Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

SSTrack: Sample-interval Scheduling for Lightweight Visual Object Tracking

Yutong Kou[1,2,3], **Shubo Lin**[1,2,3], **Liang Li**[5], **Bing Li**[1,3,6], **Weiming Hu**[1,2,3,4] and **Jin Gao**[1,2,3*]
[1]State Key Laboratory of Multimodal Artificial Intelligence Systems (MAIS), CASIA
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
[3]Beijing Key Laboratory of Super Intelligent Security of Multi-Modal Information
[4]School of Information Science and Technology, ShanghaiTech University
[5]Beijing Institute of Basic Medical Sciences
[6]People AI, Inc
{kouyutong2021,linshubo2023}@ia.ac.cn, {jin.gao,bli,wmhu}@nlpr.ia.ac.cn, liang.li.brain@aliyun.com

Abstract

In recent years, CPU real-time object tracking has gained significant attention due to its broad applications such as UAV-tracking. To maintain computational efficiency, most existing CPU real-time object trackers rely on lightweight backbones and employ a single initial template image without intermediate online templates. Although the appearance variance between the template and the search is larger under this single template setting, the representation ability of lightweight backbones is weaker which poses a challenge when training lightweight object trackers. To address this issue, we propose SSTrack, a new easier-to-harder training schedule for the lightweight object tracker. From the data perspective, our method designed a success-aware sample scheduler that gradually increases difficult training samples with longer template-search time intervals and reduces the amount of the easier samples so the training cost remains unchanged. From the optimization perspective, we utilized a gradient scaling strategy that retains the original training objective of easier samples despite the reduction in their quantities. With the collective effort from both perspectives, our method achieves State-of-the-Art CPU-real-time accuracy on 5 UAV-tracking benchmarks and 5 general object tracking benchmarks. Codes and models will be available at https://github.com/Kou-99/SSTrack.

1 Introduction

Visual object tracking is a fundamental computer vision problem that aims to locate an arbitrary object in a video sequence given its first frame location. With the advance of deep learning techniques, visual object tracking has seen significant progress in recent years. Most deep-learning object trackers follow a template matching paradigm [Bertinetto *et al.*, 2016] that matches the search region and the first frame template with a deep neural network (DNN). Although this simple paradigm proves to be effective, it also poses a high demand on the representation quality of the DNN, especially when the time interval between the template image and the search image is large. To solve this issue, some methods introduce more complicated matching networks like transformer matching [Chen *et al.*, 2021], learning-based matching [Zhang *et al.*, 2021], one-stream matching [Ye *et al.*, 2022], etc. Another approach is leveraging different kinds of temporal information, including dynamic online template [Yan *et al.*, 2021a], temporal memory readout [Fu *et al.*, 2021b], spatiotemporal prompts [Wei *et al.*, 2023], dense temporal token [Zheng *et al.*, 2024], etc.

Despite the high performance, the complex network and the additional temporal information inevitably bring heavy computational overhead which hinders their application in real-world applications like UAV-tracking. To satisfy real-world applications, many strategies are proposed to improve the efficiency of object trackers, including model design [Yan *et al.*, 2021b; Kang *et al.*, 2023; Cao *et al.*, 2021; Gopal and Amer, 2024], model distillation [Cui *et al.*, 2024], dynamic inference [Li *et al.*, 2023; Li *et al.*, 2024], pre-

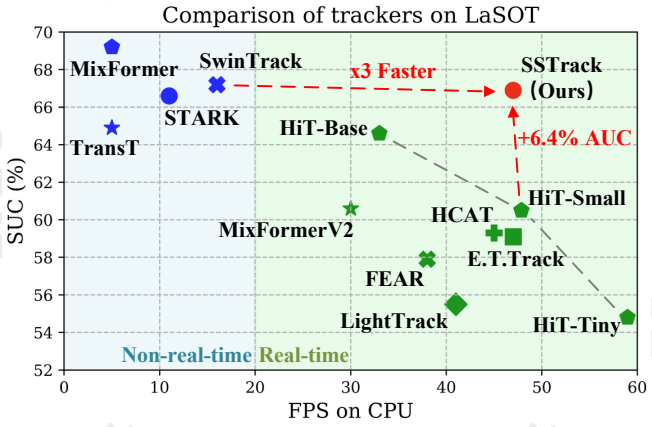

Figure 1: Comparison of the proposed SSTrack with other trackers on LaSOT in terms of CPU tracking speed and success score (SUC). Our method pushes forward the speed-accuracy trade-off by a large margin.

*Corresponding author

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

processing [Kou *et al.*, 2023]. Although the strategies are diverse, most of these methods only use one template from the initial frame and employ a lightweight backbone to ensure high efficiency. Without temporal context, this single template setting poses a high demand for representation quality, especially when the time interval between the template and search image is large. However, lightweight backbones are usually more struggled to capture large appearance variations compared with bigger backbones. This conflict between the hard training setting and weak model capability can be troublesome during training. Thus, it is natural to ask the question: ***Is it possible to customize a training schedule for lightweight trackers to compensate for their weaker representation ability?***

To this end, we proposed a customized training framework, termed **S**ample-Interval **S**cheduling **Track** (SSTrack), for lightweight object trackers that emphasize compensating for the weaker representation ability of lightweight models via easier-to-harder sample schedule. As shown in Fig3, the proposed framework handles the challenge from both data and optimization perspectives. From the data perspective, we identify training samples with longer template-search time intervals as harder samples due to their larger appearance variance. As a result, the training difficulty can be adjusted by changing the sampling frequency of samples with different template-search time intervals. To determine the ideal sampling frequency for every epoch, we proposed a novel success-aware sample scheduler that considers both the model's learning progress and the predefined easier-to-harder behavior. The success-aware sample scheduler only uses the latest epoch's training statistics so no significant overhead is introduced. To keep the training cost unchanged, the number of easier short-interval samples is inevitably reduced. From the optimization perspective, we employed a gradient scaling strategy to compensate for this reduction. By up-scaling the gradient of samples with reduced sampling frequency, the original training objective on easier samples can be retained. The gradient of samples with increased sampling frequency is unchanged to enjoy the benefit of increased harder examples. The collective effort from both data and optimization perspectives enables our method to achieve State-Of-The-Art CPU-real-time tracking performance on 5 general object tracking benchmarks and 5 UAV-tracking benchmarks.

Our contribution can be concluded as follows: (**i**) We analyze the difficulty of training lightweight trackers on samples with long template-search time intervals at early training stages owing to the weaker representation ability of lightweight backbones; (**ii**) We designed a success-aware sample scheduler that gradually introduces more hard long-interval samples based on both the model's learning progress and the predefined easier-to-harder behavior. To compensate for the reduced easy samples, we additionally employed a gradient scaling strategy that up-scales the gradient of short-interval samples to retain their original training objective; (**iii**) Our method achieves State-of-the-Art CPU real-time tracking performance on 5 general object tracking and 5 UAV-tracking benchmarks.

2 Related Work

2.1 Lightweight Object Tracking

In real-world applications like UAV-tracking or robotics, object trackers are deployed to edge devices with limited computational power and memory. Thus, many efforts have been made to boost the efficiency of object trackers. Early attempts focus on finding more efficient architectures for Siamese trackers. LightTrack [Yan *et al.*, 2021b] uses NAS to search for efficient subnets for object tracking. FEAR [Borsuk *et al.*, 2022] designs a dual-template representation and a pixel-wise fusion block to improve both the efficiency and the accuracy of Siamese trackers. With the rise of transformer trackers, some methods attempt to design more efficient transformers for visual tracking. HCAT [Chen *et al.*, 2022] designs a hierarchical cross-attention transformer for lightweight feature interaction, while Kang *et al.* explores modern lightweight transformers for efficient one-stream tracking. Some methods also employ model compression techniques to reduce model size. MixFormerV2 [Cui *et al.*, 2024] presents a distillation-based model reduction paradigm that sequentially reduces the width and depth of the tracker. Another widely explored approach for lightweight tracking is the dynamic inference of the tracking model. Aba-ViTrack [Li *et al.*, 2023] adaptively discards image tokens based on learned halting probabilities, whereas AVTrack [Li *et al.*, 2024] dynamically activates transformer blocks to balance the efficiency and accuracy. AVTrack [Li *et al.*, 2024] also proposed a mutual information maximization loss to facilitate learning view-invariant representations. These methods mainly focus on improving the model design or the supervision, while our method focuses on improving the training data and the optimization.

2.2 Sampling Strategy in Object Tracking

Many object tracking methods [Bertinetto *et al.*, 2016; Ye *et al.*, 2022; Li *et al.*, 2024; Zheng *et al.*, 2024] follow a random sampling strategy that first randomly samples a template frame as the base frame, then randomly samples search frames after the base frame. The maximum time interval between the initial template frame and the search frame (temporal range) is restricted to be lower than a threshold to avoid unstable training. To increase the temporal range without disturbing training, some methods [Yan *et al.*, 2021a; Cui *et al.*, 2022] introduce online frames only ensuring the maximum time interval between the search image and the latest online frame is within a threshold. The temporal range can be safely increased because the online frames provide sufficient recent appearance information. Following this paradigm, TCTrack++ [Cao *et al.*, 2023] proposed an easy-to-hard curriculum strategy that increases the number of online frames at 33% and 50% of total epochs to improve accuracy. However, their strategy cannot be applied to the vast majority of lightweight trackers without online frames and they switch to harder samples at predefined timesteps. In contrast, our method is customized for lightweight trackers without online frames and automatically adjusts difficulty based on the training status every epoch.

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

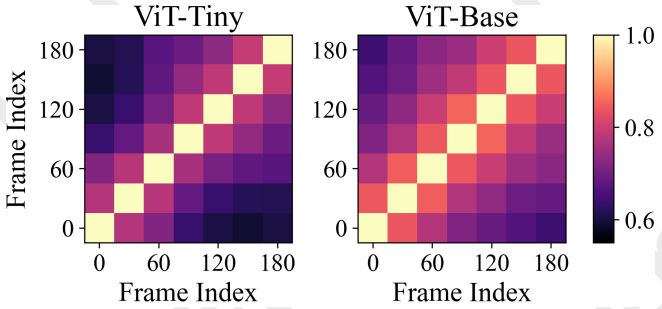

Figure 2: Representation similarity between the same object at different times as heatmaps. Brighter blocks mean the representation is more similar. The result is averaged across videos from LaSOT [Fan *et al.*, 2019] training split.

3 Method

We present SSTrack, a customized training schedule for lightweight object trackers aiming to compensate for the weaker representation ability of lightweight models from both data and optimization perspective. This section first presents an analysis of the necessity of an easier-to-harder training schedule for lightweight models. Next, we detailed the proposed success-aware sampler scheduler for easier-to-harder training, followed by the introduction of the gradient scaling strategy which retains the original training objective on reduced easier samples.

3.1 Problem Analysis

Object trackers usually follow an image pair matching paradigm [Bertinetto *et al.*, 2016]. Specifically, a lightweight visual object tracker $\phi_\theta(z, x)$ takes a template image patch z and a search image patch x as input and predicts the target bounding box. Instead of training from random initialization, object trackers typically fine-tune a pretrained backbone network and incorporate an additional localization head to leverage the robust feature representations from large-scale pretraining. For the sake of tracking speed, lightweight object trackers use lightweight pretrained backbones which often yield less robust representations due to their reduced parameters and computational budget. To quantify this difference in representation quality, we compute the representation similarity of the same object across different times in videos with a lightweight model ViT-Tiny [Wang *et al.*, 2023] and a heavy model ViT-Base [He *et al.*, 2022] using Centered Kernel Alignment(CKA) [Cortes *et al.*, 2012; Nguyen *et al.*, 2020]. The result is shown in Fig. 2. For both models, the representation similarity decreases as the time interval between images increases. Compared with the large model(ViT-Base), the representation extracted by the lightweight model (ViT-Tiny) is less consistent for the same object across time, especially when the time interval is large (top-left corner and bottom-right corner). The messages from the above observations are twofold: (**i**) The training pairs with larger time intervals are naturally more difficult. (**ii**) The lightweight pretrain model is more inadequate to capture large appearance variation, especially under longer time intervals, which makes training on long-interval samples difficult in the early epochs.

Following the above analysis, we aim to design an easy-to-hard training schedule that gradually increases harder training samples with longer template-search time intervals to compensate for the weaker representation ability of lightweight backbones. Next, we detail our proposed success-aware sample scheduler and gradient scaling strategy.

3.2 Easier-to-harder Training via Success-aware Sample Scheduling

In this section, we demonstrate a success-aware sample scheduler for lightweight trackers. We start with an interval-based sampler that allows for adjusting training difficulty by changing the sampling frequency of different samples. Then, we elaborate on the success-aware sample schedule that alters the training difficulty based on both the model's learning progress and the predefined easier-to-harder behavior.

Interval-based Sampler

To generate training samples during training, most methods follow a random sampling strategy which first randomly selects a template frame at t_z and then selects a search frame at t_x ensuring the interval $t_x - t_z \in [0, M)$, where M is the maximum sample interval. Under this random sampling strategy, the distribution of the interval $t_x - t_z$ is fixed throughout training thus the difficulty level remains the same.

To dynamically alter the difficulty, we proposed to use an interval-based sampler that generates image pairs according to a sampling distribution $Q = (q_1, \cdots, q_r), 0 \leq q_i \leq 1, \sum_i^r q_i = 1$. Specifically, given the maximum sample interval M, we part $[0, M)$ into bins with equal spacing m, resulting in $r = \lceil M/m \rceil$ bins such that i-th bin $\mathtt{b}_i$ correspond to $[l_i, u_i)$, where the lower bound is $l_i = mi$ and the upper bond is $u_i = \min(m(i+1), M)$. With probability q_i the interval-based sampler generate a sample with interval $t_x - t_z \in [l_i, u_i)$ by first uniformly samples a training video (denote its length as L), then sample the template image at $t_z \in [0, L - l_i] \cup [l_i, L]$ and the search image at $t_x \in [max(t_z - u_i + 1, 0), max(t_z - l_i + 1, 0)) \cup [min(t_z + l_i, L), min(t_z + u_i, L))$.

Given this interval-based sampler, we are able to adjust the training difficulty by changing the sampling frequency Q. Next, we will demonstrate a success-aware sample scheduler that dynamically adjusts training difficulty according to the training dynamics during training.

Success-aware Sample Scheduler

We begin by outlining the workflow of the sample scheduler. After each training epoch, the scheduler evaluates the model's performance at the current difficulty level and adjusts the difficulty for the next epoch based on this measurement. The key to the sample scheduler's design lies in its performance measurement which should consider both the model's learning progress and the predefined easier-to-harder behavior. Additionally, to minimize computational overhead, it is optimal to perform performance measurement solely using training statistics, eliminating the need for additional validation data.

Inspired by the commonly used object tracking metric: success score, we proposed an expected success measurement to satisfy the above design philosophy. To minimize

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

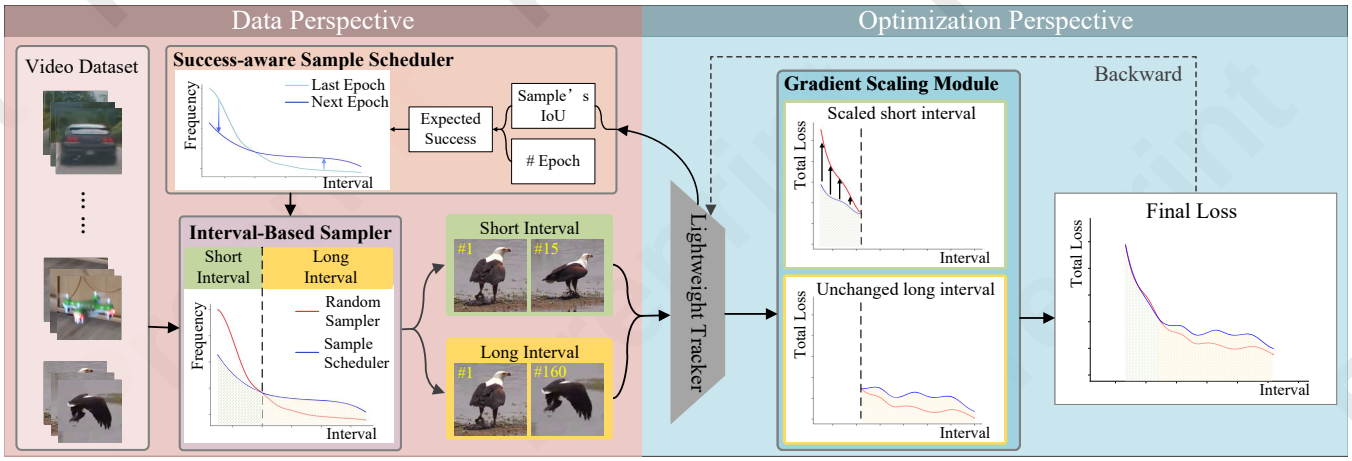

Figure 3: The framework of SSTrack. The Interval-based Sampler generates easier-to-harder training samples following the sample distribution determined by the Success-aware sample scheduler. The gradient scaling strategy scales the gradient of the short-interval samples whose sampling frequency is reduced and keeps other samples' gradients unchanged.

computational overhead, we use the training samples' IoU to estimate the training progress of the corresponding samples. Specifically, given a success threshold τ, we calculate the success rate of all the latest epoch training samples from bin $\mathtt{b}_i$ and use the success rate as an estimation of the probability(denoted as p_i^τ) that the current model successfully tracks any sample within $\mathtt{b}_i$. With this estimation, we can treat all the samples within $\mathtt{b}_i$ as a single "general frame" f_i with probability p_i^τ to success. The proposed measurement is the expected success over the "general video" $\{f_1, \cdots, f_r\}$. Specifically, the success score of the current model on each "general frame" can be seen as a random variable $X_i^\tau \in \{0, 1\}$. The sequence of success score formulates a discrete-time stochastic process $\{X_i^\tau\}$ with countable state space $E = \{0, 1\}$. As the current frame prediction solely relies on the current frame and the previous frame's prediction, we can assume $\{X_i^\tau\}$ follows the Markov property and the transition matrix T_i^τ from time i to $i+1$ can be written as follows

$$T_i^\tau = \begin{pmatrix} p_i^\tau & 1 - p_i^\tau \\ dp_i^\tau & 1 - dp_i^\tau \end{pmatrix}, \tag{1}$$

where $d \in [0, 1]$ is a hyperparameter that can be intuitively understood as the probability of re-detecting the target after tracking failure.

The state distribution at i can be calculated as $\pi_i^\tau = \pi_0^\tau T_1 \cdots T_{i-1}$, where $\pi_0^\tau = (1, 0)$ is the initial distribution. With the distribution π_i^τ, the expected success at i is $\mathbb{E}\{X_i^\tau\} = \pi_i^\tau (1, 0)^T$. And the expected success over the "general video" $\{f_1, \cdots, f_r\}$ is $S^\tau = \sum_{i=1}^r \mathbb{E}\{X_i^\tau\}/r$.

Next, we adjust the difficulty by generating a new sample distribution Q. Samples that contribute more to S^τ should be sampled more frequently so that they are better trained. Therefore, we use $\partial S^\tau / \partial p_i^\tau$ to determine the sampling frequency for bin $\mathtt{b}_i$. The final sampling distribution is obtained as $q_i = \frac{\partial S^\tau / \partial p_i^\tau}{\sum_{i=1}^r \partial S^\tau / \partial p_i^\tau}$.

There are two hyperparameters τ and d in the formulation of the expected success. First, we use τ to consider the model's learning progress. The success threshold τ affects the difficulty by changing p_i^τ. Ideally, τ should grow as the model gains better tracking ability. We set τ to the mean IoU of the latest epoch's training samples. The underlying reason is that the mean IoU can provide precise information about the training dynamics of the tracker which facilitates the sample scheduler to determine a more appropriate difficulty.

Secondly, we use d to inject predefined easier-to-harder behavior. When $d \to 0, \partial S^\tau / \partial p_i^\tau \to 1/r$, so q_i is the same for every i. Consider $p_i = c$ for every i, when $d \to 1, \partial S^\tau / \partial p_i^\tau \to \sum_{m=i-1}^{r-1} c^m / r$, so q_i significantly decreases as i grows up. Therefore, by adjusting d from 0 to 1, an easier-to-harder behavior can be obtained. The pace of introducing harder samples can be regulated by controlling how quickly d approaches 1. We use $d(e) = \sqrt[p]{\frac{e}{E}}$ as the schedule function, where e is the current epoch and E is the total epoch, p is the order of the schedule function. The benefit of choosing this schedule function is that we can control the pace of increasing hard samples with a single parameter. Specifically, a large p means introducing harder samples more quickly and vice versa.

The success-aware sample scheduler only uses training samples' IoUs and the current training epoch as the indicators to decide the sample schedule. Since the IoU is a byproduct of the loss calculation, no significant additional computation is introduced during training. The wall clock time for training is almost the same as compared with training using the random sampler.

3.3 Retain Original Training Objective on Easier Samples via Gradient Scaling

To keep the number of total training samples fixed, the easier-to-harder sample scheduling inevitably reduces the number of short-interval easy samples to increase harder long-interval samples causing a shifted optimization objective, which can potentially undermine the short-term tracking accuracy. Although carefully designed sample schedulers can partly alle-

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

viate this issue, inspired by infoBatch [Qin *et al.*, 2023], we propose a gradient scaling technique to further mitigate this problem.

Denoting the data sampled by the Random Sampler (RS) with sampling distribution $q^R(i)$ as $\mathcal{D}^R$ and data sampled by Interval-based sampler (IS) at epoch t with sampling distribution $q_t^I(i)$ as $\mathcal{D}_t^I$. $\mathcal{D}_t^I$ can be seen as a scaled version of $\mathcal{D}^R$, where the sampling frequency for bin $\mathfrak{b}_i$ is scaled by $\mathfrak{s}_t(i) = (q^R(i) - q_t^I(i))/q^R(i)$. For a sample $v \in \mathfrak{b}_i$, it is a pruned sample if $\mathfrak{s}_t(i) \geq 0$, otherwise an enhanced sample. We define $\mathcal{S}_t(v) = \mathfrak{s}_t(i), v \in \mathfrak{b}_i$ as the scaling ratio of a sample v. Consequently, $\mathcal{P}_t(v) = max(\mathcal{S}_t(v), 0)$ is the pruning ratio and $\mathcal{E}_t(v) = \mathcal{S}_t(v) - \mathcal{P}_t(v)$ is the enhance ratio. The gradient scaling strategy enlarges the gradient of pruned samples and keeps the gradient of enhanced samples unchanged. Specifically, The scaling ratio of each sample v is determined by a scaling function $\Gamma_t(v) = 1/(1 - \mathcal{P}_t(v))$. The scaling function is directly multiplied by the original loss $\mathcal{L}(v, \theta)$ as $\Gamma_t(v)\mathcal{L}(v, \theta)$ to achieve gradient scaling. We next provide a theoretical analysis of how gradient scaling retains the training objective on pruned short-interval samples while maintaining the enhanced training objective on increased long-interval samples.

Theoretical Analysis

Following InfoBatch [Qin *et al.*, 2023], we interpret the training objective as minimizing empirical risk $\mathcal{L}$. We assume the discrete sample distribution for $\mathcal{D}^R$ as $\rho^R(v)$. Then the discrete sample distribution for $\mathcal{D}_t^I$ can be derived as:

$$\rho_t^I(v) = \frac{(1 - \mathcal{S}_t(v))\rho^R(v)}{\sum_{v \in \mathcal{D}^R}(1 - \mathcal{S}_t(v))\rho^R(v)}. \tag{2}$$

At each epoch t, $\mathcal{D}^R$ can be divided into two disjoint subset: $\mathcal{D}_t^{Rp} = \{v | S_t(v) \geq 0\}$ for pruned samples and $\mathcal{D}_t^{Re} = \{v | S_t(v) < 0\}$ for enhanced samples.

The training objective on $\mathcal{D}^R$ is:

$$\arg\min_{\theta \in \Theta} \mathop{\mathbb{E}}_{v \in \mathcal{D}^R}[\mathcal{L}(v, \theta)] = \sum_{v \in \mathcal{D}^R} \mathcal{L}(v, \theta)\rho^R(v). \tag{3}$$

The training objective with gradient scaling on $\mathcal{D}_t^I$ is:

$$\arg\min_{\theta \in \Theta} \mathop{\mathbb{E}}_{v \in \mathcal{D}_t^I}[\Gamma_t(v)\mathcal{L}(v, \theta)] = \sum_{v \in \mathcal{D}_t^I} \Gamma_t(v)\mathcal{L}(v, \theta)\rho_t^I(v). \tag{4}$$

By substituting Eq. 2 into Eq. 4, we have

$$\begin{aligned} &\arg\min_{\theta \in \Theta} \mathop{\mathbb{E}}_{v \in \mathcal{D}_t^I}[\Gamma_t(v)\mathcal{L}(v, \theta)] \\ =&\frac{1}{c_t} \underbrace{\sum_{v \in \mathcal{D}_t^{Rp}} \mathcal{L}(v, \theta)\rho^R(v)}_{Original\ Objective} \\ &+ \frac{1}{c_t}(\underbrace{\sum_{v \in \mathcal{D}_t^{Re}} \mathcal{L}(v, \theta)\rho^R(v)}_{Original\ Objective} + \underbrace{\sum_{v \in \mathcal{D}_t^{Re}} \mathcal{L}(v, \theta)\rho^E(v)}_{Enhancement\ Objective}), \end{aligned} \tag{5}$$

where $c_t = \sum_v (1 - \mathcal{S}_t(v))\rho^R(v)$ is a constant at epoch t, $\rho^E(v) = -\mathcal{E}_t(v)\rho^R(v)$ is the data distribution for enhanced long-interval samples.

As shown in Eq. 5, for the short-interval samples pruned in epoch t, after applying the gradient scaling, the objective becomes a constant-resealed version of the original objective in Eq. 3. For the long-interval samples, the new objective is composed of the original objective and an enhancement objective brought by the extra long-interval samples. So the tracker still benefits from the increased harder samples.

4 Experiments

4.1 Implementation Details

We select OSTrack [Ye *et al.*, 2022] as the baseline model considering its simplicity and publicly available training and inference code. To make it lightweight, we replace the ViT-Base backbone with ViT-Tiny [Dosovitskiy *et al.*, 2020] and adopt the D-MAE pre-train weight from [Wang *et al.*, 2023; Gao *et al.*, 2025]. All the training and testing settings, except for the training data sampler, are kept the same as OSTrack-256. For the interval-based sampler, we set the maximum sample interval $M = 200$ following OSTrack. We part M into 7 bins with equal spacing 30, which is the frame rate for the training video datasets. An additional bin $[0, 1)$ is added to hold samples from the image dataset COCO [Lin *et al.*, 2014]. The success-aware scheduler uses the average iou from the latest epoch as the success threshold and uses the schedule function with order 2. The model is trained on 4 NVIDIA V100 GPU and the inference speed is measured on Intel Xeon Gold 6146 CPU.

4.2 State-of-the-Art Comparison

Comparison on General Tracking Benchmarks

We first compared our method with 6 State-of-the-Art (SOTA) CPU-real-time object trackers on 5 general object tracking datasets, namely LaSOT [Fan *et al.*, 2019], LaSOT$_{ext}$ [Fan *et al.*, 2021], GOT-10k [Huang *et al.*, 2019], TrackingNet [Muller *et al.*, 2018], NFS [Kiani Galoogahi *et al.*, 2017]. We only include CPU real-time object trackers with reported CPU speed over 30 FPS for a fair comparison. The results are shown in Tab. 1. Our method outperforms all the previous real-time SOTA methods on all 5 datasets, demonstrating the powerful performance of our method. textbf LaSOT is a diverse tracking dataset with over 280 long-term video sequences. As shown in Tab. 1, our method outperforms previous SOTA HiT-B [Kang *et al.*, 2023] by 2.3% Success and 3.7% Precision. **LaSOT**$_{ext}$ is the extension of LaSOT with an additional 150 videos containing 15 novel classes. In Tab. 1, our method achieves 2.2% gain in Success over the previous SOTA HiT-B [Kang *et al.*, 2023]. **GOT-10k** requires trackers to be trained solely on the training split to verify generalization ability. We follow this protocol and the result is reported in Tab. 1. According to the results in Tab. 1, our method improves the SOTA performance by 4.5%, 2.6% and 5.1%in AO, SR$_{0.5}$, SR$_{0.75}$ respectively. **TrackingNet** is a large-scale short-term tracking dataset with 511 testing videos. As displayed in Tab. 1, our method lifts

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

	Tracker	Source	LaSOT		$\text{LaSOT}_{\text{ext}}$		GOT-10k			TrackingNet		NFS		#Param	FLOPs	Speed
			SUC	PRE	SUC	PRE	AO	$\text{SR}_{0.5}$	$\text{SR}_{0.75}$	SUC	PRE	SUC	PRE			
Real-time	SSTrack	**ours**	**66.9**	**71.8**	**46.3**	**51.8**	**69.0**	**79.1**	**63.2**	**81.6**	**79.5**	**65.7**	**78.9**	8.0M	1.9G	**47**
	HiT-B	ICCV23	64.6	68.1	44.1	-	64.0	72.1	58.1	80.0	77.3	63.6	-	42.1M	4.3G	33
	SMAT	WACV24	61.7	64.6	43.6	-	64.5	74.7	57.8	78.6	75.6	62.0	74.6	-	-	37
	MixFormerV2-S	NIPS23	60.6	60.4	43.6	46.2	-	-	-	75.8	70.4	-	-	16.2M	4.4G	30
	FEAR-L	ECCV22	57.9	60.9	-	-	64.5	-	-	-	-	61.8	75.3	33.7M	-	38
	HCAT	ECCV22	59.3	61.0			65.1	76.5	56.7	76.6	72.9	63.5	-	6.8M	1.3G	45
	E.T.Track	WACV22	59.1	-	-	-	-	-	-	75.0	70.6	59.0	-	7.0M	1.7G	**47**
	LightTrack-LB	CVPR21	55.5	56.1	-	-	62.3	72.6	-	72.5	69.5	55.3	-	**3.1M**	**0.8G**	-
Non-Real-time	EVPTrack_{224}	AAAI24	70.4	77.2	-	-	73.3	83.6	70.7	83.5	-	-	-	73.7M	21.8G	9
	ZoomTrack	NIPS23	70.2	76.2	50.5	57.4	73.5	83.6	70.0	83.2	82.2	-	-	92.1M	21.5G	11
	ARTrack_{256}	CVPR23	70.4	76.6	46.4	52.3	73.5	82.2	70.9	84.2	83.5	64.3	-	173.1M	40.3G	3
	SwinTrack_{224}	NIPS22	67.2	70.8	47.6	53.9	71.3	81.9	64.5	81.1	78.4	-	-	23.0M	6.4G	16
	MixFormer-22k	CVPR22	69.2	74.7	-	-	72.6	82.2	68.8	83.1	81.6	-	-	35.6M	23.0G	5
	STARK-ST50	ICCV21	66.6	-	-	-	68.0	77.7	62.3	81.3	-	-	-	28.2M	12.8G	11
	TransT	CVPR21	64.9	69.0	-	-	67.1	76.8	60.9	81.4	80.3	65.7	-	23.0M	-	5

Table 1: Comparison with CPU real-time tracking algorithms on general visual tracking datasets, including LaSOT [Fan *et al.*, 2019], $\text{LaSOT}_{\text{ext}}$ [Fan *et al.*, 2021], GOT-10k [Huang *et al.*, 2019], TrackingNet [Muller *et al.*, 2018], and NFS [Kiani Galoogahi *et al.*, 2017]. The speed is measured by *fps* on CPU. The best CPU-real-time performance is highlighted in **Bold**.

Tracker	Source	DTB70		UAVDT		UAV123		UAV123@10fps		UAVTrack112L		#Param	FLOPs	Speed
		SUC	PRE	SUC	PRE	SUC	PRE	SUC	PRE	SUC	PRE			
SSTrack	**ours**	**66.4**	85.5	**61.0**	82.8	**67.6**	**87.6**	**66.4**	**85.4**	**65.8**	**83.0**	8.0M	1.9G	47
AVTrack	ICML24	65.0	84.3	58.7	82.1	66.8	84.8	65.8	83.2	-	-	**6.2M**	**1.8G**	**55**
TATrack	TGRS24	66.1	85.5	59.6	82.4	67.1	85.0	66.1	83.4	-	-	8.3M	2.4G	49
Aba-ViTrack	ICCV23	**66.4**	**85.9**	59.9	**83.4**	66.4	86.4	65.5	85.0	64.2	81.1	-	-	50
P-SiamFC++	ICME23	60.4	80.3	55.6	80.7	48.9	74.5	54.9	73.1	53.1	70.4	10.4M	2.2G	51
TCTrack	CVPR22	62.2	81.2	53.0	72.5	60.5	80.0	59.9	78.0	58.3	78.6	9.8M	8.8G	-
HiFT	ICCV21	59.4	80.2	47.5	65.2	59.0	78.7	57.0	74.9	55.1	73.4	10.4M	7.3G	-
SiamAPN++	IROS21	59.4	78.9	55.6	76.9	58.2	76.8	58.1	76.4	-	-	14.7M	8.2G	-

Table 2: Comparison with UAV-tracking algorithms on UAV-tracking datasets, including DTB70 [Li and Yeung, 2017], UAVDT [Du *et al.*, 2018], UAV123&UAV123@10fps [Mueller *et al.*, 2016] and UAVTrack112L [Fu *et al.*, 2021a]. The speed is measured by *fps* on CPU. The best performance is highlighted in **Bold**.

the SOTA performance by 1.6% in Success and 2.2% in Precision. **NFS** is a high frame-rate dataset containing challenging fast-moving objects with 100 video sequences. As reported in Tab. 1, our method achieves 65.7% Success, improving previous SOTA by 2.1%.

We further list 7 non-real-time SOTA trackers in Tab. 1 for a more comprehensive comparison. It is worth noting that our method outperforms STARK [Yan *et al.*, 2021a] using only 28% parameters and 14% computation.

Comparison on UAV-tracking Benchmarks

We next compare our method with 7 State-of-the-Art(SOTA) UAV trackers on 5 UAV-tracking benchmarks, namely DTB70 [Li and Yeung, 2017], UAVDT [Du *et al.*, 2018], UAV123 and UAV123@10fps [Mueller *et al.*, 2016], UAVTrack112L [Fu *et al.*, 2021a]. The results are shown in Tab. 2. Our method outperforms or on par with previous SOTA trackers on these 5 UAV-tracking datasets with similar overhead and speed, showing a strong generalization ability to real-world scenarios. textbf DTB70 is a UAV tracking dataset with 70 videos which primarily addresses the challenge of severe UAV motion. As shown in Tab. 2, our method achieves 66.4% SUC and 85.5% PRE. **UAVDT** is a challenging dataset mainly focus on vehicle tracking with various weather conditions, flying altitudes and camera views. As reported in Tab. 2, our method outperforms previous SOTA Aba-ViTrack [Li *et al.*, 2023] in SUC by 1.1%. **UAV123** is a UAV-tracking dataset containing 123 videos captured by low-altitude UAVs. From the results in Tab. 2, it can be seen that our method improves the previous SOTA by 0.5% SUC and 1.2% precision. **UAV123@10fps** is constructed from UAV123 by resampling from 30 FPS to 10FPS causing more abrupt motion and scale variation. In Tab. 2, our method boosts the SOTA performance by 0.3% SUC and 0.4% precision. **UAV112L** is the current largest long-term UAV-tracking dataset containing 45 videos. As displayed in Tab. 2, our method advances the SOTA performance by 1.6% in SUC and 1.9% in Precision, showing that our method is capable of

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

#	SS	GS	Epoch	LaSOT		UAVTrack112L	
				SUC	PRE	SUC	PRE
①			300	65.7	69.7	64.6	80.5
②	✔		300	66.3	70.7	65.3	81.9
③	✔	✔	300	**66.9**	**71.8**	**65.8**	**83.0**

Table 3: Ablation study of different components of the proposed method on LaSOT and UAVTrack112L dataset. SS: Success-aware Sample Scheduler, GS: Gradient Scaling.

#	p	τ	Update Every Epoch	LaSOT		UAVTrack112L	
				SUC	PRE	SUC	PRE
①	1	Mean IoU	✔	66.2	70.4	64.0	80.2
②	2	Mean IoU	✔	**66.9**	**71.8**	**65.8**	**83.0**
③	3	Mean IoU	✔	65.9	70.4	63.7	80.1
④	2	0.5	✔	65.7	69.8	64.0	80.0
⑤	2	0.75	✔	65.9	69.9	64.4	80.3
⑥	2	Mean IoU	✗	65.3	69.1	62.8	79.3

Table 4: Ablation study of different choices of the sample scheduler on LaSOT and UAVTrack112L dataset

long-term tracking.

Efficiency Analysis

We comprehensively evaluate the efficiency of our method via the number of parameters, the computational overhead and the inference speed. The results are reported in both Tab. 1 and Tab. 2. Compared with CPU real-time trackers, as displayed in Tab. 1, our method achieves 47FPS on CPU and only uses 1.9G Multiply–Accumulate Operations (MACs) which is comparable to previous CPU real-time trackers. As for parameters, our method only uses 8.0M parameters, which is significantly less than previous SOTA CPU real-time trackers, making it ideal for deploying on edge devices. Compared with UAV trackers, as presented in Tab. 2, our method has similar efficiency with these trackers customized for deployment on UAVs, while having on-par or better accuracy.

4.3 Ablation Study

We conduct the ablation experiments on two challenging long-term tracking datasets, namely LaSOT [Fan *et al.*, 2019] and UAVTrack112L [Fu *et al.*, 2021a], to demonstrate the effectiveness of our method.

Component-wise Analysis

To evaluate the effect of different components of our method, we conduct a component-wise analysis, starting from the baseline (①). The results are reported in Tab. 3. Directly employing the success-aware sample scheduler on the baseline can improve the Precision by 1.0% on LaSOT and 1.4% on UAVTrack112L (①*v.s.*②), demonstrating the effectiveness of the proposed easier-to-harder sample schedule. Next, we add the gradient scaling strategy, the Precision is further boosted by 1.1% on both datasets (②*v.s.*③), proving the effectiveness of the gradient scaling strategy.

Different Sample Scheduler Setting

As shown in Tab. 4, we ablate different settings of the success-aware sample scheduler. In the first set of experiments (①, ② and ③), we fix the success threshold τ as the mean IoU of the latest epoch and change the speed of introducing more harder samples via changing the order of schedule function p. In Tab. 4, selecting $p = 2$ achieves the best accuracy, demonstrating the importance of gradually introducing more long-interval samples. In the second set of experiments (②, ④ and ⑤), we set the order of the schedule function fixed to $p = 2$, and explore the necessity of using dynamic success threshold τ. As displayed in Tab. 4, using fixed success threshold (④ and ⑤) performs worse than using mean IoU as dynamic success threshold (②), proving the significance of considering the model's learning progress when designing sample scheduler. In the last experiment (⑥), we only update the difficulty at 33% and 50% of the total epoch following [Cao *et al.*, 2023]. Compared with ②, the success score and precision drop significantly on both datasets, showing the importance of our update-every-epoch design.

Tracker	Backbone (Pretrain)	Ours	LaSOT		UAVTrack112L	
			SUC	PRE	SUC	PRE
OSTrack	ViT-T (D-MAE)		65.7	69.7	64.6	80.5
		✔	$66.9_{\uparrow 1.2}$	$71.8_{\uparrow 2.1}$	$65.8_{\uparrow 1.2}$	$83.0_{\uparrow 2.5}$
AVTrack	ViT-T (D-MAE)		63.8	67.2	61.8	78.1
		✔	$64.9_{\uparrow 1.1}$	$68.2_{\uparrow 1.0}$	$64.3_{\uparrow 2.5}$	$80.9_{\uparrow 2.8}$
HiT	LeViT-B (DeiT)		63.5	65.4	62.1	78.5
		✔	$64.7_{\uparrow 1.2}$	$67.5_{\uparrow 2.1}$	$63.3_{\uparrow 1.2}$	$80.3_{\uparrow 1.8}$

Table 5: Ablation study of applying the proposed method to different trackers on LaSOT and UAVTrack112L dataset[1].

Generalization Ability

To demonstrate the generalization ability of our approach, we incorporate our method to other SOTA lightweight trackers: AVTrack [Li *et al.*, 2024] and HiT [Kang *et al.*, 2023]. As displayed in Tab. 5, our method achieves consistent gains on multiple lightweight trackers with different backbones and pretrain methods, showing a strong generalization ability. Note that all the hyperparameters in our method remain consistent across all datasets and trackers, while other hyperparameters shared with the baselines are kept unchanged.

5 Conclusion

In this work, we present SSTrack, a novel training schedule designed to enhance the performance of lightweight object trackers. Aiming to compensate for the weaker representation ability of lightweight backbones, we developed a success-aware sample scheduler that progressively increases difficult long-interval samples and reduces easy short-interval samples to keep the training cost unchanged. Furthermore, we incorporated a gradient scaling strategy to maintain the positive influences of easier samples, even as their quantity decreases. This combined approach has enabled us to achieve state-of-the-art accuracy on 5 UAV-tracking benchmarks and 5 general object tracking benchmarks.

[1]Baseline results are obtained by training with the official code from their GitHub repository to ensure a fair comparison.

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

Acknowledgements

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions. This work was supported in part by the Natural Science Foundation of China (Grant No. 62036011, 62422317, U22B2056, 62192782, U24A20331, U2441241, 62172413), the Beijing Natural Science Foundation (Grant No. JQ22014, L223003), the Key Research and Development Program of Xinjiang Urumgi Autonomous Region under Grant No. 2023B01005, the Project of Beijing Science and Technology Committee (Project No. Z231100005923046). Bing Li is also supported in part by the Youth Innovation Promotion Association, CAS.

References

[Bertinetto *et al.*, 2016] Luca Bertinetto, Jack Valmadre, Joao F Henriques, Andrea Vedaldi, and Philip HS Torr. Fully-convolutional siamese networks for object tracking. In *Computer Vision–ECCV 2016 Workshops: Amsterdam, The Netherlands, October 8-10 and 15-16, 2016, Proceedings, Part II 14*, pages 850–865. Springer, 2016.

[Borsuk *et al.*, 2022] Vasyl Borsuk, Roman Vei, Orest Kupyn, Tetiana Martyniuk, Igor Krashenyi, and Jiři Matas. Fear: Fast, efficient, accurate and robust visual tracker. In *European Conference on Computer Vision*, pages 644–663. Springer, 2022.

[Cao *et al.*, 2021] Ziang Cao, Changhong Fu, Junjie Ye, Bowen Li, and Yiming Li. Hift: Hierarchical feature transformer for aerial tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 15457–15466, 2021.

[Cao *et al.*, 2023] Ziang Cao, Ziyuan Huang, Liang Pan, Shiwei Zhang, Ziwei Liu, and Changhong Fu. Towards real-world visual tracking with temporal contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[Chen *et al.*, 2021] Xin Chen, Bin Yan, Jiawen Zhu, Dong Wang, Xiaoyun Yang, and Huchuan Lu. Transformer tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8126–8135, 2021.

[Chen *et al.*, 2022] Xin Chen, Ben Kang, Dong Wang, Dongdong Li, and Huchuan Lu. Efficient visual tracking via hierarchical cross-attention transformer. In *European Conference on Computer Vision*, pages 461–477. Springer, 2022.

[Cortes *et al.*, 2012] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. Algorithms for learning kernels based on centered alignment. *The Journal of Machine Learning Research*, 13:795–828, 2012.

[Cui *et al.*, 2022] Yutao Cui, Cheng Jiang, Limin Wang, and Gangshan Wu. Mixformer: End-to-end tracking with iterative mixed attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13608–13618, 2022.

[Cui *et al.*, 2024] Yutao Cui, Tianhui Song, Gangshan Wu, and Limin Wang. Mixformerv2: Efficient fully transformer tracking, 2024.

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[Du *et al.*, 2018] Dawei Du, Yuankai Qi, Hongyang Yu, Yifan Yang, Kaiwen Duan, Guorong Li, Weigang Zhang, Qingming Huang, and Qi Tian. The unmanned aerial vehicle benchmark: Object detection and tracking. In *Proceedings of the European conference on computer vision (ECCV)*, pages 370–386, 2018.

[Fan *et al.*, 2019] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5374–5383, 2019.

[Fan *et al.*, 2021] Heng Fan, Hexin Bai, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Harshit, Mingzhen Huang, Juehuan Liu, et al. Lasot: A high-quality large-scale single object tracking benchmark. *International Journal of Computer Vision*, 129:439–461, 2021.

[Fu *et al.*, 2021a] Changhong Fu, Ziang Cao, Yiming Li, Junjie Ye, and Chen Feng. Onboard real-time aerial tracking with efficient siamese anchor proposal network. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–13, 2021.

[Fu *et al.*, 2021b] Zhihong Fu, Qingjie Liu, Zehua Fu, and Yunhong Wang. Stmtrack: Template-free visual tracking with space-time memory networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13774–13783, 2021.

[Gao *et al.*, 2025] Jin Gao, Shubo Lin, Shaoru Wang, Yutong Kou, Zeming Li, Liang Li, Congxuan Zhang, Xiaoqin Zhang, Yizheng Wang, and Weiming Hu. An experimental study on exploring strong lightweight vision transformers via masked image modeling pre-training. *International Journal of Computer Vision*, pages 1–33, 2025.

[Gopal and Amer, 2024] Goutam Yelluru Gopal and Maria A Amer. Separable self and mixed attention transformers for efficient object tracking. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 6708–6717, 2024.

[He *et al.*, 2022] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[Huang *et al.*, 2019] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE transactions*

Preprint – IJCAI 2025: This is the accepted version made available for conference attendees.
Do not cite. The final version will appear in the IJCAI 2025 proceedings.

on pattern analysis and machine intelligence, 43(5):1562–1577, 2019.

[Kang *et al.*, 2023] Ben Kang, Xin Chen, Dong Wang, Houwen Peng, and Huchuan Lu. Exploring lightweight hierarchical vision transformers for efficient visual tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9612–9621, 2023.

[Kiani Galoogahi *et al.*, 2017] Hamed Kiani Galoogahi, Ashton Fagg, Chen Huang, Deva Ramanan, and Simon Lucey. Need for speed: A benchmark for higher frame rate object tracking. In *Proceedings of the IEEE international conference on computer vision*, pages 1125–1134, 2017.

[Kou *et al.*, 2023] Yutong Kou, Jin Gao, Bing Li, Gang Wang, Weiming Hu, Yizheng Wang, and Liang Li. Zoomtrack: Target-aware non-uniform resizing for efficient visual tracking. In *Advances in Neural Information Processing Systems*, volume 36, pages 50959–50977. Curran Associates, Inc., 2023.

[Li and Yeung, 2017] Siyi Li and Dit-Yan Yeung. Visual object tracking for unmanned aerial vehicles: A benchmark and new motion models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

[Li *et al.*, 2023] Shuiwang Li, Yangxiang Yang, Dan Zeng, and Xucheng Wang. Adaptive and background-aware vision transformer for real-time uav tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13989–14000, 2023.

[Li *et al.*, 2024] Yongxin Li, Mengyuan Liu, You Wu, Xucheng Wang, Xiangyang Yang, and Shuiwang Li. Learning adaptive and view-invariant vision transformer for real-time uav tracking. In *Forty-first International Conference on Machine Learning*, 2024.

[Lin *et al.*, 2014] 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 *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[Mueller *et al.*, 2016] Matthias Mueller, Neil Smith, and Bernard Ghanem. A benchmark and simulator for uav tracking. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, pages 445–461. Springer, 2016.

[Muller *et al.*, 2018] 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 *Proceedings of the European conference on computer vision (ECCV)*, pages 300–317, 2018.

[Nguyen *et al.*, 2020] Thao Nguyen, Maithra Raghu, and Simon Kornblith. Do wide and deep networks learn the same things? uncovering how neural network representations vary with width and depth. *arXiv preprint arXiv:2010.15327*, 2020.

[Qin *et al.*, 2023] Ziheng Qin, Kai Wang, Zangwei Zheng, Jianyang Gu, Xiangyu Peng, Zhaopan Xu, Daquan Zhou, Lei Shang, Baigui Sun, Xuansong Xie, et al. Infobatch: Lossless training speed up by unbiased dynamic data pruning. *arXiv preprint arXiv:2303.04947*, 2023.

[Wang *et al.*, 2023] Shaoru Wang, Jin Gao, Zeming Li, Xiaoqin Zhang, and Weiming Hu. A closer look at self-supervised lightweight vision transformers. In *International Conference on Machine Learning*, pages 35624–35641. PMLR, 2023.

[Wei *et al.*, 2023] Xing Wei, Yifan Bai, Yongchao Zheng, Dahu Shi, and Yihong Gong. Autoregressive visual tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9697–9706, 2023.

[Yan *et al.*, 2021a] Bin Yan, Houwen Peng, Jianlong Fu, Dong Wang, and Huchuan Lu. Learning spatio-temporal transformer for visual tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10448–10457, 2021.

[Yan *et al.*, 2021b] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15180–15189, 2021.

[Ye *et al.*, 2022] Botao Ye, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Joint feature learning and relation modeling for tracking: A one-stream framework. In *ECCV*, 2022.

[Zhang *et al.*, 2021] Zhipeng Zhang, Yihao Liu, Xiao Wang, Bing Li, and Weiming Hu. Learn to match: Automatic matching network design for visual tracking. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13339–13348, 2021.

[Zheng *et al.*, 2024] Yaozong Zheng, Bineng Zhong, Qihua Liang, Zhiyi Mo, Shengping Zhang, and Xianxian Li. Odtrack: Online dense temporal token learning for visual tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 7588–7596, 2024.