Learning Tracking Representations via Dual-Branch Fully Transformer Networks

Fei Xie[1*], Chunyu Wang[2], Guangting Wang[2], Wankou Yang[1], Wenjun Zeng[2]
[1]School of Automation, Southeast University, China
[2]Microsoft Research Asia
jaffe03@seu.edu.cn, chnuwa@microsoft.com
flylight@ustc.edu.cn, wkyang@seu.edu.cn, wezeng@microsoft.com

Abstract

We present a Siamese-like Dual-branch network based on solely Transformers for tracking. Given a template and a search image, we divide them into non-overlapping patches and extract a feature vector for each patch based on its matching results with others within an attention window. For each token, we estimate whether it contains the target object and the corresponding size. The advantage of the approach is that the ***features are learned from matching, and ultimately, for matching****. So the features are aligned with the object tracking task. The method achieves better or comparable results as the best-performing methods which first use CNN to extract features and then use Transformer to fuse them. It outperforms the state-of-the-art methods on the GOT-10k and VOT2020 benchmarks. In addition, the method achieves real-time inference speed (about* 40 *fps) on one GPU. The code and models are released at* https://github.com/phiphiphi31/DualTFR.

1. Introduction

Visual Object Tracking (VOT) is a fundamental problem in computer vision which aims at tracking an object of interest in a video given its bounding box in the first frame [50]. This is generally addressed by looking for the location in the search image whose features have the largest correlation with those in the template image. Introducing of deep Convolutional Neural Network (CNN) has notably boosted the tracking accuracy because of the improved features for matching [2, 4].

The core of VOT is to extract features that are not only robust to appearance variation of the same object in different frames, but also discriminative among different objects. To achieve the target, most of the recent tracking methods [27, 58, 7, 18] manually select "optimal" features from either shallow or deep layers of CNN, or their fusion based on empirical experience [21]. But our experiments show that the features computed in this way are not optimal mainly because CNN is not specifically designed for the matching purpose. Instead, it only looks for the presence of certain image features of the interested classes but does not understand the structural dependency among regions of different objects in the image.

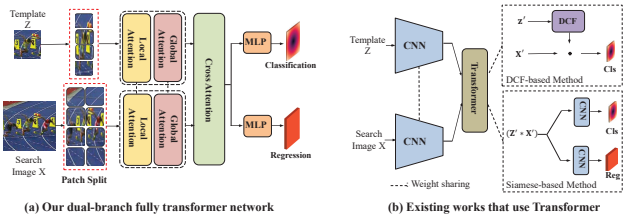

Figure 1: Comparison of our full Transformer method (a) and the existing "CNN+Transformer" based methods (b) which first use CNN to extract features and then fuse them with Transformer networks. Siamese-based [2] and DCF-based [3, 12] methods are two popular pipelines in tracking.

Vision Transformer (ViT) [14], which divides an image into regular tokens, adds positional embedding, and encodes each token based on token-to-token similarities, is a promising approach to extract features for visual object tracking because it is aware of the dependency among all tokens (objects) in all encoding layers. In other words, it extracts features from matching, and for matching, which is consistent with the ultimate task.

Some recent works have already applied Transformer to VOT [37, 47, 7, 54]. But most of them still heavily rely on CNN to extract features and only use Transformer in the last layer to fuse them by global attention. Although they have dramatically boosted the tracking accuracy on benchmark datasets, a natural question arises— can Transformer also benefit the earlier feature extraction step since it can model the structural dependency among different regions? We aim

*Interns at MSRA

to answer this question in this work.

In this work, we present the first study of using pure Transformers to extract features for tracking. To that end, a Siamese-like dual-branch network is proposed as shown in Fig. 1 (a). It divides the template and search images into tokens and extracts a feature for each based on its matching results with others in the same image. This is achieved by mixed efficient local attention blocks and powerful global attention blocks as shown in Fig. 2. In addition, we propose *cross attention blocks* which fuse the tokens between the template and search image. This helps to learn features which are robust to variation in videos. We do not use cross attention in every layer because it is expensive and not necessary— the template and search images are usually from neighboring frames thus having similar patterns which can be captured by the local and global attention blocks. We find using a single cross attention block at the final layer is sufficient in our experiments.

To achieve a good balance between accuracy and speed, we use local attention model on the high-resolution feature maps of most shallow layers, and global attention model only on the low-resolution feature maps as shown in Fig. 2. This notably improves the inference speed (about 40fps on a single 2080Ti GPU). On top of the computed features for each token, we add a MLP layer to estimate whether it is the target (classification head) and the size of the target box in current frame (regression head). Without bells and whistles, this simple approach already outperforms the state-of-the-art methods on multiple tracking benchmarks. We provide extensive ablation studies to validate different factors of the approach. In particular, we find that the use of transformer to extract tracking features is critical to the success of the approach. The main contributions of this work are summarized as follows:

- We present the first attempt to use pure transformer network to compute tracking features, which according to our experiments, is superior to the dominant "CNN+Transformer" based methods.
- We introduce a very simple dual-branch architecture which consists of local attention blocks and global attention blocks in each branch, respectively, and cross attention block to fuse features between the template and search image. The approach achieves a good accuracy/speed trade-off.
- The proposed approach outperforms the state-of-the-art methods, including the "CNN+Transformer" based methods, on multiple tracking benchmarks. In addition, we provide a lot of empirical experience to researcher/engineers in this field with extensive ablation studies.

2. Related Work

2.1. Visual Object Tracking

We classify the state-of-the-art object trackers into two classes. The first class is the Siamese-based methods which generally consist of three steps: CNN-based feature extraction for the template and search images, feature correlation, and a prediction head. For example, SiamFC [2], which is the pioneer work of the series of Siamese methods, directly locates the target object at the position with the largest correlation. SiamFC obtains promising results but it cannot estimate the size of the bounding box. SiamRPN applies a proposal network [28, 41] to the correlation map to find the object location and size which is more powerful than SiamFC. In addition, many works are introduced to improve Siamese trackers such as Feature Pyramid Network [27, 31], deeper backbone [58], anchor-free detection [53, 19] and feature-alignment [59].

The second class of methods are DCF-based [23, 11, 3, 13, 1, 60] which utilize online DCF to classify the target. A response map is generated by computing the correlation between the online DCF and the features in the search region. Current DCF methods are also heavily dependent on CNN-based deep features [4, 12] and linear correlation filter. Our approach belongs to the first class. But different from the previous works, we use fully Transformer networks to extract features.

2.2. Vision Transformer

The success of transformer in natural language processing has drawn wide attention from the computer vision community. The main advantage of ViT [14] over CNN is that the global dependency can be easily captured. A variety of ViTs [61, 44, 33, 10, 57] have been proposed which achieve state-of-the-art performance on many downstream computer vision tasks, such as object detection, semantic segmentation and human keypoint detection. Among them, DeepVit [44] attempts to make the ViT structure go deeper for more powerful representations. PVT [49] adopts pyramid structure like CNNs to better adapt ViT to image tasks. SwinT [33] restricts the self-attention operation within a local window which avoids quadratic complexity. CrossVit [6] proposes a dual-path transformer-based structure to extract multi-scale features for enhanced visual representations.

2.3. Transformer in Object Tracking

Some recent works have already explored to use Transformer in VOT [47, 7, 54]. In general, they still use CNN to extract features for the template and search image and use Transformer to enhance the CNN features which is the main difference from our fully transformer-based method. For example, TransMTracker [47] attempts to enhance features

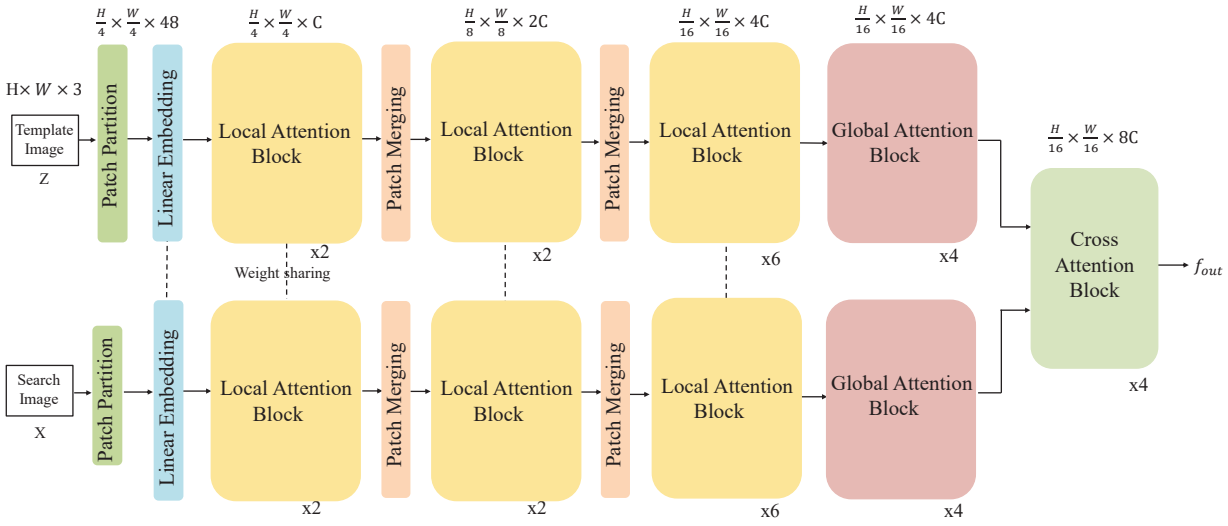

Figure 2: The architecture of our dual-branch fully Transformer-based pipeline (DualTFR).

of the search image by correlating them with the features of multiple historical templates. TransT [7] and Stark [54] enhance features of both the template image and search image based on attention which is more powerful than the standard linear correlation operation in Siamese tracker [2, 29, 27]. The above methods outperform the previous state-of-the-art methods by a notable margin. Our work differs from [47, 7, 54] in that we discard CNN and use pure Transformer to extract features.

3. Dual-branch TransFormeR (DualTFR)

This section presents the technical details of DualTFR. Section 3.1 first gives an overview. Then we dive into the details of DualTFR including local/global attention and cross attention in section 3.2. In section 3.3, we describe multiple variants of DualTFR .

3.1. Architecture Overview

As in Fig. 2, there are two branches in DualTFR, one for the search image x and the other for the template image z. Both are split into non-overlapping patches of equal size (4×4 pixels), respectively. Each of the patches is treated as a token. In total, there are $\frac{H}{4} \times \frac{W}{4}$ tokens, with each having a 48 dimensional feature vector.

Transformer-based Feature Extraction We first apply a linear projection layer to increase the feature dimension from 48 to C for all tokens. Then the resulting template feature maps $f_z \in \mathbb{R}^{4C \times \frac{H_z}{16} \times \frac{W_z}{16}}$ and search feature maps $f_x \in \mathbb{R}^{4C \times \frac{H_x}{16} \times \frac{W_x}{16}}$ are fed to Local Attention Blocks (LAB). The LAB weights are shared between the two branches. Note that LAB only computes attention within a small window with 7×7 tokens in order to reduce the computation time. A number of LAB and patch merging layers are stacked as shown in Fig. 2.

The patch merging layer is used to decrease the spatial resolution and increase the channel dimension of the feature maps both by a factor of two. The resolutions of the template and search feature maps after the LAB stage are $f_z \in \mathbb{R}^{4C \times \frac{H_z}{16} \times \frac{W_z}{16}}$ and $f_x \in \mathbb{R}^{4C \times \frac{H_x}{16} \times \frac{W_x}{16}}$, respectively. Then the two feature maps are fed to two Global Attention Blocks (GAB), respectively. Different from LAB, GAB computes attention among all tokens of the same image which allows to capture long-range dependency. Finally, they go into the Cross Attention Block (CAB) which computes attention among tokens from both images. The final resolution of the search feature maps remains the same as input. In practice, we concatenate the output features from the last two layers. So the resolution of the output feature is $f_{\text{out}} \in \mathbb{R}^{8C \times \frac{H_x}{16} \times \frac{W_x}{16}}$. We feed them to the prediction head, which will be described in detail in the subsequent section, to estimate the target location and shape.

Prediction Head Similar to Siamese-based trackers [30, 53], we add a prediction head to the output features f_{out} to estimate whether each token (location) contains the target object and its offset and size. The first is formulated as a binary classification task while the second as a continuous regression task. In particular, the size of the object is represented by normalized width and height as in DETR [5]. Both are realized by multi-layer perception (MLP) network which consists of three linear projection layers and ReLU layers, respectively.

3.2. Local, Global and Cross Attention

Multi-Head Attention After the image is split into tokens, we use pure attention operators to extract features fol-

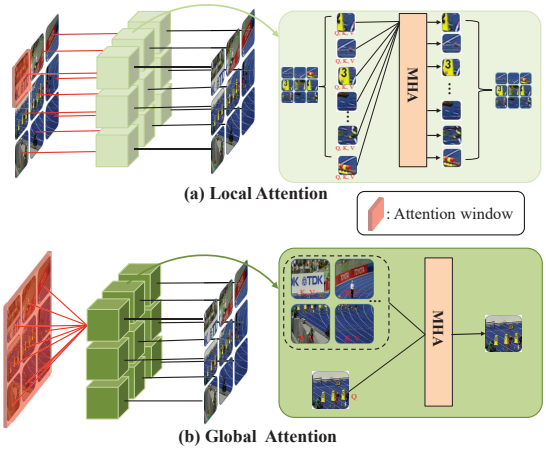

Figure 3: Local Attention and Global Attention. MHA denotes multi-head attention. Attention computation only occurs inside the attention window.

lowing ViT [14]. The Multi-Head Attention (MHA) is the core of the approach so we briefly describe it to make the paper self-contained. MHA takes its input in the form of three parameters, known as Query $\mathbf{Q}$, Key $\mathbf{K}$ and Value $\mathbf{V}$. The three parameters are similar in structure. MHA is computed as:

$$\text{MHA}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\mathbf{H}_1, ..., \mathbf{H}_{n_h})\mathbf{W}^O, \quad (1)$$

$$\mathbf{H}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V), \quad (2)$$

where $\mathbf{W}_i^Q \in \mathbb{R}^{d_m \times d_k}$, $\mathbf{W}_i^K \in \mathbb{R}^{d_m \times d_k}$, $\mathbf{W}_i^V \in \mathbb{R}^{d_m \times d_v}$, and $\mathbf{W}^O \in \mathbb{R}^{n_h d_v \times d_m}$ are learnable parameters. More details about attention can be referred to [45].

Local Attention The main difference between LAB and GAB lies in the size of the window to compute attention. For local attention, we only compute attention for tokens among a small window $M \times M$. In our experiments, M is set to be 7. Suppose an image is split into N non-overlapping local windows and each window has $M \times M$ tokens. Then the computation cost is:

$$FLOP_{Local} = 4NC^2 + 2(M \times M)^2 C, \quad (3)$$

Global Attention Global attention compute attention for all tokens in the same image. It has the capability to model long-range dependency across the whole image. But it also brings heavy computation burden. In specific, the computation cost is:

$$FLOP_{Global} = 4NC^2 + 2M^2 NC, \quad (4)$$

where C is the number of channel dimension. The complexity of global attention is quadratic to the total number of tokens. In contrast, the complexity of the local attention is linear to the total number of tokens. Since the number of tokens M^2 within a small local window is fixed, the whole complexity is linear to the image size. As a result, we only use global attention when the resolution of the feature map is small.

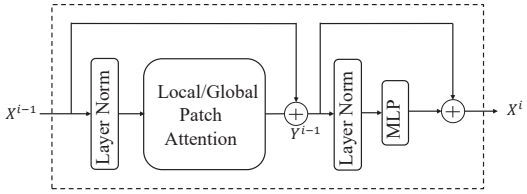

Figure 4: The network structure of a Global or Local Attention Block. MLP denotes Multi-Layer Perception.

Fig. 4 shows the structure of a GAB or LAB. Layer normalization, multi-layer perception (MLP) and residual connection are used as in standard transformers. Mathematically, it is computed as:

$$\begin{aligned} \hat{\mathbf{Y}}^i &= \text{MHSA}\left(\text{LN}\left(\mathbf{X^{i-1}}\right)\right) + \mathbf{X^{i-1}}, \\ \mathbf{X^i} &= \text{MLP}\left(\text{LN}\left(\hat{\mathbf{Y}}^\mathbf{i}\right)\right) + \hat{\mathbf{Y}}^\mathbf{i}, \end{aligned} \quad (5)$$

where $\mathbf{X^i}$ denotes the output from the block i and $\mathbf{Y^i}$ denotes the output from MLP in block i. MLP represents multi-layer perception. MHSA denotes multi-head self attention. In practice, we add the shifted-window mechanism following the [33] for enhanced multi-scale feature representation.

Cross Attention After we compute features for the template and search image, respectively, with LAB and GAB, we propose dual-branch Cross Attention to fuse features between the two images. It is similar to GAB except that we compute attention among tokens from both images. More specifically, in template branch, the template tokens are as key and value and search tokens are as query. Search branch is on the contrary. The operation allows to smooth appearance variations in neighboring frames which notably improves the tracking accuracy according to our experiments. Since we only use Cross Attention when the resolutions of the feature maps are small, the additional computation burden is not significant.

3.3. Architecture Details

In order to strike a good balance between tracking accuracy and speed, we evaluate multiple parameter choices. We choose the following hyper-parameters which achieves 40fps inference speed on a single Nvidia 2080Ti GPU. We set the projection dimension C to be 128. The window size in LAB is 7×7. Each token has 4×4 pixels. The numbers

of layers in LA, GA, CA blocks are $10, 6, 4$, respectively. Details can be seen in Fig. 2.

4. Details, Datasets and Metrics

This section describes implementation details, datasets and metrics, and the results of the state-of-the-art methods. To validate the effectiveness of DualTFR, we all adopt fixed template strategy with no other tricks except for VOT2021 benchmark.

4.1. Implementation Details

Training We train the model in two steps. In the first, we pre-train LAB on the large scale ImageNet-1K [43] dataset in the context of classification. The dataset contains $1.28M$ training images from 1000 classes. Following [33], we employ an AdamW [34] optimizer and train the model for 300 epochs. The batch size is 512 and the learning rate is 10^{-5} with 0.05 weight decay.

Next, we finetune the whole model on the tracking datasets. In particular, for each pair of search/template images from the training dataset, we compute the losses based on the classification and regression outputs from the prediction head. We use standard cross-entropy loss for the classification loss: all pixels within the ground-truth box are regarded as positive samples and the rest are negative. We use GIoU [42] loss and L_1 loss for the regression loss. We load the pre-trained LAB parameters and initialize the rest of the parameters by Xavier [17]. We use 8 tesla V100 GPUs and set the batch size to be 480. The search area factor of template and search image is set to 1.5 and 3, respectively. The total sample pairs of each epoch is 40 million. The learning rate is set to be 10^{-5} for the pre-trained weights, and 10^{-4} for the rest. The learning rate decays by a factor of 10 at the 40_{th} epoch. We finetune the model for 100 epochs.

The training datasets include the train subsets of LaSOT [15], GOT-10K [24], COCO2017 [32], and TrackingNet [38]. All the forbidden sequences defined by the VOT2019 challenge are removed. The pairs of training images in each iteration are sampled from one video sequence. On static images, we also construct an image pair by applying data augmentation like flip, brightness jittering and target center jittering.

Inference Details During inference, the regression head and classification head generate feature maps which contain estimated box shapes and location confidence values. The maximum confidence value and its corresponding bounding box size are chosen to be final prediction result. The template and search image size are set to 112×112 and 224×224, respectively. We also evaluate our approach with two tricks on VOT2021 benchmark. The approach with the first trick is the spatio-temporal version. Inspired by [54], we obtain a global context vector from the search branch feature in the previous frame by global average pooling and add it as a new token to the template token set. The update interval is set to one. The second is the online version, an online correlation filter is added to the model. The response map from online filter is added to the classification map with the weight value of 0.2. Note that the two tricks are complementary to our approach.

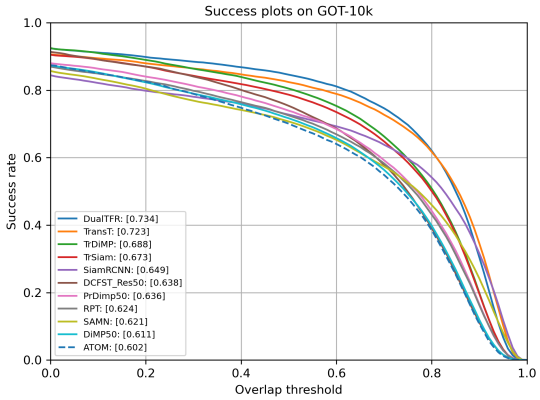

Figure 5: Comparisons on GOT-10k test set.

4.2. Evaluation

We compare DualTFR to the state-of-the-art trackers on five tracking benchmarks. Moreover, we also report results on the recently introduced VOT2021 benchmark.

GOT-10K We only compare to the trackers which use additional training datasets for fair comparison. Results are obtained from the official evaluation server. As shown in Table 1 and Fig. 5, our tracker outperforms all competing trackers in terms of three metrics and achieves the best AO score of 73.5. We also compare to a transformer-based tracker TransT [7], our tracker improves the SR_{75} by 1.8 points while raises the AO by 1.2 points. As for the fully CNN-based Siamese correlation trackers, DualTFR outperforms Ocean [59] by 12.4 points in terms of AO. The results validate the values of using Transformers to extract features.

TrackingNet TrackingNet contains 511 test video sequences. We report the Success (AUC) and Precision (P_{norm}) results in Table 2, DualTFR achieves comparable results with STARK-S50 [54]. Please note that DualTFR adopts 224×224 as search image size which is smaller than 320×320 in STARK. Both network stride of DualTFR and STARK is 16. Thus, we claim that the discriminative ability in smaller image size of DualTFR is more powerful. SiamAttn [56] is a Siamese tracker with attention generated from convolution operation. DualTFR improves SiamAttn by 4.9 points in terms of AUC and 3.2 points in precision.

Trackers		SiamFC++ [53]	SiamRPN++ [27]	ATOM[12]	DiMP-50[3]	D3S[35]	Ocean[59]	SAMN[52]	STARK-S50 [54]	TransT[7]	Ours
GOT-10K	AO↑	59.5	51.8	55.6	61.1	59.7	61.1	61.5	67.2	72.3	73.5
	$SR_{.50}$↑	69.5	32.5	63.4	71.7	67.6	72.1	69.7	76.1	83.7	84.8
	$SR_{.75}$↑	47.9	61.6	40.2	49.2	46.2	47.3	52.2	61.2	68.1	69.9

Table 1: Results on GOT-10K. Top-3 results of each dimension (row) are colored in red, blue and green, respectively.

Trackers		SiamRPN++ [27]	SiamFC++	DiMP50	MAML-FCOS [46]	SAMN [52]	SiamAttn [22]	PrDiMP50[13]	STARK-S50 [54]	Ours
TrackingNet	AUC(%)↑	73.3	75.4	74.0	75.7	74.2	75.2	75.8	80.3	80.1
	P_{norm}(%)↑	80.0	80.0	80.1	82.2	79.4	81.7	81.6	85.1	84.9

Table 2: Results on TrackingNet.

Trackers		MDnet[39]	ECO[11]	ATOM	SiamBAN [8]	SiamCAR[19]	MAML [46]	PrDiMP [26]	SiamFC++	Ocean [59]	TransT [7]	Ours
LaSOT	AUC ↑	39.7	32.4	51.5	51.4	50.7	52.3	59.8	54.4	56.0	64.9	63.5
	P_{norm} ↑	46.0	33.8	57.6	59.8	60.0	-	68.8	62.3	-	73.8	72.0
	P↑	37.3	30.1	50.5	52.1	51.0	-	60.8	54.7	56.6	69.0	66.5

Table 3: Results on LaSOT.

Trackers		DPMT[51]	SuperDiMP [1] [20]	DiMP	ATOM	SiamMask [48]	STM [40]	DET50 [25]	Ocean	TransT[7]	Stark-S50 [54]	Ours
VOT-20	Acc.↑	0.492	0.492	0.457	0.462	0.624	0.751	0.679	0.693	-	0.761	0.755
	Rob.↑	0.745	0.745	0.740	0.734	0.648	0.574	0.787	0.754	-	0.749	0.836
	EAO↑	0.303	0.305	0.274	0.271	0.321	0.308	0.441	0.430	0.495	0.462	0.528

Table 4: Results on VOT2020. We use AlphaRefine[55] to generate mask for VOT benchmark.

	ATOM	SiamRPN++	DiMP	STMTrack [16]	SiamRN [9]	Ours
AUC	64.3	61.3	65.3	64.7	64.8	68.2

Table 5: Results on UAV123.

	Baseline			Realtime		
	EAO	Acc.	Rob.	EAO	Acc.	Rob.
Baseline	0.525	0.748	0.826	0.509	0.746	0.815
ST	0.536	0.755	0.836	0.512	0.751	0.816
On	0.539	0.757	0.837	0.395	0.681	0.741

Table 6: Results on VOT2021. ST denotes the spatio-temporal version of DualTFR. "On" denotes the online version of DualTFR.

LaSOT LaSOT contains 280 long-term video sequences for testing. The evaluation protocol we adopted is one-pass evaluation. The success rate (AUC) and precision (P) of recent sota trackers are presented in Table 3. DualTFR achieves comparable results with TransT [7] and surpasses remaining trackers in three metrics. The main reason that DualTFR does not perform better than TransT lies in the network stride. TransT adopts stride 8 and 32×32 output size while DualTFR adopts stride 16 and 14×14 output size. Smaller stride is preferred as addressed in [58]. DualTFR can be improved with smaller stride in the future.

UAV123 UAV123 contains 123 aerial video sequences of small objects captured from low latitude UAVs. One pass evaluation protocol is adopted (AUC denotes success rate). Table. 5 shows the results. Compared to the recent SOTA trackers SiamRN[9] and STMTrack[16], DualTFR has over 3.4 improvements on AUC and achieves better performance than the remaining trackers.

VOT2020 VOT2020 adopts an anchor-based evaluation protocol which conducts multiple tests for one video sequence without reset operation. VOT2020 accepts axis-aligned, rotated box or binary segmentation mask format. The final metric for ranking is the Expected Average Overlap (EAO). Here, we use the alpha-refine [55] for mask generation. The VOT2020 top performers are RPT [36] and Ocean [59], two recent sota transformer-based trackers Stark [54] and TransT [7], and classical deep trackers SiamRPN++ [27], ATOM [12], DiMP [3] and sota video obeject segmentation method STM [40] are compared with our tracker. Table. 4 shows that our tracker outperforms all trackers on all three measures. In terms of EAO, DualTFR outperforms the strongest SOTA Stark-50 by 2.3 points and TransT by 3.3 points. DualTFR has larger improvement over other methods. Note that transformer-based trackers (Stark, TransT, DualTFR) are nearly or higher than 0.50 EAO, it shows the superiority of attention-based models towards current fully CNN-based trackers.

VOT2021 The evaluation metrics and protocol on the VOT2021 dataset is the same with VOT2020 benchmark. A certain number of hard video sequences are chosen to replace the easy video sequences on VOT2020. As described in Sec. 4.1, we presents three versions of DualTFR in Table. 6 to show its wide applicability. DualTFR achieves very competing performance on the VOT2021 benchmark.

5. Ablation Studies

In this section, we discuss the potential of the fully attention-based model in visual object tracking by a number of ablation studies.

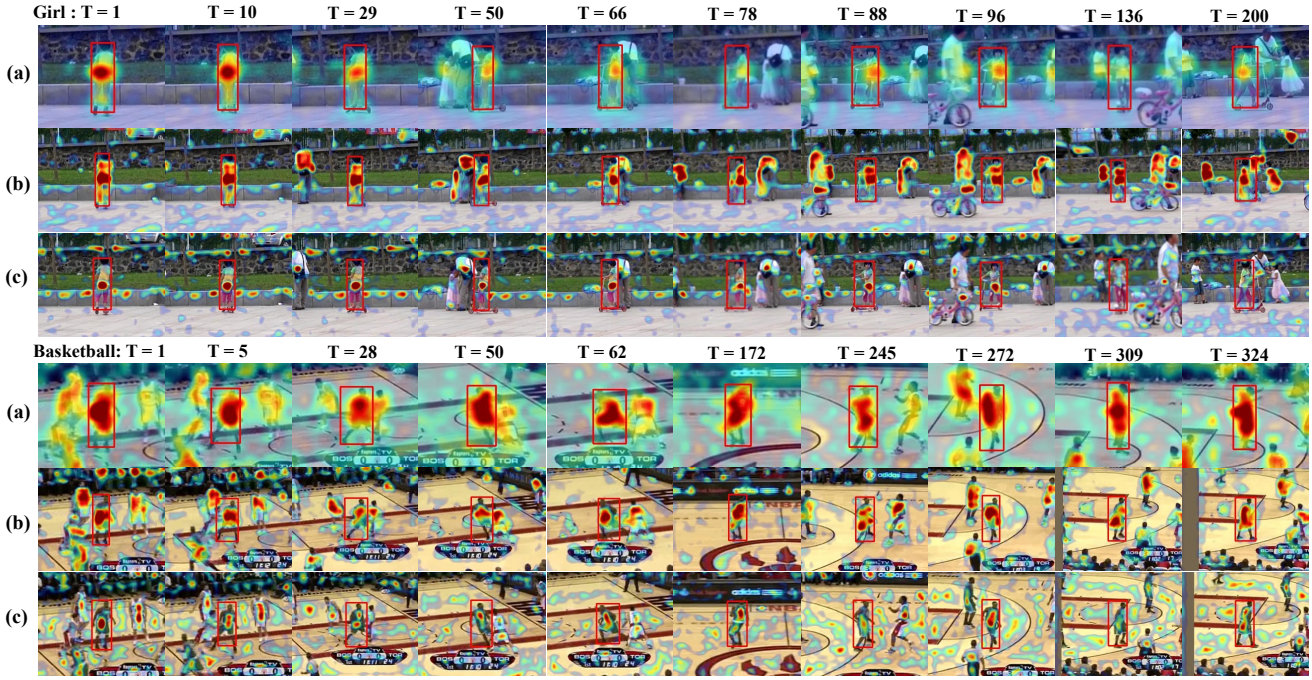

Figure 6: Visualization results. The cosine similarities between center point of template and the whole search-region feature. Features are from the the last layer of CNN backbone or the last block of LAB. (a) (b) (c) are referred to the results of DualTFR, TransT, SiamRPN++. Note that the search area scale of DualTFR is smaller, but it does not influence our analysis.

5.1. Transformer Features vs. CNN Features

To investigate why transformer-based features are better than the CNN-based features, we visualize the attention maps of the template and the search images before cross attention block. We select three trackers which also follow the Siamese framework for comparison. DualTFR belongs to the fully attention-based tracking. TransT is a representative method which combines CNN-based feature extraction and attention-based fusion, SiamRPN++ is a pure CNN-based Siamese tracker.

Instance-Discriminative Features As shown in Fig. 6, the visualization results of the CNN-based trackers (last two rows) all have large responses on all instances having similar appearance. See the two human instances in 245_{th} frame in the basketball example. Note that the response map indicates the similarity between the feature of the template center point and all features from the search image. As for the attention-based features, the similarities between template center and distractor objects are much lower which greatly enhances the discriminative ability of tracking model. Another interesting phenomenon is that the high responses inside the target instance gradually expands from pure CNN-based, CNN+transformer to fully Transformer-based model. For the pure CNN-based tracker SiamRPN++, the response values are very focus and narrow which means the template center only shares high similarity with the exact center point of target instance. In contrast, the response map of DualTFR has high values almost over the whole target area. It indicates that attention-based feature network is more focus on inter-instance difference rather than intra-instance. We name it instance-discriminative features. Thus, we argue that the attention-based features learned from matching are more suitable for instance-level tasks. In Table. 9, we replace the LAB and GAB in DualTFR by ResNet-50 which has comparable parameters. The performance on GOT-10k drops 2.7 points in terms of AO from 73.5% to 70.8%. This validates the superiority of attention-based feature.

Attention-Based Progressive Fusion Manner As illustrated in TransT [7], the transformer-based fusion performs better than linear convolution operation. Here, we further stress that the progressive manner of attention-based model are the main distinctions towards CNN-based pipelines. The attention-based CABs can gradually exchange the information between template and search branch which allows for the progressive refinement. As shown in Fig. 7, the attention in search-region gradually focuses on the target and distinct the distractors on the nearby spatial location. In the meantime, the feature embedding of template is adaptive to the search-region feature . More specifically, the template feature refines itself to be a more abundant feature bank for matching the search-region feature. In Table. 9, we replace the CABs with depth-wise correlation which results in a di-

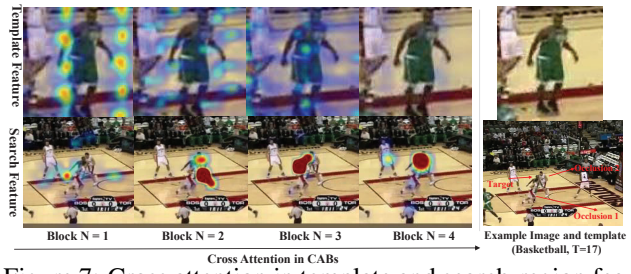

Figure 7: Cross attention in template and search-region feature when the CAB goes deeper.

method	Backbone	Param.	FLOPs	EAO
SiamRPN++	ResNet-50	53.9M	59.5G	0.356
STARK-ST101	ResNet-101	42.4M	18.5 G	0.497
TransT	ResNet-50	23M	19.1G	0.495
DualTFR	LAB	44.1M	18.9G	0.528

Table 7: Comparison of parameters and flops. EAO denotes the performance on VOT2020 benchmark. The EAO of SiamRPN++ comes from the SiamMargin in [25].

ImageNet.Pre	✓	✗	✓	✓
Trans.Extraction	✓	✓	✗	✓
Conv.Extraction	✗	✗	✓	✗
Trans.Fusion	✓	✓	✓	✗
Conv.Fusion	✗	✗	✗	✓
$AO\uparrow$	73.5	48.5	70.8	54.2

Table 8: Ablation study on GOT-10k. ✓denotes the choice between two modules. ✗represents model does not choose this module. Trans.Fusion denotes transformer-based feature fusion. Conv.Fusion denotes depth-wise correlation in Siamese tracker. Extraction denotes the type of backbone. Trans. denotes transformer-based while Conv. denotes CNN-based.

rect drop of performance from 73.5% to 51.2% in GOT-10k. This demonstrates the advantage of attention-based fusion.

5.2. ImageNet Pre-training Vs. Train from Scratch

For traditional deep trackers, the backbone with ImageNet pretrained parameters is vital for learning tracking representations. As shown in Table. 9, if the weight-sharing LAB part is pre-trained in ImageNet dataset, the whole performance rises from 48.5% to 73.5%. This indicates that the fully transformer-based pipeline needs the prior knowledge from ImageNet pretraining. However, it brings extra training burden which we hope to bridge the gap between training from scratch and ImageNet-pretraining.

5.3. Impacts of LAB, GAB and CAB

We further investigate different configurations of blocks. For convenience and fair comparison, the ablation settings are all train-from-scratch and follow GOT-10k training protocol. With 2 GABs connected to the LAB, the performance of AO rises from 42.1 to 46.1 comparing to no GAB settings. This is mainly because the global modelling enhances the feature representation of LAB. More stacked CABs which means more comprehensive feature fusion also brings improvements (42.1 to 46.1). Though more LABs can provide better feature extraction ability, the parameters and flops increases sharply. With 4.7 improvements in AO, 9 LABs brings extra 15M parameters and 12G flops. For a trade-off, we choose the fouth settings which has comparable parameters and flops to STARK to implement our DualTFR.

LAB	(2,2,6)	(2,2,6)	(2,2,6)	(2,2,6)	(2,2,18)
GAB	✗	✗	2	4	4
CAB	4	2	4	4	4
Param.	34.1 M	29.2 M	39.2 M	44.1 M	72.4 M
FLOPs.	14.2 M	11.3 G	16.5 G	18.9 G	30.0 G
$AO\uparrow$	42.1	40.4	46.1	48.5	53.2

Table 9: Ablation study of LAB, GAB, and CAB in GOT-10k. The triplets of LAB denotes the number of blocks in 3 different stages (See Fig. 2). ✗denotes not used.

5.4. Future Work

As shown in Table 7, DualTFR has comparable parameters and flops while outperforms the STARK in VOT2020 benchmark. It is worth noting that the parameters of DualTFR is twice of the TransT ($44.1M$ vs. $23M$). It is mainly due to the redundant independent module design. However, Siamese-style design may results in large computation flops. Inspired by ViT design manner, a unified dual-path block which can be stacked to formulate the whole tracking model may reduce the parameters and flops. In the future, DualTFR can be implemented by a more integral block.

6. Conclusion

In this work, we propose a dual-branch fully transformer-based tracking architecture. Through the dedicated design of GAB, LAB and CAB modules, we achieve a good balance between the computation cost and tracking performance. Furthermore, we prove the superiority of fully attention-based paradigm to the traditional CNN-based tracking paradigm. In the future, the fully transformer-based tracking model can further be more light-weight through dedicated block design. Extensive experiments show that DualTFR performs at the state-of-the-art level while running at a real-time speed. We hope this work could provides some insights on developing more powerful fully transformer-based trackers.

Acknowledgment

We would like to thanks for advices from Chenyan Wu. This work was supported by NSFC (No.61773117 and No.62006041).

References

[1] https://github.com/visionml/pytracking.

[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.

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

[4] Goutam Bhat, Joakim Johnander, Martin Danelljan, Fahad Shahbaz Khan, and Michael Felsberg. Unveiling the power of deep tracking. In *ECCV*, 2018.

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

[6] Chun-Fu Chen, Quanfu Fan, and Rameswar Panda. Crossvit: Cross-attention multi-scale vision transformer for image classification. *arXiv preprint arXiv:2103.14899*, 2021.

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

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

[9] 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.

[10] 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.

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

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

[13] Martin Danelljan, Luc Van Gool, and Radu Timofte. Probabilistic regression for visual tracking. In *CVPR*, 2020.

[14] 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.

[15] 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.

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

[17] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. ICAIS, 2010.

[18] Dongyan Guo, Yanyan Shao, Ying Cui, Zhenhua Wang, Liyan Zhang, and Chunhua Shen. Graph attention tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9543–9552, 2021.

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

[20] Fredrik K Gustafsson, Martin Danelljan, Radu Timofte, and Thomas B Schön. How to train your energy-based model for regression. *arXiv preprint arXiv:2005.01698*, 2020.

[21] Anfeng He, Chong Luo, Xinmei Tian, and Wenjun Zeng. A twofold siamese network for real-time object tracking. In *CVPR*, pages 4834–4843, 2018.

[22] Anfeng He, Chong Luo, Xinmei Tian, and Wenjun Zeng. A twofold siamese network for real-time object tracking. In *CVPR*, 2018.

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

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

[25] Matej Kristan, Ales Leonardis, Jiri Matas, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kamarainen, Luka Čehovin Zajc, Martin Danelljan, Alan Lukezic, Ondrej Drbohlav, Linbo He, Yushan Zhang, Song Yan, Jinyu Yang, Gustavo Fernandez, and et al. The eighth visual object tracking vot2020 challenge results, 2020.

[26] Matej Kristan, Jiri Matas, Ales Leonardis, Michael Felsberg, Roman Pflugfelder, Joni-Kristian Kamarainen, Luka Čehovin Zajc, Ondrej Drbohlav, Alan Lukezic, Amanda Berg, Abdelrahman Eldesokey, Jani Kapyla, and Gustavo Fernandez. The seventh visual object tracking vot2019 challenge results, 2019.

[27] 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.

[28] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018.

[29] 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.

[30] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In *CVPR*, 2018.

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 2117–2125, 2017.

[32] 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.

[33] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin trans-

former: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021.

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

[35] Alan Lukezic, Jiri Matas, and Matej Kristan. D3S-a discriminative single shot segmentation tracker. In *CVPR*, 2020.

[36] 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.

[37] Tim Meinhardt, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer. TrackFormer: Multi-object tracking with transformers. *arXiv preprint arXiv:2101.02702*, 2021.

[38] 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.

[39] Hyeonseob Nam and Bohyung Han. Learning multi–domain convolutional neural networks for visual tracking. In *CVPR*, 2016.

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

[41] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R–CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[42] 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.

[43] 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.

[44] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. *arXiv preprint arXiv:2103.17239*, 2021.

[45] 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.

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

[47] 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.

[48] 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.

[49] 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.

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

[51] Fei Xie, Ning Wang, Yuncong Yao, Wankou Yang, Kaihua Zhang, and Bo Liu. Hierarchical representations with discriminative meta-filters in dual path network for tracking. In *Chinese Conference on Pattern Recognition and Computer Vision (PRCV)*, 2020.

[52] Fei Xie, Wankou Yang, Bo Liu, Kaihua Zhang, Wanli Xue, and Wangmeng Zuo. Learning spatio-appearance memory network for high-performance visual tracking. *arXiv preprint arXiv:2009.09669*, 2020.

[53] 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.

[54] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. *arXiv preprint arXiv:2103.17154*, 2021.

[55] Bin Yan, Xinyu Zhang, Dong Wang, Huchuan Lu, and Xiaoyun Yang. Alpha-refine: Boosting tracking performance by precise bounding box estimation. *arXiv preprint arXiv:2012.06815*, 2020.

[56] Yuechen Yu, Yilei Xiong, Weilin Huang, and Matthew R Scott. Deformable siamese attention networks for visual object tracking. In *CVPR*, 2020.

[57] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Francis EH Tay, Jiashi Feng, and Shuicheng Yan. Tokens-to-token vit: Training vision transformers from scratch on imagenet, 2021.

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

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

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

[61] Daquan Zhou, Bingyi Kang, Xiaojie Jin, Linjie Yang, Xiaochen Lian, Zihang Jiang, Qibin Hou, and Jiashi Feng. Deepvit: Towards deeper vision transformer. *arXiv preprint arXiv:2103.11886*, 2021.