatrix} \times 2$
Head	Classification: MMB $\times$ 2 Regression: MMB $\times$ 2			
EoC-CA	[2, 4, 6]	[2, 4, 6]	[2, 4, 6, 8, 10]	[6, 8, 10, 12, 14, 16, 18]
Params	3.03 M	13.80 M	21.27 M	35.20 M
FLOPs	3.81 G	11.92 G	19.27 G	31.46 G
Speed	62 FPS	50 FPS	37 FPS	24 FPS

Table 2. Model settings of SBT in four scales. "$\mathrm{Conv}(k, c, s)$" means convolution layers with kernel size k, output channel c and stride s. "$\mathrm{MLP\_}c$" is the MLP with hidden channel $4c$ and output channel c. "$\mathrm{EoCA\_}n\_r$" is the EoC attention computation with the number of heads n and down-sampling ratio r. EoC-CA blocks are in the third stage. We report the speed in single Tesla V100 GPU.

5. Single Branch Transformer Driven Tracking

Beyond exploring SBT experimentally, we theoretically analyze SBT from the perspective of general VOT. Then, we design four versions of SBT and integrate them into typical trackers to show generality.

5.1. Theoretical Analysis on SBT for Tracking

SBT overcomes the intrinsic restrictions in deep trackers. Deep trackers have an intrinsic requirement for strict translation invariance, $f\left(c, x\left[\Delta\tau_j\right]\right) = f(c, x)\left[\Delta\tau_j\right]$ where $\Delta\tau_j$ is the shift window operator, c denotes the template/online filter in Siamese/DCF tracking and f denotes correlation operation. Modern backbones [17, 48] can provide more expressive features while their padding is inevitable to destroy the translation invariance. Thus, deep trackers [22, 58] crop out padding-affected features and adopt spatial-aware sampling training strategy to keep the translation invariance. Theoretically, padding in SBT can be removed completely or only exists in patch embedding for easy implementation. Moreover, the flattened feature tokens has permutation invariance which makes EoC block completely translation invariance. As the EoC block provides global receptive field, SBT can enjoy arbitrary size of template/search image and larger search area scale. Thus, we argue that SBT driven tracking can overcome the intrinsic restrictions in classical deep trackers theoretically by using brand-new network modules.

Cross-Attention is more than twice as effective as depth-wise correlation. We first prove that Cross-Attention can be decomposed into dynamic convolutions (D-Conv). CA which performs as feature correlation is mathematically equivalent to two D-Convs and a SoftMax layer. For simplicity, we annotate the encoded $\{q, k, v\}$ features to their original feature as the projection matrix is 1×1 position-wise convolutional filters. So the CA for query from search feature x to template feature z is:

$$\begin{aligned}
&\mathrm{Inter} = \mathrm{RS}(z)^T x + \mathbf{0} = \boldsymbol{W}_1(z)^T x + \boldsymbol{b}_\mathbf{1},\\
&\mathrm{Attn}_{xz} = \mathrm{Softmax}(\mathrm{Inter}),\\
&\tilde{f}_{xz} = \mathrm{Attn_{xz}} z + x = \boldsymbol{W}_\mathbf{2}(z, x)^T x + \boldsymbol{b}_2(x),
\end{aligned} \tag{8}$$

where $\boldsymbol{W}(a, b), \boldsymbol{b}(a, b)$ is the weight matrix and bias vector of dynamic filters generated by $\{a, b\}$ and RS denotes reshape. To obtain the correlation feature $\tilde{f}_{xz}$, the search feature x goes through a D-Conv layer generated by z, a SoftMax layer and another D-Conv layer generated by z and x. Two D-Conv layers come from the reshape of z along channel and spatial dimension. The depth-wise correlation (DW-Corr) or pixel-wise correlation (Pix-Corr) [51] is only equivalent to one D-Conv layer. Thus, CA is twice as effective as DW-Corr or Pix-Corr with the same template feature as dynamic parameters.

Hierarchical feature utilization is embedded in serial pipeline. Siamese trackers [6, 22] perform correlation for each hand-selected feature pair and feed them into parallel prediction heads. Then, prediction results are aggregated by a weighted sum. Comparing to the hand-craft layer-wise aggregation, SBT structure explores multi-level feature correlation intrinsically. We take three-level feature utilization as an example:

$$\begin{aligned}
&x_i, z_i = \phi_{ca}^i(\tilde{x}_i, \tilde{z}_i), \quad i \in \{0, 1, 2\}\\
&x_2, z_2 = \phi_{ca}^2(\phi_{ca}^1(\phi_{ca}^0(x_0, z_0))),\\
&S_{sbt} = \varphi^p(x_2),
\end{aligned} \tag{9}$$

where $\{0, 1, 2\}$ represents shallow, intermediate and deep level, $\{\tilde{x}, \tilde{z}\}$ are the previous layer features of $\{x, z\}$, $\{\phi_{ca}, \varphi^p\}$ denote EoC-CA block and prediction head. Though in a serial pipeline, the prediction result S_{sbt} naturally contains hierarchical feature correlation results.

5.2. Four Versions of SBT network

Following the guidelines from Sec. 4, our four versions of SBT is described in Tab. 2. For pre-training, we add extra fourth model stage and modify the network stride as [26]. For fine-tune on tracking, we only use three-stage model and replace the prediction head.

	SiamRPN++ [22]	ATOM [11]	DiMP [3]	SAMN [47]	AutoMatch [57]	Tr Siam [40]	TransT [5]	Stark s50 [50]	Stark st101 [50]	SBT light	SBT small	SBT base	SBT large
$AO\uparrow$	51.8	55.6	61.1	61.5	65.2	66.0	67.1	67.2	**68.8**	60.2	66.8	**69.9**	**70.4**
$SR_{50}\uparrow$	61.6	63.4	71.7	69.7	76.6	76.6	76.8	76.1	**78.1**	68.5	77.3	**80.4**	**80.8**
$SR_{75}\uparrow$	32.5	40.2	49.2	52.2	54.3	57.1	60.9	61.2	**64.1**	53.0	59.2	**63.6**	**64.7**

Table 3. Comparison on the GOT-10k [20] test set.

	SiamRPN++ [22]	ATOM [11]	DiMP [3]	AutoMatch [57]	DualTFR [46]	Tr Siam [40]	TransT [5]	DTT [52]	Stark s50 [50]	SBT light	SBT small	SBT base	SBT large
AUC ↑	49.6	51.5	56.9	58.3	63.5	62.4	64.9	60.1	**65.8**	56.5	61.1	**65.9**	**66.7**
Prec↑	49.1	50.5	56.7	59.9	66.5	60.0	69.0	-	**69.7**	57.1	63.8	**70.0**	**71.1**

Table 4. Comparison on the LaSOT [13] test set.

	Ocean [59]	ATOM [11]	SiamMask [42]	SuperDiMP [1]	STM [30]	DET50 [21]	AlphaRef [51]	Stark s50 [50]	Stark st101 [50]	SBT light	SBT small	SBT base	SBT large
Acc.↑	0.693	0.462	0.624	0.492	0.751	0.679	0.754	**0.761**	**0.763**	0.742	0.750	0.752	**0.753**
Rob.↑	0.754	0.734	0.648	0.745	0.574	0.787	0.777	0.749	**0.789**	0.712	0.775	**0.825**	**0.834**
EAO↑	0.430	0.271	0.321	0.305	0.308	0.441	0.482	0.462	**0.497**	0.415	0.477	**0.515**	**0.529**

Table 5. Results on VOT2020. We use AlphaRefine [51] to generate mask for VOT benchmark.

	DiMP	SiamFC++	DualTFR	TransT	DTT	STARK-s50	SBT-light	SBT-small	SBT-base	SBT-large
AUC ↑	74.0	75.4	80.1	**81.4**	79.6	80.3	68.2	78.2	**81.9**	**82.2**
Norm.Prec↑	80.1	80.0	84.9	**86.7**	85.0	85.1	74.5	83.0	**87.1**	**87.5**

Table 6. Comparison on the TrackingNet test set.

5.3. Correlation-Aware Feature for Other Trackers

We replace the backbone in four typical trackers with SBT, which are named as **Correlation-Aware Trackers**.

6. Experiments

This section describes the implement details, comparisons to the state-of-the-art (sota) trackers and improvements in CATs. Exploration studies are also provided.

6.1. Implementation Details

ImageNet pre-training. We firstly train 4-stage SBT with classification head on the ImageNet [33]. Similar to the network for image classification, our model structure and data flow is one-stream. The setting mostly follows [35] and [43]. We employ the AdamW [27] optimizer for 300 epochs. The input image is resized to 224×224 and the augmentation and regularization strategies of [35] are adopted.

Finetune on tracking. Next, the pre-trained weights are to initialize our tracking model. By arranging EoC-SA/EoC-CA blocks, the model is still one-stream in structure but two-stream in data flow. For each image pair, We compute standard cross-entropy loss for the classification and GIoU [32] loss and L_1 loss for the regression. We use 8 tesla V100 GPUs and set the batch size to be 160. The template and search image size are set to 128×128 and 256×256. The sample pairs of each epoch is $50,000$ and the total epoch is 600. The learning rate is set to be 10^{-4} for the head, and 10^{-5} for the rest and it decays by a factor of 10 at the $200^{th}, 400^{th}$ epoch. The training datasets include the train subsets of LaSOT [13], GOT-10K [19], COCO2017 [25], and TrackingNet [29]. Other settings are the same with [5, 46]. Details are in supplement.

Box-level Tracker	Params(M)	Flops(G)	SR_{50}	SR_{75}	AO
SiamFCpp	13.9	19.8	69.5	47.9	59.5
SiamFCpp-CA	16.3	14.1(5.7↓)	74.8(5.3↑)	54.5(6.6↑)	64.7(5.2↑)
DiMP	26.1	-	71.7	49.2	61.1
DiMP-CA	26.3	-	74.1(2.4↑)	56.8(7.2↑)	65.2(4.1↑)
STARK	23.3	11.5	76.1	61.2	67.2
STARK-CA	23.6	8.7(2.8↓)	77.8(1.7↑)	62.7(1.5↑)	68.3(1.1↑)
Pixel-level Tracker	**Params(M)**	**Flops(G)**	$\mathcal{J}$	$\mathcal{F}$	Mean
STM	24.5	-	69.2	74.0	71.6
STM-CA	25.1	-	72.8(3.6↑)	75.6(1.6↑)	74.2(2.6↑)

Table 7. Improvements of CATs over baselines on GOT-10k [19] and DAVIS17 [31] benchmarks. $\mathcal{J}/\mathcal{F}$ denotes the mean of the region similarity/contour accuracy.

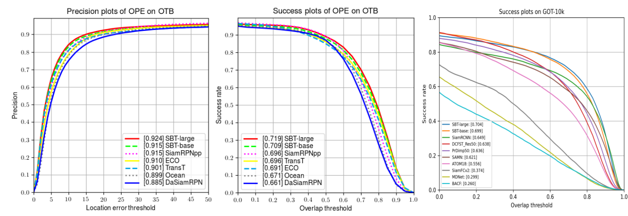


Figure 5. Comparisons on (a) OTB-100 [45] and (b) GOT-10k [19] test set in terms of success plot.

Testing. For SBT/Siamese, we adopt fixed template as [46]. For DCF/STM, the inputs are firstly fused with template by SBT.

6.2. Comparison to State-of-the-Art Trackers

GOT-10K. GOT-10K [19] is a large-scale benchmark which has the zero overlap of object classes between training and testing. We follow the official policy without extra training data. As shown in Tab. 3 and Fig. 5, in a fair comparison scenario, our base and large version outperform other top-performing trackers such as STARK-st101, TransT, TrSiam, and DiMP, verifying the strong generalization to unseen objects. Our light and small version also achieve competitive results with much smaller size.

OTB100/VOT2020/LaSOT. We refer the reader to [13, 21, 45] for detailed descriptions of datasets. In challenging short-term benchmarks (VOT2020 and OTB100), Tab. 5 shows that SBT-small achieves competitive result, which is better than SuperDiMP. After increasing the model variants, SBT-base obtains an EAO of 0.515, being superior to other top-performing trackers. With a much simpler pipeline, our SBT-large is even closed to the winner of VOT2020 challenge RPT [28] (0.530 EAO). Fig. 5 shows our base and large version achieves sota results in OTB. In long-term benchmark LaSOT, with the comparable model size and no online update, SBT-base outperforms the recent strong Transformer-based methods (STARK-s50 and TransT).

6.3. Improvement over Baselines

Box-Level tracking. In Tab. 7, our correlation-aware features improve other tracking pipelines with comparable model size and less computation burden.

Pixel-Level tracking. In multi-object video object segmentation (VOS) benchmark DAVIS17 [31], STM-CA im-

Setting	Fea.Ext	Corr.Emd	Fea.Corr	Low	Mid	High	AO
①	ResNet-50	✗	DW-Cor	S2	S3	S4	56.2
②	ResNet-50	✗	CA	S2	S3	S4	57.5
③	ResNet-50	✗	DCF[1]	-	-	S4	30.3
④	SBT-base	✗	DW-Corr	S3B6	S3B8	S3B10	60.1 (3.9↑)
⑤	SBT-base	✗	CA	S3B6	S3B8	S3B10	61.5 (4.0↑)
⑥	SBT-base	✗	DCF	-	-	S3B10	31.5 (1.2↑)
⑦	SBT-base	✓	-	S3B6	S3B8	S3B10	65.0 (3.5↑/7.5↑)
⑧	SBT-base	✓	DW-Cor	S3B6	S3B8	S3B10	65.9 (4.8↑/9.7↑)
⑨	SBT-base	✓	DCF	S3B6	S3B8	S3B10	35.2 (3.7↑/4.9↑)

Table 8. Ablation studies on GOT-10k [19]. S3B6 denotes the third stage 6^{th} block. Fea.Ext denotes the feature extraction network. Corr.Emd denotes whether network embeds correlation into extraction layers. Fea.Cor denotes the feature correlation method. For DCF, we integrate SBT-base to ECO [10].

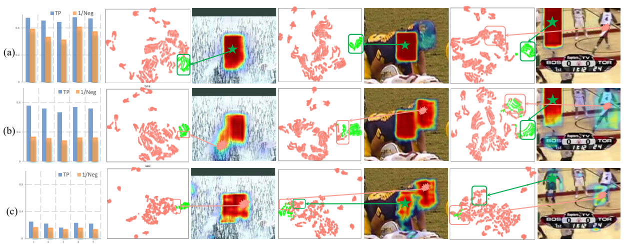


Figure 6. (a), (b), (c) denote three trackers (refer to Sec. 6.4). The first sub-figure indicates the average true positive rate and average negative numbers of negative objects. The other sub-figures denote the T-SNE and classification maps.

proves STM by 3.6% in terms of $\mathcal{J}$, proving that VOS methods can benefit from our discriminative embeddings.

6.4. Exploration Study

We further explore the characteristics of our SBT feature by training it on the GOT-10k [19] training split.

Correlation-embedded structure. As shown in Tab. 8, correlation-embedded SBT (⑦, ⑧, ⑨) significantly improves the tracking performance on all correlation cases (④, ⑤, ⑥). Comparing to the layer-wise aggregation, correlation-embedded trackers outperform the CNN-based trackers or attention-based trackers (65.9% of ⑧ Vs. 60.1% of ④, 65.0% of ⑦ Vs. 61.5% of ⑤, 39.2% of ⑨ Vs. 30.3% of ③). It clearly verifies that SBT structure is more effective on multi-level feature utilization. We also prove that CA works better than DW-Corr in feature correlation (60.1% of ④ Vs. 61.5% of ⑤). Fig. 7 also shows the superiority of correlation-embedded structure.

Target-dependent feature embedding. We further explore the features of three different settings in two folds: one is to maintain spatial location information while another is to classify the target from distractor objects. We begin by training three models with Cls head only to localize the target: (a) Correlation-embedded tracker. (b) Siamese correlation with SBT. (c) Siamese correlation with ResNet-50. We select the five hard videos from OTB [44] benchmark. The search image randomly jitters around target. We only evaluate the Cls map for localization. In Fig. 6, the true positive rate of target ground-truth indicates that (a), (b) can preserve more spatial information than CNN (c). The T-SNE/Cls map also show the target-dependent characteristic of (a) features. The average negative objects (largest connected components) of (a) is higher than (b) which indicates that correlation-embedded is critical to filter out distractors.

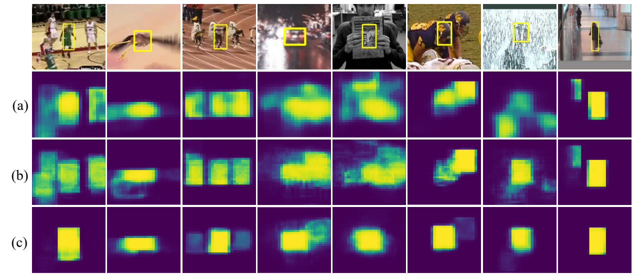
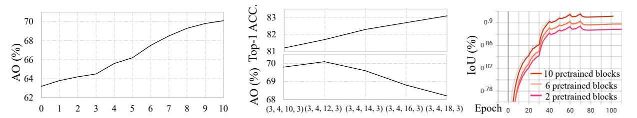


Figure 7. Visualization of classification (Cls) map ons SBT-base tracker with three different settings. (a): layer-wise aggregation with DW-Corr; (b): layer-wise aggregation with EoC-CA block; (c): correlation-embedded.



(a) Pretrained blocks in third stage (b) ImageNet pretrained settings (c) IoU curve during training

Figure 8. Tracking performance of SBT with various pretrained settings. (a) denotes the pretrained block number. In (b), different models are to initialize the tracking model. (c) denotes the IoU curves during training epochs.

Benefits from pre-training. Comparing to the existing trackers [5, 46, 50], our tracking model except prediction heads can be directly benefit from the ImageNet [33] pretrained weights. As shown in Fig. 8 (a), there is a significant correlation between the number of pre-trained blocks and tracking performance. We also investigate the impacts of model variants of SBT. In Fig. 8 (b), the SBT tracking model prefers consistent block numbers for pre-training. We also observe that SBT converges faster and the stabilized IoU value rises with more pre-trained model weights.

7. Conclusion

In this work, we are the first to propose a target-dependent feature network for VOT. Our SBT greatly simplifies the tracking pipeline and converges much faster than recent Transformer-based trackers. Then, we conduct a systematic study on SBT tracking both experimentally and theoretically. Extensive experiments demonstrate that our method achieves sota results and can be applied to other tracking pipelines as dynamic feature network.

Acknowledgment

This work was supported by NSFC (No.61773117 and No.62006041) and the National Key Research and Development Program of China under Grant 2021YFC2802002.

References

[1] https://github.com/visionml/pytracking. 3, 7

[2] Luca Bertinetto, Jack Valmadre, João F Henriques, Andrea Vedaldi, and Philip H S Torr. Fully-convolutional siamese networks for object tracking. In *ECCVW*, 2016. 2

[3] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *ICCV*, 2019. 1, 2, 7

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 4

[5] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *CVPR*, pages 8126–8135, 2021. 1, 2, 3, 6, 7, 8

[6] Zedu Chen, Bineng Zhong, Guorong Li, Shengping Zhang, and Rongrong Ji. Siamese box adaptive network for visual tracking. In *CVPR*, 2020. 6

[7] Siyuan Cheng, Bineng Zhong, Guorong Li, Xin Liu, Zhenjun Tang, Xianxian Li, and Jing Wang. Learning to filter: Siamese relation network for robust tracking. In *CVPR*, pages 4421–4431, 2021. 2

[8] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. *arXiv preprint arXiv:2104.13840*, 2021. 4

[9] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021. 4

[10] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. ECO: Efficient convolution operators for tracking. In *CVPR*, 2017. 8

[11] Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *CVPR*, 2019. 2, 7

[12] 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, 2020. 3, 4

[13] 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*, 2019. 7

[14] Heng Fan and Haibin Ling. Siamese cascaded region proposal networks for real-time visual tracking. In *CVPR*, 2019. 2

[15] Dongyan Guo, Yanyan Shao, Ying Cui, Zhenhua Wang, Liyan Zhang, and Chunhua Shen. Graph attention tracking. In *CVPR*, pages 9543–9552, 2021. 2

[16] Dongyan Guo, Jun Wang, Ying Cui, Zhenhua Wang, and Shengyong Chen. SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In *CVPR*, 2020. 2

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1, 3, 6

[18] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. In *ICVS*, 2008. 1, 2

[19] Lianghua Huang, Xin Zhao, and Kaiqi Huang. GOT-10k: A large high-diversity benchmark for generic object tracking in the wild. *TPAMI*, 2019. 1, 4, 7, 8

[20] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 7

[21] Matej Kristan, Aleš Leonardis, Jiří Matas, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kämäräinen, Martin Danelljan, Luka Čehovin Zajc, Alan Lukežič, Ondrej Drbohlav, et al. The eighth visual object tracking vot2020 challenge results. In *ECCVW*, 2020. 7

[22] 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*, 2019. 1, 2, 3, 6, 7

[23] 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, 2018. 1, 2

[24] Xin Li, Chao Ma, Baoyuan Wu, Zhenyu He, and Ming-Hsuan Yang. Target-aware deep tracking. In *CVPR*, pages 1369–1378, 2019. 2

[25] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014. 7

[26] 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. *arXiv preprint arXiv:2103.14030*, 2021. 3, 4, 6

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 7

[28] Ziang Ma, Linyuan Wang, Haitao Zhang, Wei Lu, and Jun Yin. Rpt: Learning point set representation for siamese visual tracking. *arXiv preprint arXiv:2008.03467*, 2020. 7

[29] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 2018. 7

[30] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. In *ICCV*, 2019. 7

[31] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *CVPR*, pages 724–732, 2016. 7

[32] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, 2019. 7

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, and Michael Bernstein. ImageNet Large scale visual recognition challenge. *IJCV*, 2015. 2, 7, 8

[34] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1, 3

[35] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021. 7

[36] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008. 2

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 2, 3

[38] Guangting Wang, Chong Luo, Xiaoyan Sun, Zhiwei Xiong, and Wenjun Zeng. Tracking by instance detection: A meta-learning approach. In *CVPR*, 2020. 2

[39] Guangting Wang, Chong Luo, Zhiwei Xiong, and Wenjun Zeng. Spm-tracker: Series-parallel matching for real-time visual object tracking. In *CVPR*, 2019. 2

[40] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, pages 1571–1580, 2021. 3, 7

[41] Qiang Wang, Zhu Teng, Junliang Xing, Jin Gao, Weiming Hu, and Stephen Maybank. Learning attentions: residual attentional siamese network for high performance online visual tracking. In *CVPR*, pages 4854–4863, 2018. 2

[42] Qiang Wang, Li Zhang, Luca Bertinetto, Weiming Hu, and Philip H. S. Torr. Fast online object tracking and segmentation: A unifying approach. In *CVPR*, 2019. 7

[43] 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, 2021. 3, 4, 7

[44] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 8

[45] Yi Wu, Jongwoo Lim, and Ming Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 7

[46] Fei Xie, Chunyu Wang, Guangting Wang, Yang Wankou, and Wenjun Zeng. Learning tracking representations via dual-branch fully transformer networks. In *ICCVW*, 2021. 3, 7, 8

[47] Fei Xie, Wankou Yang, Bo Liu, Kaihua Zhang, Guangting Wang, and Wangmeng Zuo. Learning spatio-appearance memory network for high-performance visual tracking. *ICCVW*, 2021. 7

[48] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017. 6

[49] Yinda Xu, Zeyu Wang, Zuoxin Li, Ye Yuan, and Gang Yu. SiamFC++: towards robust and accurate visual tracking with target estimation guidelines. In *AAAI*, 2020. 2

[50] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. *ICCV*, 2021. 1, 3, 6, 7, 8

[51] Bin Yan, Xinyu Zhang, Dong Wang, Huchuan Lu, and Xiaoyun Yang. Alpha-refine: Boosting tracking performance by precise bounding box estimation. *CVPR*, 2021. 6, 7

[52] 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*, 2021. 3, 7

[53] 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 9856–9865, 2021. 3

[54] Yuechen Yu, Yilei Xiong, Weilin Huang, and Matthew R Scott. Deformable siamese attention networks for visual object tracking. In *CVPR*, pages 6728–6737, 2020. 2

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

[56] Qinglong Zhang and Yubin Yang. Rest: An efficient transformer for visual recognition. *arXiv preprint arXiv:2105.13677*, 2021. 3, 4

[57] Zhipeng Zhang, Yihao Liu, Xiao Wang, Bing Li, and Weiming Hu. Learn to match: Automatic matching network design for visual tracking. 2021. 7

[58] Zhipeng Zhang and Houwen Peng. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, 2019. 1, 2, 3, 6

[59] Zhipeng Zhang, Houwen Peng, Jianlong Fu, Bing Li, and Weiming Hu. Ocean: Object-aware anchor-free tracking. In *ECCV*, 2020. 2, 7

[60] Linyu Zheng, Ming Tang, Yingying Chen, Jinqiao Wang, and Hanqing Lu. Learning feature embeddings for discriminant model based tracking. In *ECCV*, 2020. 2

[61] Jinghao Zhou, Peng Wang, and Haoyang Sun. Discriminative and robust online learning for siamese visual tracking. In *AAAI*, volume 34, pages 13017–13024, 2020. 2