Two-stream Beats One-stream: Asymmetric Siamese Network for Efficient Visual Tracking

Jiawen Zhu[1], Huayi Tang[2], Xin Chen[1], Xinying Wang[1], Dong Wang[1], Huchuan Lu[1]*

[1]Dalian University of Technology, Dalian, China
[2]University of Pennsylvania, Philadelphia, USA
jiawen@mail.dlut.edu.cn, huayit@seas.upenn.edu
{chenxin3131, wangxinying}@mail.dlut.edu.cn, {wdice, lhchuan}@dlut.edu.cn

Abstract

Efficient tracking has garnered attention for its ability to operate on resource-constrained platforms for real-world deployment beyond desktop GPUs. Current efficient trackers mainly follow precision-oriented trackers, adopting a one-stream framework with lightweight modules. However, blindly adhering to the one-stream paradigm may not be optimal, as incorporating template computation in every frame leads to redundancy, and pervasive semantic interaction between template and search region places stress on edge devices. In this work, we propose a novel asymmetric Siamese tracker named **AsymTrack** for efficient tracking. AsymTrack disentangles template and search streams into separate branches, with template computing only once during initialization to generate modulation signals. Building on this architecture, we devise an efficient template modulation mechanism to unidirectional inject crucial cues into the search features, and design an object perception enhancement module that integrates abstract semantics and local details to overcome the limited representation in lightweight tracker. Extensive experiments demonstrate that AsymTrack offers superior speed-precision trade-offs across different platforms compared to the current state-of-the-arts. For instance, AsymTrack-T achieves 60.8% AUC on LaSOT and 224/81/84 FPS on GPU/CPU/AGX, surpassing HiT-Tiny by 6.0% AUC with higher speeds.

Introduction

As a long-standing fundamental topic, visual tracking aims at pinpointing the position of a target object in video frames. Promising advancements have been achieved, attributable to increasingly powerful designs of deep models (He et al. 2016; Vaswani et al. 2017; Dosovitskiy et al. 2021; Diao et al. 2024). However, in practical application scenarios, current high-performance trackers (Ye et al. 2022; Cui et al. 2022; Wei et al. 2023; Zhu et al. 2023; Chen et al. 2024; Lin et al. 2024) often fail to meet the low computational latency requirements, particularly on resource-constrained platforms. Thus, designing efficient tracker is critical and recently attracts extensive research in industry and academia.

Mainstream efficient trackers can be broadly categorized into two types: siamese (two-stream) networks (Yan et al. 2021b; Chen et al. 2022b; Blatter et al. 2023) and one-stream networks (Kang et al. 2023; Li et al. 2023b; Cui et al. 2024). Efficient Siamese trackers build upon the success of the precision-oriented Siamese trackers (Li et al. 2019; Chen et al. 2021), where two symmetric branches with shared parameters are utilized to extract features from template and search region, respectively. Subsequently, the designed interaction module performs feature correlation, as illustrated in Fig. 1 (a). A typical advantage of Siamese tracking pipeline is that the detached streams allow the model to drop the template branch (except initialization) during inference, thereby reducing unnecessary latency. More recently, state-of-the-art precision-oriented trackers (Ye et al. 2022; Cui et al. 2022; Wei et al. 2023; Chen et al. 2023)

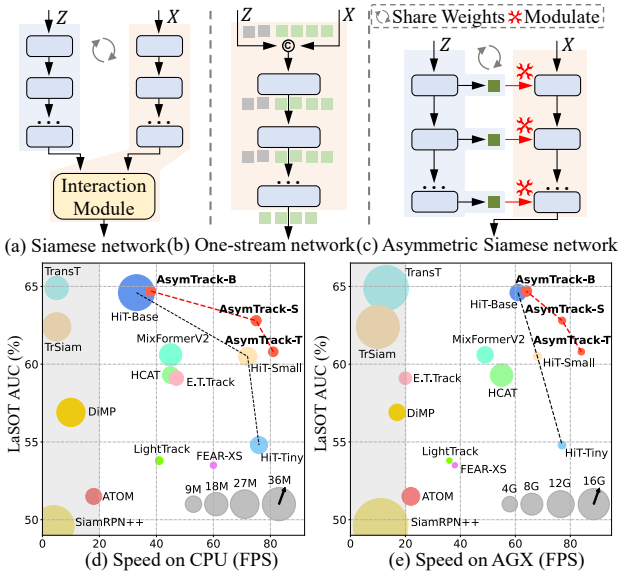

Figure 1: AsymTrack vs. other frameworks and trackers. (a)-(c) represent Siamese (two-stream) network, one-stream network and our asymmetric Siamese network, respectively. colors represent networks in initialization and inference phrases, respectively. Diagrams (d)&(e) display comparisons of speed-precision trade-offs on CPU and Jetson AGX Xavier platforms. The parameters and FLOPs are represented by the area of circles in (d) and (e), respectively.

*Corresponding author.

have evolved to universally adopt a one-stream transformer, *beat the previous two-stream architecture*, dominating the tracking field. The success of this paradigm lies in its globally receptive feature extraction and the interaction between the template and search region across the model. Naturally, some efficient trackers follow these trackers and also adopt a one-stream architecture (as illustrated in Fig. 1 (b)), replacing the original computationally intensive components with lightweight backbones or modules. For example, building on MixFormer (Cui et al. 2022), MixFormerV2 (Cui et al. 2024) proposed a one-stream fully transformer framework that utilizes distillation and transformer compression techniques, achieving real-time CPU speed.

However, the prevalent one-stream network may not be optimal for efficient visual tracking, since 1) one-stream network redefines the template and search region branches as a unified structure, incorporating template computation in every video frame, which introduces significant redundancy. 2) the pervasive relation modeling between the template and search region overlooks the fact that this computationally intensive process imposes substantial stress on edge devices. To address these issues, we propose a novel asymmetric Siamese tracker named **AsymTrack** for efficient tracking. *We delve into the strengths and weaknesses of the two- and one-stream paradigms, enabling the designed AsymTrack to combine the efficiency of Siamese trackers with the precision benefits of one-stream trackers.* As shown in Fig. 1 (c), AsymTrack employs two separate branches, avoiding repeated template computations during inference. The template features are transformed into modulation signals and injected into the search region branch for relation modeling.

Specifically, we design an efficient template modulation (ETM) mechanism to establish the relation modeling in two-stream architecture, which eliminates the additional interaction module as in Siamese trackers and avoids the involvement of repeated template computation as in one-stream trackers. Furthermore, to overcome the limited representation capability in lightweight networks, we design an object perception enhancement (OPE) module. It efficiently merges abstract semantic features and local details into the base features. Through ingenious design, the OPE module can be flexibly re-parameterized as a single-layer convolution during inference, which improves precision while minimizing latency. As presented in Fig. 1, the proposed AsymTrack demonstrates excellent speed-precision trade-offs, while having fewer parameters and lower computational requirements compared to other competitors. For instance, compared to HiT-Tiny (Kang et al. 2023), AsymTrack-T achieves a 6.0% higher AUC precision on LaSOT (Fan et al. 2019) and exhibits faster speeds on both CPU and Jetson AGX Xavier. Relative to TransT (Chen et al. 2021), AsymTrack-B maintains a comparable AUC (64.7% vs. 64.9%), while operates 6.6 times faster on CPU and 4.7 times faster on AGX. Our contributions are threefold:

- We propose a novel asymmetric Siamese framework named AsymTrack. It combines the high efficiency and high precision of two- and one-stream trackers, surpassing the current prevailing one-stream pipeline and offering new insights for efficient tracking architecture.
- We propose an efficient template modulation (ETM) mechanism for relation modeling within our asymmetric Siamese tracking architecture, along with an object perception enhancement (OPE) module to boost the limited object representation capabilities of lightweight tracker.
- AsymTrack comprises a family of efficient trackers and extensive experiments demonstrate its effectiveness. Notably, the AsymTrack series, tailored for resource-constrained platforms, leads in both accuracy and speed compared to other state-of-the-art efficient trackers.

Related Works

Precision-Oriented Tracking

Siamese tracking pipeline (Bertinetto et al. 2016; Li et al. 2019; Zhang and Peng 2019; Chen et al. 2020, 2021; Yan et al. 2021a; Song et al. 2022; Lin et al. 2022; Liu et al. 2024) has long been dominant. In these models, the template and search region are fed into two branches with shared parameters to perform feature extraction, which are then processed together through designed interaction component to achieve fusion and target object matching. With the advent of the transformer (Vaswani et al. 2017; Dosovitskiy et al. 2021), some trackers, such as TransT (Chen et al. 2021) and SwinTrack (Lin et al. 2022), have incorporated more powerful transformer blocks for feature fusion or representation. Vision transformers serialize inputs into patch embeddings, allowing patches of different sizes to be concatenated along the token dimension, enabling joint modeling of the template and search region early in the model. Consequently, a series of trackers based on a one-stream framework, such as OSTrack (Ye et al. 2022), MixFormer (Cui et al. 2022), and others (Xie et al. 2022; Chen et al. 2022a; Wei et al. 2023; Gao, Zhou, and Zhang 2023), have refreshed the state-of-the-art performance and become the mainstream paradigm for precision-oriented tracking. Despite impressive performance, these methods are limited to desktop GPUs and often fail to meet speed requirements on resource-constrained platforms. For example, the high-performance tracker ARTrack (Wei et al. 2023) runs at just 9 FPS on CPUs and 8 FPS on AGX, failing to meet basic real-time requirements.

Efficiency-Oriented Tracking

Efficiency-oriented tracking has recently gained attention for its potential to propel trackers toward practical applications. The development of efficient tracking is closely tied to precision-oriented tracking. A common approach involves reducing the computational load of high-performance trackers by incorporating lightweight modules. Early trackers like ECO (Danelljan et al. 2017) and ATOM (Danelljan et al. 2019) designed lightweight structures to reduce computational complexity in the discriminative correlation filter model (Lukezic et al. 2017). Following Siamese tracking pipeline, numerous Siamese efficient trackers such as LightTrack (Yan et al. 2021b), ETTrack (Blatter et al. 2023), FEAR (Borsuk et al. 2022), SMAT (Gopal and Amer 2024), and LiteTrack (Wei et al. 2024) emerged. For instance, LightTrack (Yan et al. 2021b) utilized NAS (Pham

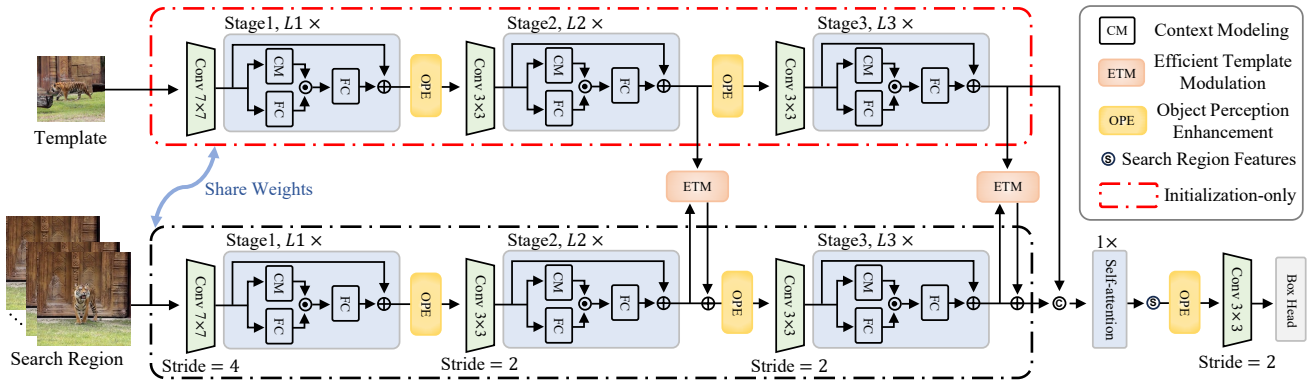

Figure 2: Overview of AsymTrack. It employs an asymmetric Siamese pipeline, where the template branch runs once during initialization, generating features and prototype that are unidirectionally fed to the search region branch for online inference.

et al. 2018) to discover lightweight backbone and head networks, significantly reducing the model's parameters and FLOPs. To improve efficiency, FEAR (Borsuk et al. 2022) introduced compact feature extraction and fusion blocks, while E.T.Track (Blatter et al. 2023) proposed a computationally friendly Exemplar Transformer for target localization. As one-stream architectures like OSTrack (Ye et al. 2022) and MixFormer (Cui et al. 2022) have demonstrated excellent performance in precision-oriented tracking, some works (Cui et al. 2024; Kang et al. 2023; Li et al. 2023b) have begun exploring high-speed tracking within the one-stream framework. Building on MixFormer (Cui et al. 2022), MixFormerV2 (Cui et al. 2024) lightens a one-stream fully transformer network through techniques such as distillation (Hinton, Vinyals, and Dean 2015), achieving real-time CPU speed. HiT (Kang et al. 2023) achieves impressive speeds on edge computing platforms by using lightweight hierarchical transformers and incorporating shallow features to compensate for information loss from large-stride downsampling. However, despite its success in precision-oriented tracking, the one-stream architecture faces significant limitations in efficient tracking. For instance, computing the template for every video frame during inference introduces considerable redundancy. Additionally, one-stream networks rely on ViT (Dosovitskiy et al. 2021; Graham et al. 2021; Yu et al. 2024) backbones, where quadratic complexity relative to input size and frequent transformer attention calculations present major challenges for edge computing devices. To this end, we delve into efficient tracker design and propose a novel asymmetric Siamese tracker that combines the strengths of both two- and one-stream trackers.

Methodology

Preliminaries and Notation

Siamese Tracking Pipeline. Given the target template $\boldsymbol{Z} \in \mathbb{R}^{H_z \times W_z \times 3}$ and search region $\boldsymbol{X} \in \mathbb{R}^{H_x \times W_x \times 3}$, the tracker aims to estimate the object bounding box $\boldsymbol{B} \in \mathbb{R}^4$ in $\boldsymbol{X}$. A typical Siamese tracker mainly consists of two symmetric feature encoder $\mathcal{F}(\cdot)$ branches with shared weights, a feature interaction module $\mathcal{I}(\cdot)$, and a box prediction head $\varphi(\cdot)$. In the initialization phase, $\boldsymbol{Z}$ is fed into $\mathcal{F}(\cdot)$ to generate the template features by $\boldsymbol{H}_Z = \mathcal{F}(\boldsymbol{Z})$. In the subsequent inference phase, $\mathcal{F}(\cdot)$ extracts features from $\boldsymbol{X}$, while $\boldsymbol{H}_Z$ is directly fed into $\mathcal{I}(\cdot)$ for interaction computation and $\varphi(\cdot)$ for box prediction. The subsequent inference process can be represented as:

$$\boldsymbol{B} = \varphi(\mathcal{I}(\mathcal{F}(\boldsymbol{X}), \boldsymbol{H}_Z)). \tag{1}$$

One-stream Tracking Pipeline. Relative to Siamese trackers, one-stream trackers have a simpler architecture, typically consisting of a backbone network $\mathcal{F}(\cdot)$ that simultaneously performs feature extraction and interaction, along with a box prediction head $\varphi(\cdot)$. The template and search region are first converted into patches through a patch embedding layer and flattened to 1D tokens $\boldsymbol{Z}_o \in \mathbb{R}^{N_z \times D}$ and $\boldsymbol{X}_o \in \mathbb{R}^{N_x \times D}$. These token sequences are then concatenated along the token dimension and fed into $\mathcal{F}(\cdot)$, which is generally a transformer network. Leveraging the global long-range modeling of transformers, the concatenated tokens undergo extensive interaction, substantially improving the accuracy of object modeling. The inference process can be described as:

$$\boldsymbol{B} = \varphi((\mathcal{F}(concat(\boldsymbol{Z}_o, \boldsymbol{X}_o)))). \tag{2}$$

Asymmetric Siamese Architecture

Asymmetric Siamese Structure. Siamese tracker has efficiency advantages due to the elimination of redundant template computations, but the insufficient relation modeling between the template and search region limits its performance. One-stream tracker demonstrates superior performance, but involving the template in every frame inference and adopting dense transformer attention layer with quadratic complexity hinder its deployment on edge computing platforms. Building on the above analysis, we propose an asymmetric Siamese tracking architecture that unites the speed of Siamese trackers with the superior performance of one-stream trackers. To avoid redundant template computation, we first adopt a two-stream structure, using separate encoders to extract features from the template and the search region. Since the architecture is not symmetrical, we denote these encoders as $\mathcal{F}_z$ and $\mathcal{F}_x$, respectively. Notably, $\mathcal{F}_z$ is a subset of the $\mathcal{F}_x$ network, so the weights

of $\mathcal{F}_z$ are integrated into and shared with $\mathcal{F}_x$. To compensate for the lack of relation modeling within the two-stream structure, we propose the concept of template modulation. During initialization, $\mathcal{F}_z$ extracts the template features and generates template prototype by $\boldsymbol{H}_z, \boldsymbol{P}_z = \mathcal{F}_z(\boldsymbol{Z})$. $\boldsymbol{P}_z$ is then unidirectionally fed into the search region branch for modulation purposes. This process facilitates cross-branch information communication throughout different stages of the backbone. The inference process of the asymmetric Siamese tracking pipeline can be described as:

$$\boldsymbol{B} = \varphi((\mathcal{F}_x(\boldsymbol{X}, \boldsymbol{H}_z, \boldsymbol{P}_z))). \tag{3}$$

Based on this pipeline, we present our efficient tracker AsymTrack, the overall framework is shown in Fig. 2.

Lightweight Backbone. We adopt a lightweight hierarchical model EfficientMod (Ma et al. 2024) as our backbone. To balance speed and accuracy, we selected the first three stages and integrated them into AsymTrack. As shown in Fig. 2, the backbone network is duplicated into two parameter-sharing branches. The input images first pass through a 7×7 convolutional layer for $4\times$ downsampling, followed by a 3×3 convolutional layer after each stage for $2\times$ downsampling. The i-th stage consists of L_i encoder blocks, each comprising a Context Modeling (CM) block and fully connected (FC) layers with a residual connection. The CM block, structured as FC-Conv-FC, aggregates visual contexts, while the parallel FC layer projects the input into a new space. They are fused by element-wise multiplication, mimicking the dynamics of self-attention, followed by a linear projection after fusion. Following three hierarchical encoder stages, and inspired by recent advances (Li et al. 2023a) combining convolution and attention, we added a transformer attention layer to enhance interaction between the template and search region. The attention layer is introduced only after the last stage, where the feature size is relatively small.

Box Prediction Head. We use a simple corner head to predict the object bounding box. Following STARK (Yan et al. 2021a) but in a more streamlined way, we feed the search region features into a few stacked Conv-BN-ReLU layers to estimate the target's top-left and bottom-right coordinates. Our box head requires no extra post-processing e.g., window penalty (Li et al. 2018).

Efficient Template Modulation

Interaction between the template and search region is pivotal for accurate tracking, as demonstrated by a series of one-stream trackers (Ye et al. 2022; Cui et al. 2022). However, designing an efficient interaction method in two-stream network without relying on dense and heavy transformer layers remains a challenge. To address this, we introduce an efficient template modulation mechanism, as shown in Fig. 3.

Prototype Generation. We first selectively extract key information from the template and search region features $\boldsymbol{H}_z \in \mathbb{R}^{h_z \times w_z \times C}$, $\boldsymbol{H}_x \in \mathbb{R}^{h_x \times w_x \times C}$ using a linear layer $\boldsymbol{W}_k \in \mathbb{R}^{C \times N}$ and aggregate it into the corresponding kernels, $\boldsymbol{S}_{z,x} \in \{\boldsymbol{S}_z, \boldsymbol{S}_x\}$ where $N<C$. Next, feature contraction is performed using a dot product operation, resulting in the corresponding prototypes $\boldsymbol{P}_z, \boldsymbol{P}_x \in \mathbb{R}^{N \times C}$. This process can be expressed as:

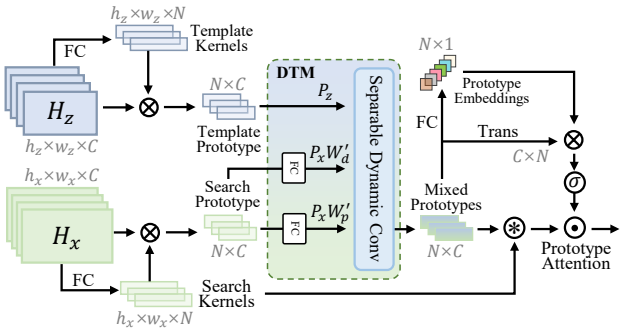

Figure 3: Efficient template modulation (ETM) mechanism.

$$\boldsymbol{P}_{z,x} = r(\boldsymbol{H}_{z,x} * \boldsymbol{W}_k) r(\boldsymbol{H}_{z,x})^\top, \tag{4}$$

where $r(\cdot)$ denotes flattening the spatial dimension from $h \times w$ to hw. Notably, template prototype generation can be completed during initialization to save latency.

Dynamic Template Modulation. Inspired by (Choromanski et al. 2021; Hu et al. 2023), we utilize 1D dynamic convolution to achieve the effect of multi-head cross-attention in transformer for prototype relation modeling, making the process more lightweight. The mixed prototypes after template modulation can be generated by:

$$\begin{aligned} \boldsymbol{P}_{mix} = DyConv_{1d}(\boldsymbol{P}_z, \boldsymbol{K}) = \boldsymbol{P}_z * \boldsymbol{K}, \\ \boldsymbol{K} = \boldsymbol{P}_x \boldsymbol{W}_c, \end{aligned} \tag{5}$$

where $\boldsymbol{W}_c$ is weight to dynamically generate the kernel $\boldsymbol{K}$ conditioned on $\boldsymbol{P}_x$. The standard convolution can be factorized into a depthwise convolution and a pointwise convolution (Howard et al. 2017). Following this manner, we further achieve dynamic template modulation in a separable form,

$$\begin{aligned} \boldsymbol{P}_{mix} &= SeDyConv_{1d}(\boldsymbol{P}_z, \boldsymbol{K}) \\ &= (\boldsymbol{P}_z * \boldsymbol{K}\boldsymbol{W}_d) * (\boldsymbol{K}\boldsymbol{W}_p) \\ &= (\boldsymbol{P}_z * \boldsymbol{P}_x\boldsymbol{W}_d^{'}) * (\boldsymbol{P}_x\boldsymbol{W}_p^{'}), \end{aligned} \tag{6}$$

where $\boldsymbol{W}_d^{'}$ and $\boldsymbol{W}_p^{'}$ are the weights of depthwise convolution and pointwise convolution. The modulated search cues $\boldsymbol{V}$ can be obtained by convolving $\boldsymbol{P}_{mix}$ with $\boldsymbol{S}_x$.

Prototype Attention. Finally, we designed prototype attention as a form of post attention to be applied to the modulated search features. A linear layer is applied to compress the mixed prototype to $N \times 1$ dimension prototype embeddings $\boldsymbol{P}_{mix}^{emb}$, which represent the global information in each prototype. We multiply $\boldsymbol{P}_{mix}^{emb}$ by the transposed prototype $\boldsymbol{P}_{mix}^{'}$ to generate an attention vector in $C \times 1$ dimension, which where be weighted to output modulated cues in channel dimension to emphasize important features and enhance representation. The calculation of the whole process is:

$$\boldsymbol{V}^{'} = \sigma(\boldsymbol{P}_{mix}^{'}\boldsymbol{P}_{mix}^{emb}) \odot \boldsymbol{V}, \tag{7}$$

where $\odot$ represents the broadcast Hadamard product and $\sigma(\cdot)$ indicates the Sigmoid function.

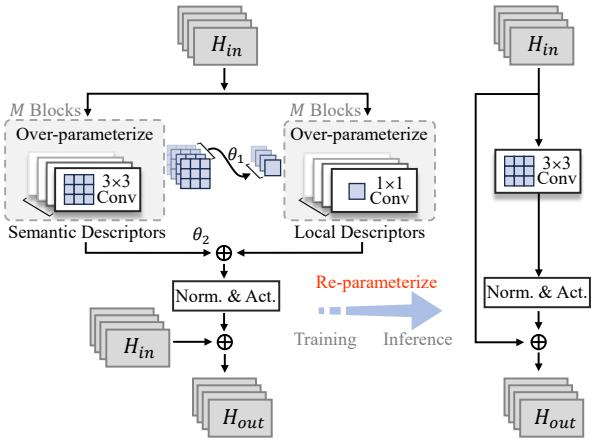

Figure 4: Object perception enhancement (OPE) module.

Object Perception Enhancement

Joint Semantic and Local Representation. To tackle the bottleneck of light-weight tracker's representation optimization, we propose an object perception enhancement (OPE) module, detailed in Fig. 4. High-level task encourages standard convolution to consolidate the abstract semantics (Li et al. 2010) and local cues can be extracted from gradient information (Su et al. 2021), OPE enhances backbone features by combining the representations of semantic and local descriptors, whch are achieved by 3×3 and 1×1 convolutional kernels, respectively. Specifically, the local descriptor is a parameter reusing mechanism generated by the linear transformation of the integrating gradient cues around shared semantic descriptors. The detail cues e.g., object boundary and texture cues can be captured by the local descriptors. Denote a single semantic descriptor as $\boldsymbol{W}_{sem} \in \mathbb{R}^{C_{out}\times C_{in}\times3\times3}$, where C_{out}/C_{in} is the output/input channel dimension, a local descriptor $\boldsymbol{W}_{loc}$ can be generated by,

$$\boldsymbol{W}_{loc} = -\theta_1 \sum_{p_n\in\mathbb{S}} \boldsymbol{W}_{sem}(p_n), \tag{8}$$

where p_n is the n-th weight value in 3×3 convolution $\mathbb{S}$. θ_1 is a learnable parameter projection factor. The enhanced representation is obtained by linearly weighting the extracted local features and semantic features with a weight θ_2. Furthermore, we utilize the over-parameterization technique (Guo, Alvarez, and Salzmann 2020) to further enhance perceptual performance. The semantic descriptor branch and the local descriptor branch are implemented as M parallel branches.

Re-parameterization Inference. For inference, we leverage the homogeneity and additivity of the convolutions to fold semantic and local descriptors into a single 3×3 convolution, and convert over-parameterization parallel branches into an equivalent single branch through a linear transformation $\boldsymbol{W} = \sum_{i=1}^{M} \boldsymbol{W}_i$, without performance degradation.

Optimization and Tracker Inference

Optimization. The training objective consists of a $\mathcal{L}_1$ loss and GIoU loss (Rezatofighi et al. 2019) $\mathcal{L}_G$,

$$\mathcal{L} = \lambda_1\mathcal{L}_1(\boldsymbol{B}, \boldsymbol{B}_{gt}) + \lambda_G\mathcal{L}_G(\boldsymbol{B}, \boldsymbol{B}_{gt}), \tag{9}$$

where $\boldsymbol{B}_{gt}$ is the ground truth, $\lambda_1 = 5$ and $\lambda_G = 2$.

Tracker Inference. AsymTrack features an asymmetric Siamese structure. The template branch runs only once during initialization, after which only the search region branch is needed for inference, with the template's extracted features and modulation cues injected.

Model		AsymTrack-T	AsymTrack-S	AsymTrack-B
Encoder Blocks		[2, 2, 1]	[2, 2, 3]	[2, 2, 3]
Input Sizes		[128, 256]	[128, 256]	[192, 384]
Inference Speed (FPS)	GPU	224	200	197
	CPU	81	75	38
	AGX	84	78	64
Params (M)		3.05	3.36	3.36
FLOPs (G)		0.7	0.8	1.8

Table 1: Detailed configurations of our AsymTrack variants.

Experiments

Implementation Details

AsymTrack Model Family. We present three variants of the AsymTrack model: AsymTrack-T, AsymTrack-S, and AsymTrack-B. Tab. 1 details their configurations, also including parameters, FLOPs, and inference speeds across different platforms: GPU (Nvidia 2080ti), CPU (Intel i7-9700KF@3.6G Hz), and edge device (Jetson AGX Xavier).

Training Details. We used training splits of four datasets for training, including LaSOT (Fan et al. 2019), TrackingNet (Muller et al. 2018), COCO2017 (Lin et al. 2014), and GOT10K (Huang, Zhao, and Huang 2019). Common augmentation such as flipping and jittering are applied. The template and search region images are resized to 128×128 and 256×256 for AsymTrack-T and AsymTrack-S, and to 192×192 and 384×384 for AsymTrack-B. We trained the model for 500 epochs using the AdamW (Loshchilov and Hutter 2018) optimizer with an initial learning rate of 4e-4 on 2 NVIDIA A800 GPUs, with each epoch consisting of 60,000 randomly sampled image pairs.

Comparison with State-of-the-arts

We conduct a comprehensive comparison across seven widely used benchmarks. Trackers are categorized as either real-time or non-real-time based on their speed (20 FPS) on the Jetson AGX Xavier, in accordance with the VOT (Kristan et al. 2021) real-time criteria.

GOT-10k. GOT-10k (Huang, Zhao, and Huang 2019) is a large-scale tracking dataset with over 10,000 video sequences featuring diverse objects and scenes. As shown in Tab. 2, AsymTrack-B achieves the highest real-time AO score of 67.7%. It surpasses the recent efficient tracker HiT-Base (Kang et al. 2023) by a large margin of 3.7% while achieving faster speeds across all test platforms. Besides, our fastest variant, AsymTrack-T, stands out as the fastest among all real-time trackers, with comparable precision.

LaSOT. LaSOT (Fan et al. 2019) is a large-scale long-term benchmark comprising 1,400 video sequences, each averaging over 2,500 frames, with 280 sequences reserved for testing. As shown in Tab. 2, AsymTrack-B achieved the best

	Method	GOT-10k			LaSOT			TrackingNet			PyTorch Speed (*fps*)		
		AO	$SR_{0.5}$	$SR_{0.75}$	AUC	P_{Norm}	P	AUC	P_{Norm}	P	GPU	CPU	AGX
Real-time	AsymTrack-B (ours)	**67.7**	**76.6**	**61.4**	**64.7**	73.0	67.8	**80.0**	**84.5**	**77.4**	197	38	64
	AsymTrack-S (ours)	65.5	74.8	58.9	62.8	71.2	64.8	77.9	82.2	74.0	200	**75**	**78**
	AsymTrack-T (ours)	62.3	71.3	54.7	60.8	68.7	61.2	76.2	80.9	71.6	**224**	**81**	**84**
	HiT-Base (Kang et al. 2023)	64.0	72.1	58.1	64.6	**73.3**	**68.1**	**80.0**	84.4	77.3	175	33	61
	TCTrack (Cao et al. 2022)	66.2	75.6	61.0	60.5	69.3	62.4	74.8	79.6	73.3	140	45	41
	MixFormerV2 (Cui et al. 2024)	61.9	71.7	51.3	60.6	69.9	60.4	75.8	81.1	70.4	167	45	49
	HCAT (Chen et al. 2022b)	65.1	76.5	56.7	59.3	68.7	61.0	76.6	82.6	72.9	195	45	55
	HiT-Small (Kang et al. 2023)	62.6	71.2	54.4	60.5	68.3	61.5	77.7	81.9	73.1	192	72	68
	E.T.Track (Blatter et al. 2023)	-	-	-	59.1	-	-	75.0	80.3	70.6	40	47	20
	FEAR (Borsuk et al. 2022)	61.9	72.2	-	53.5	-	54.5	-	-	-	105	60	38
	LightTrack (Yan et al. 2021b)	61.1	71.0	-	53.8	-	53.7	72.5	77.8	69.5	128	41	36
	ATOM (Danelljan et al. 2019)	55.6	63.4	40.2	51.5	57.6	50.5	70.3	77.1	64.8	83	18	22
	HiT-Tiny (Kang et al. 2023)	52.6	59.3	42.7	54.8	60.5	52.9	74.6	78.1	68.8	**204**	**76**	**77**
	ECO (Danelljan et al. 2017)	31.6	30.9	11.1	32.4	33.8	30.1	55.4	61.8	49.2	**240**	15	39
Non-real-time	ARTrack (Wei et al. 2023)	73.5	82.2	70.9	70.4	79.5	76.6	84.2	88.7	83.5	26	9	8
	MixFormer-L (Cui et al. 2022)	75.6	85.7	72.8	70.1	79.9	76.3	83.9	88.9	83.1	18	-	-
	TransT (Chen et al. 2021)	72.3	82.4	68.2	64.9	73.8	69.0	81.4	86.7	80.3	63	5	13
	OSTrack-256 (Ye et al. 2022)	71.0	80.4	68.2	69.1	78.7	75.2	83.1	87.8	82.0	105	11	19
	Sim-B/16 (Chen et al. 2022a)	68.6	78.9	62.4	69.3	78.5	-	82.3	-	86.5	87	10	16
	STARK-ST50 (Yan et al. 2021a)	68.0	77.7	62.3	66.6	-	-	81.3	86.1	-	50	7	13
	TrSiam (Wang et al. 2021)	67.3	78.7	58.6	62.4	-	60.6	78.1	82.9	72.7	40	5	10
	DiMP (Bhat et al. 2019)	61.1	71.7	49.2	56.9	65.0	56.7	74.0	80.1	68.7	77	10	17
	SiamFC++ (Xu et al. 2020)	59.5	69.5	47.9	54.5	-	54.7	75.4	80.0	70.5	-	12	-
	SiamRPN++ (Li et al. 2019)	51.7	61.6	32.5	49.6	56.9	49.1	73.3	80.0	69.4	56	4	11

Table 2: State-of-the-art comparison on the TrackingNet, LaSOT, and GOT-10k benchmarks. The top two real-time results are highlighted in **bold** and underlined, respectively. The top three speed across different platforms are highlighted in **bold**.

AUC score of 64.7% and outperformed the previous state-of-the-art HiT-Base (Kang et al. 2023) in speed across all platforms. While AsymTrack-T leads in speed on the AGX (84 FPS), it ranks fourth among real-time trackers. Notably, it outperformed HiT-Tiny by 6.0% in AUC, 8.2% in normalized precision, and 6.7% in precision.

TrackingNet. TrackingNet (Muller et al. 2018) is a large-scale short-term benchmark with 511 test video sequences, covering a wide range of object categories and scenes. As shown in Tab. 2, AsymTrack-B delivered top-tier performance with an AUC of 80.0%, normalized Precision of 84.5%, and precision of 77.4%. The smaller variant, AsymTrack-S, is also highly competitive, surpassing the latest one-stream tracker MixFormerV2 (Cui et al. 2024) by 2.1% in AUC and 3.6% in precision.

Speed Comparison. We conducted speed tests across three different platforms. As we can see, ECO (Danelljan et al. 2017) and our AsymTrack-T are the two fastest trackers, with AsymTrack-T achieving 224 FPS on GPU, 81 FPS on CPU, and 84 FPS on AGX. On resource-constrained platforms like the CPU and AGX, AsymTrack-T outperforms ECO in speed while far surpassing it in precision. The smaller variant, AsymTrack-S, is 1.2× faster on GPU, 1.7× faster on CPU, and 1.6× faster on AGX compared to the recent one-stream tracker MixFormerV2-S (Cui et al. 2024). The base variant, AsymTrack-B, is 22 FPS faster on GPU, 5 FPS faster on CPU, and 3 FPS faster on AGX compared to its counterpart, HiT-Base (Kang et al. 2023). In summary, AsymTrack delivers impressive speed across multiple platforms, making it highly suitable for application scenarios e.g., UAVs and embodied robots.

NFS. NFS (Kiani Galoogahi et al. 2017) is a high-frame-rate dataset focused on fast-motion object scenarios. As shown in Tab. 3, on 30 FPS version of NFS, our tracker excels in these challenging conditions, achieving the top two AUC scores.

UAV123. UAV123 (Mueller, Smith, and Ghanem 2016) aims to focus on the challenges unique to UAV-based tracking. AsymTrack-B achieves the highest AUC score of 66.5% As shown in Tab. 3, outperforming MixFormerV2 (Kang et al. 2023) by 0.7% and HiT-Base (Kang et al. 2023) by 0.9%.

LaSOT$_{ext}$. LaSOT$_{ext}$ (Fan et al. 2021), an extension of LaSOT for more challenging tracking evaluations, further demonstrates the strength of our approach. AsymTrack-B ranks first with an AUC of 44.6%, and AsymTrack-T outperforms HiT-Small by 2.1% with a speed advantage.

	Method	NFS	UAV123	LaSOT$_{ext}$
Real-time	AsymTrack-B (ours)	64.4	**66.5**	**44.6**
	AsymTrack-S (ours)	**64.9**	65.6	43.3
	AsymTrack-T (ours)	63.3	64.6	42.5
	MixFormerV2 (Cui et al. 2024)	-	65.8	43.6
	HiT-Base (Kang et al. 2023)	63.6	65.6	44.1
	HCAT (Chen et al. 2022b)	63.5	62.7	-
	HiT-Small (Kang et al. 2023)	61.8	63.3	40.4
	E.T.Track (Blatter et al. 2023)	59.0	62.3	-
	FEAR (Borsuk et al. 2022)	61.4	-	-
	ATOM (Danelljan et al. 2019)	58.4	64.2	37.6
	LightTrack (Yan et al. 2021b)	55.3	62.5	-
	HiT-Tiny (Kang et al. 2023)	53.2	58.7	35.8
	ECO (Danelljan et al. 2017)	46.6	53.2	22.0
Non-real-time	GRM (Sun et al. 2023)	65.6	70.2	-
	ARTrack (Wei et al. 2023)	64.3	67.7	46.4
	OSTrack-256 (Ye et al. 2022)	64.7	68.3	47.4
	TrSiam (Wang et al. 2021)	65.8	67.4	-
	TransT (Chen et al. 2021)	65.7	69.1	-
	PrDiMP (Danelljan et al. 2020)	63.5	68.0	-
	DiMP (Bhat et al. 2019)	62.0	65.3	39.2
	SiamRPN++ (Li et al. 2019)	50.2	61.6	34.0

Table 3: State-of-the-art comparison on more benchmarks.

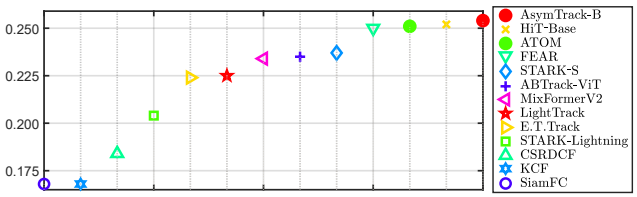

Figure 5: VOT real-time testing on Jetson AGX Xavier.

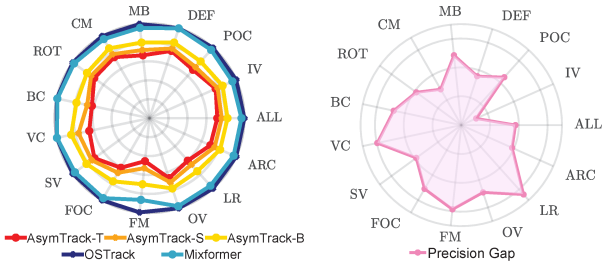

Figure 6: Gap analysis between AsymTrack and precision-oriented trackers across different attributes on LaSOT.

VOT2021. We also conducted real-time experiments on VOT2021 (Kristan et al. 2021) challenge benchmark running on the edge device of Jetson AGX Xavier. As shown in Fig.5, AsymTrack-B achieves 25.4% in terms of EAO score, surpassing all other real-time efficient trackers.

Exploration Studies

We further conduct experiments to explore the characteristics of our AsymTrack. LaSOT (Fan et al. 2019) and GOT-10k (Huang, Zhao, and Huang 2019) are employed as the evaluation datasets. AsymTrack-S is used as the baseline model by default, and unless otherwise stated, the experimental settings are kept the same as the baseline.

Gap Analysis with Precision-Oriented Trackers. We compare AsymTrack with precision-oriented trackers and, as shown in Fig. 6, there is still a significant performance gap between AsymTrack and alternatives like OSTrack (Ye et al. 2022). The right graph shows the average AUC gap between AsymTrack variants and precision-oriented trackers across 14 attributes. The largest gaps appear in low resolution, viewpoint change, and fast motion, where the model's representation capability is more challenged. We hope future designs for efficient tracking will further narrow the gap.

Necessity of Template Modulation. To validate the effectiveness of template modulation, we compare AsymTrack with models without this feature and evaluate their performance and speed. In Tab. 4, model#2 and model#4 outperform model#1 with AUC gains of 2.5%/4.0% on GOT-10k and 2.4%/3.7% on LaSOT, confirming the benefits of template modulation. Moreover, the speed remains high across all platforms, reflecting a good speed-precision trade-off.

Effectiveness of Perception Enhancement. To further validate the effectiveness of OPE, we conducted another comparative analysis between AsymTrack and models lacking this enhancement. The results in Tab. 4 (model#1 and model#3) show that incorporating OPE improves the

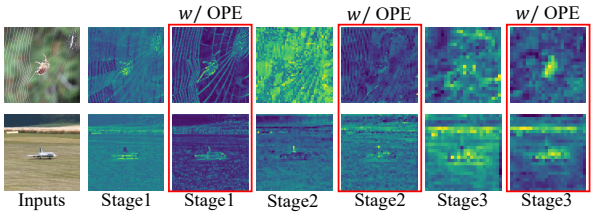

Figure 7: Visualization maps between w/o and $w/$ OPE.

Model	Components		Model Speed (FPS)			GOT-10k	LaSOT
	ETM	OPE	GPU	CPU	AGX		
1	✘	✘	260	95	100	60.4	58.8
2	✔	✘	220	80	82	62.9	61.2
3	✘	✔	240	86	92	62.1	60.7
4	✔	✔	200	75	78	64.4	62.5

Table 4: Component-wise study on performance and speed.

AO on GOT-10k by 1.7% and the AUC on LaSOT by 1.9%. More importantly, thanks to the introduction of re-parameterization inference, the OPE module adds minimal latency across different platforms. Fig. 7 also demonstrates that with our OPE, more discriminative features, particularly crucial detail cues, are obtained across different stages.

Ablation on ETM Designs. As shown in Tab. 5 (a), we explore different ETM designs to validate its effectiveness. Replacing our DTM with vanilla multi-head cross attention (MHCA) yields baseline-level performance but with lower efficiency. Prototype attention in ETM also proves effective. Adding ETM in early stage1 brings little improvement, suggesting early interaction has limited benefit.

Location Analysis of OPE. We further investigate the effect of placing OPE at different stages of the model. As shown in Tab. 5 (b), applying perception enhancement in the early stages is more effective than doing so at later stages, and applying it at every stage yields the best results.

Method	GOT-10k	LaSOT
Baseline	64.4	62.5
DTM→MHCA	64.1	62.6
− Prototype Att	63.9	62.3
+ Stage1 ETM	64.6	62.4

(a) Ablation on ETM designs

Method	GOT-10k	LaSOT
Baseline	64.4	62.5
{*s1*,*s2*}	64.2	62.2
{*s1*}	63.9	61.9
{*s3*}	63.3	61.4

(b) Different locations for OPE

Table 5: Ablation Study of ETM designs and OPE location.

Conclusion

In this work, we present AsymTrack, a new family of efficient tracking models. Departing from the prevalent one-stream architectures, AsymTrack utilizes a novel asymmetric Siamese framework that integrates the efficiency of two-stream trackers with the performance benefits of one-stream designs. AsymTrack broadens the possibilities for real-time visual tracking on resource-constrained platforms, offering a viable alternative to one-stream architectures. We envision the AsymTrack family becoming a dependable visual tracking solution for real-world deployment, bridging the gap between academic research and industrial applications.

Acknowledgments

The paper is supported by the National Natural Science Foundation of China under grant No. 62293540, 62293542, 62106149, Liao Ning Province Science and Technology Plan No.2023JH26/10200016 and Dalian City Science and Technology Innovation Fund No. 2023JJ11CG001.

References

Bertinetto, L.; Valmadre, J.; Henriques, J. F.; Vedaldi, A.; and Torr, P. H. 2016. Fully-convolutional siamese networks for object tracking. In *ECCVW*, 850–865.

Bhat, G.; Danelljan, M.; Van Gool, L.; and Timofte, R. 2019. Learning discriminative model prediction for tracking. In *ICCV*, 6182–6191.

Blatter, P.; Kanakis, M.; Danelljan, M.; and Van Gool, L. 2023. Efficient visual tracking with exemplar transformers. In *WACV*, 1571–1581.

Borsuk, V.; Vei, R.; Kupyn, O.; Martyniuk, T.; Krashenyi, I.; and Matas, J. 2022. FEAR: Fast, efficient, accurate and robust visual tracker. In *ECCV*, 644–663.

Cao, Z.; Huang, Z.; Pan, L.; Zhang, S.; Liu, Z.; and Fu, C. 2022. TCTrack: Temporal contexts for aerial tracking. In *CVPR*, 14798–14808.

Chen, B.; Li, P.; Bai, L.; Qiao, L.; Shen, Q.; Li, B.; Gan, W.; Wu, W.; and Ouyang, W. 2022a. Backbone is All Your Need: A Simplified Architecture for Visual Object Tracking. In *ECCV*, 375–392.

Chen, X.; Kang, B.; Wang, D.; Li, D.; and Lu, H. 2022b. Efficient visual tracking via hierarchical cross-attention transformer. In *ECCVW*, 461–477.

Chen, X.; Peng, H.; Wang, D.; Lu, H.; and Hu, H. 2023. Seqtrack: Sequence to sequence learning for visual object tracking. In *CVPR*, 14572–14581.

Chen, X.; Yan, B.; Zhu, J.; Wang, D.; Yang, X.; and Lu, H. 2021. Transformer tracking. In *CVPR*, 8126–8135.

Chen, Z.; Zhang, L.; Hu, P.; Lu, H.; and He, Y. 2024. MaskTrack: Auto-Labeling and Stable Tracking for Video Object Segmentation. *TNNLS*.

Chen, Z.; Zhong, B.; Li, G.; Zhang, S.; and Ji, R. 2020. Siamese box adaptive network for visual tracking. In *CVPR*, 6668–6677.

Choromanski, K.; Likhosherstov, V.; Dohan, D.; Song, X.; Gane, A.; Sarlos, T.; Hawkins, P.; Davis, J.; Mohiuddin, A.; Kaiser, L.; et al. 2021. Rethinking attention with performers. In *ICLR*, 1–38.

Cui, Y.; Jiang, C.; Wang, L.; and Wu, G. 2022. Mixformer: End-to-end tracking with iterative mixed attention. In *CVPR*, 13608–13618.

Cui, Y.; Song, T.; Wu, G.; and Wang, L. 2024. Mixformerv2: Efficient fully transformer tracking. *NeurIPS*, 36: 58736–58751.

Danelljan, M.; Bhat, G.; Khan, F. S.; and Felsberg, M. 2017. ECO: Efficient Convolution Operators for Tracking. In *CVPR*, 6638–6646.

Danelljan, M.; Bhat, G.; Khan, F. S.; and Felsberg, M. 2019. Atom: Accurate tracking by overlap maximization. In *CVPR*, 4660–4669.

Diao, H.; Zhang, Y.; Gao, S.; Zhu, J.; Chen, L.; and Lu, H. 2024. Gssf: Generalized structural sparse function for deep cross-modal metric learning. *TIP*.

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N. 2021. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. *ICLR*, 1–12.

Fan, H.; Bai, H.; Lin, L.; Yang, F.; Chu, P.; Deng, G.; Yu, S.; Huang, M.; Liu, J.; Xu, Y.; et al. 2021. Lasot: A high-quality large-scale single object tracking benchmark. *IJCV*, 129(2): 439–461.

Fan, H.; Lin, L.; Yang, F.; Chu, P.; Deng, G.; Yu, S.; Bai, H.; Xu, Y.; Liao, C.; and Ling, H. 2019. LaSOT: A High-Quality Benchmark for Large-Scale Single Object Tracking. In *CVPR*, 5374–5383.

Gao, S.; Zhou, C.; and Zhang, J. 2023. Generalized relation modeling for transformer tracking. In *CVPR*, 18686–18695.

Gopal, G. Y.; and Amer, M. A. 2024. Separable self and mixed attention transformers for efficient object tracking. In *WACV*, 6708–6717.

Graham, B.; El-Nouby, A.; Touvron, H.; Stock, P.; Joulin, A.; Jégou, H.; and Douze, M. 2021. Levit: a vision transformer in convnet's clothing for faster inference. In *ICCV*, 12259–12269.

Guo, S.; Alvarez, J. M.; and Salzmann, M. 2020. Expandnets: Linear over-parameterization to train compact convolutional networks. *NeurIPS*, 33: 1298–1310.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Howard, A. G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; and Adam, H. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*.

Hu, J.; Huang, L.; Ren, T.; Zhang, S.; Ji, R.; and Cao, L. 2023. You only segment once: Towards real-time panoptic segmentation. In *CVPR*, 17819–17829.

Huang, L.; Zhao, X.; and Huang, K. 2019. GOT-10k: A large high-diversity benchmark for generic object tracking in the wild. *TPAMI*, 43(05): 1562–1577.

Kang, B.; Chen, X.; Wang, D.; Peng, H.; and Lu, H. 2023. Exploring lightweight hierarchical vision transformers for efficient visual tracking. In *ICCV*, 9612–9621.

Kiani Galoogahi, H.; Fagg, A.; Huang, C.; Ramanan, D.; and Lucey, S. 2017. Need for speed: A benchmark for higher frame rate object tracking. In *ICCV*, 1125–1134.

Kristan, M.; Matas, J.; Leonardis, A.; Felsberg, M.; Pflugfelder, R.; Kämäräinen, J.-K.; Chang, H. J.; Danelljan, M.; Cehovin, L.; Lukežič, A.; et al. 2021. The ninth visual

object tracking vot2021 challenge results. In *ICCVW*, 2711–2738.

Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; and Yan, J. 2019. SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks. In *CVPR*, 4282–4291.

Li, B.; Yan, J.; Wu, W.; Zhu, Z.; and Hu, X. 2018. High Performance Visual Tracking With Siamese Region Proposal Network. In *CVPR*, 8971–8980.

Li, J.; Hassani, A.; Walton, S.; and Shi, H. 2023a. Convmlp: Hierarchical convolutional mlps for vision. In *CVPR*, 6307–6316.

Li, L.-J.; Su, H.; Fei-Fei, L.; and Xing, E. 2010. Object bank: A high-level image representation for scene classification & semantic feature sparsification. *NeurIPS*, 23.

Li, S.; Yang, Y.; Zeng, D.; and Wang, X. 2023b. Adaptive and background-aware vision transformer for real-time uav tracking. In *ICCV*, 13989–14000.

Lin, L.; Fan, H.; Zhang, Z.; Wang, Y.; Xu, Y.; and Ling, H. 2024. Tracking meets lora: Faster training, larger model, stronger performance. In *ECCV*, 300–318. Springer.

Lin, L.; Fan, H.; Zhang, Z.; Xu, Y.; and Ling, H. 2022. Swintrack: A simple and strong baseline for transformer tracking. In *NeurIPS*, 16743–16754.

Lin, T.-Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; and Zitnick, C. L. 2014. Microsoft COCO: Common objects in context. In *ECCV*, 740–755.

Liu, C.; Yuan, Y.; Chen, X.; Lu, H.; and Wang, D. 2024. Spatial-temporal initialization dilemma: towards realistic visual tracking. *Visual Intelligence*, 35. https://link.springer.com/article/10.1007/s44267-024-00068-5.

Loshchilov, I.; and Hutter, F. 2018. Decoupled Weight Decay Regularization. In *ICLR*, 1–8.

Lukezic, A.; Vojir, T.; ˇCehovin Zajc, L.; Matas, J.; and Kristan, M. 2017. Discriminative correlation filter with channel and spatial reliability. In *CVPR*, 6309–6318.

Ma, X.; Dai, X.; Yang, J.; Xiao, B.; Chen, Y.; Fu, Y.; and Yuan, L. 2024. Efficient Modulation for Vision Networks. *ICLR*, 1–19.

Mueller, M.; Smith, N.; and Ghanem, B. 2016. A benchmark and simulator for UAV tracking. In *European conference on computer vision*, 445–461.

Muller, M.; Bibi, A.; Giancola, S.; Alsubaihi, S.; and Ghanem, B. 2018. TrackingNet: A large-scale dataset and benchmark for object tracking in the wild. In *ECCV*, 300–317.

Pham, H.; Guan, M.; Zoph, B.; Le, Q.; and Dean, J. 2018. Efficient neural architecture search via parameters sharing. In *ICML*, 4095–4104. PMLR.

Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I. D.; and Savarese, S. 2019. Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In *CVPR*, 658–666.

Song, Z.; Yu, J.; Chen, Y.-P. P.; and Yang, W. 2022. Transformer Tracking with Cyclic Shifting Window Attention. In *CVPR*, 8791–8800.

Su, Z.; Liu, W.; Yu, Z.; Hu, D.; Liao, Q.; Tian, Q.; Pietikäinen, M.; and Liu, L. 2021. Pixel difference networks for efficient edge detection. In *ICCV*, 5117–5127.

Sun, M.; Wang, P.; Xu, J.; Li, X.; and Di, R. 2023. GRM: Gaussian response module for visual tracking. *Displays*, 79: 102509.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *NeurIPS*, 5998–6008.

Wang, N.; Zhou, W.; Wang, J.; and Li, H. 2021. Transformer meets tracker: Exploiting temporal context for robust visual tracking. In *CVPR*, 1571–1580.

Wei, Q.; Zeng, B.; Liu, J.; He, L.; and Zeng, G. 2024. LiteTrack: Layer pruning with asynchronous feature extraction for lightweight and efficient visual tracking. In *ICRA*, 4968–4975. IEEE.

Wei, X.; Bai, Y.; Zheng, Y.; Shi, D.; and Gong, Y. 2023. Autoregressive visual tracking. In *CVPR*, 9697–9706.

Xie, F.; Wang, C.; Wang, G.; Cao, Y.; Yang, W.; and Zeng, W. 2022. Correlation-aware deep tracking. In *CVPR*, 8751–8760.

Xu, Y.; Wang, Z.; Li, Z.; Yuan, Y.; and Yu, G. 2020. SiamFC++: Towards Robust and Accurate Visual Tracking with Target Estimation Guidelines. In *AAAI*, 12549–12556.

Yan, B.; Peng, H.; Fu, J.; Wang, D.; and Lu, H. 2021a. Learning spatio-temporal transformer for visual tracking. In *ICCV*, 10448–10457.

Yan, B.; Peng, H.; Wu, K.; Wang, D.; Fu, J.; and Lu, H. 2021b. Lighttrack: Finding lightweight neural networks for object tracking via one-shot architecture search. In *CVPR*, 15180–15189.

Ye, B.; Chang, H.; Ma, B.; Shan, S.; and Chen, X. 2022. Joint feature learning and relation modeling for tracking: A one-stream framework. In *ECCV*, 341–357.

Yu, C.; Liu, X.; Wang, Y.; Zhang, P.; and Lu, H. 2024. TF-CLIP: Learning text-free CLIP for video-based person reidentification. In *AAAI*, 6764–6772.

Zhang, Z.; and Peng, H. 2019. Deeper and wider siamese networks for real-time visual tracking. In *CVPR*, 4591–4600.

Zhu, J.; Lai, S.; Chen, X.; Wang, D.; and Lu, H. 2023. Visual prompt multi-modal tracking. In *CVPR*, 9516–9526.