WHERE TO INTERACT AND WHEN TO SKIP: EFFICIENT TRANSFORMER FOR UNDERWATER VISUAL TRACKING

Anonymous authors
Paper under double-blind review

ABSTRACT

Underwater visual tracking is challenged by degraded imaging conditions such as light attenuation and scattering, as well as by limited computational resources on underwater platforms that restrict real-time deployment. Existing Transformer trackers typically perform dense template–search interactions and execute all Transformer layers for every frame, resulting in considerable redundant computation. To address these issues, we propose W^{2}Track, an efficient Transformer-based underwater tracker that allocates computation adaptively in both token interaction and network depth, determining where token interactions should occur and when redundant Transformer blocks can be skipped. Specifically, W^{2}Track contains two key components: (i) Stage-aware Attention Interaction (SAI), which reorganizes template-search interactions across Transformer stages. In shallow stages, SAI removes search-to-template interactions to prevent ambiguous information propagation caused by dense and visually similar underwater targets. In deeper stages, SAI eliminates redundant intra-frame interactions and focuses on bidirectional template-search correspondence to improve target matching and localization. (ii) Adaptive Layer Execution (ALE), which dynamically adjusts Transformer depth according to the tracking state. Since underwater sequences exhibit varying target visibility, motion patterns, and image quality, ALE estimates scene complexity and skips redundant deep-layer computation in easy cases where additional layers provide limited benefits. Experiments on UTB180, UOT100, and VMAT demonstrate that W^{2}Track achieves competitive tracking accuracy and real-time efficiency, reaching 210.52 FPS on GPU while outperforming existing real-time trackers.

1 INTRODUCTION

Underwater visual tracking (Alawode et al., 2022; Panetta et al., 2021; Kezebou et al., 2019) is an important branch of visual object tracking, with broad applications in marine ecological monitoring, marine ranch and underwater safety perception. However, compared with terrestrial visual tracking, underwater tracking faces unique challenges caused by light attenuation, scattering, and other complex environmental factors. Moreover, underwater edge devices are often limited by strict power budgets and computational resources, posing significant challenges for efficient model deployment. Therefore, an effective underwater tracker should maintain robust tracking accuracy while achieving real-time inference under degraded underwater environments.

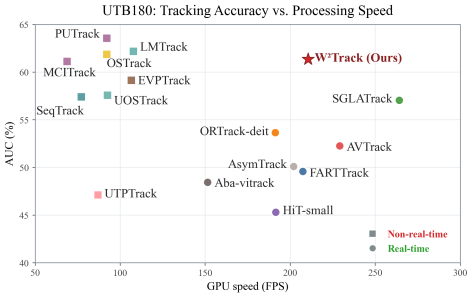

Figure 1: Accuracy-speed comparison on UTB180. W^{2}Track achieves a favorable trade-off among real-time trackers.

Recently, Transformer-based architectures have become the dominant paradigm in visual object tracking. Among them, OSTrack (Ye et al., 2022) and MixFormer (Cui et al., 2022) adopt a one-stream framework that unifies feature learning and relation modeling by directly connecting the template and search images, substantially improving computational efficiency. Building on this paradigm, subsequent studies further reduce computational overhead through token pruning (Liang et al., 2022; Rao et al., 2021), sparse attention (Gopal & Amer, 2024), and dynamic depth adjustment (Yin et al., 2022; Xue et al., 2025). However, their domain-agnostic designs generalize poorly to degraded underwater conditions. Consequently, several trackers have been developed specifically for underwater scenarios. For example, PUTrack

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(Zhang et al., 2025) enhances underwater tracking by mining underwater-specific prompts and progressively propagating them through all semantic layers of an open tracker; UOSTrack (Li et al., 2023b) introduces a hybrid training strategy that combines underwater images with open-air sequences to improve tracking robustness. Despite these improvements, such methods remain too cumbersome for real-time deployment. As shown in Figure 1, existing trackers still struggle to achieve a satisfactory balance between tracking accuracy and inference speed.

To address these issues, we propose an efficient Transformer-based underwater tracker, dubbed W^{2}Track, which reduces redundant computation by determining "Where" token interactions should occur and "When" Transformer blocks can be skipped. First, we introduce Stage-aware Attention Interaction (SAI), which adopts different token interaction patterns between template and search tokens across stages. Since underwater biological targets are often densely distributed and visually similar, SAI removes the search-to-template interaction in shallow layers to avoid premature cross-frame matching and prevent ambiguous information propagation from the search region to template representations. Meanwhile, the template-to-search interaction is retained, enabling template tokens to incorporate target-related contextual cues from the search region and providing richer representations for subsequent correspondence modeling. In deeper layers, once discriminative template and search representations have been established, SAI removes redundant intra-frame interactions and focuses on bidirectional template–search correspondence to improve target matching and localization. This stage-wise allocation reduces redundant interactions while preserving effective target association. Second, we introduce Adaptive Layer Execution (ALE), which dynamically adjusts Transformer depth according to the tracking state. As underwater sequences vary substantially in target visibility, motion, and image quality, tracking difficulty differs across frames. ALE estimates scene complexity and skips unnecessary deep-layer computation for easy cases, where additional Transformer layers offer limited gains. Extensive experiments on multiple popular datasets demonstrate an improved accuracy–speed trade-off. As shown in Figure 1, our tracker achieves an AUC of 61.35% at 210 FPS on UTB180 (Alawode et al., 2022).

The main contributions of this paper are summarized as follows:

- We propose an efficient Transformer-based framework for real-time underwater object tracking, which reduces redundant computation while maintaining robust tracking performance in challenging underwater environments.
- By exploiting the dense target distribution and varying tracking complexity in underwater environments, we develop a Stage-aware Attention Interaction strategy and an Adaptive Layer Execution strategy to eliminate redundant token interactions and unnecessary Transformer layers, achieving an improved accuracy-efficiency trade-off.
- Extensive experiments on multiple benchmark datasets demonstrate that our tracker achieves superior tracking performance while maintaining real-time inference speed.

2 Related Work

2.1 Lightweight Tracking

Classical trackers achieve efficient appearance matching through correlation filters (Bolme et al., 2010; Henriques et al., 2015), Siamese networks (Bertinetto et al., 2016; Li et al., 2018; 2019), and online model adaptation (Danelljan et al., 2019; Bhat et al., 2019). These advances have paved the way for Transformer-based trackers (Chen et al., 2021; Yan et al., 2021; Cui et al., 2022; Ye et al., 2022), which further enhance visual tracking through global context modeling and improved template-search interaction. However, their reliance on dense self-attention and multiple Transformer blocks introduces high computational and parameter costs, limiting deployment on resource-constrained platforms and in real-time scenarios.

To address this issue, existing lightweight Transformer trackers mainly improve computational efficiency from three aspects: attention simplification, token reduction, and adaptive layer execution. Attention simplification reduces the cost of self-attention through lightweight similarity computation or low-cost operations, as adopted by methods such as SMAT (Gopal & Amer, 2024) and FERMT (Zheng et al., 2024). Token reduction methods, including EViT (Liang et al., 2022) and DynamicViT (Rao et al., 2021), remove less informative tokens at different network depths to reduce the computational burden of subsequent attention operations. However, dynamic token selection may introduce irregular memory access and reduce the practical acceleration on hardware platforms. Adaptive layer execution skips unnecessary Transformer layers according to the current feature state.

Compared with token pruning, layer skipping preserves the token structure and is generally more hardware-friendly. Methods such as AVTrack (Li et al., 2024a) and SGLATrack (Xue et al., 2025) have demonstrated the effectiveness of adaptive layer selection for efficient tracking.

Other efficiency-oriented approaches include the mobile-friendly MobileViT backbone (Mehta & Rastegari, 2022), background-aware UAV tracking (Li et al., 2023a), LiteTrack with layer pruning and asynchronous template extraction (Wei et al., 2023a), and the convolutional LightFC tracker with a ViT-backed variant (Li et al., 2024b). Asymmetric Siamese streams (Zhu et al., 2025) and occlusion-robust UAV Transformers (Wu et al., 2025) explore further accuracy-efficiency trade-offs under different tracking conditions. These approaches motivate studying which interactions and layers are needed for each frame, without implying that their computational costs or deployment settings are directly comparable.

Building on these studies, our method combines stage-wise computation with dynamic layer execution. Shallow layers use stable fixed skipping to reduce redundant computation, whereas deep layers employ an attention-aware adaptive strategy to allocate computation based on the current feature state. This design better balances tracking accuracy and inference speed, making the tracker more suitable for challenging underwater environments and resource-constrained platforms.

2.2 Underwater Visual Tracking

Wavelength-dependent light attenuation, scattering, and other complex underwater environmental factors distort target appearance and background statistics. Marine targets may also deform non-rigidly and resemble coral, rocks, plants, or other animals, requiring trackers to learn robust target features while suppressing confusing backgrounds. Computational efficiency is also crucial because underwater edge devices face limited resources and real-time constraints. Although enhancement and restoration can improve visual quality, they may add latency. Several underwater-specific trackers have attempted to address these challenges. UOSTrack (Li et al., 2023b) combines hybrid training with motion modeling, while PUTrack (Zhang et al., 2025) introduces underwater-specific prompts for domain adaptation. However, both retain relatively high inference cost.

Lightweight Transformer trackers provide a promising solution by modeling global context and template-search relations with reduced computation. However, directly applying generic lightweight trackers to underwater scenes remains challenging, because degraded imaging conditions require both sufficient feature interaction and efficient computation allocation. Our W^2Track addresses this setting by using stage-aware Transformer interaction for shallow feature extraction and deep template-search matching, together with dynamic layer execution to skip redundant computation. It aims to balance robustness, accuracy, and real-time efficiency in challenging underwater environments.

3 Methodology

3.1 Overview

As illustrated in Figure 2, our tracker follows a one-stream Transformer framework (Ye et al., 2022; Cui et al., 2022), where the template and search images are first processed by a shared CNN block and converted into token sequences, which are then jointly fed into the Transformer encoder. To reduce redundant computation, we propose two complementary strategies from both attention interaction and network depth perspectives: (i) the Stage-aware Attention Interaction strategy. It adopts stage-specific token interaction patterns between template and search frames. (ii) the Adaptive Layer Execution strategy. It dynamically estimates the tracking complexity and information requirement of the current search frame based on intermediate token representations, determining whether subsequent Transformer layers should be executed. Finally, the resulting search features are fed into a center-based head for response map generation and bounding box prediction.

3.2 Stage-aware Attention Interaction

In visual object tracking, one-stream Transformer-based trackers jointly model template and search frames through attention (Ye et al., 2022; Cui et al., 2022). Repeatedly computing intra-frame and cross-frame interactions across encoder layers can incur substantial cost. This issue becomes more pronounced in underwater object tracking. Underwater biological targets often exhibit dense distributions and high inter-object appearance similarity, making search-to-template interactions prone to propagating ambiguous information from visually similar instances and background regions into template representations. Related lightweight tracking work has separated feature extraction from relation modeling to reduce repeated computation (Zheng et al., 2024). Moreover, as Transformer

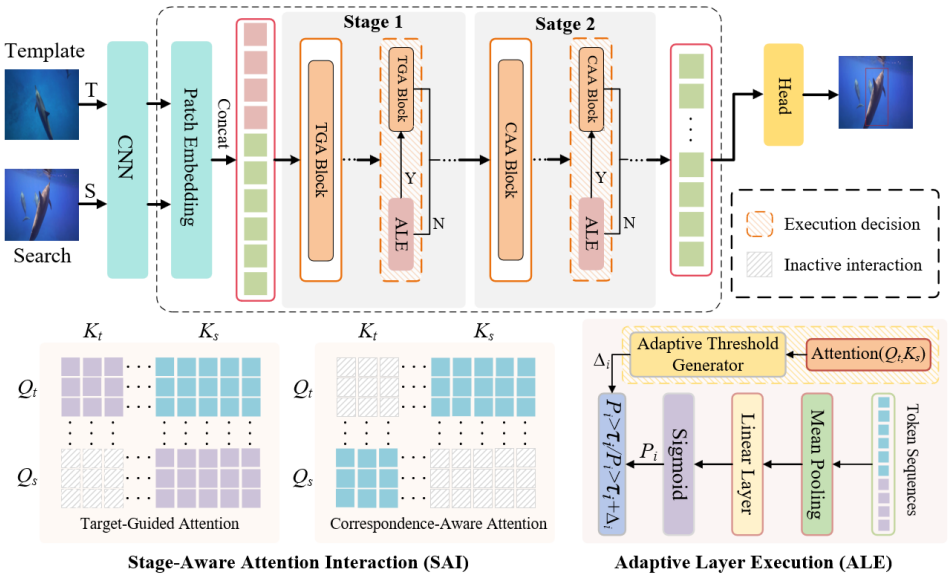

Figure 2: Overview of the proposed W^{2}Track. It introduces two key components: (i) Stage-aware Attention Interaction (SAI) with target-guided attention (TGA) and correspondence-aware attention (CAA) to alleviate ambiguous token interactions from visually similar underwater targets, and (ii) Adaptive Layer Execution (ALE) to dynamically skip redundant Transformer blocks based on scene complexity for a better accuracy-efficiency trade-off.

layers deepen, template and search representations gradually achieve sufficient intra-frame modeling, making further intra-frame interactions less beneficial while cross-frame correspondence modeling becomes increasingly important. Therefore, we propose a Stage-aware Attention Interaction strategy that adopts different token interaction patterns across Transformer stages according to their distinct computational requirements, including Target-guided Attention in shallow stages and Correspondence-aware Attention in deeper stages.

Vanilla Self-Attention. Existing one-stream Transformer trackers typically concatenate template and search tokens, then apply joint self-attention to model target-search relationships. Given the concatenated tokens $X = [X_t; X_s]$, the query, key, and value matrices follow the standard attention formulation (Vaswani et al., 2017):

$$Q = [Q_t; Q_s], \qquad K = [K_t; K_s], \qquad V = [V_t; V_s]. \tag{1}$$

Accordingly, the vanilla template-search self-attention can be formulated as follows:

$$\mathrm{Attn}(Q, K, V) = \mathrm{Softmax}\big(QK^\top\big)\, V = \mathrm{Softmax}\left(\begin{bmatrix} Q_t K_t^\top & Q_t K_s^\top \\ Q_s K_t^\top & Q_s K_s^\top \end{bmatrix}\right) \begin{bmatrix} V_t \\ V_s \end{bmatrix}, \tag{2}$$

where $Q_t K_t^\top$ and $Q_s K_s^\top$ capture intra-frame dependencies within template and search frames, while $Q_t K_s^\top$ and $Q_s K_t^\top$ establish cross-frame interactions between template and search frames. For simplicity, the scaling factor $1/\sqrt{d}$ is omitted in the attention formulation.

Target-guided Attention. We replace the vanilla self-attention with the proposed Target-guided Attention (TGA) in the shallow Stage 1 of the Transformer. Since shallow layers mainly focus on learning fundamental representations of the template and search branches, TGA preserves intra-frame interactions ($Q_t K_t^\top$ and $Q_s K_s^\top$) to capture their intrinsic visual characteristics. Meanwhile, considering that underwater biological targets often exhibit dense distributions and high inter-object appearance similarity, early cross-frame matching between search queries and template keys may introduce ambiguous responses from visually similar instances when feature representations are not yet sufficiently discriminative. Therefore, TGA removes the $Q_s K_t^\top$ interaction to prevent ambiguous information propagation from the search region to template representations. Meanwhile, the $Q_t K_s^\top$ interaction is retained, enabling the template features to perceive target-related information in the search region and providing a representational basis for subsequent cross-frame correspondence

modeling. Accordingly, the proposed TGA is formulated as:

$$\text{Attn}_{\text{TGA}} = \text{Softmax}\left(\begin{bmatrix} Q_t K_t^\top & Q_t K_s^\top \\ & Q_s K_s^\top \end{bmatrix}\right)\begin{bmatrix} V_t \\ V_s \end{bmatrix}. \tag{3}$$

Correspondence-aware Attention. In the deep Stage 2 of the Transformer encoder, we replace the vanilla self-attention with the proposed Correspondence-aware Attention (CAA). After sufficient intra-frame representation learning in the shallow stages, template and search features have already established discriminative visual characteristics. Therefore, redundant intra-frame interactions ($Q_t K_t^\top$ and $Q_s K_s^\top$) are removed, and attention computation is concentrated on template-search correspondence modeling. Specifically, $Q_t K_s^\top$ enables template queries to identify target-related regions in the search frame, while $Q_s K_t^\top$ allows search tokens to retrieve target identity information from the template representation. Their bidirectional interaction facilitates accurate target matching and localization. Accordingly, the proposed CAA is formulated as:

$$\text{Attn}_{\text{CAA}} = \text{Softmax}\left(\begin{bmatrix} & Q_t K_s^\top \\ Q_s K_t^\top & \end{bmatrix}\right)\begin{bmatrix} V_t \\ V_s \end{bmatrix}. \tag{4}$$

Computational Analysis. Let m_t and m_s denote the numbers of template and search tokens, respectively, and let N_1 and N_2 denote the numbers of Transformer blocks in Stage 1 and Stage 2. The vanilla self-attention across the entire encoder has a complexity of

$$\mathcal{O}\left((N_1 + N_2)\left(m_t^2 + 2m_t m_s + m_s^2\right)\right). \tag{5}$$

In contrast, SAI applies TGA in Stage 1 and CAA in Stage 2, resulting in an overall complexity of

$$\mathcal{O}\left(N_1\left(m_t^2 + m_t m_s + m_s^2\right) + 2N_2 m_t m_s\right). \tag{6}$$

Thus, SAI reduces the computational complexity to $\mathcal{O}\left(N_1 m_t m_s + N_2\left(m_t^2 + m_s^2\right)\right)$ by removing redundant token interactions while preserving stage-specific feature modeling.

3.3 Adaptive Layer Execution

Although Stage-aware Attention Interaction reduces redundant token interactions, existing Transformer trackers still execute all layers for every frame, regardless of tracking difficulty. Underwater sequences exhibit significant variations in target visibility, motion patterns, and imaging quality, leading to different levels of representation requirements across frames. Therefore, a fixed-depth Transformer may introduce unnecessary computation for simple cases while limiting computational flexibility in challenging scenarios. Our proposed Adaptive Layer Execution (ALE) strategy dynamically determines whether to execute subsequent Transformer layers based on intermediate token representations. Specifically, for the i-th candidate block, the input feature X_{i-1} is first aggregated through global average pooling to obtain a compact representation:

$$f_i = \text{MeanPool}(X_{i-1}). \tag{7}$$

Then, we predict the execution probability of the current block through a learnable linear projection followed by a sigmoid function σ:

$$P_i = \sigma(W_i f_i + b_i), \tag{8}$$

where W_i and b_i are learnable parameters. A larger P_i indicates a higher necessity of executing the current block, which usually corresponds to challenging underwater conditions requiring further feature refinement. The execution decision is determined by comparing P_i with a threshold τ_i:

$$d_i = \mathbb{I}(P_i > \tau_i), \tag{9}$$

where $d_i = 1$ indicates block execution and $d_i = 0$ denotes skipping. Since the binary decision is non-differentiable, we adopt a soft execution score during training:

$$s_i = \sigma\left(\gamma(P_i - \tau_i)\right), \tag{10}$$

where $\gamma = 12$ controls the approximation sharpness. A straight-through estimator is then applied:

$$\widetilde{d}_i = \text{stopgrad}(d_i - s_i) + s_i. \tag{11}$$

During the forward pass, this gate behaves exactly like the binary decision d_i. During backpropagation, it uses the gradient of s_i, allowing the tracking loss to update the routing module. The output of the candidate block is therefore

$$X_i = \widetilde{d}_i B_i(X_{i-1}) + (1 - \widetilde{d}_i) X_{i-1}. \tag{12}$$

where $B_i(\cdot)$ denotes the i-th Transformer block. Thus, executed samples use $B_i(X_{i-1})$, while skipped ones retain X_{i-1}.

Notably, different threshold strategies are adopted for Stage 1 and Stage 2. Stage 1 employs a fixed threshold τ_f because shallow layers mainly focus on fundamental representation learning, where a stable execution criterion is preferred. In contrast, Stage 2 introduces an adaptive threshold strategy. Since deeper layers have accumulated more discriminative target appearance and contextual information, the template-search cross-attention response provides a more reliable indicator of target correspondence and can be used to adaptively adjust the execution threshold for eliminating unnecessary computation. Specifically, as shown in Figure 2, we calculate the normalized entropy of the template-search cross-attention response $\mathcal{A}_{i-1} = \mathrm{Softmax}\left(Q_t K_s^\top\right)$. A higher entropy indicates a more dispersed attention response with uncertain target correspondence, while a lower entropy reflects a concentrated attention distribution with clearer target matching. Formally, the adaptive threshold in Stage 2 is calculated as:

$$\tau_i = \tau_f + \Delta_i = \tau_f + \alpha(1 - \mathrm{Mean}(-\frac{1}{\log N_k}\sum_k \mathcal{A}_{i-1,k}\log \mathcal{A}_{i-1,k})), \tag{13}$$

where N_k denotes the number of keys. α is a scaling factor set to 0.1, controlling the magnitude of threshold adjustment. In our implementation, ALE is applied to the later blocks of each Transformer stage, including blocks $i \in \{3, 4, 5\}$ in Stage 1 and $i \in \{8, 9, 10, 11\}$ in Stage 2, since deeper layers tend to exhibit feature saturation and provide diminishing refinement benefits.

3.4 Loss Function

During training, the bounding-box regression is supervised by the generalized intersection-over-union loss (Rezatofighi et al., 2019) and the L_1 loss, while the center heatmap is optimized using focal loss (Lin et al., 2017). Formally,

$$\mathcal{L}_{\text{track}} = \lambda_{\text{giou}}\mathcal{L}_{\text{giou}} + \lambda_1\mathcal{L}_1 + \lambda_{\text{focal}}L_{\text{focal}}, \tag{14}$$

where $\lambda_{\text{giou}} = 2$, $\lambda_1 = 5$, and $\lambda_{\text{focal}} = 1$, respectively. To prevent excessive execution or over-skipping during training, a regularization term $\mathcal{L}_{reg}$ is introduced to constrain the layer execution behavior and stabilize the adaptive routing process. For more details on the formulation of this regularization term, please refer to (Li et al., 2024a). The overall loss function is defined as:

$$\mathcal{L} = \mathcal{L}_{\text{track}} + \mathcal{L}_{\text{reg}}. \tag{15}$$

4 Experiments

4.1 Implementation Details

For fair comparison, our proposed approach and all competing trackers are trained using the same WebUOT (Zhang et al., 2024) and VMAT (Cai et al., 2023) training protocols. During training, we adopt standard data augmentation techniques, including random scaling, translation, and brightness jittering. The model is optimized with AdamW (Loshchilov & Hutter, 2019) using an initial learning rate of 4×10^{-4} and a weight decay of 1×10^{-4}. All experiments are conducted on a single RTX 5090 GPU with an Intel Core i7-14650HX processor. The model is trained for 120 epochs with a batch size of 32, and the learning rate is decayed by a factor of 10 after the 100th epoch. We adopt ViT-Tiny (Touvron et al., 2021), a lightweight pretrained Vision Transformer, as the backbone. Stage 1 and Stage 2 each consist of six Transformer blocks. The template and search regions are resized to 128×128 and 256×256, respectively. For Adaptive Layer Execution, the initial execution threshold τ_f is set to 0.65. During inference, we follow the common practice (Ye et al., 2022) and employ the Hamming window to incorporate positional priors.

4.2 State-of-the-Art Comparisons

To evaluate the effectiveness of our approach, we compare it with representative real-time and non-real-time trackers on three widely used underwater tracking benchmarks: UTB180 (Alawode et al., 2022), UOT100 (Panetta et al., 2021; Kezebou et al., 2019), and VMAT (Cai et al., 2023). We report AUC, Precision, PNorm, GPU/CPU inference speed, GMACs, and the number of parameters to jointly evaluate accuracy, efficiency, and model complexity. Notably, trackers exceeding 20 GMACs are categorized as non-real-time methods.

UTB180. UTB180 is a high-quality underwater object tracking benchmark containing 180 underwater video sequences and more than 58,000 manually annotated frames. This dataset covers typical

	Methods	Source	UTB180			UOT100			VMAT			Speed (FPS)		Complexity	
			AUC	Precision	P_{Norm}	AUC	Precision	P_{Norm}	AUC	Precision	P_{Norm}	GPU	CPU	GMACs	Params
Real-time	W^{2}Track	Ours	61.35	54.01	71.79	61.64	58.04	77.08	46.06	57.30	55.06	210.52	27.09	1.553	4.870
	FARTrack (Wang et al., 2026)	ICLR26	49.57	41.73	57.32	52.99	50.19	68.69	38.36	46.55	46.56	207.50	24.02	1.694	6.821
	ORTrack-deit (Wu et al., 2025)	CVPR25	53.64	45.70	61.94	52.96	48.09	66.13	36.30	42.65	42.92	191.30	40.12	1.538	5.310
	SGLATrack (Xue et al., 2025)	CVPR25	57.03	50.56	66.22	57.79	55.37	73.77	41.75	50.13	50.47	264.23	25.26	1.680	5.807
	AsymTrack (Zhu et al., 2025)	AAAI25	50.09	39.73	55.85	59.98	54.85	72.10	41.31	47.88	45.96	202.72	43.63	0.807	3.363
	AVTrack-deit (Li et al., 2024a)	ICML24	52.25	44.15	60.10	47.04	39.46	56.35	26.64	27.02	28.86	229.21	36.21	1.964	6.648
	SMAT (Gopal & Amer, 2024)	WACV24	37.24	24.13	41.55	43.54	31.32	48.36	20.52	20.31	21.46	200.91	–	1.870	7.300
	LightFC-ViT (Li et al., 2024b)	KBS24	48.87	39.52	57.50	55.54	47.50	66.20	28.34	31.13	32.77	168.13	–	0.950	3.165
	Aba-vitrack (Li et al., 2023a)	ICCV23	48.43	38.20	56.31	49.93	42.91	62.63	28.74	31.37	33.78	151.50	24.60	2.396	7.979
	HiT-small (Cao et al., 2023)	ICCV23	45.29	34.49	49.66	53.33	43.98	62.24	32.90	34.45	35.99	191.59	–	1.131	11.025
Non-real-time	UTPTrack (Wu et al., 2026)	CVPR26	47.12	37.31	54.25	50.30	42.34	61.16	27.55	31.35	32.18	86.91	8.05	23.860	92.121
	PUTrack (Zhang et al., 2025)	TII25	63.56	58.61	73.55	67.38	66.52	84.35	53.37	65.07	62.07	60.15	4.76	21.563	92.063
	LMTrack (Xu et al., 2025)	AAAI25	62.19	58.89	71.17	56.50	57.87	73.27	44.67	55.10	55.94	107.82	4.07	29.970	92.121
	MCITrack (Kang et al., 2025)	AAAI25	61.12	60.66	69.07	63.52	63.07	78.04	53.54	66.91	63.67	68.73	5.14	38.460	88.029
	AQATrack (Xie et al., 2024)	CVPR24	49.21	41.86	56.09	50.89	45.10	63.03	32.12	36.86	37.98	133.59	7.48	25.831	71.980
	EVPTrack (Shi et al., 2024)	AAAI24	59.15	57.45	67.04	60.98	62.28	78.26	47.87	59.59	57.59	106.55	8.69	21.778	73.697
	UOSTrack (Li et al., 2023b)	Ocean Eng23	57.57	49.28	67.03	65.47	61.97	78.97	37.92	42.64	42.80	96.56	8.16	21.517	92.493
	SeqTrack (Chen et al., 2023)	CVPR23	57.41	54.90	65.24	61.12	61.59	77.85	49.66	60.51	58.20	77.14	4.51	65.862	89.111
	ARTrack (Wei et al., 2023b)	CVPR23	40.27	31.13	44.63	47.04	39.46	56.35	26.64	27.02	28.86	54.08	5.21	36.990	173.105
	OSTrack (Ye et al., 2022)	ECCV22	61.87	59.31	71.29	61.94	62.20	78.72	49.78	61.41	59.61	71.29	8.57	65.363	92.121

Table 1: Comparison with state-of-the-art trackers on the UTB180, UOT100 and VMAT datasets.

challenges such as low visibility, contrast degradation, non-uniform illumination, suspended particle interference, and underwater imaging blur, making it suitable for evaluating tracker robustness in real underwater scenarios. W^{2}Track achieves 61.35% AUC, 54.01% Precision, and 71.79% PNorm, obtaining the best results among the compared real-time trackers. It surpasses the runner-up by 4.32, 3.45, and 5.57 percentage points in AUC, Precision, and PNorm, respectively, showing stronger robustness in challenging underwater scenes. Compared with underwater-specific trackers, our method achieves competitive performance with PUTrack and significantly surpasses UOSTrack.

UOT100. UOT100 contains 104 underwater video sequences and more than 74,000 annotated frames from natural and artificial underwater scenes. It also covers diverse underwater degradation conditions, including color attenuation, low contrast, underexposure, scattering, and refraction. W^{2}Track achieves 61.64% AUC, 58.04% Precision, and 77.08% PNorm, ranking first among real-time trackers. Its three metrics improve by 1.66, 2.67, and 3.31 percentage points, respectively, compared with the current best-performing results.

VMAT. VMAT is a real-world marine-animal tracking dataset with non-rigid deformation, irregular motion, complex backgrounds, camouflage, illumination variation, water scattering, and camera motion. W^{2}Track achieves 46.06% AUC, 57.30% Precision, and 55.06% PNorm, obtaining the best performance among real-time trackers. Compared with SGLATrack, it improves the three metrics by 4.31, 7.17, and 4.59 percentage points. We observe that all trackers exhibit degraded performance on the VMAT dataset compared with UTB180 and UOT100, indicating the increased difficulty of tracking real-world marine animals.

Speed and Complexity. W^{2}Track runs at 210.52 FPS on GPU and 27.09 FPS on CPU, with only 1.553 GMACs and 4.870M parameters. Although SGLATrack and AVTrack-DeiT achieve higher GPU inference speeds, W^{2}Track consistently delivers superior accuracy across all three benchmarks while requiring fewer GMACs and parameters. Moreover, our method achieves competitive performance against non-real-time trackers while providing substantially higher efficiency. These results show that W^{2}Track achieves a favorable balance among accuracy, speed, and complexity for practical underwater tracking. Overall, W^{2}Track achieves a favorable balance of accuracy, speed, and complexity for practical underwater tracking.

4.3 Ablation Study and Analysis

Impact of Key Components.

Our tracker introduces two key components: the Stage-aware Attention Interaction strategy and the Adaptive Layer Execution strategy, which reduce computational redundancy from the attention interaction and network depth perspectives, respectively. To evaluate their individual contributions, we remove each component separately, and the results are reported in Table 2. SAI improves AUC from 55.43% to 60.93% while reducing GMACs from 2.95 to 2.78. Further introducing ALE improves the performance to 61.35% AUC with only 1.55 GMACs and

Table 2: Ablation for important components: SAI and ALE on the UTB180 dataset.

SAI	ALE	AUC	FPS	GMACs
✗	✗	55.43	188.29	2.95
✓	✗	60.93	198.72	2.78
✗	✓	60.88	201.08	1.98
✓	✓	61.35	210.52	1.55

210.52 FPS, demonstrating the complementary contributions of the two components to accuracy-efficiency optimization.

Beyond the overall improvement, we further examine whether SAI and ALE adapt to underwater tracking characteristics by analyzing target density and scene difficulty, respectively. As shown in Table 3, SAI achieves larger gains on dense-target subsets than on non-dense ones, with AUC improvements of 2.354 and 1.195, respectively. This indicates that reducing redundant search-to-template interactions is particularly effective for densely distributed and visually similar underwater targets. Moreover, the improvements on non-dense subsets indicate that reducing redundant token interactions remains effective in sparse scenarios, where discriminative target information is concentrated within target regions and excessive interactions may introduce irrelevant background cues. Meanwhile, Table 4 shows that ALE dynamically adjusts computation according to tracking difficulty. The layer-skipping ratio increases from 40.25% on hard sequences to 50.33% on easy ones, suggesting that ALE retains deeper feature refinement for challenging scenarios while avoiding unnecessary computation in simpler cases.

Table 3: Performance gain of SAI under different target densities on the UTB180 dataset.

Density	Metric	SAI	Vanilla Attn	Gain
Dense	AUC	**65.102**	62.748	+2.354
	Precision	**46.207**	44.163	+2.044
Sparse	AUC	**58.179**	56.984	+1.195
	Precision	**69.988**	69.070	+0.918

Table 4: Relationship between sample difficulty and adaptive layer execution on the UOT100 dataset.

Level	AUC	Avg. Skipped Layers	Skip Ratio
Hard	28.61	2.817	40.25%
Medium	70.39	3.165	45.13%
Easy	86.46	3.523	50.33%

Impact of Attention Interaction Patterns.

To validate the stage-specific interaction design, we evaluate different attention variant combinations for Stage 1 and Stage 2. Each variant is represented by a four-bit binary mask. Specifically, the four bits in each mask correspond to $[Q_tK_t^\top, Q_tK_s^\top; Q_sK_t^\top, Q_sK_s^\top]$, respectively. Accordingly, the five attention settings $\mathcal{S}_1$, $\mathcal{S}_2$, $\mathcal{S}_3$, $\mathcal{S}_4$, and $\mathcal{S}_5$ are defined by the masks 1001, 1011, 1111, 1101, and 0110, respectively. As shown in Table 5, different attention settings yield different accuracy-efficiency trade-offs. Compared with vanilla self-attention ($\mathcal{S}_3$, $\mathcal{S}_3$), the proposed combination ($\mathcal{S}_4$, $\mathcal{S}_5$) improves AUC from 60.88% to 61.35% while reducing GMACs from 1.98 to 1.55. Specifically, $\mathcal{S}_4$ validates removing search-to-template interaction in shallow layers, while $\mathcal{S}_5$ shows that deep layers should focus on template-search correspondence. These results verify the necessity of stage-specific token interaction designs.

Table 5: Comparison of different attention interaction pattern combinations on the UTB180 dataset.

Method	AUC	Precision	GMACs
$\mathcal{S}_3$, $\mathcal{S}_3$	60.88	53.78	1.98
$\mathcal{S}_1$, $\mathcal{S}_5$	58.77	49.94	1.48
$\mathcal{S}_2$, $\mathcal{S}_5$	60.30	53.23	1.59
$\mathcal{S}_3$, $\mathcal{S}_5$	60.31	52.79	1.79
$\mathcal{S}_4$, $\mathcal{S}_3$	60.69	54.13	1.66
$\boldsymbol{\mathcal{S}_4}$, $\boldsymbol{\mathcal{S}_5}$	**61.35**	**54.01**	**1.55**

Table 6: Comparison of different Adaptive Layer Execution placements on the UTB180 dataset.

Stage 1	Stage 2	AUC	GMACs	Stage 1	Stage 2	AUC	GMACs	Stage 1	Stage 2	AUC	GMACs
None	None	60.93	2.78	2,3,4,5	9,10,11	57.67	1.66	3,4,5	6,7,8,9	60.08	1.70
3,4,5	None	54.56	2.13	2,3,4	8,9,10	57.09	1.76	0,1,2	8,9,10,11	59.30	1.75
None	8,9,10,11	58.89	1.94	2,3,4,5	8,9,10	60.69	1.69	**3,4,5**	**8,9,10,11**	**61.35**	**1.55**

Influence of ALE Location. To investigate the impact of ALE placement, we evaluate different layer execution configurations across Stage 1 and Stage 2. (i) As shown in the first group of Table 6, introducing ALE in both stages reduces computational cost while maintaining competitive accuracy. (ii) We further test alternative layer placements to examine the sensitivity of ALE to different execution locations. Although some configurations achieve lower computational costs, they yield inferior performance, indicating that improper layer selection may hinder feature refinement. (iii) We finally examine whether ALE is more suitable for later Transformer layers. The rightmost group of Table 6 shows that applying ALE to shallow blocks degrades performance, whereas applying it to later blocks in both stages achieves a better accuracy-efficiency trade-off. These results suggest that ALE is more effective in deeper layers, where feature representations are sufficiently established and further refinement provides diminishing returns.

Impact of Threshold. The threshold controls the execution criterion of ALE for deciding whether a Transformer block is executed. A higher threshold imposes a stricter execution requirement, resulting

Table 7: Ablation study of different fixed threshold values on the UTB180 dataset.

Threshold	AUC	FPS	GMACs
0.75	56.91	216.92	1.49
0.70	59.39	211.43	1.54
0.65	**61.35**	**210.52**	**1.55**
0.60	60.51	200.92	1.61

Table 8: Ablation study of adaptive threshold placement across Transformer stages on the UTB180 dataset.

#	Stage 1	Stage 2	AUC	FPS	GMACs
1	✗	✗	59.49	204.05	1.61
2	✓	✗	60.46	208.75	1.63
3	✓	✓	60.10	216.39	1.61
4	✗	✓	**61.35**	**210.52**	**1.55**

in more skipped blocks and lower computational cost, while a lower threshold encourages more block execution for feature refinement. As shown in Table 7, $\tau_f = 0.65$ achieves the best trade-off, obtaining 61.35% AUC with 1.55 GMACs and 210.52 FPS.

To further verify the proposed adaptive-threshold strategy, we evaluate combinations of fixed and adaptive thresholds in Stages 1 and 2. As shown in Table 8, applying the adaptive threshold only in Stage 1 slightly improves the baseline, whereas introducing it in Stage 2 yields a larger gain. These results suggest that adaptive thresholds are more beneficial in deeper Transformer stages, where attention responses provide more reliable feedback for execution decisions. Accordingly, we use a fixed threshold in Stage 1 and an adaptive threshold in Stage 2.

4.4 Qualitative Comparison and Visualization

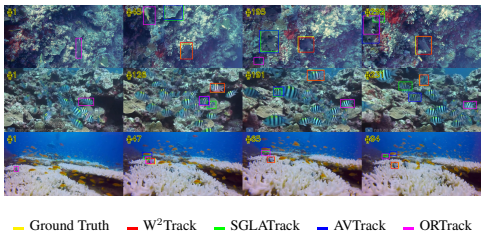

Figure 3: Qualitative comparison on representative sequences.

Qualitative Results. To intuitively demonstrate the effectiveness of our approach, we present qualitative comparisons with representative trackers in Figure 3. The selected sequences cover three challenging underwater scenarios: similar background interference, dense and visually similar targets, and severe color distortion. Under these conditions, our tracker consistently maintains robust and accurate target localization and outperforms the compared methods. These qualitative results further demonstrate the effectiveness and robustness of our underwater-oriented design in complex underwater environments.

Visualization of Attention. Figure 4 further visualizes the attention responses of W^{2}Track with and without SAI and ALE on representative underwater sequences. Without SAI, vanilla self-attention produces scattered responses over both targets and backgrounds due to exhaustive token interactions, introducing ambiguous cues. In contrast, SAI suppresses irrelevant information propagation and generates more concentrated target responses. Similarly, ALE dynamically skips redundant Transformer blocks, preserving discriminative target representations while reducing unnecessary feature refinement, thereby improving localization reliability with lower computational cost.

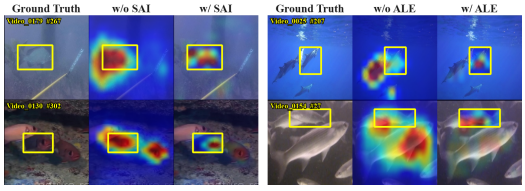

Figure 4: Attention map visualization on representative sequences.

5 Conclusion

This paper addresses two major challenges: complex underwater environments and limited computational resources on edge platforms. We propose W^{2}Track, an efficient Transformer tracker for underwater visual tracking, which combines Stage-aware Attention Interaction for stage-specific template–search modeling with Adaptive Layer Execution for input-dependent computation. The former allocates template–search interactions according to network stages, while the latter adaptively skips redundant blocks based on tracking difficulty. Experiments on UTB180, UOT100, and VMAT demonstrate competitive tracking accuracy with real-time efficiency, achieving 210.52 FPS and 1.55 GMACs on UTB180. These results validate stage-aware interaction and adaptive depth allocation for efficient underwater tracking.

AI USE STATEMENT

In this work, we used generative Al tools for language polishing and improving the readability of the manuscript.

REFERENCES

Basit Alawode, Yuhang Guo, Mehnaz Ummar, Naoufel Werghi, Jorge Dias, Ajmal Mian, and Sajid Javed. Utb180: A high-quality benchmark for underwater tracking. In *Proceedings of the Asian Conference on Computer Vision*, pp. 3326–3342, 2022.

Luca Bertinetto, Jack Valmadre, Joao F. Henriques, Andrea Vedaldi, and Philip H. S. Torr. Fully-convolutional siamese networks for object tracking. In *European Conference on Computer Vision Workshops*, 2016.

Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019.

David S. Bolme, J. Ross Beveridge, Bruce A. Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

Levi Cai, Nathan E. McGuire, Roger Hanlon, T. Aran Mooney, and Yogesh Girdhar. Semi-supervised visual tracking of marine animals using autonomous underwater vehicles. *arXiv preprint arXiv:2302.07344*, 2023. URL https://arxiv.org/abs/2302.07344.

Ziang Cao, Changhong Fu, Junjie Ye, Bowen Li, and Yiming Li. Hit: Hierarchical transformer with mixed attention for visual tracking. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023. URL https://arxiv.org/abs/2308.06904.

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*, 2021.

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*, 2023.

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*, 2022.

Martin Danelljan, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Atom: Accurate tracking by overlap maximization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

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*, pp. 6708–6717, 2024.

Joao F. Henriques, Pedro Caseiro, Rui Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2015.

Ben Kang, Xin Chen, Simiao Lai, Yang Liu, Yi Liu, and Dong Wang. Exploring enhanced contextual information for video-level object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

L. Kezebou, V. Oludare, K. Panetta, and S. S. Agaian. Underwater object tracking benchmark and dataset. In *2019 IEEE International Symposium on Technologies for Homeland Security*, pp. 1–6, Woburn, MA, USA, 2019. doi: 10.1109/HST47167.2019.9032954.

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*, 2018.

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*, 2019.

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*, pp. 13989–14000, 2023a.

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 *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 28403–28420, 2024a.

Yunfeng Li, Bo Wang, Ye Li, Zhuoyan Liu, Wei Huo, Yueming Li, and Jian Cao. Underwater object tracker: UOSTrack for marine organism grasping of underwater vehicles. *Ocean Engineering*, 285: 115449, 2023b.

Yunfeng Li, Bo Wang, Xueyi Wu, Zhuoyan Liu, and Ye Li. Lightweight full-convolutional siamese tracker. *Knowledge-Based Systems*, 286:111439, 2024b. doi: 10.1016/j.knosys.2024.111439.

Youcheng Liang, Chunjiang Ge, Zhan Tong, Yibing Song, Jue Wang, and Pengtao Xie. Not all patches are needed: Expediting vision transformers via token reorganizations. In *International Conference on Learning Representations*, 2022.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2019.

Sachin Mehta and Mohammad Rastegari. Mobilevit: Light-weight, general-purpose, and mobile-friendly vision transformer. In *International Conference on Learning Representations*, 2022.

K. Panetta, L. Kezebou, V. Oludare, and S. S. Agaian. Comprehensive underwater object tracking benchmark dataset and underwater image enhancement with gan. *IEEE Journal of Oceanic Engineering*, June 2021. doi: 10.1109/JOE.2021.3086907. URL https://ieeexplore.ieee.org/document/9499961.

Yongming Rao, Wenbo Zhao, Zhengzhong Liu, et al. Dynamicvit: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems*, 2021.

Seyed Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

Liangtao Shi, Bineng Zhong, Qihua Liang, Ning Li, Shengping Zhang, and Xianxian Li. Explicit visual prompts for visual object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers and distillation through attention. In *International Conference on Machine Learning*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, et al. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017.

Guijie Wang, Tong Lin, Yifan Bai, Anjia Cao, Shiyi Liang, Wangbo Zhao, and Xing Wei. FARTrack: Fast autoregressive visual tracking with high performance. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=lq7Zfr8kAS.

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 *arXiv preprint arXiv:2309.09249*, 2023a. URL https://arxiv.org/abs/2309.09249.

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*, pp. 9697–9706, 2023b.

Hao Wu, Xudong Wang, Jialiang Zhang, Junlong Tong, Xinghao Chen, Junyan Lin, Yunpu Ma, and Xiaoyu Shen. Utptrack: Towards simple and unified token pruning for visual tracking, 2026. URL https://arxiv.org/abs/2602.23734.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

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*, pp. 19300–19309, 2024.

Chenlong Xu, Bineng Zhong, Qihua Liang, Yaozong Zheng, Guorong Li, and Shuxiang Song. Less is more: Token context-aware learning for object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. URL https://arxiv.org/abs/2501.00758.

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 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6730–6740, 2025.

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*, 2021.

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*, 2022.

Hongxu Yin, Arash Vahdat, Jose M. Alvarez, Arun Mallya, Jan Kautz, and Pavlo Molchanov. A-vit: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Chunhui Zhang, Li Liu, Guanjie Huang, Hao Wen, Xi Zhou, and Yanfeng Wang. Webuot-1m: Advancing deep underwater object tracking with a million-scale benchmark. *arXiv preprint arXiv:2405.19818*, 2024. URL https://arxiv.org/abs/2405.19818.

Qiuyang Zhang, Wei Song, Cong Liu, and Minghua Zhang. PUTrack: Improved underwater object tracking via progressive prompting. *IEEE Transactions on Industrial Informatics*, 21(5):3986–3995, 2025.

Jikai Zheng, Mingjiang Liang, Shaoli Huang, and Jifeng Ning. Feature extraction and relation modeling transformer for visual tracking. In *European Conference on Computer Vision*, 2024.

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*, 2025.