Adaptive Capacity Autoregressive Visual Tracking

Tong Lin[1*] Yifan Bai[2*‡] Shiyi Liang[1] Ruigang Niu[2] Xing Wei[1†]
[1]School of Software Engineering, Xi'an Jiaotong University [2]DAMO Academy, Alibaba Group
{lintong1220, sy_liang2023}@stu.xjtu.edu.cn
{baiyifan.byf, niuruigang.nrg}@alibaba-inc.com weixing@mail.xjtu.edu.cn

Abstract

*We present **ARTrack-AC**, a new step in the autoregressive tracking paradigm that introduces adaptive capacity inference to achieve both temporal consistency and dynamic efficiency. While existing autoregressive trackers predict object states sequentially with fixed inference capacity, they fail to accommodate the fluctuating temporal difficulty of real videos. ARTrack-AC addresses this limitation by equipping the tracker with the ability to **modulate its inference capacity over time**. A diffusion-based trajectory estimator anticipates the stability of upcoming segments, guiding a controller to switch between an **accurate** (high-capacity) and an **efficient** (low-capacity) mode while maintaining autoregressive consistency. This system-level autoregression extends conventional sequence modeling beyond "what to predict" toward "how to predict," forming a self-regulated tracking process that aligns inference cost with temporal complexity. Despite its simplicity, ARTrack-AC achieves state-of-the-art accuracy–speed trade-off on major benchmarks—66.7% AUC on LaSOT and 47.5% AUC on LaSOText—running 2.9× faster than its predecessor. Source code is available at: https://github.com/MIV-XJTU/ARTrackAC.git.*

1. Introduction

Visual object tracking [5, 18, 28, 35, 40, 41, 43, 48] is a cornerstone problem in computer vision, requiring continuous target localization under dynamic appearance changes and frequent occlusions—all while meeting real-time constraints. These real-time demands have intensified the need for efficient tracking [6, 19, 38], where the objective is to increase throughput and lower computational overhead while maintaining reliable performance in challenging scenarios.

In recent years, **autoregressive (AR) tracking** has emerged as a powerful paradigm that formulates tracking as sequence modeling [1, 10, 29, 42]: each prediction depends on the model's own previous outputs rather than solely on the current frame. This formulation establishes consistent train-test objectives, naturally encodes temporal dependency, and unifies tracking and prediction within a single

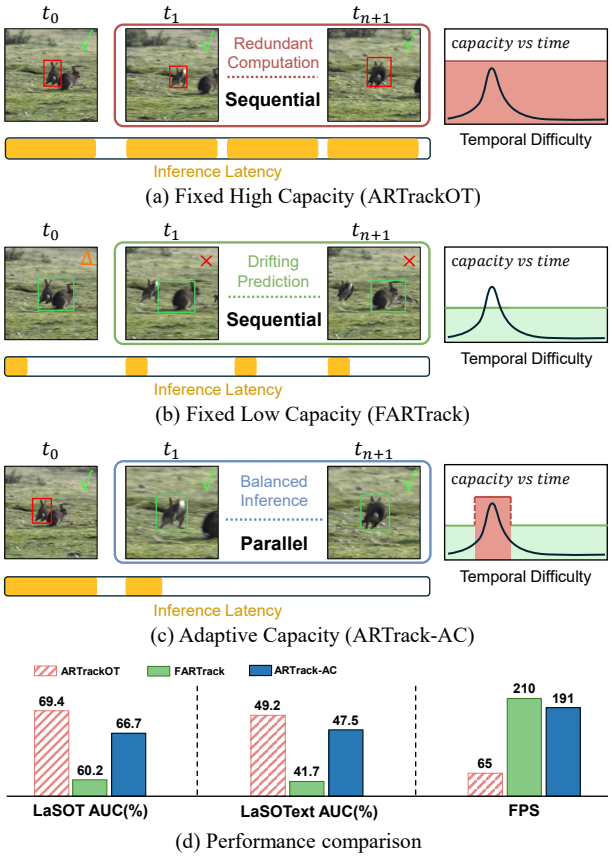


Figure 1. **Tracking paradigms and performance comparison.** (a) Accurate-oriented tracker: Fixed high capacity, robust yet redundant, strictly sequential inference with high inference latency. (b) Efficient-oriented tracker: Fixed low capacity, fast but less robust, also sequential with low latency. (c) Adaptive capacity tracker: Dynamically adjusts inference capacity, performs parallel inference on stable segments, achieving a balanced accuracy–speed trade-off. (d) Performance comparison.

* Equal Contribution. † Corresponding Author. ‡ Project Lead.

framework. Early work, such as ARTrack [42], demonstrated that inter-frame autoregression can achieve strong temporal consistency by sequentially generating object coordinates conditioned on historical states. ARTrackV2 [1] advanced this paradigm by jointly evolving both trajectory and appearance, allowing the tracker to "read out" where an object is and "retell" what it looks like. These efforts have proven that autoregressive modeling offers a principled foundation for robust tracking. However, both methods share an implicit assumption: the **inference capacity**, i.e., the computational depth and reasoning strength used for each frame, is fixed.

In practice, this assumption is mismatched with real-world videos. The temporal difficulty of a sequence is highly dynamic: stable segments with smooth motion require minimal reasoning, while sudden motion, heavy occlusion, or cluttered backgrounds demand richer temporal modeling. A fixed-capacity tracker either wastes computation on easy segments or fails to cope with abrupt challenges. Although heuristic strategies, such as periodic template updates [30, 49, 50] or frame skipping [16, 25], attempt to alleviate the issue, they ignore the underlying temporal uncertainty and easily break AR consistency.

To bridge this gap, we introduce **Adaptive Capacity Autoregressive Visual Tracking (ARTrack-AC)**, which extends autoregressive tracking from fixed-capacity modeling to an adaptive capacity inference paradigm. The key idea is to make the tracker autoregressive not only in predicting the target's state but also in regulating its own inference capacity over time. Specifically, ARTrack-AC incorporates a **difficulty-aware controller** that forecasts upcoming temporal complexity and modulates computational resources accordingly. A lightweight diffusion-based estimator models future stability conditioned on recent observations and trajectories. Its prediction drives a **dual-mode inference system**—an accurate mode with higher capacity for challenging segments and an efficient mode for stable ones—thus aligning model effort with temporal difficulty and improving overall efficiency.

This design yields three major benefits. (i) **Difficulty awareness**: the diffusion estimator captures multi-modal future uncertainty, providing a reliable unsupervised signal for determining when higher capacity is necessary. (ii) **Capacity adaptivity**: a single tunable threshold continuously controls the trade-off between accuracy and speed, enabling flexible deployment under different latency budgets. (iii) **Autoregressive integrity**: both state prediction and capacity selection are conditioned on the tracker's historical context, preserving temporal causality and train–test alignment. By dynamically allocating computational effort according to temporal difficulty, the tracker achieves resource efficiency without compromising temporal coherence. Such adaptive capacity also yields an emergent benefit: it permits flexible scheduling and partial parallelism in stable periods, reflecting a broader system-level advantage of self-regulated inference.

It is worth clarifying that the term "adaptive capacity" here refers to the tracker's ability to dynamically modulate its computational capacity, rather than to any spatial multi-scale feature representation. The paradigm thus operates at the system level, augmenting the autoregressive process with self-regulated inference decisions.

In summary, ARTrack-AC advances the autoregressive tracking family in a new direction from modeling "what to predict" toward deciding "how to predict." By coupling temporal autoregression with adaptive capacity control, it achieves superior efficiency without sacrificing temporal coherence, offering a principled path toward self-regulated, resource-aware tracking systems.

2. Related Work

Efficient Visual Object Tracking. Efficient visual tracking [7, 24, 37, 51, 52] methods center on single-capacity designs, following Siamese-based [17, 26, 27, 38] or one-stream [12, 13, 22, 48] architectures where pipeline remains fixed during inference. To improve efficiency, works compress model scale through lightweight design or distillation. MixFormerV2 [13] lightens a Transformer network through teacher–student distillation. Methods also improve efficiency by reducing input redundancy. LightTrack [46] employs neural architecture search (NAS) to discover a backbone, reducing parameters. AsymTrack [53] disentangles template and search streams within an asymmetric Siamese framework, performing template computing only once during initialization, while FEAR [4] adopts an update mechanism to avoid frequent full-network updates. Although these trackers balance efficiency, they remain bound by a fixed-capacity design and fail to exploit GPU bandwidth during inference. In contrast, our method reframes efficient tracking from an inference-paradigm perspective, allocating computation at the video-level and enabling multi-frame parallel inference to utilize GPU bandwidth.

Autoregressive Tracking Framework. Autoregressive tracking formulates tracking as sequence modeling, where each prediction depends on previous outputs to capture temporal dependency [1, 9, 10, 29, 42]. Representative works include ARTrack [39] for inter-frame coordinate generation, ARTrackV2 [1] for joint trajectory–appearance evolution, and FARTrack [36] for trajectory-aware self-distillation. A shared limitation is fixed inference capacity, which mismatches temporal difficulty in real videos, wasting compute on easy segments while under-handling abrupt challenges. Our approach preserves autoregressive integrity by conditioning both state prediction and capacity selection on the same causal history.

Adaptive Visual Object Tracking. Adaptive tracking has been explored by modulating inference cost using different control signals. EAST [20] learns RL-based layer-wise early exit, where the signal is derived from per-frame matching confidence or representation quality. ABTrack [47] adaptively bypasses ViT blocks via learned skip decisions conditioned on current representations. SGLATrack [45] uses inter-layer representation similarity as a redundancy cue to selectively deactivate layers during inference. ACFN [11] leverages CF response validation scores together with temporal prediction to select modules. Overall, these signals are predominantly appearance-driven and are used for frame-wise, reactive allocation. In this work, we extend adaptive tracking beyond per-frame, appearance-only control signals by integrating motion dynamics to form a forward-looking scheduling signal under temporal difficulty.

3. Tracking as Adaptive Capacity Regulation

3.1. System-Level Autoregression

Conventional autoregressive (AR) trackers predict target states under a fixed inference capacity, which mismatches the fluctuating temporal difficulty of real-world videos. We therefore propose **System-Level Autoregression**: a self-regulated inference paradigm in which *state prediction* and *capacity selection* are tightly coupled over time. At each step t, the tracker conditions on its historical states and current visual observation to predict the next state, and simultaneously adjusts its inference capacity according to the difficulty inferred from its own recent reasoning. This coupling aligns computational effort with temporal complexity while preserving the causal, sequence-level nature of AR tracking.

3.2. Autoregressive Tracker

An autoregressive tracker formulates tracking as sequence modeling: each prediction depends on the model's previous outputs rather than solely on the current frame. This can be expressed as a conditional probability:

$$p(Y^t|Y^{t-N:t-1}, (C, Z, X^t)), \tag{1}$$

where Z and X^t are the template and search images at time step t, C is the command token, and Y denotes the target sequence associated with X. In ARTrack-AC, accurate mode, efficient mode, and difficulty-aware controller operate within a shared trajectory space. This unified representation enables seamless mode switching without disrupting temporal coherence or autoregressive consistency.

3.3. Adaptive Capacity Inference Paradigm

System-level autoregression is instantiated by the following three cooperating components:

- **Accurate Mode (high capacity).** A high-capacity AR tracker is activated on hard frame to ensure robustness via sequential inference.
- **Efficient Mode (low capacity).** A low-capacity AR tracker is deployed on stable segment to improve efficiency via multi-frame parallel inference.
- **Difficulty-Aware Controller.** A lightweight, diffusion-based estimator forecasts the difficulty of the upcoming segment, determines the capacity for the next step, and provides trajectory priors for cropping.

As illustrated in Fig. 2, at each step the controller uses current visual observation and historical trajectories to forecast the upcoming segment. It then (i) selects the appropriate capacity (accurate vs. efficient) and (ii) determines the cropping strategy. For stable segments, crops derived from the diffusion-predicted trajectory priors enable multi-frame batching in the efficient mode. For hard frame, crops are instead obtained from the previous state, and the tracker proceeds in a sequential frame-by-frame manner.

3.4. Diffusion Trajectory Modeling

To modulate capacity over time, we model short-term future motion with a conditional diffusion process that captures the multi-modal uncertainty of trajectories. Given an observation window of N frames, the diffusion model conditions on the previous trajectories $Y^{t-N+1:t}$ and predicts the next N trajectories $Y^{t+1:t+N}$.

At time t, the accurate mode produces the visual observation V^t and the historical trajectory segment $H^t = Y^{t-N+1:t}$. We project and concatenate them to obtain the condition:

$$v^t = \phi_v(V^t), \quad h^t = \phi_h(H^t), \quad \mathbf{C}^t = [v^t; h^t], \tag{2}$$

where ϕ_v and ϕ_h are learnable projection heads.

Starting from Gaussian noise $\mathbf{x}_K \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, the reverse process iteratively denoises to obtain a future trajectory sample:

$$p_\theta(\mathbf{x}_{k-1} \mid \mathbf{x}_k, \mathbf{C}^t) = \mathcal{N}\big(\boldsymbol{\mu}_\theta(\mathbf{x}_k, k, \mathbf{C}^t),\ \sigma_k^2\mathbf{I}\big), \tag{3}$$

where $\boldsymbol{\mu}_\theta$ is predicted by a conditional noise network and σ_k follows the noise schedule. At each denoising step k, we decode $\mathbf{x}_k$ into a trajectory hypothesis:

$$Y_k^{t+1:t+N} = \psi(\mathbf{x}_k, \mathbf{C}^t), \tag{4}$$

and the final sample $Y_0^{t+1:t+N}$ serves as the short-term future trajectory used for difficulty assessment and crop prediction.

3.5. Difficulty Estimation Criterion

We derive difficulty directly from the *convergence behaviour* of diffusion denoising, avoiding additional heads or supervision.

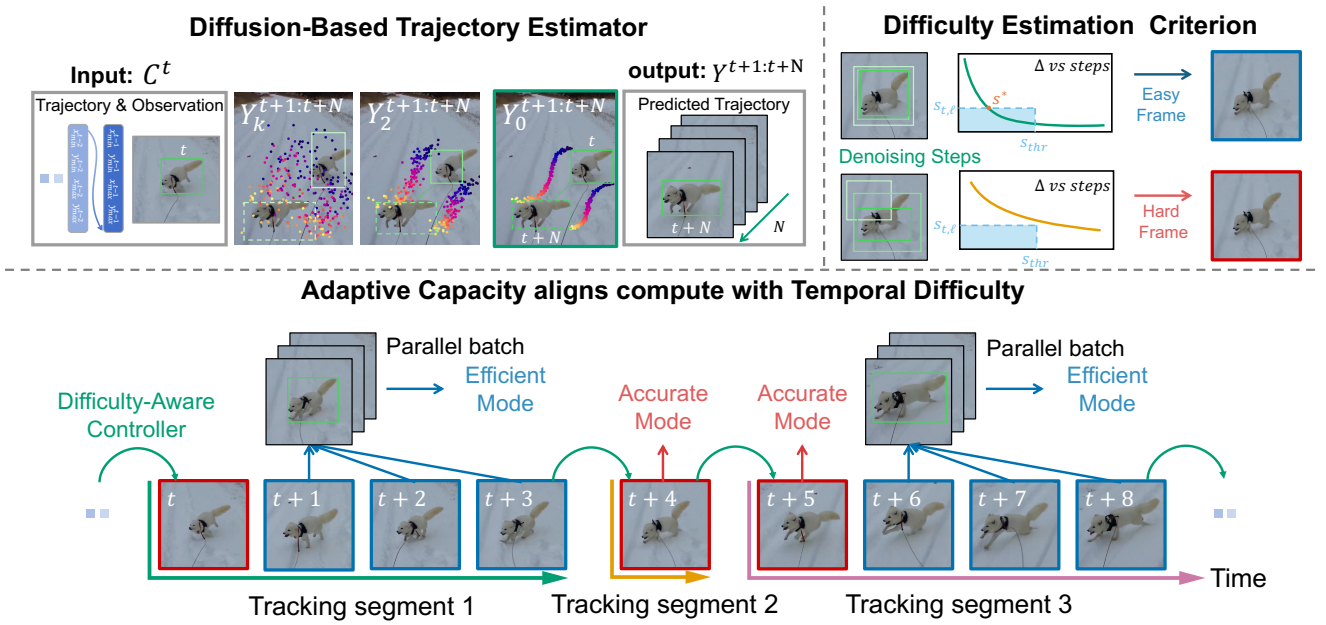


Figure 2. **Overview of ARTrack-AC.** The proposed framework dynamically aligns computational capacity with temporal difficulty. (Top) The diffusion-based trajectory estimator observes recent trajectories and visual features to model future motion uncertainty. Through denoising-based convergence analysis, it identifies easy and hard segments without any explicit supervision. (Bottom) The adaptive capacity paradigm leverages the estimated difficulty to regulate system-level autoregression: efficient mode performs parallel inference for stable segments, while accurate mode sequentially handles complex regions. The difficulty-aware controller adaptively switches between modes and supplies diffusion-predicted trajectory priors for multi-frame parallel tracking, effectively balancing robustness and efficiency across time.

Stability signal. For a predicted future horizon of N frames and diffusion steps indexed by k, let

$$Y_k^{t+1:t+N} = \{\mathbf{y}_{t+1}^{(k)}, \ldots, \mathbf{y}_{t+N}^{(k)}\}$$

denote the decoded trajectory hypothesis at denoising step k. For each future frame $t+\ell$ ($\ell \in \{1, \ldots, N\}$), we measure the adjacent-step change as

$$\Delta_{t,\ell}^{(k)} = \left\| \mathbf{y}_{t+\ell}^{(k)} - \mathbf{y}_{t+\ell}^{(k-1)} \right\|_\infty, \tag{5}$$

capturing how quickly the predicted trajectory stabilizes during denoising.

The stability score for frame $t+\ell$ is defined as the maximum change within an early window of denoising steps:

$$S_{t,\ell} = \max_{k=1,\ldots,s_{\text{thr}}} \Delta_{t,\ell}^{(k)}, \tag{6}$$

where s_{thr} is a small cutoff chosen for efficiency.

Stability threshold and stable-segment length. A frame is considered easy if $S_{t,\ell} \leq \tau_\Delta$ and hard otherwise, with τ_Δ an amplitude threshold. Starting from $t+1$, the number of consecutive easy frames defines the easy-segment length. Frames outside this run are treated as hard and scheduled to the accurate mode.

3.6. Training

During segment-level training, only the diffusion model is optimized, with the accurate tracker (high capacity) frozen to provide visual observations as conditioning information. The overall loss function is formulated as:

$$\mathcal{L} = \mathcal{L}_{\text{MSE}} + \lambda_1 \mathcal{L}_{\text{L1}} + \lambda_2 \mathcal{L}_{\text{SIoU}}, \tag{7}$$

where $\mathcal{L}_{\text{MSE}}$ represents the diffusion denoising loss, and $\mathcal{L}_{\text{L1}}$ and $\mathcal{L}_{\text{SIoU}}$ impose geometric constraints. λ_1 and λ_2 are weighting hyperparameters.

4. Experiments

4.1. Implementation Details

Setup. The diffusion model is trained on 4 NVIDIA RTX A6000 GPUs with a total training time of approximately 15 hours. The inference process runs on NVIDIA Titan Xp GPU and Intel Xeon Gold 6230R CPU @ 3.00GHz.

Table 1. Efficiency of Each Component in ARTrack-AC.

Component	GPU	CPU	Params (M)	MACs (G)
ARTrackOT (Accurate)	65	16	108.05	17.36
FARTrack$_{\text{nano}}$ (Efficient)	210	77	4.59	1.78
Diffusion Model	283	234	0.51	0.05

Components of the inference paradigm. ARTrack-AC consists of three components: an accurate (high capacity) tracker, an efficient (low-capacity) tracker, and a diffusion model. The accurate tracker is the single-template version of ARTrackV2 [1] (denoted as ARTrackOT), while the efficient tracker corresponds to the 10-layer pico variant in FARTrack [36]. The diffusion model is configured with an observation window of size 5, with an embedding dimension of 96 and a single transformer layer while performing denoising through a 4-step DDIM [34] sampling process. The computational efficiency of these three components is summarized in Table 1.

Training Strategy. The diffusion model is trained on GOT-10k [21], TrackingNet [32], and LaSOT [14]. We used the AdamW optimizer [31] with a weight decay of 1×10^{-4} and an initial learning rate of 1×10^{-4}. The model follows the velocity-prediction formulation [33]. Training was performed for 300 epochs, each consisting of 76,800 randomly sampled frame–trajectory pairs.

4.2. Main Results

We evaluate ARTrack-AC across four standard benchmarks and report three variants, as summarized in Table 2:

- **ARTrack-AC$_{\text{para10}}$**: observation window of 10, with the efficient mode running in parallel.
- **ARTrack-AC$_{\text{para}}$**: observation window of 5, with the efficient mode running in parallel.
- **ARTrack-AC$_{\text{seq}}$**: observation window of 5, with the efficient mode running sequentially.

LaSOT [14]. LaSOT contains 1,400 training and 280 test sequences captured at 30 FPS, providing long-term, diverse, and challenging scenarios. As demonstrated in Table 2, ARTrack-AC$_{\text{para}}$ matches the accuracy of ARTrack-AC$_{\text{seq}}$ while running 1.5× faster. It delivers a remarkable balance between accuracy and efficiency, setting a new state of the art while running at 191 FPS.

TrackingNet [32]. TrackingNet contains over 30,000 short-term training videos and 511 held-out test sequences, all captured at 30 FPS. As summarized in Table 2, our method achieves superior performance in both accuracy and speed. Even in short sequences, ARTrack-AC effectively allocates computation, outperforming specialized efficient trackers in accuracy while maintaining inference speed.

GOT-10k [21]. GOT-10k contains over 10,000 videos captured at 10 FPS with frame-level annotations. ARTrack-AC$_{\text{para}}$ achieves the best accuracy and GPU speed among all trackers. Table 2 demonstrates that ARTrack-AC$_{\text{seq}}$ matches the runtime of AsymTrack-B on both GPU and CPU while exceeding it by 9.7% AO, showing that our inference paradigm remains robust even on low-frame-rate datasets. ARTrack-AC$_{\text{para10}}$ exhibits a noticeable accuracy drop, as a 10-frame batch at 10 FPS spans a wide temporal gap that weakens the reliability of trajectory-guided cropping.

LaSOText [15]. LaSOText extends LaSOT with 150 additional videos from 15 new categories, featuring dense distractors, frequent occlusions, and small fast-moving targets, all captured at 30 FPS. As shown in Table 2, all three variants of ARTrack-AC surpass existing efficient trackers across all three metrics, even under the 10-frame parallel setting. This demonstrates the suitability of our inference paradigm for challenging scenarios and further confirms its ability to align computation with temporal difficulty.

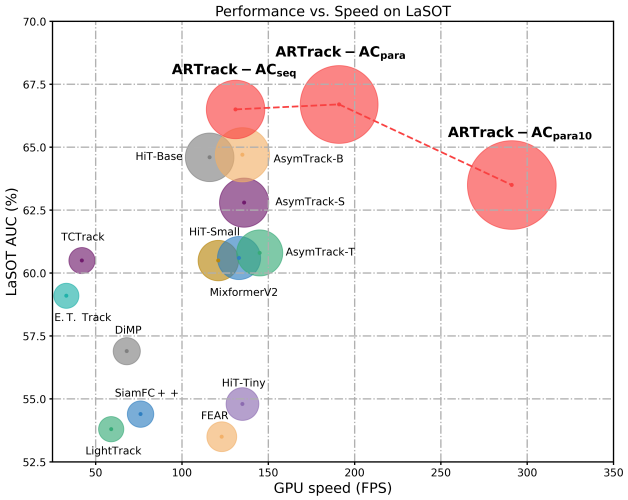


Figure 3. Comparison of accuracy vs. latency trade-off for different tracking methods in LaSOT.

4.3. Accuracy-Latency Trade-off

We conducted a comparative analysis of state-of-the-art trackers on the LaSOT benchmark, as depicted in Figure 3 our ARTrack-AC$_{\text{para}}$ establishes a new state of the art, achieving an impressive AUC of 66.7% and delivering competitive performance at 191 FPS, surpassing all other trackers in both accuracy and efficiency. Moreover, ARTrack-AC$_{\text{para10}}$ runs 2× faster than AsymTrack-T while yielding a 4.5% improvement in AUC.

4.4. Analysis of the Inference Paradigm

We analyze the proposed Adaptive Capacity Inference Paradigm on the LaSOT dataset. Unless otherwise specified, all experiments follow a consistent evaluation protocol, and the default settings are marked in gray.

Switching Strategy. In our inference paradigm, the proportion of steps executed in the efficient mode controls the overall inference speed, while the switching strategy governs the resulting accuracy–speed balance. As illustrated

Table 2. State-of-the-art comparison on LaSOT [14], TrackingNet [32], GOT-10k [21], and LaSOText [15]. Best in **bold**, second best underlined, and third best underwave. The inference speed of ARTrack-AC is averaged over four datasets under a unified threshold.

Methods	GPU FPS	CPU FPS	LaSOT			TrackingNet			GOT-10k			LaSOText		
			AUC(%)	P_{Norm}(%)	P(%)	AUC(%)	P_{Norm}(%)	P(%)	AO(%)	$SR_{0.5}$(%)	$SR_{0.75}$(%)	AUC(%)	P_{Norm}(%)	P(%)
DiMP [2]	68	22	56.9	65.0	56.7	74.0	80.1	70.6	-	-	-	-	-	-
SiamFC++ [44]	76	28	54.4	62.3	54.7	75.4	80.0	68.7	-	-	-	-	-	-
LightTrack [46]	59	27	53.8	60.5	53.7	72.5	77.8	69.5	61.1	71.0	54.3	-	-	-
TCTrack [8]	42	29	60.5	69.3	62.4	74.8	79.6	73.3	66.2	75.6	61.0	-	-	-
FEAR [4]	123	28	53.5	59.7	54.5	70.2	80.8	71.5	61.9	72.2	52.5	-	-	-
E.T. Track [3]	33	16	59.1	66.8	60.1	72.5	77.8	69.5	56.6	64.6	42.5	-	-	-
HiT-Tiny [22]	135	42	54.8	60.5	52.9	74.6	78.1	68.8	52.6	59.3	42.7	35.8	41.4	35.5
HiT-Small [22]	121	35	60.5	68.3	61.5	77.7	81.9	73.1	62.6	71.2	54.4	40.4	-	-
HiT-Base [22]	116	30	64.6	73.3	68.1	80.0	84.4	77.3	64.0	72.1	58.1	44.1	-	-
MixformerV2-S [13]	133	31	60.6	69.9	60.4	75.8	81.1	70.4	61.9	71.7	51.3	43.6	-	46.2
AsymTrack-T [53]	145	55	60.8	68.7	61.2	76.2	80.9	71.6	62.3	71.3	54.7	42.5	-	-
AsymTrack-S [53]	136	48	62.8	71.2	64.8	77.9	82.2	74.0	65.5	74.8	58.9	43.3	-	-
AsymTrack-B [53]	135	32	64.7	73.0	67.8	80.0	84.5	77.4	67.7	76.6	61.4	44.6	-	-
ARTrack-AC$_{para10}$	**291**	**57**	63.5	73.0	66.0	79.7	84.1	74.5	62.6	72.8	53.2	45.2	55.2	50.5
ARTrack-AC$_{para}$	191	42	**66.7**	**76.5**	71.5	81.8	86.5	78.4	72.3	82.4	67.3	**47.5**	**57.9**	**53.3**
ARTrack-AC$_{seq}$	131	42	66.5	76.3	**72.0**	**82.6**	**87.7**	**80.0**	**74.3**	**84.8**	**70.2**	46.6	56.7	52.8

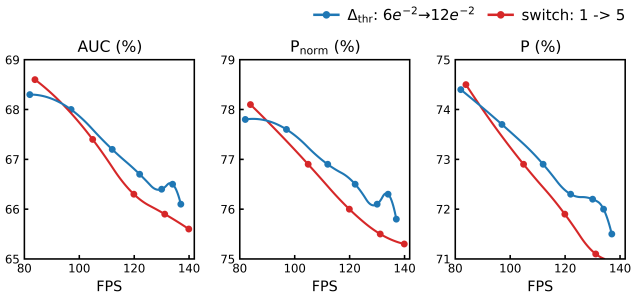


Figure 4. **Comparison of switching strategies.** We compare our difficulty-aware switching (DAS) against fixed-cycle switching (FCS) in a non-parallel setting. DAS controls the proportion of efficient-mode steps by varying the stability threshold from 6×10^{-2} to 12×10^{-2}, whereas FCS activates the accurate mode at a fixed interval, switching every 1 to 5 frames.

in Fig. 4, We compare two switching strategies: fixed-cycle switching, which activates the accurate mode every n frames, and difficulty-aware (self-regulated) switching, triggered by a stability threshold. Fixed-cycle switching ignores temporal difficulty and acts as a blind scheduling rule, when dominated by efficient mode, drifting predictions in challenging segments contaminate the historical context and undermine autoregressive consistency, leading to noticeable degradation. In contrast, difficulty-aware switching aligns computational capacity with temporal difficulty, activating the accurate mode selectively based on predicted stability, thereby reducing the likelihood of corrupting the historical context while maintaining high inference efficiency.

Role of the Diffusion Model: Prior vs. Predictor. We investigate whether diffusion-predicted trajectories can serve as final tracking outputs versus acting as priors for localization-based tracking. The diffusion model predicts trajectories under a segment-level autoregression framework, where each segment's prediction depends on the previous one. In self-conditioned, the diffusion-predicted segment is directly used as conditioning for the next. In refined-conditioned, FARTrack$_{nano}$ further refines the diffusion-predicted trajectory at the frame level before it is reused as conditioning for the following segment. When the refined trajectories are adopted as final outputs, the setup corresponds to the ARTrack-AC$_{para}$ variant.

Table 3. Role of Diffusion Model.

tracking paradigm	AUC	P_{Norm}	P
self-conditioned (generative)	38.5	36.4	26.7
refined-conditioned (generative)	56.9	65.4	56.6
FARTrack$_{nano}$ (localization-based)	60.2	69.0	64.2
ARTrack-AC$_{para}$ (localization-based)	**66.7**	**76.3**	**72.0**

As shown in Table 3, self-conditioned suffers from compounding errors, accumulating drift across segments and degrading accuracy versus localization-based trackers. Refined-conditioned alleviates drift via per-frame refinement and improves performance, yet diffusion trajectories ensure coverage rather than precise alignment, leaving a consistent gap to localization methods. Thus, the diffusion model is most effective as a trajectory prior to maintain temporal coherence, rather than serving as the final predictor.

Table 4. Trajectory Predictor comparison.

trajectory predictor	AUC	P_{Norm}	P
regression model	53.9	61.4	48.6
diffusion model	**56.9**	**65.4**	**56.6**

Trajectory Predictor. In this subsection, we delve into trajectory predictors within our inference paradigm. We compare two trajectory predictors that share the same transformer-based decoder and are conditioned on both the current visual observation and the previous trajectory, as

shown in Table 4 under the refined-conditioned setting. Diffusion consistently outperforms regression, providing more reliable priors for tracking. While regression performs well on simple and steady motion, it tends to average possible motion modes and lacks calibrated uncertainty under rapid or abrupt target and camera changes, leading to delayed responses and tracking drift. In contrast, diffusion captures multi-modal motion, adapts to high-frequency transitions, and yields stronger temporal consistency and robustness on challenging segments.

Table 5. Difficulty estimation signal comparison.

difficulty signal	AUC	P_{Norm}	P	Eff. Ratio
fixed-cycle	65.8	75.5	71.1	80.0%
cosine-distance (learned)	65.9	75.7	71.5	80.7%
stability-signal (training-free)	**66.5**	**76.3**	**72.0**	**81.1%**

Difficulty Estimation Signals. The design of the difficulty estimator signal dictates how effectively the self-regulated tracking paradigm aligns computational expenditure with temporal complexity. We compare two signal designs under a similar compute budget against a non-adaptive fixed-cycle baseline: a supervised cosine-distance regressor and our implicit stability-signal. As shown in Table 5, the stability-signal yields clear gains over fixed-cycle, effectively anticipating motion volatility and sequence-level instability by sensing the diffusion model's own uncertainty—slower convergence implies more complex situations. In contrast, cosine-distance yields only marginal improvement despite supervision and extra computation, since it mainly reflects local trajectory quality and poorly anticipates upcoming challenges. These results support convergence-based implicit signals as a more adaptive and efficient mechanism for dynamic capacity modulation.

Cross-Tracker Generalization. We evaluate the adaptive capacity inference paradigm under two settings while keeping the coordination mechanism unchanged. In AR-Sub, we substitute the accurate mode with a higher-capacity (slower) AR tracker and the efficient mode with a lower-capacity (faster) AR tracker. In NonAR-Sub, we replace both modes with Non-AR trackers of comparable accuracy–speed characteristics.

As shown in Table 6, increasing the capacity of the accurate AR mode brings little benefit, whereas lowering the capacity of the efficient AR mode still yields a baseline-level improvement, indicating broad applicability on AR trackers and that gains originate from adaptive coordination rather than absolute capacity. In NonAR-Sub, swapping the accurate mode for a high-capacity Non-AR tracker keeps performance near baseline because that mode covers nearly 20% of the overall inference and mainly corrects trajectories to preserve temporal and autoregressive consistency. Replacing the efficient mode with a lower-capacity Non-AR tracker reduces the improvement. Notably, low-capacity Non-AR trackers (HiT-Tiny) benefit markedly under parallel inference, where diffusion-predicted trajectory crops act as trajectory priors, enabling Non-AR trackers to adapt predictions with temporal context.

Predicted Crop. To assess whether the predicted trajectory can reliably guide search region cropping, we adopt a mixed-cropping scheme under a fixed stability threshold, where the first m frames use Predicted Crop (cropping search regions guided by the predicted trajectory for parallel inference), and the remaining $5-m$ frames use Tracked Crop (based on the previous bounding box, sequential inference). We denote this setting as PC@ m/5 . As shown in Table 7, Predicted Crop does not significantly degrade accuracy (and even yields minor gains in some settings), while relaxing the strict temporal dependency between consecutive frames to enable multi-frame parallel inference, which substantially boosts inference speed.

Table 7. Predicted Crop.

PC@m/5	AUC	P_{Norm}	P	GPU
0/5	66.5	76.3	72.0	134
1/5	65.9	75.6	71.3	134
2/5	66.6	76.2	72.0	149
3/5	66.1	75.8	71.2	166
4/5	66.9	76.8	71.9	181
5/5	**66.7**	**76.5**	**71.5**	**198**

Table 8. Observation Window.

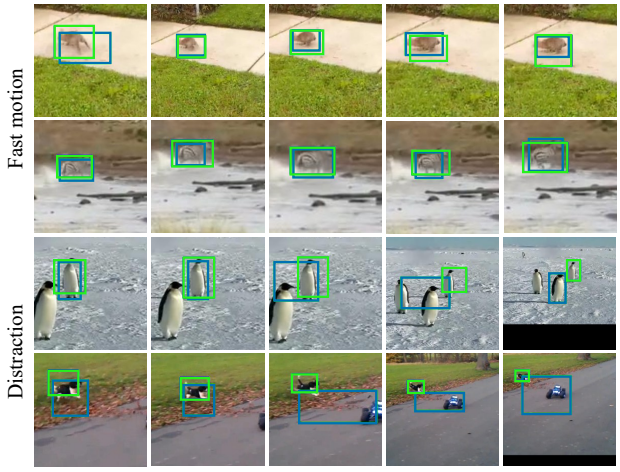
window	AUC	P_{Norm}	P	GPU
5	**66.7**	**76.5**	**71.5**	**198**
6	65.2	74.9	69.2	226
7	64.8	74.4	68.2	246
8	64.7	74.4	68.0	269
9	64.0	73.6	66.8	279
10	63.5	73.0	66.0	291

Observation Window. The observation window defines the diffusion model's trajectory receptive horizon and sets the upper limit of parallelization. While the stability threshold provides fine-grained control over the speed–accuracy trade-off, the window size governs the paradigm's coarse-level adjustment. We vary the window size to assess its influence on the capacity modulation dynamics of our adaptive inference paradigm. As shown in Table 8, larger windows extend the prediction horizon but also amplify uncertainty and crop bias, increasing the proportion of efficient mode steps and reducing accuracy on challenging segments. This produces a clear speed–accuracy trade-off: small windows (5–6) favor accuracy with minimal overhead, medium windows (7–8) offer balanced performance, and large windows (9–10) prioritize speed at the cost of moderate accuracy degradation.

Trajectory Visualization. We visualize and compare the trajectories predicted by the diffusion model and those produced by the low-capacity tracker under two representative scenarios: Fast Motion and Distraction. As depicted in Figure 5, in the Fast Motion case, the diffusion-predicted trajectory successfully covers the moving target throughout the

Table 6. **Cross-Tracker Generalization.** Overview of tracker configurations used to evaluate the adaptive capacity inference paradigm across *autoregressive* (AR) and *non-autoregressive* (Non-AR) tracking paradigms. The table lists the substitutions for accurate and efficient modes with varying capacities and architectures. Non-AR trackers are masked in blue

Methods		GPU	CPU	Eff.	LaSOT			TrackingNet			GOT-10k			LaSOText		
Accurate	Efficient	FPS	FPS	Ratio	AUC(%)	P_{Norm}(%)	P(%)	AUC(%)	P_{Norm}(%)	P(%)	AO(%)	$SR_{0.5}$(%)	$SR_{0.75}$(%)	AUC(%)	P_{Norm}(%)	P(%)
Fixed High Capacity																
ARTrackv2$_{256}$		52	13	0%	71.5	80.5	77.8	84.6	88.9	83.8	76.9	86.0	75.3	51.1	61.9	58.3
ARTrackOT		65	16	0%	69.4	77.8	74.8	84.1	88.3	82.7	74.5	83.4	72.5	49.2	58.7	55.1
OSTrack$_{256}$		73	18	0%	69.1	78.7	75.2	83.1	87.8	82.0	73.7	84.0	69.8	47.4	57.3	53.3
Fixed Low Capacity																
	FARTrack$_{nano}$	210	77	100%	60.2	69.0	64.2	79.1	84.5	75.6	68.9	79.5	60.9	42.4	52.5	47.1
	FARTrack$_{pico}$	343	121	100%	58.7	67.3	59.8	75.8	81.4	71.0	62.8	72.7	50.4	42.0	51.6	46.3
	HiT-Tiny	135	42	100%	54.8	60.5	52.9	74.6	78.1	68.8	52.6	59.3	42.7	35.8	41.4	35.5
Adaptive Capacity (Sequential)																
ARTrackv2$_{256}$	FARTrack$_{nano}$	121	38	80.7%	67.8	78.0	72.8	82.1	86.7	78.6	74.9	86.1	70.1	49.2	60.2	55.9
ARTrackOT	FARTrack$_{nano}$	131	42	79.8%	66.5	76.3	72.0	82.6	87.7	80.0	74.3	84.8	70.2	46.6	56.7	52.8
OSTrack$_{256}$	FARTrack$_{nano}$	140	46	81.2%	65.9	75.9	71.6	82.1	87.2	79.6	73.3	83.1	68.7	46.1	56.5	52.1
ARTrackOT	FARTrack$_{pico}$	162	50	79.8%	63.4	73.3	67.1	79.5	84.9	76.1	66.6	76.6	57.2	46.3	56.8	51.5
ARTrackOT	HiT-Tiny	104	31	80.9%	57.1	63.2	56.4	75.8	79.3	70.6	57.7	64.7	48.0	38.2	44.9	39.7
Adaptive Capacity (Parallel)																
ARTrackv2$_{256}$	FARTrack$_{nano}$	173	38	81.2%	67.0	77.1	72.1	81.9	86.4	78.4	72.7	83.2	67.6	47.1	57.7	52.4
ARTrackOT	FARTrack$_{nano}$	191	42	80.1%	66.7	76.5	71.5	81.8	86.5	78.4	72.3	82.4	67.3	47.5	57.9	53.3
OSTrack$_{256}$	FARTrack$_{nano}$	214	46	81.5%	65.8	75.8	70.3	80.9	85.7	77.3	71.7	82.0	66.8	46.3	56.6	51.7
ARTrackOT	FARTrack$_{pico}$	210	50	80.0%	63.5	73.0	65.8	78.6	83.3	73.9	64.6	74.2	54.8	44.9	54.4	48.7
ARTrackOT	HiT-Tiny	216	31	80.9%	59.4	66.2	59.6	76.9	80.5	71.8	56.2	62.7	47.2	38.9	45.5	40.9



(a) time t (b) time $t+1$ (c) time $t+2$ (d) time $t+3$ (e) time $t+4$

Figure 5. **Visualization of diffusion-predicted trajectories.** The blue boxes indicate the predictions of the low-capacity tracker, while the green boxes represent the diffusion-predicted trajectories. (a)-(e) show the search regions and the corresponding predicted boxes.

sequence, demonstrating its robustness to rapid displacement but showing limited focus on precise object boundaries. In contrast, during Distraction, the low-capacity tracker is misled by nearby distractors, resulting in drifting predictions. The diffusion model, by modeling the target's temporal trajectory, maintains stable and accurate localization, preserving temporal consistency even under strong visual interference.

4.5. Limitation Analysis

A major limitation of our inference paradigm is its sensitivity to the video frame rate. While the parallel inference variant preserves performance under high-frame-rate settings, its accuracy degrades as the frame rate decreases, as illustrated in Table 9. This degradation primarily results from reduced fidelity in the diffusion-predicted trajectories, rather than from the difficulty-aware controller that schedules Accurate and Efficient modes. Since the diffusion estimator focuses on modeling the trajectory stability distribution and motion volatility, rather than regressing frame-level precise coordinates, it can still provide a robust stability signal for dual-mode switching even under low-frame-rate conditions.

Table 9. Impact of frame rate on performance across NFS [23] (240/30 fps) and GOT-10k [21] (10 fps).

Methods	Eff.	NFS (240 fps)		NFS (30 fps)		GOT-10k (10 fps)	
	Ratio	AUC	P	AUC	P	AO	$SR_{0.75}$
ARTrackOT	0%	69.5	84.9	65.4	78.5	74.5	72.5
FARTrack$_{nano}$	100%	67.1	81.9	62.5	75.1	68.9	60.9
ARTrack-AC$_{seq}$	81.5%	69.9	85.9	65.8	79.7	74.3	70.2
ARTrack-AC$_{para}$	81.4%	69.4	84.7	64.8	77.3	72.3	67.3

5. Conclusion

In this paper, we have revisited autoregressive visual tracking from a system-level perspective, moving beyond deciding *what to predict* to regulating *how much capacity* is needed over time. ARTrack-AC couples temporal autoregression with a diffusion-based trajectory estimator, whose convergence behavior serves as a difficulty-aware scheduling signal for adaptive capacity allocation across segments, aligning computation with temporal complexity while preserving AR consistency. This shift from fixed-capacity designs to difficulty-aware, self-regulated inference offers a principled path toward more resource-aware tracking systems.

References

[1] Yifan Bai, Zeyang Zhao, Yihong Gong, and Xing Wei. Artrackv2: Prompting autoregressive tracker where to look and how to describe. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 19048–19057, 2024. 1, 2, 5

[2] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In ICCV, 2019. 6

[3] Philippe Blatter, Menelaos Kanakis, Martin Danelljan, and Luc Van Gool. Efficient visual tracking with exemplar transformers. In WACV, 2023. 6

[4] Vasyl Borsuk, Roman Vei, Orest Kupyn, Tetiana Martyniuk, Igor Krashenyi, and Jiři Matas. Fear: Fast, efficient, accurate and robust visual tracker. In ECCV, 2022. 2, 6

[5] Wenrui Cai, Qingjie Liu, and Yunhong Wang. Hiptrack: Visual tracking with historical prompts. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19258–19267, 2024. 1

[6] Wenrui Cai, Qingjie Liu, and Yunhong Wang. Spmtrack: Spatio-temporal parameter-efficient fine-tuning with mixture of experts for scalable visual tracking. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 16871–16881, 2025. 1

[7] Yidong Cai, Jie Liu, Jie Tang, and Gangshan Wu. Robust object modeling for visual tracking. In ICCV, 2023. 2

[8] Ziang Cao, Ziyuan Huang, Liang Pan, Shiwei Zhang, Ziwei Liu, and Changhong Fu. Tctrack: Temporal contexts for aerial tracking. In CVPR, 2022. 6

[9] Xin Chen, Ben Kang, Jiawen Zhu, Dong Wang, Houwen Peng, and Huchuan Lu. Unified sequence-to-sequence learning for single-and multi-modal visual object tracking. arXiv preprint arXiv:2304.14394, 2023. 2

[10] Xin Chen, Houwen Peng, Dong Wang, Huchuan Lu, and Han Hu. Seqtrack: Sequence to sequence learning for visual object tracking. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 14572–14581, 2023. 1, 2

[11] Jongwon Choi, Hyung Jin Chang, Sangdoo Yun, Tobias Fischer, Yiannis Demiris, and Jin Young Choi. Attentional correlation filter network for adaptive visual tracking. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4807–4816, 2017. 3

[12] 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. 2

[13] Yutao Cui, Tianhui Song, Gangshan Wu, and Limin Wang. Mixformerv2: Efficient fully transformer tracking. In NeurIPS, 2023. 2, 6

[14] Heng Fan, Liting Lin, Fan Yang, Peng Chu, Ge Deng, Sijia Yu, Hexin Bai, Yong Xu, Chunyuan Liao, and Haibin Ling. Lasot: A high-quality benchmark for large-scale single object tracking. In CVPR, 2019. 5, 6

[15] 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 (2):439–461, 2021. 5, 6

[16] Fei Gao, Shengzhe You, Yisu Ge, and Shifeng Zhang. Gaussian-based adaptive frame skipping for visual object tracking. The Visual Computer, 40(10):6897–6912, 2024. 2

[17] 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. 2

[18] Mingzhe Guo, Weiping Tan, Wenyu Ran, Liping Jing, and Zhipeng Zhang. Dreamtrack: Dreaming the future for multimodal visual object tracking. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 7201–7210, 2025. 1

[19] Lingyi Hong, Jinglun Li, Xinyu Zhou, Shilin Yan, Pinxue Guo, Kaixun Jiang, Zhaoyu Chen, Shuyong Gao, Runze Li, Xingdong Sheng, et al. General compression framework for efficient transformer object tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 13427–13437, 2025. 1

[20] Chen Huang, Simon Lucey, and Deva Ramanan. Learning policies for adaptive tracking with deep feature cascades. In Proceedings of the IEEE international conference on computer vision, pages 105–114, 2017. 3

[21] Lianghua Huang, Xin Zhao, and Kaiqi Huang. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. PAMI, 2019. 5, 6, 8

[22] Ben Kang, Xin Chen, Dong Wang, Houwen Peng, and Huchuan Lu. Exploring lightweight hierarchical vision transformers for efficient visual tracking. In ICCV, 2023. 2, 6

[23] 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. 8

[24] 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 NeurIPS, 2023. 2

[25] Yun Gu Lee. Confidence-guided frame skipping to enhance object tracking speed. Sensors, 24(24):8120, 2024. 2

[26] Bo Li, Junjie Yan, Wei Wu, Zheng Zhu, and Xiaolin Hu. High performance visual tracking with siamese region proposal network. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8971–8980, 2018. 2

[27] Bo Li, Wei Wu, Qiang Wang, Fangyi Zhang, Junliang Xing, and Junjie Yan. Siamrpn++: Evolution of siamese visual tracking with very deep networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pages 4282–4291, 2019. 2

[28] Xiaohai Li, Bineng Zhong, Qihua Liang, Zhiyi Mo, Jian Nong, and Shuxiang Song. Dynamic updates for language adaptation in visual-language tracking. In Proceedings of the

Computer Vision and Pattern Recognition Conference, pages 19165–19174, 2025. 1

[29] Shiyi Liang, Yifan Bai, Yihong Gong, and Xing Wei. Autoregressive sequential pretraining for visual tracking. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 7254–7264, 2025. 1, 2

[30] Jing Liu, Yating Wang, Xiangdong Huang, and Yuting Su. Tracking by dynamic template: Dual update mechanism. Journal of Visual Communication and Image Representation, 84:103456, 2022. 2

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

[32] Matthias Muller, Adel Bibi, Silvio Giancola, Salman Alsubaihi, and Bernard Ghanem. Trackingnet: A large-scale dataset and benchmark for object tracking in the wild. In ECCV, 2018. 5, 6

[33] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In International Conference on Learning Representations, 2022. 5

[34] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In International Conference on Learning Representations, 2021. 5

[35] Yuedong Tan, Jiawei Shao, Eduard Zamfir, Ruanjun Li, Zhaochong An, Chao Ma, Danda Paudel, Luc Van Gool, Radu Timofte, and Zongwei Wu. What you have is what you track: Adaptive and robust multimodal tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 3455–3465, 2025. 1

[36] Guijie Wang, Tong Lin, Yifan Bai, Anjia Cao, Shiyi Liang, Wangbo Zhao, and Xing Wei. FARTrack: Fast autoregressive visual tracking with high performance. In The Fourteenth International Conference on Learning Representations, 2026. 2, 5

[37] Jun Wang, Changwang Lai, Yuanyun Wang, and Wenshuang Zhang. Emat: Efficient feature fusion network for visual tracking via optimized multi-head attention. Neural Networks, 172:106110, 2024. 2

[38] Qingmao Wei, Bi Zeng, Jianqi Liu, Li He, and Guotian Zeng. Litetrack: Layer pruning with asynchronous feature extraction for lightweight and efficient visual tracking. In 2024 IEEE International Conference on Robotics and Automation (ICRA), pages 4968–4975. IEEE, 2024. 1, 2

[39] 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. 2

[40] You Wu, Xucheng Wang, Xiangyang Yang, Mengyuan Liu, Dan Zeng, Hengzhou Ye, and Shuiwang Li. Learning occlusion-robust vision transformers for real-time uav tracking. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 17103–17113, 2025. 1

[41] Fei Xie, Zhongdao Wang, and Chao Ma. Diffusiontrack: Point set diffusion model for visual object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19113–19124, 2024. 1

[42] Jinxia Xie, Bineng Zhong, Zhiyi Mo, Shengping Zhang, Liangtao Shi, Shuxiang Song, and Rongrong Ji. Autoregressive queries for adaptive tracking with spatio-temporal transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 19300–19309, 2024. 1, 2

[43] Mengjie Xu, Yitao Zhu, Haotian Jiang, Jiaming Li, Zhenrong Shen, Sheng Wang, Haolin Huang, Xinyu Wang, Han Zhang, Qing Yang, et al. Mitracker: Multi-view integration for visual object tracking. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 27176–27185, 2025. 1

[44] Yinda Xu, Zeyu Wang, Zuoxin Li, Ye Yuan, and Gang Yu. Siamfc++: Towards robust and accurate visual tracking with target estimation guidelines. In AAAI, 2020. 6

[45] Chaocan Xue, Bineng Zhong, Qihua Liang, Yaozong Zheng, Ning Li, Yuanliang Xue, and Shuxiang Song. Similarity-guided layer-adaptive vision transformer for uav tracking. In Proceedings of the Computer Vision and Pattern Recognition Conference, pages 6730–6740, 2025. 3

[46] 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 CVPR, 2021. 2, 6

[47] Xiangyang Yang, Dan Zeng, Xucheng Wang, You Wu, Hengzhou Ye, Qijun Zhao, and Shuiwang Li. Adaptively bypassing vision transformer blocks for efficient visual tracking. Pattern Recognition, 161:111278, 2025. 3

[48] Botao Ye, Hong Chang, Bingpeng Ma, Shiguang Shan, and Xilin Chen. Joint feature learning and relation modeling for tracking: A one-stream framework. In European conference on computer vision, pages 341–357. Springer, 2022. 1, 2

[49] Kibum Yun, Kyujin Shim, Kangwook Ko, and Changick Kim. Dynamic template update for visual object tracking. In 2022 IEEE International Conference on Image Processing (ICIP), pages 3111–3115. IEEE, 2022. 2

[50] Mingyang Zhang, Kristof Van Beeck, and Toon Goedemé. Object tracking with multiple dynamic templates updating. In International Conference on Image and Vision Computing New Zealand, pages 144–158. Springer, 2022. 2

[51] Minghua Zhang, Qiuyang Zhang, Wei Song, Dongmei Huang, and Qi He. Promptvt: Prompting for efficient and accurate visual tracking. IEEE Transactions on Circuits and Systems for Video Technology, 34(8):7373–7385, 2024. 2

[52] Xinyu Zhou, Pinxue Guo, Lingyi Hong, Jinglun Li, Wei Zhang, Weifeng Ge, and Wenqiang Zhang. Reading relevant feature from global representation memory for visual object tracking. In NeurIPS, 2023. 2

[53] Jiawen Zhu, Huayi Tang, Xin Chen, Xinying Wang, Dong Wang, and Huchuan Lu. Two-stream beats one-stream: asymmetric siamese network for efficient visual tracking. In Proceedings of the AAAI Conference on Artificial Intelligence, pages 10959–10967, 2025. 2, 6