Available Online: 24 November 2026 DOI: 10.54254/2977-3903/2026.39361

Adaptive visual tracking method for UAV real-time tracking task

*Yanglong Guo, Zhiyong Zhang**

School of Artificial Intelligence, Changchun Normal University, Changchun, China

*Corresponding Author. Email: 1426994702@qq.com

Abstract. Visual object tracking for Unmanned Aerial Vehicle (UAV) applications requires an effective balance between tracking accuracy and real-time inference speed under limited computational resources. Although Vision Transformer (ViT)-based trackers achieve strong representation capability through global feature modeling, they usually perform full multi-layer computation on all tokens, resulting in considerable computational redundancy. To address this issue, this paper proposes a dual-dimensional adaptive visual tracking framework that jointly exploits token-level and layer-level redundancy. The proposed framework retains the background-aware token halting mechanism in the shallow Transformer stages to suppress redundant background tokens, while introducing a similarity-guided layer adaptation strategy in the deep stages to dynamically select representative Transformer blocks, thereby reducing redundant layer computation. To overcome the inconsistency between dynamic layer-skipping paths and the original full-layer distribution prior, a shallow distribution prior is proposed to constrain token halting only during the continuous execution stage of shallow layers, preserving stable token evolution while remaining compatible with dynamic layer adaptation. The experimental results demonstrate that the proposed dual-adaptive framework substantially improves inference efficiency while maintaining competitive tracking accuracy. Compared with the baseline background-adaptive tracker, the final model achieved 95.63 ± 4.38 FPS, while the baseline achieved 70.98 ± 2.12 FPS, corresponding to a 34.73% increase in mean inference speed, while maintaining comparable tracking performance with only a marginal reduction in AUC. These results demonstrate that jointly modeling token-level and layer-level redundancy provides an effective solution for achieving a better trade-off between tracking accuracy and computational efficiency in real-time UAV visual tracking.

Keywords: Visual object tracking, Dynamic computation, Layer adaptation, Token halting, Similarity-guided learning, Distribution prior

1. Introduction

UAV visual tracking refers to the process in which an unmanned aerial vehicle captures aerial video sequences using an onboard visual camera. Given the target state specified in the initial frame of the input image sequence, the algorithm infers and predicts the position of the target in subsequent aerial frames [1]. Real-time visual tracking refers to the continuous monitoring and tracking of a target at a speed of no less than 30 frames

Copyright: © 2026 by the authors. Submitted for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

per second. Therefore, visual learning algorithms for real-time UAV tracking are required to achieve a balance between accuracy and speed on UAV platforms with limited computational resources.

The technological development of real-time tracking has evolved through three stages: the era of Discriminative Correlation Filters (DCFs), the era of lightweight Convolutional Neural Networks (CNNs), and the era of Vision Transformers (ViTs) [2-4]. In recent years, Transformers have been widely applied to visual object tracking tasks due to their strong capability for global relation modeling, which enables them to effectively capture long-range dependencies between the template and search regions [5-8]. Trackers based on the one-stream architecture typically concatenate template tokens and search tokens and feed them jointly into a Transformer backbone, enabling deep interaction between the target template and the search region [9-12]. However, such methods usually require full multi-layer Transformer computation over all tokens, resulting in high computational cost [10-13]. This makes it particularly difficult to meet the requirements of real-time performance and low power consumption in UAV scenarios [1, 13]. Therefore, it is crucial to design a lightweight adaptive visual tracking method for real-time UAV tracking [14, 15]. Motivated by this, this paper proposes a dual-dimensional adaptive visual tracking framework. The main contributions of this paper are summarized as follows:

• Propose a dual-dimensional adaptive tracking framework based on both tokens and layers. Building upon the background-adaptive mechanism, a similarity-guided layer-adaptive mechanism is introduced to simultaneously suppress redundant background tokens and redundant deep layers.

• We propose a mask-aware similarity-guided layer adaptation mechanism, which introduces the active mask generated by background-adaptive token halting into layer-selection supervision, enabling deep layer selection to focus more effectively on still-valid search tokens.

• To address the mismatch between dynamic layer-skipping paths and the prior distribution of the original full-layer architecture, we design a shallow distribution prior that constrains the token-halting distribution only during the continuous execution stage of shallow layers. This design preserves the stability of prior regularization while avoiding conflicts with dynamic layer selection.

• The effectiveness of the proposed mechanisms is validated through three groups of core experiments: a single-dimensional background-adaptive mechanism, a dual-adaptive mechanism without distribution prior, and a dual-adaptive mechanism with the shallow distribution prior retained. Experimental results demonstrate that the model equipped with the adaptive similarity layer mechanism maintains comparable accuracy while achieving further acceleration compared with the original scheme.

2. Related work

2.1. Efficient visual transformer

The Transformer architecture has gradually become an important alternative to traditional convolutional neural networks in deep learning, especially in the field of computer vision, due to its efficient self-attention mechanism [5-7]. In 2020, the Google research team first proposed the ViT, which was the first model built entirely upon the Transformer architecture and surpassed convolutional neural networks on mainstream visual tasks [8, 16, 17].

The main workflow of the ViT-based tracking model is as follows. In the initial stage, the target region in the first frame of a video sequence is manually or automatically annotated and used as the target template input to the model [11, 12]. The ViT model first performs initial feature extraction and memorization of the target. During the subsequent online tracking stage, for each incoming frame, a larger search region centered around the target position in the previous frame is cropped to continuously localize the target and output its

updated position [10-12]. Nowadays, ViT models and their variants have gradually become core components of modern vision foundation models and have been widely applied to nearly all computer vision tasks, including image classification, object detection, and many other visual understanding applications [18-21].

2.2. Visual object tracking based on transformer

With the widespread application of the Transformer architecture in visual tasks, trackers based on the ViT have demonstrated strong global modeling capabilities in complex scenarios [10-12]. However, Transformer-based models usually require multi-layer self-attention computation over a large number of template tokens and search tokens, resulting in high computational complexity [22, 23].

To address this issue, one line of research focuses on token-level optimization [13-15, 24]. For example, Li et al. [25] proposed a background-adaptive mechanism that reduces unnecessary computation by identifying redundant background tokens. In this mechanism, tokens corresponding to background regions can terminate computation at shallow layers, while target-related tokens continue to participate in subsequent feature interactions [15, 25]. This approach effectively reduces the computational redundancy of background tokens and constrains token-halting behavior through ponder loss and distribution prior loss, thereby improving the balance between tracking speed and accuracy to some extent [26].

However, the background-adaptive mechanism still has a limitation: it mainly focuses on token-level redundancy and does not explicitly address hierarchical redundancy among deep Transformer blocks [25]. In practical tracking scenarios, feature representations may become increasingly stable after passing through several shallow Transformer blocks, leading to strong similarity among subsequent deep blocks [27, 28]. If all layers are still executed sequentially with a fixed depth, additional computational overhead will be introduced [13, 29-31].

Another line of research addresses redundancy from the layer dimension [30-33]. For example, Xue et al. [34] proposed a similarity-guided layer-adaptive mechanism, which dynamically selects a representative deep block for execution by measuring feature similarity between the saturated layer and candidate subsequent layers, thereby skipping other redundant layers. However, similarity-guided mechanisms usually compute similarity based on overall token representations, which can easily be affected by a large number of background tokens in visual tracking tasks. Since the background region often occupies most of the search area, conventional similarity measures tend to reflect the consistency of background regions rather than the effective representation of target-related regions [25].

To address the above issues, this paper proposes a dual-dimensional adaptive visual tracking framework built upon the background-adaptive mechanism. In the shallow stages, the framework preserves background-aware token halting, while in the deep stages, a similarity-guided layer-adaptive mechanism is introduced, enabling the model to dynamically select representative layers from candidate deep Transformer blocks. In addition, the proposed framework fully utilizes the token-halting results obtained in the shallow stages and constructs a mask-aware similarity target, where inter-layer similarity is computed only on the still-active search tokens. This design effectively reduces the interference of halted background tokens on layer-selection supervision.

3. Method

3.1. Overall framework

The method in this paper is based on the background adaptive model Aba-ViTrack, and adopts the tracking framework of one-stream Transformer. Given template image z and search image x, the model first divides

them into patch tokens, which are spliced to form a complete token sequence, as shown in Eq. (1).

$$T = [t_{cls}, T_z, T_x] \tag{1}$$

t_{cls} denotes the global CLS token, T_z denotes the template tokens, and T_x denotes the search tokens. In this article, the template size is set to 128×128, the search size is 256×256, and the patch size is 16, as shown in Eqs. (2)–(4).

$$| T_z | = 8 \times 8 = 64 \tag{2}$$

$$| T_x | = 16 \times 16 = 256 \tag{3}$$

$$| T | = 1 + 64 + 256 = 321 \tag{4}$$

The model contains 12 Transformer blocks. In this article, it is divided into two stages, as shown in Eqs. (5) and (6).

$$B_{shallow} = \{0, 1, 2, 3, 4, 5\} \tag{5}$$

$$B_{deep} = \{6, 7, 8, 9, 10, 11\} \tag{6}$$

The shallow stages implement the background-aware token halting mechanism, while the deep stages adopt the similarity-guided layer adaptive mechanism to select a representative layer from candidate deep blocks for subsequent computation. The overall process is as shown in Eq. (7).

$$T^0 \to B_0 \to B_1 \to \cdots \to B_5 \to LayerSelector \to B_k, k \in \{6, 7, 8, 9, 10, 11\} \tag{7}$$

T^0 denotes the initial input tokens of the network, and B_k refers to the candidate layer dynamically selected by the similarity-guided layer selection module. An overview of the model is illustrated in Figure 1.

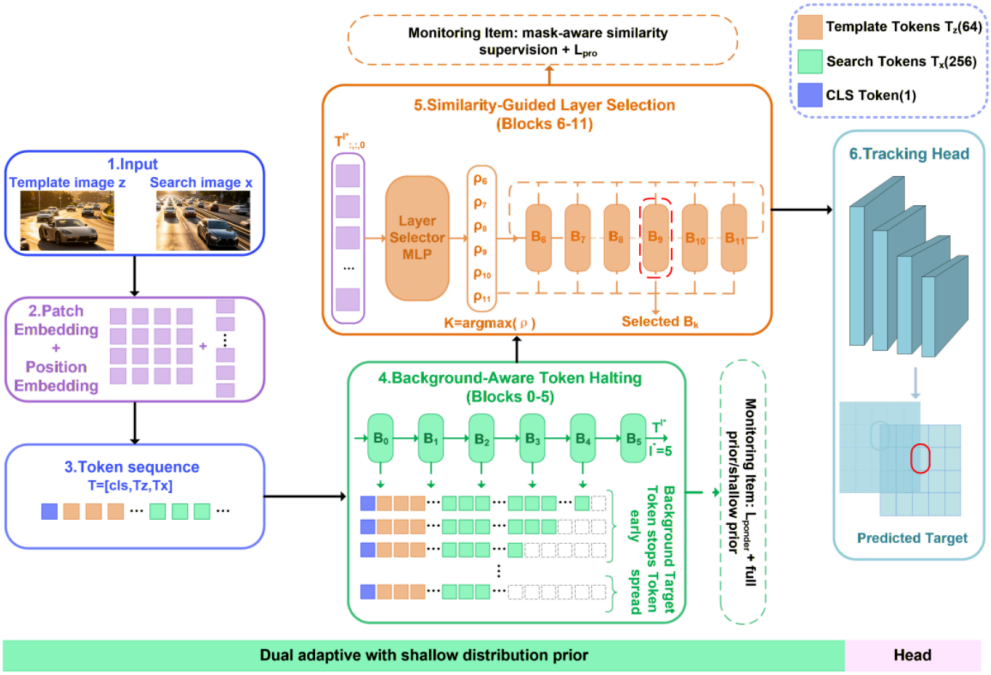

Figure 1. Overall framework of the dual-adaptive tracker

3.2. Background-aware token halting

In the shallow stages, the model performs a background-aware token-halting mechanism. After the backbone vision model receives the input sample, parameter initialization becomes an essential step in the workflow. The algorithm initializes the intermediate states of the model, the accumulated halting probabilities, and related variables to zero. In addition, a maximum number of iterations is predefined to prevent the model from falling into infinite computation.

After initialization, the model enters the iterative forward-propagation process. At each computation step, the model produces an intermediate output together with a halting probability p_t^n. The current halting probability is accumulated into the cumulative halting probability P_t^N. If the cumulative probability exceeds a predefined threshold $(1-\epsilon)$, where ϵ is a very small constant typically set to 10^{-3}, or if the maximum number of iterations is reached, the iterative computation is terminated, thereby enabling adaptive allocation of computational resources. Otherwise, if the cumulative probability does not exceed the threshold and the maximum iteration limit has not been reached, the model proceeds to the next computation step. If the maximum number of iterations is reached before the threshold condition is satisfied, the computation is forcibly terminated, and the remaining probability mass is assigned to the final iteration. After all iterative steps are completed, the model performs a weighted fusion of the intermediate feature outputs according to the halting probabilities of each layer, producing the final prediction result for the input sample.

For the token representation T^l at the l-th layer, the model predicts a halting score for each token as shown in Eq. (8).

$$h_i^l = H(T_i^l) \tag{8}$$

$H(\cdot)$ denotes the halting score predictor and i represents the token index.

The model determines whether a token should continue participating in subsequent layer computations by accumulating the halting scores. For background-region tokens, which contribute less to target localization, the model applies background-weight constraints to encourage them to stop computation earlier. In contrast, target-related tokens retain longer computation paths in order to preserve sufficient feature interaction. The background-adaptive loss is defined as shown in Eq. (9).

$$L_{ponder} = \frac{\sum_i \rho_i w_i}{\sum_i w_i + \epsilon} \tag{9}$$

ρ_i denotes the accumulated computation cost of the i-th token, and w_i represents the background weight constructed from the template/search mask. This loss encourages background tokens to reduce unnecessary computation while maintaining the representational capability of target-related tokens.

3.3. Similarity-guided layer-adaptive mechanism

This article argues that representational redundancy exists among deep Transformer blocks. Therefore, instead of always executing layers 6 to 11 sequentially, a layer-selection module is introduced to dynamically choose a representative layer. The input to the layer-selection module is derived from the token features output by the last Transformer block of the shallow background-adaptive stage. After executing Transformer blocks 0 to 5, the shallow output is denoted as shown in Eq. (10).

$$T^{(l^*)} \in R^{B \times N \times C} \tag{10}$$

$l^* = 5$ indicates the final shallow Transformer block before layer selection, and $N = 321$ denotes the complete token state containing the CLS token, template tokens, and search tokens. To balance computational efficiency and feature representativeness, this paper selects the responses of all tokens on the 0-th feature

channel, $T_{(:,0)}^{(l^*)}$, as the input to the layer-selection MLP. Compared with using the full C-dimensional token features, using only a single channel substantially reduces the computational overhead of the layer-selection module.

The output of the layer-selection module is a real-valued vector $p \in R^6$, corresponding to the candidate layers $\{6, 7, 8, 9, 10, 11\}$. The candidate layer with the highest probability is selected for execution, as shown in Eq. (11).

$$k = \arg\max_j p_j \tag{11}$$

This design differs from layer-selection approaches based solely on template or search tokens. Since the CLS token provides global tracking-state information, template tokens provide target priors, and search tokens describe the target-background distribution in the current frame, the complete token state provides richer contextual information for layer selection.

3.4. Mask-aware similarity supervision

If the overall token representation is directly used to compute similarity among candidate layers, the layer-selection target may be heavily affected by numerous background tokens [35, 36]. To address this issue, this paper proposes a mask-aware similarity supervision mechanism. After the l^*-th layer, the background-aware token-halting mechanism produces an active token mask, as shown in Eq. (12).

$$M^{(l^*)} \in \{0, 1\}^N \tag{12}$$

$M^{(l^*)}$ denotes the token mask obtained after the l^*-th layer, N is the total number of tokens and $\{0, 1\}$ indicates whether a token is active. This paper only uses the portion corresponding to search tokens, denoted as $M_x^{(l^*)}$, where the subscript x represents the search region. Candidate-layer similarity is then computed based only on the still-active search tokens.

For a candidate layer B_j, the token feature $T^{(l^*)}$ is fed into each candidate Transformer block to obtain temporary outputs as shown in Eq. (13).

$$\widehat{T}^j = B_j(T^{(l^*)}), j \in \{6, 7, 8, 9, 10, 11\} \tag{13}$$

j denotes the candidate-layer index and B_j denotes the j-th candidate Transformer block.

The masked cosine similarity on active search tokens is then computed as shown in Eq. (14).

$$S_j = \frac{\sum_i M_{x,i}^{(l^*)} \langle T_{x,i}^{(l^*)}, \widehat{T}_{x,i}^j \rangle}{\sqrt{\sum_i M_{x,i}^{(l^*)} \|T_{x,i}^{(l^*)}\|^2} \sqrt{\sum_i M_{x,i}^{(l^*)} \|\widehat{T}_{x,i}^j\|^2} + \epsilon} \tag{14}$$

S_j denotes the similarity score of the j-th candidate layer, $T_{x,i}^{(l^*)}$ represents the feature of the i-th search token after the l^*-th layer, and $\widehat{T}_{x,i}^j$ represents the temporary output feature of the i-th search token after passing through the j-th candidate block. $M_{x,i}^{(l^*)}$indicates whether the i -th search token remains active after the l^*-th layer.

The numerator represents the sum of feature inner products over all active search tokens. The inner product measures directional similarity between the features before and after the candidate layer. Multiplication by the activity mask ensures that halted background tokens do not affect the similarity computation.

The denominator represents the product of the norms of the two feature sets, normalizing the inner product so that S_j becomes a cosine similarity rather than a raw inner-product magnitude. This avoids obtaining higher similarity values simply due to larger feature magnitudes. Specifically, $M_{x,i}^{(l^*)} \|T_{x,i}^{(l^*)}\|^2$denotes the squared L_2 -

norm of the reference feature, while $M_{x,i}^{(l^*)}\|\widehat{T}_{x,i}^{j}\|^2$ denotes the squared L_2 -norm of the candidate-layer output feature. ϵ is a small constant introduced to prevent division by zero.

Unlike computing similarity over all search tokens, the proposed mask-aware similarity supervision only computes similarity on search tokens that remain active in the mask. If the output of a candidate layer is most similar to the current reference feature, it indicates that the layer introduces minimal feature perturbation and may therefore be the most suitable subsequent execution layer.

The candidate layer with the highest similarity score is then used as the one-hot supervision label, as shown in Eq. (15).

$$y_j = \begin{cases} 1, & j = argmax_m S_m, \\ 0, & otherwise. \end{cases} \tag{15}$$

S_m denotes the similarity score of the m-th candidate layer, $argmax_m$ represents the index of the candidate layer with the highest similarity, and y_j denotes the supervision label corresponding to the j-th candidate layer.

The layer-selection loss is defined as shown in Eq. (16).

$$L_{pro} = \|p - y\|_1 \tag{16}$$

This loss supervises the MLP-based layer-selection module, enabling it to predict the most suitable subsequent layer according to the complete token state.

3.5. Shallow distribution prior

In the original background-aware token-halting mechanism, a full-layer distribution prior is used to constrain token-halting behavior, forcing the token-halting distribution to approach a predefined Gaussian distribution. This prior is suitable for a fully sequential 12-layer Transformer architecture, However, in the proposed dual-adaptive model, the execution path becomes as shown in Eq. (17).

$$0 \to 1 \to 2 \to 3 \to 4 \to 5 \to k, k \in \{6, 7, 8, 9, 10, 11\} \tag{17}$$

In other words, layers 6 to 11 are no longer executed sequentially. Instead, a representative layer is dynamically selected by the layer-selection module. If the original 12-layer distribution prior is still applied, the prior target becomes inconsistent with the actual execution path. Therefore, this paper proposes a shallow distribution prior that constrains only the first six continuously executed shallow blocks, as shown in Eq. (18).

$$B_{shallow} = \{0, 1, 2, 3, 4, 5\} \tag{18}$$

Let the shallow halting-score distribution be defined as shown in Eq. (19).

$$q = Normalize([h^0, h^1, h^2, h^3, h^4, h^5]) \tag{19}$$

A Gaussian target distribution of length 6 is then constructed as shown in Eq. (20).

$$g = GaussianTarget(L = 6, d = 4) \tag{20}$$

$L = 6$ indicates that only the first six layers are constrained, and $d = 4$ indicates that the Gaussian peak is positioned relatively late in the shallow stage, thereby avoiding overly aggressive early token stopping.

The shallow distribution-prior loss is defined as shown in Eq. (21).

$$L_{dp}^{shallow} = KL(q\|g) \tag{21}$$

This design preserves the stabilizing effect of the distribution prior on token halting while avoiding conflicts between the full-layer prior and the dynamic layer-skipping mechanism.

3.6. Overall loss function

The basic tracking loss of the proposed model includes the GIoU loss, the L1 bounding-box loss, and the center-location prediction loss, as shown in Eq. (22).

$$L_{track} = \lambda_{giou} L_{giou} + \lambda_1 L_1 + \lambda_{loc} L_{loc} \tag{22}$$

For different experimental settings, the overall loss functions are defined as follows.

Experiment 1: Original Background-Adaptive Mechanism, as shown in Eq. (23).

$$L_{Exp1} = L_{track} + \alpha_p L_{ponder} + \alpha_d L_{dp}^{full} \tag{23}$$

L_{dp}^{full} denotes the full 12-layer distribution-prior loss.

Experiment 2: Dual-Adaptive Mechanism Without Distribution Prior, as shown in Eq. (24).

$$L_{Exp2} = L_{track} + \alpha_p L_{ponder} + \gamma L_{pro} \tag{24}$$

This experiment disables the distribution prior in order to verify whether the similarity-guided layer-adaptive mechanism alone can improve model performance.

Experiment 3: Dual-Adaptive Mechanism With Shallow Distribution Prior, as shown in Eq. (25).

$$L_{Exp3} = L_{track} + \alpha_p L_{ponder} + \alpha_s L_{dp}^{shallow} + \gamma L_{pro} \tag{25}$$

This experiment is designed to verify whether preserving the distribution prior only during the shallow continuous-execution stage can further stabilize token-halting behavior and improve tracking performance in the dual-adaptive framework.

4. Experimental design

4.1. Experimental settings

This paper adopts Aba-ViTrack as the baseline framework and uses an MAE-pretrained ViT backbone. The input settings are: template size 128×128 , search size 256×256, and patch size 16. Training is performed using the AdamW optimizer with a base learning rate of 4×10^{-4}. The learning-rate scaling factor for the backbone is set to 0.1. The total number of training epochs is 300, and the learning rate decays at the 240th epoch.

The training datasets comprised the GOT-10K vottrain split, LaSOT, and the COCO 2017 training split, while the validation datasets consisted of the GOT-10K votval split [37], LaSOT, and the COCO 2017 validation split. UAV123 [38], VisDrone [39], and DTB70 were used as independent test benchmarks. All experiments employed identical dataset configurations, numbers of training epochs, optimizer settings, batch sizes, and input resolutions to ensure a fair comparison.

4.2. Comparative experimental settings

This paper designs three groups of core experiments.

Experiment 1: Original Background-Adaptive Mechanism. This experiment does not include the layer-adaptive mechanism. It only uses the original background-aware token-halting mechanism while retaining the full 12-layer distribution prior. This experiment aims to answer how the model performs in terms of accuracy and efficiency when only the background-token adaptive mechanism is used.

Experiment 2: Dual-Adaptive Mechanism Without Distribution Prior. This experiment introduces the similarity-guided layer-adaptive mechanism on top of the background-adaptive mechanism while disabling the

distribution prior. This experiment aims to answer whether the similarity-guided layer-adaptive mechanism can still improve model performance without using a distribution prior.

Experiment 3: Dual-Adaptive Mechanism With Shallow Distribution Prior This experiment uses the same dual-adaptive backbone as Experiment 2 but enables the shallow distribution prior. This experiment aims to answer whether, in the dual-adaptive model, preserving the distribution prior only during the shallow continuous-execution stage can further stabilize token-halting behavior and improve tracking performance.

5. Experimental results and analysis

5.1. Comparison of main results

As shown in Table 1, compared with Exp1, which only incorporates the background-adaptive mechanism, both Exp2 and Exp3 achieve substantial improvements in inference speed after introducing the layer-adaptive mechanism. Specifically, the mean FPS increased from 70.98 ± 2.12 in Exp1 to 91.65 ± 4.07 FPS in Exp2, while Exp3 retained a similarly high mean inference speed of 95.63 ± 4.38. These results indicate that considerable layer-wise redundancy exists within deep Transformer blocks. The proposed layer-selection strategy reduces redundant deep-layer computation while preserving informative feature representations, thereby producing a substantial improvement in measured inference efficiency under the evaluated configurations.

Comparing Exp1 and Exp2, when the original full distribution prior is removed and replaced with no distribution prior, all accuracy-related metrics decline. Specifically, the AUC decreases from 59.99 to 57.25, Precision drops from 76.73 to 74.39, and Norm Precision decreases from 72.77 to 69.67. This performance degradation suggests that simply disabling the full distribution prior and retaining only the background ponder loss together with the similarity-guided layer selection loss negatively affects tracking accuracy. Therefore, the distribution prior serves as an important factor in maintaining model performance.

The observations from Exp2 further motivate the necessity of Exp3. Compared with Exp2, the introduction of the shallow distribution prior in Exp3 leads to consistent improvements across all accuracy metrics. The resulting tracking performance becomes comparable to that of Exp1, while Exp3 retains a similarly high inference speed of 95.63 ± 4.38 FPS compared with 91.65 ± 4.07 FPS for Exp2. This result indicates that the shallow distribution prior improves tracking accuracy without introducing an evident inference-time burden, thereby achieving a better trade-off between accuracy and efficiency. These results demonstrate that the proposed shallow distribution prior effectively stabilizes token halting behavior during the shallow stages of the network, providing a reliable foundation for the subsequent similarity-guided layer adaptation. Consequently, the proposed shallow distribution prior and similarity-guided layer adaptation form a complementary dual-adaptive framework that exploits both token-level and layer-level redundancy, achieving substantially improved measured inference efficiency while maintaining competitive tracking performance.

Table 1. Comparison of overall performance of three groups of experiments

Method	Background adaptation	Layer adaptation	AUC	Precision	Norm Precision	FPS (mean ± SD)
Exp1: baseline	√	×	59.99	76.73	72.77	70.98 ± 2.12
Exp2: dual	√	√	57.25	74.39	69.67	91.65 ± 4.07
Exp3: dual shallow	√	√	59.39	76.83	72.15	95.63 ± 4.38

Note: √ indicates that the corresponding component is enabled, whereas × indicates that it is disabled. FPS values are reported as the mean ± standard deviation over five complete DTB70 evaluations.

5.2. Effectiveness analysis of similarity-guided layer adaptation

As shown in Table 2, the comparison between Exp1 and Exp2 evaluates the efficiency effect of introducing layer adaptation under the no-distribution-prior configuration. Exp1 only contains background token adaptation, while Exp2 adds a similar guiding layer selection mechanism. Compared with Exp1, Exp2 exhibits a consistent decline in all accuracy metrics, demonstrating the necessity of the original full distribution prior. The results indicate that completely removing the distribution prior is not a reasonable design choice, as it adversely affects tracking performance. Meanwhile, the increase in FPS confirms that the proposed similarity-guided layer adaptation mechanism effectively reduces redundant computations in deep Transformer layers.

From a mechanistic perspective, the background-adaptive strategy primarily alleviates redundancy at the token level by suppressing unnecessary background tokens. However, it still requires all Transformer blocks to be executed at a fixed depth. In contrast, the proposed dual-adaptive model further exploits redundancy at the layer level by dynamically selecting representative blocks from deep Transformer layers for execution, thereby reducing additional computational overhead associated with redundant layer processing.

These results suggest that token-level dynamic computation alone is insufficient to fully exploit the redundancy inherent in Transformer-based tracking models. By introducing layer-level dynamic selection, the proposed framework can further improve computational efficiency while maintaining competitive tracking performance, highlighting the effectiveness of jointly modeling token-level and layer-level redundancy.

Table 2. Adaptive ablation of layers

Method	Layer adaptation	L_{pro}	AUC	Precision	Norm Precision	FPS (mean ± SD)
Exp1: baseline	×	×	59.99	76.73	72.77	70.98 ± 2.12
Exp2: dual	√	√	57.25	74.39	69.67	91.65 ± 4.07

Note: √ indicates that the corresponding component is enabled, whereas × indicates that it is disabled. FPS values are reported as the mean ± standard deviation over five complete DTB70 evaluations.

5.3. Effectiveness analysis of the shallow distribution prior

As shown in Table 3, the comparison between Exp2 and Exp3 is used to evaluate the effectiveness of the shallow distribution prior.

Compared with Exp2, Exp3 introduces the proposed shallow distribution prior while maintaining the same similarity-guided layer adaptation mechanism. The experimental results show that Exp3 achieves consistent improvements in tracking accuracy, with AUC increasing from 57.25 to 59.39 and Precision increasing from 74.39 to 76.83. Notably, the accuracy performance becomes comparable to that of Exp1, indicating that the performance degradation observed in Exp2 can be effectively alleviated by reintroducing an appropriate distribution prior. The reason lies in the interaction between token-level adaptation and layer-level adaptation. In the original background-adaptive framework, the full distribution prior constrains the halting behavior of search tokens throughout the entire network, thereby guiding the model to learn a stable token selection pattern. However, after introducing layer adaptation, the execution path of deep Transformer blocks becomes dynamic rather than fixed. Under such circumstances, directly applying the original full distribution prior may introduce inconsistencies between the assumed token distribution and the actual computation path. Conversely, completely removing the distribution prior, as in Exp2, weakens the supervision on token halting behavior, resulting in unstable token selection and degraded tracking performance. To address this issue, the proposed shallow distribution prior only constrains token halting behavior within the shallow continuous execution stage, where all Transformer blocks are still executed sequentially. Since the computation path remains stable

in this stage, the distribution prior can provide reliable supervision without conflicting with the subsequent dynamic layer selection process. As a result, the model learns a more stable token halting distribution while preserving the flexibility of deep-layer adaptation.

Furthermore, Exp3 retained a comparable high inference speed to Exp2, achieving 95.63 ± 4.38 FPS compared with 91.65 ± 4.07 FPS. This indicates that the shallow distribution prior restores tracking accuracy without introducing an evident inference-time burden. Therefore, the shallow distribution prior and similarity-guided layer adaptation form a complementary design: the former stabilizes token-level adaptation in shallow layers, while the latter exploits layer-level redundancy in deep layers. Together, they enable the model to achieve a superior balance between tracking accuracy and inference efficiency.

Table 3. Adaptive ablation of layers

Method	Distribution prior	L_{dp}	L_{pro}	AUC	Precision	FPS (mean ± SD)
Exp2: dual	none	×	0.137	57.25	74.39	91.65 ± 4.07
Exp3: dual shallow	shallow	0.525	0.137	59.39	76.83	95.63 ± 4.38

Note: "None" indicates that no distribution prior is applied, whereas "shallow" indicates that the proposed shallow distribution prior is enabled. √ and × indicate enabled and disabled loss terms, respectively. FPS values are reported as the mean ± standard deviation over five complete DTB70 evaluations.

5.4. Comparison of training losses

To further investigate the optimization behavior of the proposed models, the major training losses are recorded and compared across the three experiments. As shown in Table 4, the localization-related losses (L_{giou}, L_1, and L_{loc}) exhibit trends that are consistent with the tracking performance reported in the previous section. Compared with Exp1, Exp2 produces higher localization losses, indicating degraded localization accuracy after removing the distribution prior. After introducing the proposed shallow distribution prior, Exp3 reduces these losses, demonstrating that the proposed regularization effectively compensates for the performance degradation caused by the absence of distribution supervision.

In Exp1, L_{dp} measures the KL divergence between the learned token halting distribution across all twelve Transformer layers and the predefined Gaussian prior L_i. The relatively small value of L_{dp} (0.052) indicates that the learned halting behavior closely follows the expected distribution, suggesting that the Full Distribution Prior effectively regularizes token halting and promotes stable token selection throughout the network.

In Exp2, the original distribution prior is completely removed, and only the similarity-guided layer adaptation is optimized. Consequently, L_{pro} becomes the primary auxiliary objective for learning the layer selection strategy. This loss measures the discrepancy between the predicted layer selection and the mask-aware similarity target. The stable convergence of L_{pro} demonstrates that the MLP-based layer selector can successfully learn the similarity-guided layer selection policy. However, without explicit distribution supervision, the token evolution trajectory gradually deviates from the original optimization objective, resulting in increased localization losses and reduced tracking accuracy, as observed in Table 3.

Exp3 simultaneously optimizes both the shallow distribution prior loss (L_{dp}) and the layer selection loss (L_{pro}). Although the value of L_{dp} increases from 0.052 in Exp1 to 0.525, this increase is expected rather than detrimental. After introducing Similarity-guided Layer Adaptation, the execution path of tokens is no longer fixed because deep Transformer blocks are dynamically skipped according to the predicted layer importance. Consequently, the actual token halting distribution in the shallow stage deviates more from the original Gaussian prior, leading to a larger KL divergence and therefore a higher L_{dp}.

Nevertheless, the increased L_{dp} does not indicate unstable optimization. Instead, it reflects the fact that the proposed shallow distribution prior only constrains token halting behavior during the shallow continuous execution stage rather than enforcing the original full-layer distribution prior. As a lightweight regularization term, it provides sufficient supervision for stable token halting while avoiding conflicts with the subsequent dynamic layer adaptation. Meanwhile, the stable value of L_{pro} indicates that the similarity-guided layer selector continues to learn reliable layer selection patterns. These observations demonstrate that the shallow distribution prior and the similarity-guided layer adaptation are compatible and can be jointly optimized without interfering with each other.

Table 4. Training loss comparison

Method	L_{total}	L_{giou}	L_1	L_{loc}	L_{ponder}	L_{dp}	L_{pro}
Exp1: baseline	0.647	0.167	0.014	0.245	0.502	0.052	×
Exp2: dual	0.829	0.204	0.018	0.306	0.501	×	0.137
Exp3: dual shallow	0.760	0.188	0.016	0.276	0.501	0.525	0.137

Note: √ indicates that the corresponding component is enabled, whereas × indicates that it is disabled.

5.5. Qualitative visualization analysis

In addition to the quantitative evaluation, qualitative visualization was conducted to further investigate the tracking behavior and localization characteristics of the three experimental settings. Representative frames were selected from the Basketball and Animal1 sequences of the DTB70 dataset. To ensure a fair comparison, Exp1, Exp2, and Exp3 were evaluated using the same video frames, ground-truth annotations, search-region settings, input resolution, and visualization parameters. The qualitative analysis consists of two parts: bounding-box prediction comparison and classification response-map visualization.

5.5.1. Bounding-box tracking comparison

To intuitively compare the localization performance of the three experimental settings, the predicted bounding boxes were visualized together with the corresponding ground-truth annotations. Exp1 represents the original background-adaptive tracker with the full-layer distribution prior; Exp2 introduces similarity-guided layer adaptation while removing the distribution prior; and Exp3 combines the dual-adaptive architecture with the proposed shallow distribution prior.

As shown in Figure 2, Exp1 generally maintains stable target localization because the full-layer distribution prior provides continuous supervision for token-halting behavior throughout the Transformer backbone. However, its fixed execution of all deep Transformer blocks results in relatively low inference efficiency. Exp2 substantially reduces redundant deep-layer computation through similarity-guided layer selection, but the removal of distribution supervision makes token evolution less stable. Consequently, its predicted bounding boxes may exhibit more noticeable center-position or scale deviations in challenging frames. Compared with Exp2, Exp3 produces bounding boxes that are more consistent with the ground-truth target in terms of both target position and spatial extent. This result indicates that the shallow distribution prior provides effective regularization during the continuously executed shallow stage, allowing reliable target-related representations to be preserved before dynamic deep-layer selection. Meanwhile, because the prior is restricted to the shallow layers, it does not conflict with the subsequent layer-skipping path.

The qualitative bounding-box comparison is consistent with the quantitative results reported in Tables 1–3. Although Exp2 improves inference speed through layer adaptation, its tracking accuracy decreases after the distribution prior is completely removed. Exp3 alleviates this degradation and restores localization

performance to a level comparable to Exp1 while retaining the computational advantages of layer adaptation. Therefore, the shallow distribution prior and similarity-guided layer adaptation form a complementary mechanism for balancing tracking accuracy and inference efficiency.

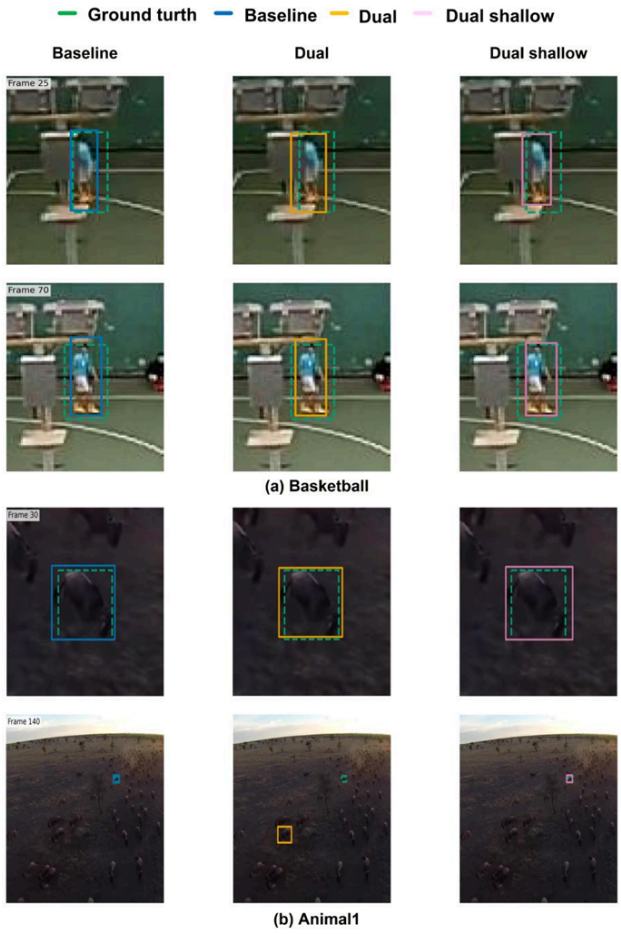

Figure 2. Qualitative comparison of bounding-box predictions for the three experimental settings on representative DTB70 sequences: (a) Basketball, illustrating target motion, appearance variation, and scale change; (b) Animal1, illustrating complex background clutter, small-target conditions, and similar distractors. The green dashed boxes denote the ground-truth targets, while the colored solid boxes denote the predictions of the corresponding models. All methods are visualized using identical frame indices and image regions for fair comparison

5.5.2. Response-map visualization

To further analyze the internal localization behavior of the three models, the raw classification score maps generated by the center prediction head were visualized for the same frames used in the bounding-box comparison. The raw score map represents the model-estimated confidence that each spatial location corresponds to the target center before the application of the Hann-window motion constraint. Warmer colors indicate higher target-location confidence, whereas cooler colors indicate lower responses. For fair comparison, all response maps were generated using the same search regions, frame indices, color mapping, response-value range, and overlay transparency. In addition, the response values were directly clipped to the common range of $[0,1]$, rather than independently applying min–max normalization to each model. This

avoids artificially amplifying weak responses and ensures that the response intensity remains directly comparable across the three experimental settings.

As illustrated in Figure 3, Exp1 generally produces a relatively concentrated response around the target region, indicating that the full distribution prior helps maintain stable token evolution and target discrimination. Nevertheless, the model still performs the complete deep-layer computation, which limits its inference speed. After the distribution prior is removed, Exp2 tends to produce a more dispersed or background-sensitive response distribution in difficult frames. Multiple competing responses or a broader activation region may reduce the certainty of target-center localization and consequently lead to bounding-box displacement or inaccurate scale estimation. In contrast, Exp3 produces a more compact and target-centered response distribution than Exp2. The high-response region is more strongly associated with the actual target, while responses from surrounding background regions are comparatively suppressed. This observation suggests that the shallow distribution prior stabilizes token-halting behavior before the layer-selection stage and preserves discriminative target features for subsequent adaptive computation. As a result, the similarity-guided layer selector can reduce redundant deep-layer computation without excessively weakening the localization representation.

The response-map visualization provides a mechanistic explanation for the quantitative performance differences among the three experiments. Exp2 achieves a clear speed improvement but suffers from reduced localization accuracy because the absence of distribution supervision weakens the stability of token selection. Exp3 restores a more reliable response distribution by introducing shallow-stage regularization, thereby improving AUC and precision while preserving high inference speed. These qualitative observations further support the effectiveness of jointly modeling token-level and layer-level redundancy in the proposed dual-adaptive framework.

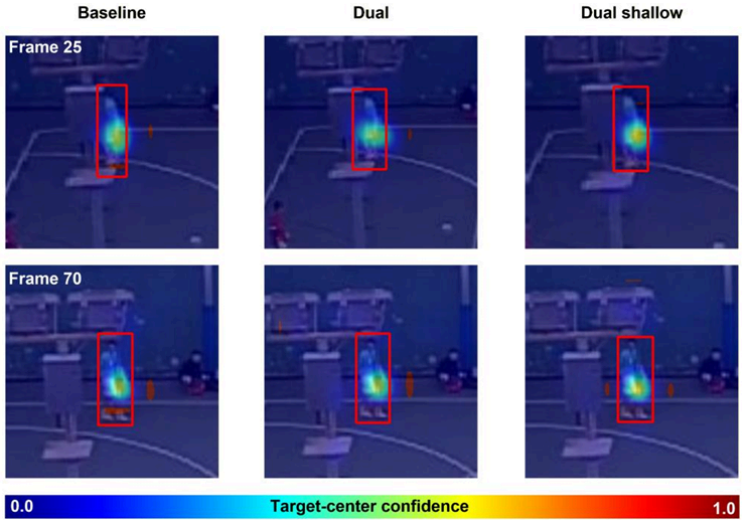

Figure 3. Visualization of the raw classification response maps generated by Exp1, Exp2, and Exp3 on representative frames from the DTB70 Basketball sequence. Warmer colors indicate higher target-center confidence, while cooler colors indicate lower confidence. The red boxes denote the predicted bounding boxes. All response maps use identical frame indices, search regions, value ranges, colormaps, and overlay parameters. The shared color bar represents response values ranging from 0.0 to 1.0.

5.6. Computational efficiency analysis

5.6.1. Average executed transformer blocks

To further quantify the computational savings introduced by the proposed layer-adaptive mechanism, we record the actual number of Transformer blocks executed for every tracking frame during inference on the DTB70 benchmark. Unlike the FPS measurement, which reflects the overall runtime of the complete tracking pipeline, the executed-block analysis directly characterizes the effective computation depth of the Transformer backbone. For each tracking frame, the number of executed blocks is recorded and subsequently aggregated across all evaluated sequences. The average number of executed blocks is calculated as shown in Eq. (26).

$$\overline{B} = \frac{1}{N} \sum_{i=1}^{N} B_i \tag{26}$$

where denotes the total number of tracked frames and represents the number of Transformer blocks actually executed for the -th frame. As shown in Table 5, Exp1, which uses the full-depth background-adaptive baseline, executes all 12 Transformer blocks for each tracking frame. The measured average executed depth is therefore 12.00 blocks per frame. This result is consistent with the fixed computation path of the baseline architecture, in which token-level adaptation is performed while the complete sequence of Transformer blocks remains available for inference.

In contrast, both Exp2 and Exp3 execute only seven Transformer blocks per tracking frame on average. The reduction from 12 to 7 executed blocks corresponds to a reduction of 41.67%. This result provides direct evidence that the proposed similarity-guided layer adaptation effectively removes redundant deep-layer computation. In the adaptive architecture, the first six shallow Transformer blocks are executed continuously, while only one representative block is selected from the subsequent candidate deep layers. Consequently, the effective Transformer execution depth is reduced from 12 blocks in Exp1 to 7 blocks in both Exp2 and Exp3.

Importantly, the identical average executed depth obtained by Exp2 and Exp3 indicates that the introduction of the shallow distribution prior does not increase the number of Transformer blocks executed during inference. This observation is consistent with the design objective of the shallow distribution prior: it constrains token-halting behavior only in the shallow continuous execution stage while preserving the layer-adaptive mechanism in the deep stage. Therefore, the additional regularization introduced in Exp3 improves tracking stability without sacrificing the computational savings achieved through layer adaptation. This is also consistent with the previously reported results, where Exp3 achieves higher tracking accuracy than Exp2 while maintaining a comparable high inference speed.

Overall, the executed-block analysis demonstrates that the proposed layer-adaptive mechanism reduces the effective Transformer computation depth from 12 to 7 blocks per frame, corresponding to a 41.67% reduction in executed Transformer blocks. Exp3 achieves the same computational depth as Exp2 while providing substantially improved tracking accuracy, demonstrating that the shallow distribution prior can improve the accuracy-efficiency trade-off without introducing additional Transformer block execution.

Table 5. Average executed Transformer blocks on DTB70

Method	Number of sequences	Total tracked frames	Average executed blocks
Exp1: baseline	70	15707	12
Exp2: dual	70	15707	7
Exp3: dual shallow	70	15707	7

5.6.2. Params comparison

To further evaluate the computational complexity of the three experimental settings, the total and trainable parameter counts and the static multiply-accumulate (MAC) operations were measured using the same model configuration and input resolution. Parameter counts characterize model storage and learnable capacity, whereas static MACs characterize the nominal computational cost of a fixed forward graph before dynamic layer-selection behaviour is taken into account.

The exact parameter counts obtained from the implemented models are 8.23M parameters for Exp1, 8.29M parameters for Exp2, and 8.29 parameters for Exp3. All parameters are trainable in the evaluated configurations. Relative to Exp1, Exp2 and Exp3 introduce only 0.06M additional parameters, corresponding to a 0.64% increase. The identical parameter counts of Exp2 and Exp3 are expected because the shallow distribution prior changes the training objective but does not add or remove inference-network parameters. Static MACs were profiled using THOP under the same input configuration. Exp1 requires 2.40 GMACs, whereas both Exp2 and Exp3 require 1.68 GMACs. The corresponding static MAC reduction is 29.74% relative to Exp1. Using the conventional approximation of two FLOPs per MAC, the corresponding static FLOP estimates are 4.79 GFLOPs for Exp1 and 3.37 GFLOPs for Exp2 and Exp3.

The parameter and static-complexity results are summarized in Table 6. The results show that the proposed adaptive architecture achieves a substantial reduction in static MACs with only a very small increase in parameter count. Moreover, the identical static complexity of Exp2 and Exp3 confirms that the shallow distribution prior does not introduce additional inference architecture or parameter overhead.

Table 6. Parameter counts and static computational complexity of the three experiments.

Method	Params (M)	Trainable Params (M)	Params	Static MACs (G)	Static FLOPs (G)
Exp1: baseline	8.23	8.23	0.00%	2.40	4.79
Exp2: dual	8.29	8.29	0.64%	1.68	3.37
Exp3: dual shallow	8.29	8.29	0.64%	1.68	3.37

6. Conclusion

This paper presents a dual-dimensional adaptive tracking method based on background-aware token adaptation and similarity-guided layer selection. Based on the background adaptive token halting mechanism, this method further adds a similarity-guided layer-adaptive module in the deep stage, so that the model can reduce the background token redundancy and the deep Transformer layer redundancy at the same time. In order to solve the problem that the prior of the original complete layer distribution does not match the dynamic layer-skipping path, this paper further designs the shallow distribution prior, which only restricts the token halting distribution in the shallow continuous execution stage.

A comparative analysis of the three experimental settings shows that Exp1, which adopts the original background-adaptive mechanism with the full distribution prior, achieves high tracking accuracy but has limited inference speed because of its fixed computation path. Exp2 introduces Similarity-Guided Layer Adaptation and achieves a substantially higher mean inference speed under the evaluated configuration; however, the absence of distribution supervision causes token evolution to drift, leading to degraded localization performance. Exp3 further incorporates the proposed Shallow Distribution Prior, which constrains token halting behavior only in the shallow layers while preserving the flexibility of dynamic layer adaptation. As a result, the localization accuracy is effectively recovered with only a marginal reduction of approximately

0.6 AUC compared with Exp1, while the mean inference speed increases substantially from 70.98 ± 2.12 FPS to 95.63 ± 4.38 FPS.

These results demonstrate that the proposed shallow distribution prior successfully reconciles the conflict between distribution regularization and dynamic layer adaptation, enabling the model to achieve a superior balance between tracking accuracy and computational efficiency.

Future work can further study the prior design of dynamic distribution and the joint optimization of mask-aware similarity and token importance estimation, thus further improving the efficiency and robustness of dynamic visual tracking models.

Author contribution

The authors confirm contribution to the paper as follows: conceptualization, Zhiyong Zhang; methodology, Zhiyong Zhang and Yanglong Guo; software, Yanglong Guo; validation, Yanglong Guo and Zhiyong Zhang; formal analysis, Yanglong Guo; investigation, Yanglong Guo; data curation, Yanglong Guo; writing—original draft preparation, Yanglong Guo; writing—review and editing, Zhiyong Zhang and Yanglong Guo; visualization, Yanglong Guo; supervision, Zhiyong Zhang; project administration, Zhiyong Zhang. All authors reviewed and approved the final version of the manuscript.

References

[1] Chen, L., & Liu, Y. (2022). UAV visual target tracking algorithms: Review and future prospect. *Information and Control, 51*(1), 23–40. https://doi.org/10.13976/j.cnki.xk.2022.1144

[2] Danelljan, M., Bhat, G., Khan, F. S., & Felsberg, M. (2017). ECO: Efficient convolution operators for tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 6638–6646). IEEE. https://doi.org/10.1109/CVPR.2017.702

[3] Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., & Torr, P. H. S. (2016). Fully-convolutional siamese networks for object tracking. In *Proceedings of the European Conference on Computer Vision* (pp. 850–865). Springer. https://doi.org/10.1007/978-3-319-48881-3_56

[4] Li, B., Yan, J., Wu, W., Zhu, Z., & Hu, X. (2018). High performance visual tracking with siamese region proposal network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 8971–8980). IEEE. https://doi.org/10.1109/CVPR.2018.00935

[5] Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., & Xu, C. (2023). A survey on vision transformer. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 45*(1), 87–110. https://doi.org/10.1109/TPAMI.2022.3152247

[6] Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., & Shah, M. (2022). Transformers in vision: A survey. *ACM Computing Surveys, 54*(10s), 1–41. https://doi.org/10.1145/3505244

[7] Khan, A., Rauf, Z., Sohail, A., Khan, A. R., Asif, H., Asif, A., & Farooq, U. (2023). A survey of the vision transformers and their CNN-transformer based variants. *Artificial Intelligence Review, 56*(3), 2917–2970. https://doi.org/10.1007/s10462-023-10595-0

[8] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16 × 16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*. https://openreview.net/forum?id=YicbFdNTTy

[9] Ye, B., Chang, H., Ma, B., Shan, S., & Chen, X. (2022). Joint feature learning and relation modeling for tracking: A one-stream framework. In *Proceedings of the 17th European Conference on Computer Vision* (pp. 338–355). Springer. https://doi.org/10.1007/978-3-031-20047-2_20

[10] Cui, Y., Jiang, C., Wang, L., & Wu, G. (2024). Unified spatio-temporal attention MixFormer for visual object tracking. *Engineering Applications of Artificial Intelligence, 134*, Article 108682. https://doi.org/10.1016/j.engappai.2024.108682

[11] Chen, X., Yan, B., Zhu, J., Wang, D., Yang, X., & Lu, H. (2021). Transformer tracking. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8122–8131. https://doi.org/10.1109/CVPR46437.2021.00803

[12] Ye, B., Peng, W., Chen, X., Zhang, T., Shan, S., & Chen, X. (2022). OSTrack: One-stream transformer tracking. In *Proceedings of the 17th European Conference on Computer Vision* (pp. 444–459). Springer. https://doi.org/10.1007/978-3-031-20047-2_26

[13] Fayyaz, M., Koohpayegani, S. A., Jafari, N., Adeli, E., Galasso, F., & Ghodrati, A. (2024). Efficient vision transformers: A survey. *ACM Computing Surveys, 57*(1), 1–38. https://doi.org/10.1145/3656571

[14] Yin, H., Vahdat, A., Alvarez, J. M., Mallya, A., Kautz, J., & Molchanov, P. (2022). A-ViT: Adaptive tokens for efficient vision transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (pp. 10799–10808). IEEE. https://doi.org/10.1109/CVPR52688.2022.01054

[15] Rao, Y., Zhao, W., Liu, B., Lu, J., Zhou, J., & Hsieh, C. J. (2021). DynamicViT: Efficient vision transformers with dynamic token sparsification. *Advances in Neural Information Processing Systems, 34*, 13937–13949. https://doi.org/10.5555/3540261.3541329

[16] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., & Zagoruyko, S. (2020). End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision* (pp. 213–229). Springer. https://doi.org/10.1007/978-3-030-58452-8_13

[17] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., & Dai, J. (2021). Deformable DETR: Deformable transformers for end-to-end object detection. In *Proceedings of the 9th International Conference on Learning Representations*.

[18] Yu, W., Si, C., Zhou, P., Luo, M., Zhou, Y., Feng, J., Yan, S., & Wang, X. (2024). MetaFormer baselines for vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 46*(2), 896–912. https://doi.org/10.1109/TPAMI.2023.3329173

[19] Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., & Jégou, H. (2021). Training data-efficient image transformers and distillation through attention. *Proceedings of Machine Learning Research, 139*, 10347–10357.

[20] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (pp. 9992–10002). https://doi.org/10.1109/ICCV48922.2021.00986

[21] Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., Wei, F., & Guo, B. (2022). Swin transformer V2: Scaling up capacity and resolution. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 11999–12009). https://doi.org/10.1109/CVPR52688.2022.01170

[22] Meng, L., Li, H., Chen, B.-C., Lan, S., Wu, Z., Jiang, Y.-G., & Lim, S.-N. (2022). AdaViT: Adaptive vision transformers for efficient image recognition. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 12299–12308). https://doi.org/10.1109/CVPR52688.2022.01199

[23] Chen, M., Peng, H., Fu, J., & Ling, H. (2021). Autoformer: Searching transformers for visual recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (pp. 12270–12280). IEEE. https://doi.org/10.1109/ICCV48922.2021.01205

[24] Ryoo, M. S., Piergiovanni, A. J., Arnab, A., Dehghani, M., & Angelova, A. (2021). TokenLearner: Adaptive space-time tokenization for videos. *Advances in Neural Information Processing Systems, 34*, 12786–12797. https://doi.org/10.5555/3540261.3541240

[25] Li, S., Yang, Y., Zeng, D., & Wang, X. (2023). 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). IEEE. https://doi.org/10.1109/ICCV51070.2023.01286

[26] Graves, A. (2016). *Adaptive computation time for recurrent neural networks*. arXiv. https://doi.org/10.48550/arXiv.1603.08983

[27] Michel, P., Levy, O., & Neubig, G. (2019). Are sixteen heads really better than one? *Advances in Neural Information Processing Systems, 32*, 14014–14024. https://doi.org/10.5555/3454287.3455544

[28] Voita, E., Talbot, D., Moiseev, F., Sennrich, R., & Titov, I. (2019). Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics* (pp. 5797–5808). Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1580

[29] Elbayad, M., Gu, J., Grave, E., & Auli, M. (2020). Depth-adaptive transformer. In *Proceedings of the 8th International Conference on Learning Representations*.

[30] Han, Y., Huang, G., Song, S., Yang, L., Wang, H., & Wang, Y. (2022). Dynamic neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 44*(11), 7436–7456. https://doi.org/10.1109/TPAMI.2021.3117837

[31] Fan, A., Grave, E., & Joulin, A. (2020). Reducing transformer depth on demand with structured dropout. In *Proceedings of the 8th International Conference on Learning Representations*.

[32] Wu, L., Wang, Y., Wang, L., & Li, H. (2024). Depth-adaptive vision transformer with dynamic layer aggregation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 46*(7), 4678–4692. https://doi.org/10.1109/TPAMI.2024.3367890

[33] Wang, H., Zhang, H., Li, Y., & Wang, X. (2023). Learning dynamic network architectures via layer selection. *IEEE Transactions on Neural Networks and Learning Systems, 34*(9), 5678–5689. https://doi.org/10.1109/TNNLS.2022.3185632

[34] Xue, C., Zhong, B., Liang, Q., Zheng, Y., Li, N., Xue, Y., & Song, S. (2025). Similarity-guided layer-adaptive vision transformer for UAV tracking. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 6730–6740). https://doi.org/10.1109/CVPR52734.2025.00631

[35] Müller, R., Kornblith, S., & Hinton, G. E. (2019). When does label smoothing help? *Advances in Neural Information Processing Systems, 32*, 4696–4705. https://doi.org/10.5555/3454287.3454709

[36] Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (pp. 2818–2826). IEEE. https://doi.org/10.1109/CVPR.2016.308

[37] Huang, L., Zhao, X., & Huang, K. (2021). GOT-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 43*(5), 1562–1577. https://doi.org/10.1109/TPAMI.2019.2957464

[38] Mueller, M., Smith, N., & Ghanem, B. (2016). A benchmark and simulator for UAV tracking. In *Proceedings of the European Conference on Computer Vision* (pp. 445–461). Springer. https://doi.org/10.1007/978-3-319-46448-0_27

[39] Zhu, P., Du, D., Wen, L., Bian, X., Ling, H., Hu, Q., Peng, T., Zheng, J., Wang, X., Zhang, Y., Bo, L., Shi, H., Zhu, R., Dong, B., Pailla, D. R., Ni, F., Gao, G., Liu, G., Xiong, H., . . . Liu, Z. (2019). VisDrone-VID2019: The vision meets drone object detection in video challenge results. *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)* (pp. 227–235). https://doi.org/10.1109/ICCVW.2019.00031