Separable Self and Mixed Attention Transformers for Efficient Object Tracking

Goutam Yelluru Gopal
g_yellur@encs.concordia.ca

Maria A. Amer
amer@ece.concordia.ca

Department of Electrical and Computer Engineering, Concordia University, Montréal, Québec, Canada

Abstract

The deployment of transformers for visual object tracking has shown state-of-the-art results on several benchmarks. However, the transformer-based models are underutilized for Siamese lightweight tracking due to the computational complexity of their attention blocks. This paper proposes an efficient self and mixed attention transformer-based architecture for lightweight tracking. The proposed backbone utilizes the separable mixed attention transformers to fuse the template and search regions during feature extraction to generate superior feature encoding. Our prediction head performs global contextual modeling of the encoded features by leveraging efficient self-attention blocks for robust target state estimation. With these contributions, the proposed lightweight tracker deploys a transformer-based backbone and head module concurrently for the first time. Our ablation study testifies to the effectiveness of the proposed combination of backbone and head modules. Simulations show that our Separable Self and Mixed Attention-based Tracker, SMAT, surpasses the performance of related lightweight trackers on GOT10k, TrackingNet, LaSOT, NfS30, UAV123, and AVisT datasets, while running at 37 fps on CPU, 158 fps on GPU, and having 3.8M parameters. For example, it significantly surpasses the closely related trackers E.T.Track and MixFormerV2-S on GOT10k-test by a margin of 7.9% and 5.8%, respectively, in the AO metric. The tracker code and model is available at https://github.com/goutamyg/SMAT.

1. Introduction

The Siamese Network-based (SN) architecture is prevalent in visual object tracking due to its simplicity and high speed [34, 39]. The SN architecture consists of a backbone to generate robust feature representation of the target template and search regions, a localization head module for target state estimation, and an optional feature fusor module for relation modeling [37]. In recent years, the transformer-based [11, 30, 33] tracking methods have unified feature extraction and relation modeling by deploying self and mixed attention blocks in their backbone [8, 34] to simplify the SN architecture further. Enabled by the computational power of GPUs, these transformer-based SN trackers achieve high frames-per-second (*fps*) during inference. However, the high computational complexity of these transformer-based tracking algorithms severely impacts the *fps* on constrained hardware, *e.g.*, CPUs, limiting the utility of these algorithms towards applications with hardware constraints.

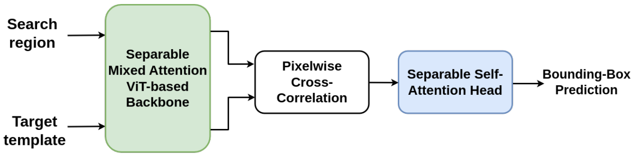
Search region
Target template
Separable Mixed Attention ViT-based Backbone
Pixelwise Cross-Correlation
Separable Self-Attention Head
Bounding-Box Prediction

Figure 1. Proposed *SMAT* architecture. The separable mixed attention Vision Transformer-based backbone jointly performs feature extraction and fusion of template and search regions. The separable transformer-based head models long-range dependencies within the fused features to predict accurate bounding boxes.

The lightweight SN-based trackers [1, 2, 35], which are specifically designed for resource-constrained environments, adopt efficient building blocks to maintain real-time speed, i.e., ≥ 30 *fps*. Therefore, these trackers cannot fully leverage the modeling power of transformers in their architecture because of the computational complexity of the standard transformers, especially the expensive matrix multiplication while computing attention. Mehta *et al.* [23] addressed this issue by replacing the costly matrix-to-matrix multiplication with separable elementwise operations to present an efficient Mobile Vision Transformer (ViT) block for vision-related tasks. Leveraging these advances, we propose a separable self and mixed attention transformer-based lightweight architecture for real-time tracking.

The architecture of the proposed tracker is shown in Figure 1. We employ a cascaded arrangement of convolutional neural network (CNN) and ViT blocks in the proposed tracker backbone. Such a hybrid design [22] combines the merits of convolutions (i.e., learning the spatially-local representations) and transformers (i.e., modeling the long-range dependencies) with fewer parameters compared to the fully transformer-based backbone architecture, such as [8, 37]. Apart from generating a robust feature representation, the proposed backbone facilitates the exchange of information between the target template and search region by computing mixed attention [8] in the ViT block without bloating the backbone latency. Our prediction head efficiently performs global contextual modeling of encoded features using separable self-attention units. Such transformer-based global modeling of encoded features improves the localization accuracy compared to the fully convolution-based methods, as shown in [1]. With these contributions, we propose a lightweight self and mixed attention transformers-based tracker, *SMAT*, running beyond real-time speed on a CPU.

2. Related Work

The introduction of transformer-based modeling has significantly improved the performance of SN-based trackers in recent years [17]. These SN trackers [6, 31, 34, 38] exploited the global contextual modeling capabilities of the transformer layers for relation modeling [37], i.e., to fuse the features extracted from the target template and the search region. The deployment of computationally expensive transformer-based backbones for SN tracking [5, 8, 13, 19, 32, 37] has improved the tracker performance further and achieved state-of-the-art results on various challenging benchmarks [12, 16, 25].

In recent years, there have been several SN-based lightweight algorithms proposed for efficient object tracking. LightTrack [35] used neural architectural search [7] to design an efficient backbone and head modules suitable for resource-constrained environments. FEAR [2] presented a compact and energy-efficient SN-based tracking method running at real-time speed on a smartphone. It uses the dual-template representation with a dynamic update scheme to model the target appearance variations. Stark-Lightning [34] proposed an efficient tracking method with a lightweight transformer-based feature fusor module. HiFT [3] used hierarchical feature transformers to achieve real-time speed on an embedded processor for aerial tracking. SiamHFFT [10] introduced a hierarchical transformer-based feature fusion module for efficient tracking on CPUs. HCAT [4] deployed a feature sparsification module and a hierarchical cross-attention transformer-based architecture to achieve real-time on edge devices. It should be noted that the transformer-based tracker [3, 4, 10, 34] employ CNN-based backbones for feature extraction and utilize the transformer layers only for relation modeling, i.e., to fuse the feature representations generated by their backbones.

Fewer lightweight SN trackers use transformer modules in their backbone or head architecture. MixFormerV2 [9] presented a fully transformer-based [11] backbone for efficient tracking, based on knowledge-distillation [15] and progressive model-depth pruning. E.T.Track [1] proposed an efficient Exemplar Transformer-based prediction head for visual tracking. It utilized a single instance-level attention layer in the transformer block to achieve real-time on a CPU. Compared to the other related trackers, [9] and [1] are the closest to our work.

Unlike the two-stream encoding approach by [1–4, 10, 34, 35] (i.e., the template and search region features are extracted independently), the proposed backbone facilitates the exchange of information between template and search regions during feature extraction. Different from [2–4, 9, 10, 34], we use a transformer-based head for target localization. In contrast to the fully transformer-based backbone by [9], we use a cascade of CNN [29] and ViT [23] blocks in our backbone. Also, the iterative nature of knowledge-distillation and model pruning by [9] requires multiple rounds of model training, whereas our tracker model needs to be trained only once. Compared to the exemplar transformer-based head module by E.T.Track [1], our separable self-attention prediction head is 3× compact in terms of parameters (5.7 Million for E.T.Track versus 1.8 Million for our *SMAT*). As a post-processing step, the related [35] and [1] refine the predicted bounding boxes by penalizing large changes in target size and aspect ratio of the predicted boxes between consecutive frames. We do not employ any bounding-box refinement techniques during post-processing.

To summarize our contributions, we are the first to propose:

- A separable mixed attention ViT-based backbone for joint feature extraction and information fusion between the template and search regions. Our approach combines the merits of CNNs and efficient ViTs to learn superior feature encoding for accurate tracking without bloating the backbone latency.

- A separable self-attention transformer-based prediction head to model the global dependencies within the fused feature encoding for accurate bounding-box prediction.

Compared to related work, for the first time, our method concurrently deploys an efficient transformer-based backbone and prediction head for lightweight tracking.

3. The Proposed Method

This section discusses the architecture and training details of the proposed *SMAT* tracker. Section 3.1 presents our tracker backbone and Section 3.2 describes the architecture of the proposed head module. Section 3.3 has details of the loss function used during training.

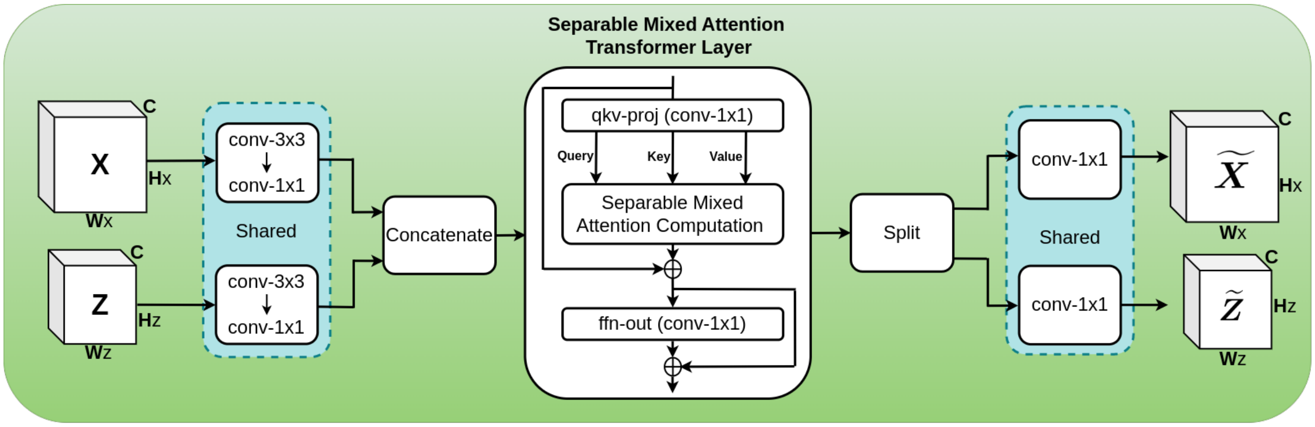

Figure 2. Proposed Separable Mixed Attention ViT block. The *qkv-proj* denotes the set of three 1 × 1 convolutional filters to generate the *Query*, *Key*, and *Value* for attention computation. The mixed attention output is passed through a 1 × 1 convolutional *ffn-out* block to generate the output of the transformer layer.

3.1. Proposed Mixed Attention ViT-based Backbone

The backbone of the proposed tracker receives two images as its input; one is the target template Z_{in}, and the other is the search region for target localization, X_{in}. First, we apply the CNN-based Inverted Residual (IR) [29] blocks on Z_{in} and X_{in}. These IR blocks generate spatially local feature representations of Z_{in} and X_{in} while being efficient compared to the regular convolutional blocks [29]. In addition, these IR blocks reduce the spatial dimensionality of input images by the pooling operation to generate low-dimensional feature representations for our separable mixed attention ViT block.

The architecture of the proposed mixed attention ViT block is shown in Figure 2. Let $Z \in R^{W_z \times H_z \times C}$ and $X \in R^{W_x \times H_x \times C}$ denote the template and search region feature representations, respectively. Inside the proposed ViT block, we initially pass Z and X through a series of 3×3 and 1×1 CNN layers with shared weights to project the number of channels in Z and X from C to d. We tokenize [22] the output of CNN blocks and concatenate them to generate a total of k tokens to learn mixed attention [8] between the template and search regions. Inside the transformer layer, we first apply a set of three 1×1 convolutional filters (denoted as *qkv-proj* in Figure 2) to generate the query $\mathcal{Q} \in R^{k \times 1}$, the key $\mathcal{K} \in R^{k \times d}$, and the value $\mathcal{V} \in R^{k \times d}$. Then, we apply the softmax operation on the query vector $\mathcal{Q}$ and broadcast along its column (i.e., the element in i^{th} row is repeated d times along the column dimension) to generate $\tilde{\mathcal{Q}} \in R^{k \times d}$. Using $\tilde{\mathcal{Q}}$ and $\mathcal{K}$, the context vector $\mathcal{A} \in R^{1 \times d}$ is computed as

$$\mathcal{A} = \sum_k \tilde{\mathcal{Q}} \odot \mathcal{K}, \quad (1)$$

where $\odot$ denotes the element-wise multiplication and $\sum_k$ indicates summation across the rows. The context vector $\mathcal{A}$ is broadcasted along its rows to create $\tilde{\mathcal{A}} \in R^{k \times d}$, which is used to compute the mixed attention $\mathcal{M} \in R^{k \times d}$ as

$$\mathcal{M} = \tilde{\mathcal{A}} \odot \mathrm{ReLU}(\mathcal{V}). \quad (2)$$

The elementwise multiplication operations in Eq. 1 and Eq. 2 reduce the latency of the separable transformer layer, shown in Figure 2, when compared to the dense matrix-to-matrix multiplication-based attention computation in standard transformers [30]. Also, computing the mixed attention on the concatenated features concurrently models the global interactions *within* (i.e., self) and *between* (i.e., cross) the target template and the search area. Therefore, mixed attention requires fewer transformer block evaluations than separately computing the self and cross-attention. Similar to [30], we employ a residual connection [14] around the attention computation block. We pass the output of the residual connection through a 1×1 convolutional feedforward network (denoted as *ffn-out* in Figure 2) to generate the output of the separable transformer layer. We split the resulting feature map to separate the template and search region features with d channels. Finally, we re-project the number of channels from d to C by applying a shared 1×1 convolutional filter on the separated feature maps to generate $\tilde{X}$ and $\tilde{Z}$ as the output.

The computation of mixed attention in the ViT block facilitates implicit relation modeling during feature extraction, thereby generating superior features compared to the two-stream encoding approach. Such feature fusion also avoids needing a parameter-heavy module for the subsequent relation modeling between the template and search region features. For a parameter-efficient relation modeling (or feature fusion) between the features $\tilde{X}$ and $\tilde{Z}$ generated by the proposed backbone, we use the *parameter-free* pixel-wise cross-correlation [36] operation (*cf.* Figure 1).

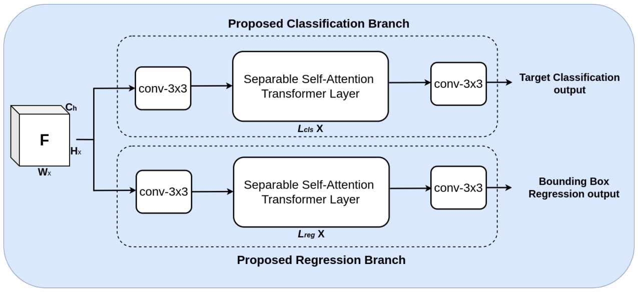
Proposed Classification Branch
conv-3x3
Separable Self-Attention Transformer Layer
conv-3x3
Target Classification output
L_{cls} X
F
C_h
H_x
W_x
conv-3x3
Separable Self-Attention Transformer Layer
conv-3x3
Bounding Box Regression output
L_{reg} X
Proposed Regression Branch

Figure 3. Proposed Separable Self-Attention Transformer-based Predictor Head. It utilizes two branches for target classification and bounding-box regression.

The fused feature encoding F is computed as,

$$F = PWCorr(\tilde{X}, \tilde{Z}). \tag{3}$$

where $PWCorr$ denotes the pixel-wise cross-correlation operation. We apply a 1×1 convolution filter on the fused encoding to transform the number of channels to C_h. To perform target localization, we pass the resulting encoding F to the proposed prediction head in Section 3.2.

3.2. Proposed Self-Attention Prediction Head

The pipeline of the proposed predictor head is shown in Figure 3. Our prediction head has two branches: target classification and bounding-box regression. Unlike a fully CNN-based approach, we implement these branches via a convolutional and transformer layers cascade. The CNN layers are suitable for modeling the local relationships within the fused feature representation but are limited in effectively capturing the non-local associations. On the other hand, the transformer layers explicitly model the long-range global interactions within the features by processing the tokenized feature encoding. Such a global modeling scheme is beneficial for localization under scenarios of drastic target shape variations and heavy occlusion [34]. The cascade of convolutional and transformer-based layers combines the best of their modeling strengths to produce high-quality tracking results on challenging videos.

First, we apply a 3×3 convolutional filter for the proposed predictor head on the fused encoding F from Eq. 3 to extract spatially local feature representations. We then tokenize the filter output and pass it through a stack of L_{cls} and L_{reg} separable self-attention transformer layers in classification and regression branches, respectively. For these transformer blocks, the pipeline of computing the attention is similar to the transformer layer in the ViT block from Section 3.1; except, here, we calculate self-attention to model long-range dependencies *within* the fused feature encoding F for robust target state estimation. The output of the self-attention transformer layer is passed through a 3×3 CNN layer to generate a score map $\mathcal{R}$ for the classification branch, and the local offset and the normalized bounding-box size for the regression branch, as in [37]. A combination of local and long-range contextual modeling by our predictor head improves tracker performance without significantly increasing the overall model latency.

3.3. Loss function for Training

We use loss functions on the classification and regression output generated by the proposed head module while training our model. We use the weighted focal loss for the output of the classification branch; for the output of the regression branch, we use the ℓ_1 loss and generalized Intersection-over-Union (IoU) loss, as in [37]. The overall loss function $\mathcal{L}_{total}$ is defined as

$$\mathcal{L}_{total} = \mathcal{L}_{focal} + \lambda_{\ell_1} \cdot \mathcal{L}_{\ell_1} + \lambda_{IoU} \cdot \mathcal{L}_{IoU}, \tag{4}$$

where $\mathcal{L}_{focal}$, $\mathcal{L}_{\ell_1}$, and $\mathcal{L}_{IoU}$ represent the focal loss, ℓ_1 loss and the generalized IoU loss functions, respectively. λ_{ℓ_1} and λ_{IoU} are the hyperparameters determining the relative impact of the respective loss functions.

4. Experimental and Ablation Results

4.1. Implementation Details

We set the size of the template and search region images, i.e., Z_{in} and X_{in} from Section 3, to 128×128 and 256×256, respectively, during training and inference. We deploy two CNN-based IR [29] blocks and two ViT-blocks in our backbone. The sequential ordering of the IR and ViT blocks is the same as MobileViTv2 [23] pipeline. During feature extraction, our backbone performs four downsampling operations, each by a factor of 2; therefore, the spatial dimension of the features generated by our backbone is 8×8 and 16×16 for the template and search regions, respectively. For the proposed head module, we set the number of channels in the fused encoding, i.e., C_h in Figure 3, to 128. We set the number of transformer layers L_{cls} and L_{reg} in the classification and regression branches to 2 and 4, respectively. The reason for defining L_{reg} two times the value of L_{cls} is because the regression head predicts twice the variables, i.e., local offset and bounding-box size, compared to the classification branch predicting the target center.

4.2. Training and Inference Details

We use the GOT10k [16], LaSOT [12], TrackingNet [25], and COCO [20] datasets to train our tracker. GOT10k, LaSOT, and TrackingNet have a non-overlapping train-test split ratio of 9335-to-180, 1120-to-280, and 30132-to-511 videos, respectively. Also, GOT10k provides 180 additional videos as the validation split. We apply data augmentation (horizontal flip and scale jittering) to generate training image pairs for the still images in the COCO train dataset. We use the combined training splits of these four datasets to train our model.

We train our model for 300 epochs, and each epoch uses 6×10^4 image pairs uniformly sampled from the training dataset. The initial learning rate (*lr*) is set to 0.0004 and is reduced by a factor of 10 after 240 epochs. The *lr* for the backbone parameters is set 0.1 times the *lr* for the remaining trainable parameters of our model. We use AdamW [21] as the network optimizer and set the weight decay to 10^{-4}. The values of hyperparameters λ_{ℓ_1} and λ_{IoU} from Eq. 4 are set to 5 and 2, respectively. These hyperparameter values used for training are derived from [37] with no additional finetuning. We initialize our backbone weights using a pretrained MobileViTv2 model provided by the authors [23]. We use PyTorch [27] for developing the tracker code. Our model is trained on a single NVidia Telsa V100 GPU (32GB memory) with a batch size of 128. We monitor the possibility of overfitting by periodically evaluating the values of loss functions $\mathcal{L}_{focal}, \mathcal{L}_{\ell_1}$, and $\mathcal{L}_{IoU}$ from Eq. 4 using the GOT10k validation videos.

During inference, we use the annotation from the first frame in the video as the target template and do not perform model update. To define the search region at frame t, we crop a region around tracker output at frame $t-1$, four times the target size. This image is resized to 256×256 and utilized as the search image at frame t. As a post-processing step, we apply a Hanning window on the classification score map $\mathcal{R}$ to penalize large target displacement predictions.

4.3. Comparison to the Related Work

To assess the performance of the proposed *SMAT*, we evaluate our tracker on the test-split of GOT10k [16], LaSOT [12], TrackingNet [25], NfS30 [18], UAV123 [24], and AVisT [26] datasets. GOT10k-test has 180 test videos having non-overlapping object classes compared to their training videos, mainly to promote the generalization of tracking algorithms towards unseen object categories. LaSOT-test has 280 videos with 14 different attributes and balanced class categories. With an average length of 2500 frames per video, LaSOT-test is effective in accessing long-term tracking capabilities. TrackingNet-test contains 511 challenging videos curated from the large-scale YouTube-BB [28] dataset with 15 attribute annotations. NfS30 has 100 test videos, predominantly containing fast-moving objects with significant motion blur. UAV123 is a low-altitude UAV tracking benchmark and has 123 videos with 12 attribute annotations. AVisT dataset has 120 challenging videos with a wide range of atmospheric adverse scenarios such as rain, fog, fire, low-light, snow, tornado, and smoke impacting the target appearance in the test videos. The datasets NfS30, UAV123, and AVisT have no training split videos.

We use the metrics recommended by the corresponding dataset authors to quantify the tracker performance during our evaluation. GOT10k uses the Average of Overlap (AO) based on the Intersection-of-Union (*IoU*) value between the groundtruth and predicted bounding boxes, averaged over all the test videos. It also uses Success Rate (SR), computing the fraction of frames having an IoU value greater than a threshold τ, with values of τ as 0.5 and 0.75. TrackingNet uses Area-Under-the-Curve (AUC), Precision (P), and Normalized-Precision (P_{norm}) as the tracker evaluation metrics. AUC is equivalent to AO [40], and P is computed based on the distance between the groundtruth and predicted bounding-box centers, measured in pixels. The metric P_{norm} is similar to P; however, P_{norm} uses normalized bounding boxes while measuring the distance between their centers. LaSOT uses the same evaluation metric as TrackingNet, whereas NfS30 and UAV123 use AUC and P for tracker evaluation. Along with the AUC, AVisT uses OP50 and OP75 as its evaluation metrics, which are equivalent to SR at thresholds 0.5 and 0.75, respectively, from GOT10k. To ensure fair tracker evaluation and avoid finetuning of parameters on the test data, GOT10k and TrackingNet sequester the ground-truth annotations for their test videos. Therefore, we generate the metrics for these datasets by submitting the raw tracker results to the remote evaluation server. The groundtruth annotations are available for the LaSOT-test, NfS30, UAV123, and AVisT datasets.

We compare the results of the proposed *SMAT* against the related lightweight trackers: LightTrack [35], Stark-Lightning [34], FEAR-XS [2], HCAT [4], E.T.Track [1], and MixFormerV2-S [9]. From Table 1, we can see that the proposed *SMAT* comprehensively outperforms the related trackers on all six test datasets: GOT10k-test, TrackingNet-test, LaSOT-test, NfS30, UAV123, and AVisT. No related tracker performs consistently second best across the six datasets. HCAT exhibits the second-best results in 10 out of 16 metrics across all datasets, while MixFormerV2-S scores the second-best in 6 out of 16 cases. Regarding *fps* under CPU, our tracker is relatively faster than MixFormerV2-S by 19% and slower than HCAT by 17%. Considering the GOT10k-test dataset (server-based evaluation), our tracker is better than MixFormerV2-S and HCAT on average by 3.5% in AO, 4% in $SR_{0.50}$, and 5.8% in $SR_{0.75}$.

Since **GOT10k-test** dataset has videos with target object categories unseen during training, it is well-suited for evaluating tracker generalization. Our *SMAT* results on the GOT10k-test demonstrate its superior generalization capability compared to the related lightweight trackers. Since E.T.Track has a transformer-based predictor head and MixFormerV2-S has a fully transformer-based backbone, these trackers are closely related to our work. By concurrently deploying a transformer-based backbone and head module, our *SMAT* outperforms both E.T.Track and MixFormerV2 on average by a significant margin of 6.8% in AO, 8.8% in $SR_{0.50}$, and 12.4% in $SR_{0.75}$. On

Tracker	GOT10k-test [16]			TrackingNet-test [25]			LaSOT-test [12]			NfS30 [18]		UAV123 [24]		AVisT [26]			*fps*
	AO	$SR_{0.50}$	$SR_{0.75}$	AUC	P_{norm}	P	AUC	P_{norm}	P	AUC	P	AUC	P	AUC	OP50	OP75	(CPU)
LightTrack [35] (CVPR'21)	0.582	0.668	0.442	0.729	0.793	0.699	0.522	0.583	0.517	0.565	0.692	0.617	0.799	0.404	0.437	0.242	42
Stark-Lightning [34] (ICCV'21)	0.596	0.696	0.479	0.727	0.779	0.674	0.578	0.660	0.574	0.596	0.710	0.620	0.820	0.394	0.431	0.223	50
FEAR-XS [2] (ECCV'22)	0.573	0.681	0.455	0.715	0.805	0.699	0.501	0.594	0.523	0.486	0.563	0.610	0.816	0.370	0.421	0.220	42
HCAT [4] (ECCV'22)	0.634	0.743	0.558	0.763	0.824	0.726	0.590	0.683	0.605	0.619	0.741	0.620	0.805	0.418	0.481	0.263	45
E.T.Track [1] (WACV'23)	0.566	0.646	0.425	0.740	0.798	0.698	0.589	0.670	0.603	0.570	0.694	0.626	0.808	0.390	0.412	0.227	44
MixFormerV2-S [9] (arXiv'23)	0.587	0.672	0.482	0.767	0.812	0.714	0.610	0.694	0.614	0.610	0.722	0.634	0.837	0.396	0.425	0.227	30
SMAT (ours)	**0.645**	**0.747**	**0.578**	**0.786**	**0.842**	**0.756**	**0.617**	**0.711**	**0.646**	**0.620**	**0.746**	**0.643**	**0.839**	**0.447**	**0.507**	**0.313**	37

Table 1. Comparison of proposed *SMAT* with the related lightweight SN trackers on GOT10k-test (server), TrackingNet-test (server), LaSOT-test, NfS30, UAV123, and AVisT datasets. The best and second-best results are highlighted in red and blue, respectively.

TrackingNet-test benchmark, the proposed *SMAT* has a better performance than all the related trackers by at least 1.9% in AUC, 1.8% in P_{norm}, and 3% in P. No single tracker consistently exhibits the second-best performance on the TrackingNet dataset. The results of our *SMAT* on the **LaSOT-test** videos show that our tracker has better long-term tracking performance than other lightweight trackers. The related E.T.Track employs a post-processing step to refine the predicted bounding boxes, which reduces the chances of target loss or drift during long-term tracking. The other related tracker, MixFormerV2, utilizes an online template update scheme to improve long-term tracking performance. Despite the absence of such bounding-box refinement step and template update scheme, our *SMAT* performs better than E.T.Track by 2.8% and MixFormerV2 by 0.7% in AUC. The results on **NfS30** dataset show that our *SMAT* is resilient to motion blur and is better-suited for tracking fast-moving objects than the related trackers. Similarly, proposed *SMAT* has a better AUC by at least 2.8% and 0.9% on **AVisT** and **UAV123** datasets, respectively, compared to the other lightweight trackers. It shows that our tracker performs better than the related trackers on videos affected by adverse visibility conditions and is robust to the challenges of airborne scenarios.

The last column of Table 1 lists the *fps* of the proposed *SMAT* and related trackers, evaluated on a 12th Gen Intel(R) Core-i9 CPU. The proposed *SMAT* tracker achieves a real-time speed of 37 *fps* by leveraging the computational efficiency of the separable self and mixed attention blocks used in its model architecture. Compared to standard transformer-based MixFormerV2-S, our tracker is faster by 19% since the proposed *SMAT* uses separable attention-based transformers in its backbone for efficient computation of attention. However, in comparison to the related lightweight trackers with a two-stream pipeline [1, 2, 4, 34, 35], our *SMAT* has a 17% lower *fps* on average. It is mainly due to the coupling of features in our tracker backbone, requiring evaluation of template and search region features at every frame during inference. On the other hand, trackers with a two-stream pipeline compute the template features only once since they do not perform feature fusion in their backbone.

Our tracker achieves a speed of 158 *fps* upon its evaluation on an Nvidia RTX 3090 GPU. With nearly 3.8 Million parameters, our model has a size of 15.3MB on disk.

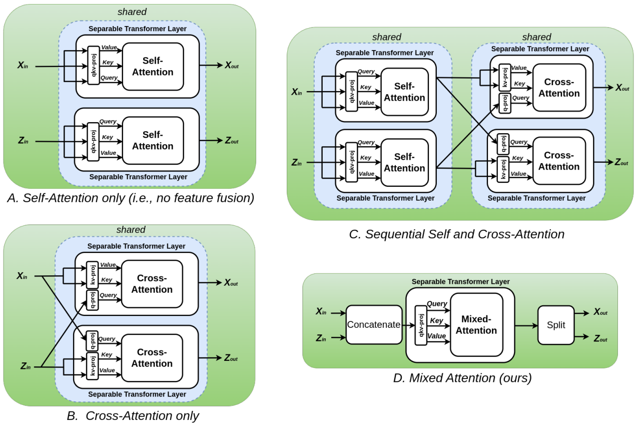
shared
Separable Transformer Layer
X_{in}
Z_{in}
X_{out}
Z_{out}
Value
Key
Query
Self-Attention
Cross-Attention
Concatenate
Mixed-Attention
Split
A. Self-Attention only (i.e., no feature fusion)
B. Cross-Attention only
C. Sequential Self and Cross-Attention
D. Mixed Attention (ours)

Figure 4. Comparing different feature fusion techniques (*A, B* and *C*) with the proposed mixed attention-based method shown in *D*.

4.4. Ablation Study

In this section, we present ablation study results quantifying the role of different components of our tracker. For this study, we use GOT10k-test [16] and LaSOT-test [12] due to their suitability towards gauging tracker generalization and long-term tracking performance, respectively.

Comparing feature fusion techniques: In Section 3.1, we mentioned that the proposed mixed attention efficiently approximates the explicit modeling of interaction *within* and *between* the target template and search regions. In this section, we experimentally verify the efficacy and efficiency of the proposed approach by comparing its results with other feature fusion methods that deploy explicit computation of self or cross-attention (or both). For the variant-*A* shown in Figure 4, we use a shared separable self-attention transformer block for the template and search regions. This approach facilitates independent computation of the template and search features for high tracking speed; however, it restricts the information exchange between the two regions (i.e., no feature fusion). For the variant-*B*, we deploy shared separable cross-attention transformers for inter-region feature fusion but no intra-region feature

Attention Mechanism	GOT10k-test [16]			LaSOT-test [12]			*fps*
	AO	$SR_{0.50}$	$SR_{0.75}$	AUC	P_{norm}	P	(CPU)
A	0.631	0.726	0.578	0.604	0.689	0.629	**39**
B	0.645	0.743	0.590	0.609	0.696	0.631	30
C	**0.654**	**0.761**	**0.598**	**0.621**	**0.717**	**0.656**	24
D (ours)	0.645	0.747	0.578	0.617	0.711	0.646	37

Table 2. Summarizing the feature fusion-based ablation study results for the proposed *SMAT* tracker. The best and second-best results are highlighted in red and blue, respectively.

modeling (i.e., no self-attention). A similar cross-feature blending approach for relation modeling has shown excellent tracking results [6]. Lastly, we implement cascaded self and cross-attention transformers for the third variant-*C* to explicitly model inter and intra-region feature fusion. The proposed mixed attention block, shown as variant-*D* in Figure 4, approximates explicit self and cross-attention computation by applying a separable transformer block on the concatenated features from the template and search regions. Table 2 summarizes the performance of these variants on GOT10k and LaSOT test datasets. From Table 2, we can see that variant-*A* has a $1.05\times$ higher *fps* than the proposed method; however, the lack of information flow between the template and search region impacts its performance. Hence, compared to the proposed tracker, it has a lower AUC score by 1.4% and 1.3% on GOT10k and LaSOT, respectively. Performance of the pure cross-attention-based variant-*B* is lower than our *SMAT* by 0.8% in AUC on the LaSOT dataset, and both approaches have comparable performance on the GOT10k dataset. However, variant-*B* has a relatively lower *fps* by 18.9% than our *SMAT*, which indicates that separately computing cross-attention for the template and search regions is slower than our mixed-attention computation on concatenated features. Variant-*C* achieves the best results on both datasets; however, explicit computation of self and cross-attention significantly impacts its tracking speed. Therefore, it has the lowest *fps* value compared to the other variants and is relatively slower than our approach by 35% on a CPU. Compared to the variant-*C*, our method approximates the inter and intra-region feature fusion by a single attention operation. Therefore, as seen from Table 2, our mixed attention-based approach provides the best trade-off between performance and speed compared to other feature fusion variants shown in Figure 4.

Convolutional vs Transformer-based head: To quantify the significance of our separable self-attention transformer-based predictor head from Section 3.2, we replace the proposed module with a fully-convolutional predictor head for classification and bounding-box regression. As seen from Table 3, this replacement decreases the performance of the proposed *SMAT* tracker by 3.5% in AO for the GOT10k-test dataset and 2.3% in AUC for the LaSOT-test dataset, highlighting the impact of global contextual modeling of encoded features by the proposed predictor head.

Predictor Head	GOT10k-test [16]			LaSOT-test [12]		
	AO	$SR_{0.50}$	$SR_{0.75}$	AUC	P_{norm}	P
Fully-Convolutional	0.610	0.709	0.540	0.594	0.682	0.612
Transformer-based (ours)	**0.645**	**0.747**	**0.578**	**0.617**	**0.711**	**0.646**

Table 3. Ablation study results for the proposed transformer-based prediction head. Best results are highlighted in red.

Mixed Attention Mechanism	GOT10k-test [16]			LaSOT-test [12]			*fps*
	AO	$SR_{0.50}$	$SR_{0.75}$	AUC	P_{norm}	P	(CPU)
Standard	**0.659**	**0.760**	**0.605**	0.600	0.687	0.630	32
Separable (ours)	0.645	0.747	0.578	**0.617**	**0.711**	**0.646**	**37**

Table 4. Comparison of tracker performance using the standard *vs* separable mixed attention mechanism in the backbone. The best results are highlighted in red.

Standard vs Separable Attention mechanism: To evaluate the efficiency of the separable attention mechanism deployed in the proposed *SMAT*, we retrain our model by replacing the separable transformer blocks in the tracker backbone with the standard transformer-based MobileViT block [22]. From Table 4, we observe that the proposed method has a higher AUC of 1.7% on the LaSOT dataset compared to the standard attention-based tracker. On the other hand, our approach has a lower AUC of 1.4% on the GOT10k dataset. However, the efficient attention evaluation enhances the speed of the proposed tracking approach by 15.6% compared to the standard transformer-based tracking, as seen from Table 4.

4.5. Attribute-based analysis

In this section, we compare the per-attribute performance of the proposed *SMAT* tracker with related lightweight trackers on the LaSOT dataset. Table 5 summarizes the tracker evaluation results on various challenging factors (or attributes) of the LaSOT dataset, namely Aspect Ratio Change (*ARC*), Background Clutter (*BC*), Camera Motion (*CM*), Deformation (*DEF*), Fast Motion (*FM*), Full Occlusion (*FOC*), Illumination Variation (*IV*), Low Resolution (*LR*), Motion Blur (*MB*), Out-of-View (*OV*), Partial Occlusion (*POC*), Rotation (*ROT*), Scale Variation (*SV*), and Viewpoint Change (*VC*).

From Table 5, we can see that our tracker has the best performance on 8 out of 14 attributes, and it has the second-best performance in 5 cases. For the attributes *IV* and *DEF*, the proposed *SMAT* performs significantly better than the second-best tracker MixFormerV2-S [9], with a higher AUC of 4.3% and 3.7%, respectively. In comparison to the related trackers that do not perform feature fusion in their backbone, i.e., [1, 2, 4, 34, 35], our tracker has a higher AUC of 4.4% for *DEF* and 2.3% for *ARC* than the sec-

Tracker	*ARC*	*BC*	*CM*	*DEF*	*FM*	*FOC*	*IV*	*LR*	*MB*	*OV*	*POC*	ROT	SV	*VC*	Overall
LightTrack [35]	0.503	0.434	0.539	0.577	0.334	0.386	0.550	0.407	0.457	0.441	0.497	0.519	0.523	0.502	0.522
Stark-Lightning [34]	0.572	0.491	0.613	0.594	0.471	0.505	0.610	0.516	0.568	0.557	0.554	0.577	0.582	0.581	0.578
FEAR-XS [2]	0.488	0.437	0.528	0.505	0.389	0.403	0.506	0.421	0.473	0.425	0.478	0.489	0.506	0.487	0.501
HCAT [4]	0.587	0.524	0.639	0.619	0.460	0.507	0.606	0.520	0.579	0.538	0.568	0.592	0.600	0.567	0.590
E.T.Track [1]	0.573	0.526	0.590	0.619	0.404	0.480	0.612	0.484	0.545	0.519	0.562	0.588	0.594	0.576	0.589
MixFormerV2-S [9]	0.603	0.519	0.642	0.626	0.507	0.539	0.619	0.556	0.604	0.574	0.586	0.603	0.617	0.630	0.610
SMAT (ours)	0.610	0.553	0.656	0.663	0.466	0.517	0.662	0.523	0.592	0.570	0.600	0.621	0.624	0.597	0.617

Table 5. Comparing the AUC values of the proposed *SMAT* with the related lightweight trackers for 14 attributes of the LaSOT dataset. The best and second-best results are highlighted in red and blue, respectively. The last column indicates the mean AUC across all videos.

ond best trackers, [1, 4] and [4], respectively. It indicates the effectiveness of transformer-based feature fusion in our tracker backbone, producing accurate bounding boxes under drastic target appearance variations. Also, in comparison to these trackers, our *SMAT* is resilient to tracking failures under *POC* and *BC*, with a higher AUC of 3.2% and 2.7%, respectively, than the second-best results. On the other hand, our *SMAT* has an inferior performance than the related trackers [9] and [34] for the attribute *FM*; we are working to improve our *SMAT* performance under *FM*.

4.6. Visualizing the attention maps

To showcase the interpretability of the proposed *SMAT* tracker, we visualize the tracker output and the corresponding attention maps in Figure 5 for four sequences chosen from the LaSOT test dataset. The images on the left contain the target template (top-left corner), the search region at frame $\#t$, and the tracker output. The images in the center and right indicate the attention maps corresponding to the transformer blocks of our tracker backbone at the spatial resolution of 32×32 and 16×16, respectively. For the examples shown in Figure 5, the target object is impacted by a challenging factor described in Section 4.5, i.e., *ARC* for *bicycle-18*, *IV* for *drone-2*, *DEF* for *drone-2*, and *POC* for *microphone-16*. Despite the influence of these attributes, our *SMAT* successfully locates the target object. For the example shown in the last row of Figure 5, the target is partially occluded by an external object. In this case, the tracker focuses on the visual cues around the target object, as seen from the attention maps in the center. This information is processed by the subsequent transformer blocks of our tracker backbone to produce stronger attention values in the target center and generate an accurate bounding box.

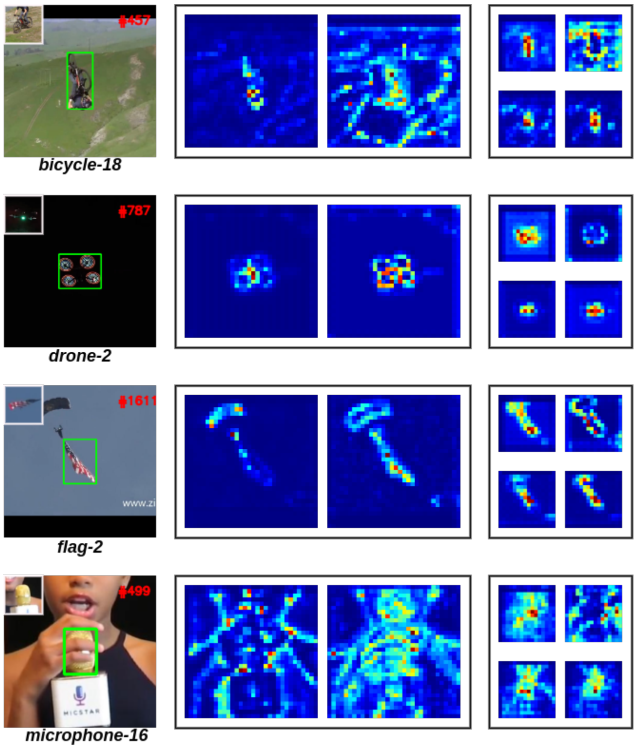

Figure 5. Visualizing the bounding box output (left) and the corresponding attention maps (center and right) for the proposed *SMAT* tracker. The larger values in the attention map are denoted by red color, while the smaller values are represented by blue color.

5. Conclusion

This paper proposed a separable self and mixed attention transformer-based architecture for lightweight tracking. The proposed backbone utilized the separable mixed attention transformer layer to facilitate the exchange of information between the target template and the search region and generate improved encoding compared to the two-stream tracking pipeline. The proposed separable self-attention transformer-based predictor head efficiently modeled long-range dependencies within the fused encoding to generate superior target classification and bounding-box prediction results. Our ablation study analyzed the accuracy-speed tradeoffs using different feature fusion methods, showcased the effectiveness of the proposed head module for accurate tracking, and demonstrated the efficiency of the separable mixed attention compared to the standard attention-based tracking. Our *SMAT* performed better than related lightweight trackers on six challenging benchmarks. The computational efficiency of the proposed architecture assisted our tracker, with 3.8M parameters, to exceed real-time speed on a CPU, while running at 158 *fps* on GPU.

References

[1] Philippe Blatter, Menelaos Kanakis, Martin Danelljan, and Luc Van Gool. Efficient visual tracking with exemplar transformers. In *IEEE Winter Conf. App. Computer Vision*, pages 1571–1581, 2023. 1, 2, 5, 6, 7, 8

[2] Vasyl Borsuk, Roman Vei, Orest Kupyn, Tetiana Martyniuk, Igor Krashenyi, and Jiřı Matas. FEAR: Fast, efficient, accurate and robust visual tracker. In *Proc. European Conf. Computer Vision*, pages 644–663. Springer, 2022. 1, 2, 5, 6, 7, 8

[3] Ziang Cao, Changhong Fu, Junjie Ye, Bowen Li, and Yiming Li. HiFT: Hierarchical feature transformer for aerial tracking. In *Proc. IEEE Int. Conf. Computer Vision*, pages 15457–15466, 2021. 2

[4] Xin Chen, Ben Kang, Dong Wang, Dongdong Li, and Huchuan Lu. Efficient visual tracking via hierarchical cross-attention transformer. In *Proc. European Conf. Computer Vision*, pages 461–477. Springer, 2022. 2, 5, 6, 7, 8

[5] Xin Chen, Houwen Peng, Dong Wang, Huchuan Lu, and Han Hu. Seqtrack: Sequence to sequence learning for visual object tracking. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 14572–14581, 2023. 2

[6] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 8126–8135, 2021. 2, 7

[7] Yukang Chen, Tong Yang, Xiangyu Zhang, Gaofeng Meng, Xinyu Xiao, and Jian Sun. DetNAS: Backbone search for object detection. *NeurIPS*, 32, 2019. 2

[8] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 13608–13618, 2022. 1, 2, 3

[9] Yutao Cui, Tianhui Song, Gangshan Wu, and Limin Wang. MixformerV2: Efficient fully transformer tracking. *arXiv preprint arXiv:2305.15896*, 2023. 2, 5, 6, 7, 8

[10] Jiahai Dai, Yunhao Fu, Songxin Wang, and Yuchun Chang. Siamese hierarchical feature fusion transformer for efficient tracking. *Frontiers in Neurorobotics*, 2022. 2

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2

[12] Heng Fan, Hexin Bai, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Mingzhen Huang, Juehuan Liu, Yong Xu, et al. LaSOT: A high-quality large-scale single object tracking benchmark. *Int. J. Computer Vision*, 129:439–461, 2021. 2, 4, 5, 6, 7

[13] Shenyuan Gao, Chunluan Zhou, and Jun Zhang. Generalized relation modeling for transformer tracking. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 18686–18695, 2023. 2

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 770–778, 2016. 3

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2

[16] Lianghua Huang, Xin Zhao, and Kaiqi Huang. GOT-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Trans. Pattern Anal. Machine Intell.*, 43(5):1562–1577, 2021. 2, 4, 5, 6, 7

[17] Sajid Javed, Martin Danelljan, Fahad Shahbaz Khan, Muhammad Haris Khan, Michael Felsberg, and Jiri Matas. Visual object tracking with discriminative filters and siamese networks: A survey and outlook. *IEEE Trans. Pattern Anal. Machine Intell.*, 45(5):6552–6574, 2023. 2

[18] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *Proc. IEEE Int. Conf. Computer Vision*, pages 1125–1134, 2017. 5, 6

[19] Liting Lin, Heng Fan, Zhipeng Zhang, Yong Xu, and Haibin Ling. Swintrack: A simple and strong baseline for transformer tracking. *Advances in Neural Information Processing Systems*, 35:16743–16754, 2022. 2

[20] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Proc. European Conf. Computer Vision*, pages 740–755. Springer, 2014. 4

[21] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. 5

[22] Sachin Mehta and Mohammad Rastegari. MobileViT: Light-weight, general-purpose, and mobile-friendly vision transformer. In *International Conference on Learning Representations*, 2022. 1, 3, 7

[23] Sachin Mehta and Mohammad Rastegari. Separable self-attention for mobile vision transformers. *Transactions on Machine Learning Research*, 2023. 1, 2, 4, 5

[24] Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for uav tracking. In *Proc. European Conf. Computer Vision*, pages 445–461. Springer, 2016. 5, 6

[25] 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 *Proc. European Conf. Computer Vision*, pages 300–317, 2018. 2, 4, 5, 6

[26] Mubashir Noman, Wafa Al Ghallabi, et al. AVisT: A benchmark for visual object tracking in adverse visibility. In *British Machine Vision Conf.*, page 817. BMVA Press, 2022. 5, 6

[27] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in Neural Information Processing Systems*, 32, 2019. 5

[28] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large

high-precision human-annotated data set for object detection in video. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 5296–5305, 2017. 5

[29] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 4510–4520, 2018. 2, 3, 4

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NeurIPS*, 30, 2017. 1, 3

[31] Ning Wang, Wengang Zhou, Jie Wang, and Houqiang Li. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 1571–1580, 2021. 2

[32] Xing Wei, Yifan Bai, Yongchao Zheng, Dahu Shi, and Yihong Gong. Autoregressive visual tracking. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 9697–9706, June 2023. 2

[33] Haiping Wu, Bin Xiao, Noel Codella, Mengchen Liu, Xiyang Dai, Lu Yuan, and Lei Zhang. CvT: Introducing convolutions to vision transformers. In *Proc. IEEE Int. Conf. Computer Vision*, pages 22–31, 2021. 1

[34] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *Proc. IEEE Int. Conf. Computer Vision*, pages 10448–10457, 2021. 1, 2, 4, 5, 6, 7, 8

[35] Bin Yan, Houwen Peng, Kan Wu, Dong Wang, Jianlong Fu, and Huchuan Lu. Lighttrack: Finding lightweight neural networks for object tracking via one-shot architecture search. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 15180–15189, 2021. 1, 2, 5, 6, 7, 8

[36] Bin Yan, Xinyu Zhang, Dong Wang, Huchuan Lu, and Xiaoyun Yang. Alpha-refine: Boosting tracking performance by precise bounding box estimation. In *Proc. IEEE Conf. Computer Vision Pattern Recognition*, pages 5289–5298, 2021. 3

[37] Botao Ye, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Joint feature learning and relation modeling for tracking: A one-stream framework. In *Proc. European Conf. Computer Vision*, pages 341–357. Springer, 2022. 1, 2, 4, 5

[38] Moju Zhao, Kei Okada, and Masayuki Inaba. TrTr: Visual tracking with transformer. *arXiv preprint arXiv:2105.03817*, 2021. 2

[39] Zheng Zhu, Qiang Wang, Bo Li, Wei Wu, Junjie Yan, and Weiming Hu. Distractor-aware siamese networks for visual object tracking. In *Proc. European Conf. Computer Vision*, pages 101–117, 2018. 1

[40] Luka Čehovin, Aleš Leonardis, and Matej Kristan. Visual object tracking performance measures revisited. *IEEE Trans. Image Process.*, 25(3):1261–1274, 2016. 5