electronics

MDPI

Article

A Pruning Strategy-Based Object Tracking Method

Peiting Gu [1,2,3], Detian Huang [4,*], Hang Liu [4] and Xintong Li [4]

[1] School of Mathmatics and Computer Science, Quanzhou Normal University, Quanzhou 362000, China; ptinggu@qztc.edu.cn
[2] Fujian Provincial Key Laboratory of Data-Intensive Computing, Quanzhou Normal University, Quanzhou 362000, China
[3] Key Laboratory of Intelligent Computing and Information Processing, Quanzhou Normal University, Quanzhou 362000, China
[4] College of Engineering, Huaqiao University, Quanzhou 362021, China
* Correspondence: huangdetian@hqu.edu.cn

Abstract

Most Transformer-based tracking methods overemphasize tracking accuracy while neglecting tracking efficiency. To address this issue, a lightweight object tracking method based on a hierarchical pruning strategy is proposed. First, an activation module is introduced to adaptively adjust the attention layers of the backbone network, avoiding unnecessary computations of attention layers. Then, based on the demands at different stages of the network, the standard attention mechanism is improved and divided into hybrid attention and cross attention, further reducing redundant computations. Experimental results on multiple benchmark datasets demonstrate that PSTrack achieves a tracking speed of 216 FPS on GPU and 48 FPS on CPU while maintaining competitive accuracy, with an AO of 68.7% on GOT-10k and an AUC of 66.1% on LaSOT. The proposed tracking method exhibits strong competitiveness in both quantitative and qualitative evaluations.

Keywords: transformer; pruning strategy; object tracking; attention mechanism

1. Introduction

In recent years, with the rapid development of deep neural networks, significant progress has been made in the field of object tracking. In particular, the Transformer, with its powerful encoding capabilities, has played a key role in the development of high-performance trackers, effectively addressing issues such as deformation, blurring, and cluttered backgrounds during object motion. From the early convolutional neural networks to the current mainstream Transformer methods, which stack a large number of dense attention modules, many high-performance trackers have emerged. However, a common and overlooked issue with the current mainstream Transformer-based trackers is that they do not fully consider tracking speed. Although these advanced trackers can achieve real-time performance on high-performance GPUs, their efficiency significantly drops when applied to devices with limited computational resources. For example, ARTrack [1], considered a top-tier tracker, achieves a tracking speed of 37 frames per second (FPS) on the NVIDIA RTX 2080 Ti GPU, but this drops to only 5 FPS on the common edge device Nvidia Jetson Orin NX. How to effectively balance performance and speed has become a key challenge that needs to be addressed in the current development of object tracking technology. By carefully examining the attention heatmaps across different layers of the Transformer, this study finds that the Transformer has already given sufficient attention to the target in some of the early layers. Moreover, in certain simple tracking scenarios—such

Check for updates

Academic Editor: Dah-Jye Lee

Received: 1 December 2025
Revised: 3 January 2026
Accepted: 13 January 2026
Published: 16 January 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

as when the target's motion is stable, and its appearance does not change significantly—the tracking network can achieve reliable tracking results using only a few encoder layers. Therefore, by flexibly pruning the number of network layers, it is possible to significantly improve the inference speed of the tracker.

Based on the above considerations, this paper investigates a lightweight object tracking method based on a hierarchical pruning strategy, named PSTrack (Pruning Strategy-based object Tracking method). Specifically, to build an efficient and fast tracking network, this method introduces an activation module into the attention layers of the Transformer, based on an existing heavyweight tracker, to selectively activate specific layers. The activation module consists of a linear layer and a nonlinear activation function, producing an activation probability that determines whether a particular attention layer should be activated. Furthermore, to further reduce computational overhead, the Transformer backbone is divided into two stages: in the shallow stage, a hybrid attention mechanism with an asymmetric structure is employed for feature extraction and one-way information transfer from the template to the search region; in the deep stage, a cross-attention mechanism is used to model the feature interaction between the template and the search region.

In summary, the main contributions of this paper are as follows:

- A visual object tracking method with a variable number of Transformer layers is proposed. By leveraging an activation module, the method can adaptively utilize different numbers of attention layers, effectively avoiding unnecessary computation and significantly improving tracking efficiency.
- A novel feature extraction and fusion approach is introduced. In the shallow stage of the Transformer, it enables feature extraction and one-way information transfer from the template to the search region. In the deep stage, it focuses solely on feature interaction between the template and the search region, greatly reducing the computational cost of feature fusion.
- Extensive experiments and in-depth analysis are conducted on several challenging tracking benchmarks, demonstrating the effectiveness of the proposed PSTrack tracker.

2. Related Work

The Transformer model [2], which initially achieved great success in the field of natural language processing (NLP), features a unique self-attention mechanism and position-based encoding scheme that enables it to efficiently handle sequential data. Vision Transformer (ViT) [3] was the first to apply Transformer architecture to image classification tasks, demonstrating outstanding performance and introducing a pure Transformer-based approach for image modeling. Transformer Tracking [4] marked the first application of Transformers to visual object tracking by designing a cross-attention-based feature fusion module to replace traditional cross-correlation operations. Since then, numerous ViT-based tracking algorithms [5–11] have emerged. Although Transformers show tremendous potential in computer vision, they often overlook tracking speed. To address this, researchers have made significant efforts to improve these high-performance but heavyweight trackers. For instance, OSTrack [12] considers pruning irrelevant background features; SwimTrack [13] reduces computational cost by removing the last four layers; MixFormerV2 [14] adopts knowledge distillation to reduce the number of feature extraction layers from 12 to 8. SparseTT [15] introduces a sparse attention mechanism to reduce the computational burden of Transformers. ViT-STS [16] applies a fast sampling algorithm that learns sparse associations between tokens and supertokens to predict supertokens, thereby achieving lightweight tracking. Recent lightweight attention mechanisms have also shown promise in reducing computational overhead. For example, RTLinearFormer [17] proposes a lightweight linear attention mechanism for semantic segmentation, replacing

standard quadratic attention with linear complexity while maintaining representation capacity. Inspired by such efficiency-driven designs, our work integrates a hierarchical pruning strategy with adaptive attention selection, further pushing the efficiency-accuracy trade-off in object tracking.

Building upon previous research, this paper proposes an object tracking method based on a pruning strategy. A new activation function is introduced, and the backbone network is hierarchically structured to reduce computational cost. Experiments conducted on a wide range of public datasets demonstrate that the proposed pruning-based tracking method is well-suited for various tracking scenarios.

In the research on lightweight Transformer tracking, existing pruning-related methods can be categorized into three types: OSTrack minimizes ineffective computations by filtering foreground-related features but relies on manually designed filtering rules, limiting its adaptability; SimTrack directly reduces computational costs by deleting the last four layers of the Transformer but fails to account for the functional importance differences across layers; MixformerV2 compresses network layers through knowledge distillation, which retains core features but increases training complexity due to the distillation process. The hierarchical pruning strategy proposed in this paper achieves adaptive and refined pruning through the synergistic design of activation modules and hierarchical attention mechanisms. The innovation of this work lies in: eliminating the need for manual intervention in determining pruning locations and ratios, as the model dynamically adjusts computational resource allocation based on tracking scenarios (e.g., target deformation or background clutter); meanwhile, the customized design of the hierarchical attention mechanism tightly couples the pruning process with network functionality, preventing performance degradation caused by the mere pursuit of efficiency.

3. Method

3.1. Network Framework

As shown in Figure 1, the proposed PSTrack network architecture is illustrated. PSTrack adopts a Transformer-based single-stream tracking structure, where each attention layer in the ViT-based backbone is embedded with an activation module. This activation module is specially trained to adaptively determine whether to activate its corresponding Transformer attention layer, thereby optimizing the inference process. During feature extraction and relationship modeling between the template and the search region, different attention modules are used in the shallow and deep stages of the network. Specifically, the shallow stage employs specialized hybrid attention blocks, while the deep stage utilizes cross-attention blocks. The hybrid attention blocks in the shallow stage enable feature extraction and one-way information flow from the template to the search region, while the cross-attention blocks in the deep stage focus solely on feature interaction, omitting redundant self-attention computations. This hierarchical attention mechanism design allows the network to process information more efficiently at different stages. Finally, the features extracted and processed from the search region are fed into the prediction head for accurate target localization.

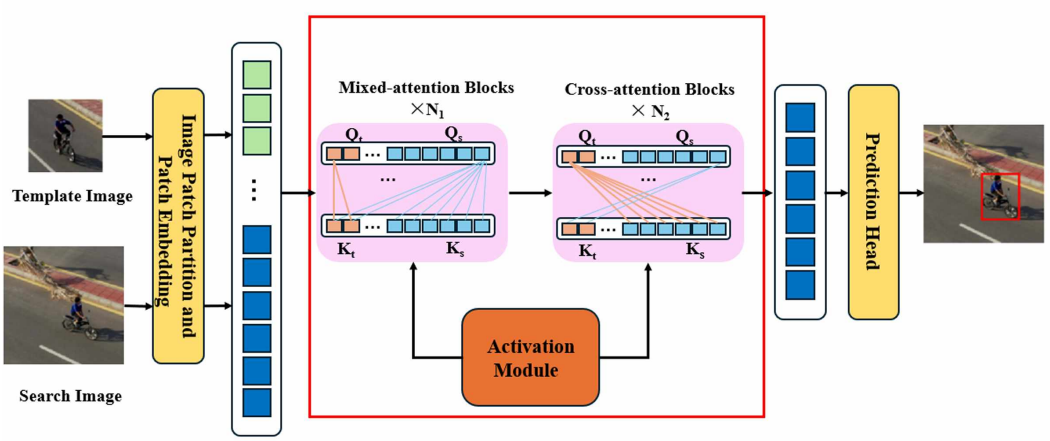

Figure 1. Network architecture of PSTrack. The regions marked by red boxes represent the novel hybrid attention modules proposed in this paper. Specifically, the hybrid attention blocks in the shallow stage enable feature extraction and one-way information flow from the template to the search region, while the cross-attention blocks in the deep stage focus solely on feature interaction, omitting redundant self-attention computations.

3.2. Activation Module

In Transformer-based visual object trackers, not all Transformer layers are equally important for every tracking task. The impact of semantic features and inter-feature relationships on tracking performance varies depending on target characteristics and scene complexity. In simple backgrounds, effective tracking can be achieved using basic features based on color contrast. However, in complex environments, support from higher-level semantic feature relationships becomes necessary. Therefore, to improve the tracking efficiency of ViT, this paper introduces activation modules to adaptively adjust the ViT architecture.

Considering efficiency, the input to the activation module is not the full information from all tokens of the template and search region, but rather a partial slice that contains key positional information. The output of the activation module is an activation probability, which determines whether the current attention layer should be activated. Specifically, for the i-th attention layer, let the total number of tokens be K, and the embedding dimension of each token be d. Then, the output tokens of the i-th layer are represented as $t_{1:K}^{i-1} \in R^{K\times d}$. Subsequently, by passing through a standard unit vector $e_1^T = [1, 0, ...0]$, the activation input $a^{i-1} = e_1^T t_{1:K}^{i-1} \in R^K$ is obtained. The vector a^{i-1} is derived from the first token embedding of the concatenated template and search region tokens. In ViT-based architectures, the first token typically corresponds to the token that aggregates global information from the entire sequence. This token is chosen as the activation input because it encapsulates high-level semantic and contextual features that are indicative of the complexity and discriminative content of the input data. By using the token, the activation module can efficiently assess whether deeper attention layers are necessary without processing all tokens, thus reducing computational overhead while preserving decision relevance. Finally, the activation probability is computed through a linear layer L^i and a sigmoid activation function: $\sigma(x) = 1/(1 + e^{-x})$

$$p^i = \sigma(L^i(a^{i-1})) \tag{1}$$

where p^i is activation probability (scalar, range [0, 1]), the linear layer L^i consists of a two-layer MLP, and σ is the sigmoid activation function. If the final computed activation probability $p^i > \tau$ (where τ is the activation probability threshold), the current attention layer is activated. Otherwise, the layer is skipped, and the output tokens from the $(i-1)$-th layer are directly passed to the $(i+1)$-th layer. It is important to note that if all attention layers in the Transformer are deactivated at the same time, the network will not perform

any feature extraction or relationship modeling between the template and the search region—an undesirable outcome. Moreover, the initial layers of a Transformer typically provide fundamental and crucial information, playing a vital role in constructing higher-level and more abstract features. Therefore, to prevent this issue, this paper enforces the activation of the first m attention layers, while the subsequent layers are selectively activated via the activation module.

Through this dynamic activation mechanism, the model can flexibly and adaptively determine the activation status of each attention layer in the Transformer based on the specific characteristics of the input data. This avoids uniform processing of all Transformer blocks during tracking, skips unnecessary computations, and improves computational efficiency. The activation mechanism is detailed in Algorithm 1, which controls the layer activation by probability threshold judgment.

Algorithm 1 Activation Module Algorithmic Procedure

Require: Tokens from the previous layer $t_{1:K}^{i-1}$

1: **Activation Input Extraction**: Compute a^{i-1}
2: **Probability Computation**: Pass through a two-layer MLP and Sigmoid: $p^i = \sigma(L^i(a^{i-1}))$
3: **Activation Decision**: If $p^i > \tau$ or layer index $i \leq m$, activate the layer; otherwise, skip.

Ensure: Tokens are either processed by the attention layer or passed through directly.

3.3. Feature Extraction and Relation Modeling

In the field of object tracking, feature extraction and relationship modeling are the core and key components for improving tracking performance. In earlier studies, trackers typically employed a pair of parameter-shared Siamese networks for feature extraction, followed by correlation operations between template and search features.

In recent years, with the rapid rise of Transformers in computer vision, trackers that utilize Transformers for feature extraction and interaction between the template and search region have demonstrated strong competitiveness. Among them, the single-stream tracking architecture represented by OSTrack has gained particular attention for its parallel processing of feature extraction and relationship modeling.

However, despite their impressive tracking accuracy, these models often suffer from large numbers of parameters and struggle to perform effectively in resource-constrained environments.

In earlier tracking algorithms, MixFormer introduced an asymmetric structure for relationship modeling. This design not only effectively reduced computational overhead but also eliminated interference from the search region to the template during tracking, significantly improving tracking accuracy. Inspired by this, the present work conducts an in-depth study of the feature extraction and relationship modeling process, aiming to explore more efficient backbone networks for fast object tracking tasks

Standard Attention Mechanism: In the attention module, the template image and the search image are denoted as t and s, respectively. Through linear transformations, we obtain Q_t, K_t, V_t (tokens from the template image) and Q_s, K_s, V_s (tokens from the search image). Let $Q = [Q_t; Q_s]$, $K = [K_t; K_s]$, $V = [V_t; V_s]$ represent the inputs to a particular attention layer. In existing single-stream tracking architectures, the attention computation process in the Transformer is as follows:

$$A(Q, K, V) = Softmax(\frac{QK^T}{\sqrt{d}})V \tag{2}$$

To further explore the relationship between feature extraction and interaction, this paper reformulates Equation (2) into Equation (3):

$$\begin{aligned} A([Q_t;Q_s],[K_t;K_s],[V_t;V_s]) &= Softmax(\frac{[Q_t;Q_s][K_t;K_s]^T}{\sqrt{d}})[V_t;V_s] \\ &= Softmax(\frac{\begin{bmatrix} Q_tK_t^T & Q_tK_s^T \\ Q_sK_t^T & Q_sK_s^T \end{bmatrix}}{\sqrt{d}})\begin{bmatrix} V_t \\ V_s \end{bmatrix} \end{aligned} \tag{3}$$

As shown in Equation (3), $Q_tK_t^T$ and $Q_sK_s^T$ belong to the self-attention module and serve the purpose of feature extraction, representing the self-attention within the template and search frames, respectively. In contrast, $Q_tK_s^T$ and $Q_sK_t^T$ are part of the cross-attention module, responsible for feature interaction—that is, modeling the relationship between the target template and the search frame.

Computational cost is one of the key factors affecting tracking speed. Since the search frame typically has a much larger image size than the template frame, the feature extraction module for the search frame contributes significantly to the attention module's overall computation. Therefore, reducing the computational load of this module is beneficial for lowering the number of parameters. Considering that the interaction between the template image and the search image is crucial for estimating the object's position in object tracking, and that as the network layers deepen, the importance of feature extraction gradually decreases while the importance of feature interaction increases. Based on this, in the first stage, the focus is primarily on feature extraction, supplemented by a certain degree of feature interaction. In the second stage, only feature interaction is retained, while the computationally intensive feature extraction for the template and search regions is omitted.

Specifically, as shown in Figure 1, the first $N1$ layers employ hybrid attention blocks for shallow feature processing. These use an asymmetric structure for feature extraction and enable one-way information flow from the template to the search region. The following $N2$ layers adopt cross-attention blocks for deep feature processing, where self-attention within the template and search regions is removed, and only the relationship between them is modeled. The two different attention computation methods are formulated in Equations (4) and (5).

$$A_{hyb} = Softmax(\frac{\begin{bmatrix} Q_tK_t^T & \\ Q_sK_t^T & Q_sK_s^T \end{bmatrix}}{\sqrt{d}})\begin{bmatrix} V_t \\ V_s \end{bmatrix} \tag{4}$$

$$A_{cro} = Softmax(\frac{\begin{bmatrix} & Q_tK_s^T \\ Q_sK_t^T & \end{bmatrix}}{\sqrt{d}})\begin{bmatrix} V_t \\ V_s \end{bmatrix} \tag{5}$$

To better illustrate the process of feature extraction and interaction, we provide a flowchart of feature extraction and relation modeling. As shown in Figure 2, the shallow layer $N1$ employs a hybrid attention mechanism to achieve unidirectional feature transfer from the template to the search region. The deep layer $N2$, on the other hand, models the feature interaction between the two solely through cross-attention. This structure corresponds to the different attention modules in the shallow and deep layers shown in Figure 1.

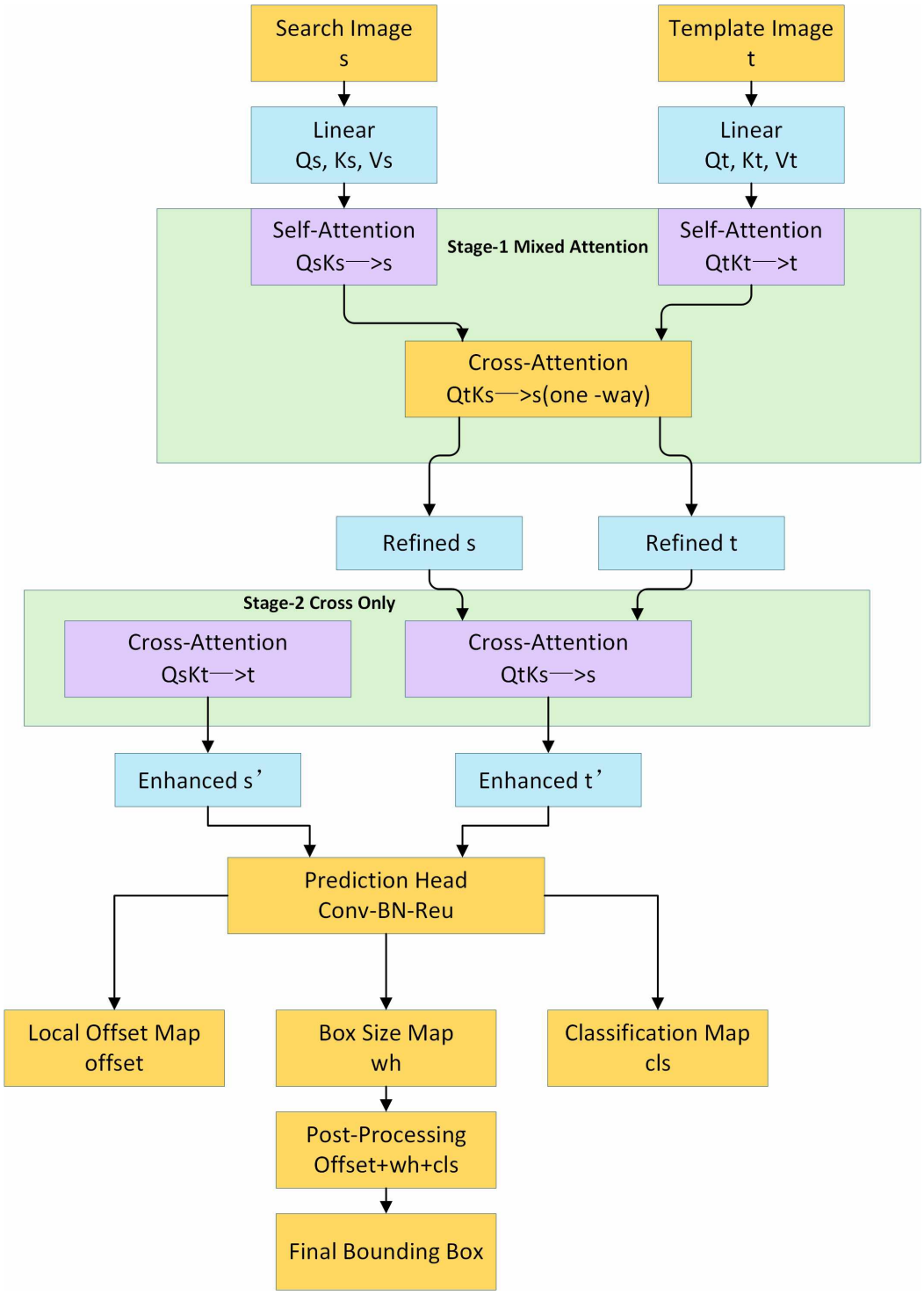

Figure 2. Flowchart of feature extraction and relation modeling in PSTrack.

3.4. Prediction Head and Loss Function

Prediction Head: The prediction head in this work adopts a structure similar to that of a corner-point detection head. It consists of several sequentially stacked Conv-BN-ReLU layers. During the prediction process, the output tokens associated with the search image are first reshaped into a 2D spatial feature map. This feature map is then fed into the prediction head. The prediction head outputs three key pieces of information: classification scores $p \in [0,1]^{2\times H_s W_s/p^2}$, bounding box sizes $s \in [0,1]^{2\times H_s W_s/p^2}$, and local offsets $o \in [0,1]^{2\times H_s W_s/p^2}$. With these outputs, the model can preliminarily estimate the target's location. Specifically, the location with the highest classification score is first

identified, which roughly indicates the most probable target region. Then, by combining the local offset and bounding box size information, the precise extent of the target is determined, yielding the final predicted bounding box.

Loss Function: The activation module determines whether each Transformer attention layer should be activated by computing an activation probability. However, there are two extreme cases that must be avoided. When all attention layers are deactivated, the model cannot compute the correlation between the template image and the search image. As a result, it fails to capture target information effectively, severely degrading tracking performance. Conversely, when all attention layers are activated, although it can reduce the classification loss and regression loss to some extent, making the model more accurate in predicting the target's category and location, it will inevitably significantly increase the computational load and reduce the model's operational efficiency. To address these issues, this paper introduces a sparsity loss function to prevent both extremes, thereby achieving a balance between computational efficiency and tracking performance. The sparsity loss is defined as shown in Equation (6):

$$L_{spa} = \left| \frac{1}{M-m} \sum_{i=m+1}^{M} p_i - \beta \right| \tag{6}$$

where M is the total number of attention layers, m is the number of forcibly activated layers (set to 2 in our implementation), and $\beta \in [0, 1]$ is the sparsity constant, controlling the desired activation ratio. The sparsity loss function penalizes high average activation probabilities across all adaptive layers, encouraging more attention layers to be deactivated on average. Specifically, when the mean activation probability is too high, the value of the sparsity loss increases. During training, this encourages the model to automatically adjust the activation probabilities, leading to more attention layers being turned off. As a result, unnecessary computations are reduced, and the overall inference efficiency is improved.

Additionally, this work adopts Focal Loss as the classification loss, and uses L_1 loss and L_{GIou} loss as the regression losses. The overall L can be expressed as follows:

$$L = L_{cls} + \eta_1 L_1 + \eta_2 L_{GIou} + \eta_3 L_{spa} \tag{7}$$

where, $\eta_1 = 5$, $\eta_2 = 2$ and $\eta_3 = 50$ are regularization parameters used to balance the weights of the classification loss, regression loss, and sparsity loss, respectively.

4. Experimental Results and Analysis

4.1. Experimental Setup

(1) Model Architecture and Training

In this study, ViT-Tiny is selected as the backbone network for the proposed tracker. The input template and search region images are cropped to resolutions of 128 × 128 and 256 × 256, respectively. The model is trained on the GOT-10k [18], LaSOT [19], TrackingNet [20], COCO [21],UAV123 [22] and NFS [23] datasets. To enhance the diversity of the training data, horizontal flipping and brightness jittering are applied as data augmentation techniques. AdamW [24] is used to optimize the network parameters, with an initial backbone learning rate of 4×10^{-5} and a weight decay of 1×10^{-4}. The model is trained for 300 epochs, with 60,000 image pairs randomly sampled per epoch. After 240 epochs, the learning rate is reduced by a factor of 10. As shown in Table 1, the network architecture configuration is presented.

Table 1. PSTrack network architecture configuration.

Component	Specification	Output Size
Backbone	ViT-Tiny	Patch size 16 × 16
Template input	128 × 128 × 3	8 × 8 × 192
Search region input	256 × 256 × 3	16 × 16 × 192
Shallow stage ($N1$)	Hybrid Attention Blocks ($N1 = 6$)	16 × 16 × 192
Deep stage ($N2$)	Cross Attention Blocks ($N2 = 6$)	16 × 16 ×192
Activation module	Two-layer MLP + Sigmoid	Scalar probability
Prediction head	Conv-BN-ReLU ×3 + Corner-point prediction	Classification + BBox
Total parameters	∼12.5 M	–

(2) Inference

During inference, the traditional Hanning window penalty strategy is adopted. Specifically, the classification score map is element-wise multiplied with a Hanning window of the same size to incorporate spatial prior information, thereby refining the classification results. Finally, the bounding box with the highest score is selected as the final tracking result.

4.2. Comparative Experiments

In this study, the proposed tracker PSTrack is comprehensively compared with state-of-the-art trackers on several benchmark datasets, including GOT-10k, LaSOT, TrackingNet, UAV123, and NFS. The comparative algorithms can be primarily categorized into two groups. The first consists of lightweight trackers implemented using traditional methods or convolutional neural networks, while the second comprises heavyweight Transformer-based trackers. Although the latter methods exhibit significantly better performance, they suffer from slower processing speeds. The detailed comparison results are presented in Tables 2 and 3. In Tables 2 and 3, the upper and lower parts of each table respectively display the performance of real-time and non-real-time trackers.

GOT-10k: GOT-10k is a large-scale and highly representative object tracking dataset, containing a total of 180 test sequences. As shown in Table 2, PSTrack demonstrates outstanding performance among real-time trackers, achieving the best score. Compared to the previously top-performing real-time tracker HCAT, PSTrack improves the score by a significant 3.6%. When compared with the non-real-time tracker OSTrack, PSTrack maintains comparable performance while offering a much faster inference speed, highlighting its efficiency and competitiveness in real-time tracking tasks.

LaSOT: LaSOT is a large-scale long-term object tracking dataset containing 1400 video sequences, with 280 sequences in the test set. As shown in Table 2, PSTrack demonstrates outstanding performance on the LaSOT dataset—achieving an AUC of 66.1%, a P_{norm} of 75.5%, and a P of 69.4%. These impressive results make PSTrack the current state-of-the-art (SOTA) among real-time trackers on LaSOT, strongly validating its stability and accuracy in long-term object tracking tasks.

TrackingNet. TrackingNet is a large-scale dataset derived from YouTube Bounding Boxes, comprising 511 video sequences in the test set. According to the results presented in Table 2, PSTrack demonstrates particularly outstanding performance in real-time tracking, achieving state-of-the-art results in all three key metrics—at 80.5%, at 85.9%, and at 79.1%. Moreover, when compared to non-real-time trackers such as SwinTrack and STARK, PSTrack achieves comparable performance, further highlighting its effectiveness.

Table 2. Comparison with State-of-the-Art Trackers on GOT-10k, LaSOT, and TrackingNet Benchmarks, Where *AO*, *SR*, *AUC*,P_{norm}, and *P* represent Average Overlap, Success Rate, Area Under Curve, Normalized Precision Rate, and Precision Rate, respectively; $SR_{0.5}$ denotes the success rate at a threshold of 0.5, and $SR_{0.75}$ denotes the success rate at a threshold of 0.75.

Method	GOT-10k			LaSOT			TrackingNet			Speed/FPS	
	AO	$SR_{0.5}$	$SR_{0.75}$	AUC	P_{norm}	P	AUC	P_{norm}	P	GPU	CPU
ECO [25]	31.6	30.9	11.1	32.4	33.8	30.1	55.4	61.8	49.2	240	15
ATOM [26]	55.6	63.4	40.2	51.5	57.6	50.5	70.3	77.1	64.8	83	18
LightTrack [27]	61.1	71.0	-	53.8	-	53.7	72.5	7.8	69.5	128	41
FEAR [28]	61.9	72.2	-	53.5	-	54.5	-	-	-	96	35
HCAT [29]	65.1	76.5	56.7	59.3	68.7	61.0	76.6	82.6	72.9	195	35
E.T.Track [30]	-	-	-	59.1	-	-	75.0	80.3	70.6	150	45
HiT [31]	64.0	72.1	58.1	64.6	73.3	68.1	80.0	84.4	77.3	175	33
PSTrack	68.7	78.3	63.8	66.1	75.5	69.4	80.5	85.9	79.1	206	48
TransT [4]	67.1	76.8	60.9	64.9	73.8	69.0	81.4	86.7	80.3	63	6
STARK [5]	68.8	78.1	64.1	67.1	77.0	-	82.0	86.9	-	50	7
SwinTrack [32]	71.3	81.9	64.5	67.2	-	70.8	81.1	-	78.4	96	16
OSTrack [12]	71.0	80.4	68.2	69.1	78.7	75.2	83.1	87.8	82.0	58	12
ARTrack [1]	73.5	82.2	70.9	70.4	79.5	76.6	84.2	88.7	83.5	28	4
OIFTrack [33]	74.6	85.6	71.9	69.6	79.5	75.4	84.1	89.0	82.8	36	9

To more intuitively demonstrate the effectiveness of the proposed PSTrack method, this paper also visualizes its tracking results on the LaSOT dataset alongside those of three other real-time trackers. As illustrated in Figure 3, the comparison involves tracking results in four distinct scenarios. The first two scenarios (airplane-1 and bird-7) represent simple cases with a clear distinction between the target and the background or a single object, while the latter two (coin-6 and giraffe-2) represent complex scenarios involving interference from multiple similar objects and occlusions. The results in the figure show that PSTrack achieves favorable tracking performance across all four sequences. This indicates that the proposed tracker not only ensures real-time performance during tracking but also significantly enhances tracking accuracy, enabling it to handle complex and varied real-world scenarios.

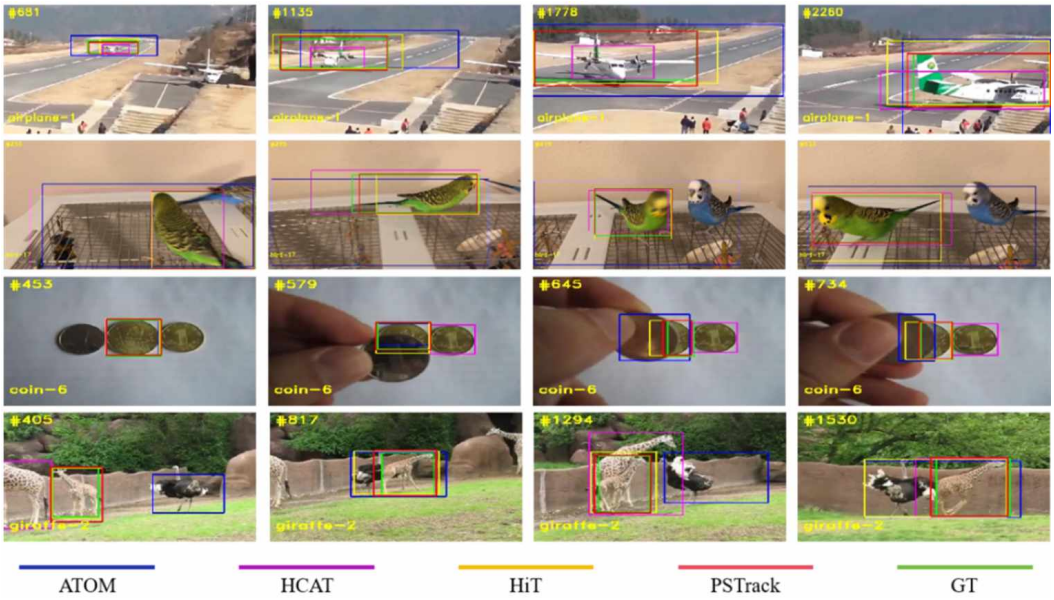

Figure 3. Representative experimental results of PRSTrack and the other four trackers: ATOM [26], HCAT [29], HiT [31], and GT. Note: Blue boxes represent the tracking results of the ATOM, purple boxes represent the tracking results of the HCAT, yellow boxes represent the tracking results of the HiT, red boxes represent the tracking results of the PRSTrack, and green boxes represent the ground truth.

NFS and UAV123: The NFS dataset consists of 100 video sequences featuring a wide range of fast-moving objects. UAV123 is a manually annotated dataset focusing on UAV-based tracking scenarios and contains 123 video sequences. As shown in Table 3, the proposed PSTrack achieves outstanding performance on both datasets. Specifically, it attains a score of 64.3 on the NFS dataset and 66.5 on the UAV123 dataset. Among all real-time trackers compared, PSTrack achieves the best performance on both benchmarks. These results demonstrate that PSTrack not only excels in general tracking tasks but is also highly effective and accurate in challenging scenarios such as high-speed motion and aerial tracking.

Table 3. Comparison with State-of-the-Art Trackers on UAV123 and NFS Benchmarks.

Method	UAV123	NFS
ECO [25]	53.2	46.6
ATOM [26]	64.2	58.4
LightTrack [27]	62.5	55.3
FEAR [28]	-	61.4
HCAT [29]	62.7	63.5
E.T.Track [30]	62.3	59.0
HiT [31]	65.6	63.6
PSTrack	66.5	64.3
TransT [4]	68.1	65.7
OSTrack [12]	70.7	66.5
ARTrack [1]	70.5	66.8
OIFTrack [33]	68.6	-

4.3. Ablation Study

In this section, we conduct detailed ablation studies and extensive analysis on the GOT-10k test dataset to verify the effectiveness of the proposed tracking method.

Effectiveness of Network Modules: To investigate the impact of the introduced activation module and the modified attention mechanism on tracking performance, we designed and carried out ablation experiments, as shown in Table 4. The results in Table 4 indicate that, although incorporating the activation module and adjusting the attention mechanism led to a slight performance drop, the inference speed on GPU was significantly improved, with gains of up to 38%. Notably, the activation module contributed most to this improvement, highlighting its critical role in optimizing the efficiency of the tracking network's inference process. Although some accuracy was sacrificed, the substantial boost in speed is particularly valuable in real-time applications, where processing efficiency is paramount.

Table 4. Ablation study on the effectiveness of network modules, Where *AO* and *SR* represent Average Overlap and Success Rate, respectively; $SR_{0.5}$ denotes the success rate at a threshold of 0.5, and $SR_{0.75}$ denotes the success rate at a threshold of 0.75. Note: ✓ indicates that the corresponding module is included in the model, and × indicates that the module is ablated.

Activation Module	Improved Attention Module	*AO*	$SR_{0.5}$	$SR_{0.75}$	Speed/FPS (GPU)	Speed/FPS (CPU)
×	×	70.2	79.7	66.5	156	36
✓	×	69.3	78.9	64.7	193	43
×	✓	69.5	79.1	65.3	186	40
✓	✓	68.7	78.3	63.8	216	48

As shown in Table 5, after incorporating the activation module and sparsity parameter to enhance the tracking network, although the tracker's performance has experienced a

slight decline, its execution speed on the GPU has been substantially improved. This is of significant importance in practical applications.

Table 5. Ablation study on the activation module and sparsity parameter, Where *AO* and *AUC* represent Average Overlap and Area Under Curve. Note: ✓ indicates that the corresponding module is included in the model, and × indicates that the module is ablated.

Activation Module	Sparsity Parameter β	GOT-10k *AO*	LaSOT AUC	UAV123 AUC	Speed/FPS (GPU)	Speed/FPS (CPU)
×	×	69.5	66.7	67.3	186	40
✓	✓	68.7	66.1	66.5	216	48

Number of Forced Activation Layers: In the design of the activation module, the initial attention layers must be forcibly activated to ensure overall model performance; otherwise, the tracking effectiveness may be adversely affected. To determine an appropriate number of such forced activation layers, we conducted targeted experiments, with the specific results shown in Table 6. As the number of forcibly activated layers mmm gradually increased, the tracker showed slight improvements in metrics such as *AO*, $SR_{0.5}$, and $SR_{0.75}$. However, it is important to note that this also led to a significant drop in FPS. Considering the trade-off between tracking accuracy and runtime speed, we ultimately set the number of forced activation layers to 2. This configuration ensures reliable model performance without excessively compromising tracking efficiency.

Table 6. Ablation study on the number of forced activated layers, Where *AO* and *SR*represent Average Overlap and Success Rate, respectively; $SR_{0.5}$ denotes the success rate at a threshold of 0.5, and $SR_{0.75}$ denotes the success rate at a threshold of 0.75.

m	*AO*	$SR_{0.5}$	$SR_{0.75}$	Speed/FPS
2	68.7	78.3	63.8	216
3	68.9	78.4	64.1	208
4	69.3	78.9	64.6	192

Attention Layer Ratio: To verify the importance of feature extraction and relationship modeling in visual object tracking, we designed an experiment to investigate the impact of different layer ratios between Mixed Attention Blocks and Cross Attention Blocks on tracking performance. Specifically, we set the ratio of mixed attention layers to cross attention layers $N1$:$N2$ to three configurations: 3:9, 6:6, and 9:3. As shown in Table 7, the tracker achieved the best performance when the ratio was 6:6. This result indicates that a balanced configuration between feature extraction (via mixed attention) and interaction modeling (via cross attention) enables the tracker to fully leverage both mechanisms, thereby achieving optimal tracking accuracy.

Table 7. Ablation study on the ratio of attention layers, Where *AO* and *SR*represent Average Overlap and Success Rate, respectively; $SR_{0.5}$ denotes the success rate at a threshold of 0.5, and $SR_{0.75}$ denotes the success rate at a threshold of 0.75.

N1:*N2*	*AO*	$SR_{0.5}$	$SR_{0.75}$
3:9	66.5	76.7	60.3
6:6	68.7	78.3	63.8
9:3	68.4	77.8	64.2

Selection of the Sparsity Parameter: In the sparse loss function of this method, by penalizing the deviation between the mean activation probability and β, the model is forced

to deactivate most blocks on average to reduce computational cost. As shown in Table 8, this approach avoids excessive sparsity, where an overly small β would lead to insufficient activation layers and inadequate feature extraction, as well as insufficient sparsity, where an overly large β would result in too many activated layers, thereby limiting efficiency gains. Ultimately, in this paper, β is set to 0.2.

Table 8. Ablation study on the sparsity parameter β, Where AO and SRrepresent Average Overlap and Success Rate, respectively; $SR_{0.5}$ denotes the success rate at a threshold of 0.5, and $SR_{0.75}$ denotes the success rate at a threshold of 0.75.

β	AO	$SR_{0.5}$	$SR_{0.75}$	Speed/FPS (GPU)	Speed/FPS (CPU)
0.1	65.9	74.5	60.8	240	54
0.2	68.7	78.3	63.8	216	48
0.3	69.2	78.9	64.7	189	40

5. Conclusions

In this work, we propose a novel object tracking network named PSTrack, which achieves a balance between tracking performance and inference speed. First, an activation module is introduced to adaptively and selectively activate the attention layers in the Transformer. This allows the model to compute only the necessary attention layers during runtime, thereby avoiding redundant computation. Additionally, we modify the standard attention mechanism in the backbone network: mixed attention is employed in the shallow layers for feature extraction and one-way information transfer from the template to the search region, while cross-attention is applied in the deeper layers solely for feature interaction, omitting unnecessary feature extraction and further reducing the computational load. Comprehensive experiments and ablation studies across multiple datasets demonstrate that the proposed tracking method offers superior inference speed without compromising tracking accuracy. To further bridge the accuracy gap with heavier non-real-time SOTA trackers, we plan to integrate the proposed pruning strategy with richer template representations and more sophisticated prediction modules. Additionally, we will explore dynamic early-layer activation mechanisms to reduce reliance on fixed m and enhance adaptability across diverse tracking scenarios.

Author Contributions: Conceptualization, P.G. and D.H.; methodology, P.G.; software, P.G.; validation, P.G. and D.H.; formal analysis, P.G.; investigation, P.G.; resources, P.G. and H.L.; data curation, H.L. and X.L.; writing—original draft preparation, P.G.; writing—review and editing, P.G. and D.H.; visualization, H.L. and X.L.; supervision, D.H.; project administration, P.G.; funding acquisition, D.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China under Grant 61901183, the Fundamental Research Funds for the Central Universities under Grant ZQN-921, the Natural Science Foundation of Fujian Province under Grant 2023J01140, the Collaborative Innovation Platform Project of Fujian Province under Grant 2021FX03, and the Key Project of Quanzhou Science and Technology Plan under Grant 2023C007R.

Data Availability Statement: The data presented in this study are available upon request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Wei, X.; Bai, Y.; Zheng, Y.; Shi, D.; Gong, Y. Autoregressive visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Vancouver, BC, Canada, 18–22 June 2023; pp. 9697–9706. [CrossRef]

2. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Adv. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008. [CrossRef]
3. Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv* **2020**, arXiv:2010.11929. [CrossRef]
4. Chen, X.; Yan, B.; Zhu, J.; Wang, D.; Yang, X.; Lu, H. Transformer tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 8126–8135. [CrossRef]
5. Yan, B.; Peng, H.; Fu, J.; Wang, D.; Lu, H. Learning spatio-temporal transformer for visual tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 10448–10457. [CrossRef]
6. Zhao, M.; Okada, K.; Inaba, M. TrTr: Visual tracking with transformer. *arXiv* **2021**, arXiv:2105.03817. [CrossRef]
7. Yu, B.; Tang, M.; Zheng, L.; Zhu, G.; Wang, J.; Feng, H.; Feng, X.; Lu, H. High-performance discriminative tracking with transformers. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 9856–9865. [CrossRef]
8. Cao, Z.; Fu, C.; Ye, J.; Li, B.; Li, Y. HiFT: Hierarchical feature transformer for aerial tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 15457–15466. [CrossRef]
9. Mayer, C.; Danelljan, M.; Bhat, G.; Paul, M.; Paudel, D.P.; Yu, F.; Van Gool, L. Transforming model prediction for tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 8731–8740. [CrossRef]
10. Xie, F.; Wang, C.; Wang, G.; Yang, W.; Zeng, W. Learning tracking representations via dual-branch fully transformer networks. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 2688–2697. [CrossRef]
11. Chen, B.; Li, P.; Bai, L.; Qiao, L.; Shen, Q.; Li, B.; Gan, W.; Wu, W.; Ouyang, W. Backbone is all your need: A simplified architecture for visual object tracking. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 375–392. [CrossRef]
12. Ye, B.; Chang, H.; Ma, B.; Shan, S.; Chen, X. Joint feature learning and relation modeling for tracking: A one-stream framework. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 341–357. [CrossRef]
13. Chen, B.; Li, P.; Bai, L.; Qiao, L.; Shen, Q.; Li, B.; Gan, W.; Wu, W.; Ouyang, W. Backbone is All Your Need: A Simplified Architecture for Visual Object Tracking. *arXiv* **2022**, arXiv:2203.05328. [CrossRef]
14. Cui, Y.; Jiang, C.; Wang, L.; Wu, G. Mixformer: End-to-end tracking with iterative mixed attention. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 13608–13618. [CrossRef]
15. Fu, Z.; Fu, Z.; Liu, Q.; Cai, W.; Wang, Y. SparseTT: Visual tracking with sparse transformers. *arXiv* **2022**, arXiv:2205.03776. [CrossRef]
16. Huang, H.; Zhou, X.; Cao, J.; He, R.; Tan, T. Vision transformer with super token sampling. *arXiv* **2022**, arXiv:2211.11167. [CrossRef]
17. Gu, Y.; Fu, C.; Song, W.; Wang, X.; Chen, J. RTLinearFormer: Semantic segmentation with lightweight linear attentions. *Neurocomputing* **2025**, *625*, 129489. [CrossRef]
18. Huang, L.; Zhao, X.; Huang, K. Got-10k: A large high-diversity benchmark for generic object tracking in the wild. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *43*, 1562–1577. [CrossRef] [PubMed]
19. Fan, H.; Lin, L.; Yang, F.; Chu, P.; Deng, G.; Yu, S.; Bai, H.; Xu, Y.; Liao, C.; Ling, H. Lasot: A high-quality benchmark for large-scale single object tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June 2019; pp. 5374–5383. [CrossRef]
20. Müller, M.; Bibi, A.; Giancola, S.; Al-Subaihi, S.; Ghanem, B. TrackingNet: A large-scale dataset and benchmark for object tracking in the wild. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Cham, Switzerlan, 2018; pp. 300–316. [CrossRef]
21. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; pp. 740–755. [CrossRef]
22. Müller, M.; Smith, N.; Ghanem, B. A benchmark and simulator for UAV tracking. In *Proceedings of the European Conference on Computer Vision (ECCV)*; Springer: Cham, Switzerland, 2016; pp. 445–461. [CrossRef]
23. Kiani Galoogahi, H.; Fagg, A.; Huang, C.; Ramanan, D.; Lucey, S. Need for speed: A benchmark for higher frame rate object tracking. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1125–1134. [CrossRef]
24. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv* **2017**, arXiv:1711.05101. [CrossRef]

25. Danelljan, M.; Bhat, G.; Shahbaz Khan, F.; Felsberg, M. Eco: Efficient convolution operators for tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 6638–6646. [CrossRef]
26. Danelljan, M.; Bhat, G.; Khan, F.S.; Felsberg, M. Atom: Accurate tracking by overlap maximization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–20 June; pp. 4660–4669. [CrossRef]
27. Yan, B.; Peng, H.; Wu, K.; Wang, D.; Fu, J.; Lu, H. Lighttrack: Finding lightweight neural networks for object tracking via one-shot architecture search. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 19–25 June 2021; pp. 15180–15189. [CrossRef]
28. Borsuk, V.; Vei, R.; Kupyn, O.; Martyniuk, T.; Krashenyi, I.; Matas, J. FEAR: Fast, efficient, accurate and robust visual tracker. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 644–663. [CrossRef]
29. Chen, X.; Kang, B.; Wang, D.; Li, D.; Lu, H. Efficient visual tracking via hierarchical cross-attention transformer. In Proceedings of the European Conference on Computer Vision, Tel Aviv, Israel, 23–27 October 2022; Springer: Berlin/Heidelberg, Germany, 2022; pp. 461–477. [CrossRef]
30. Blatter, P.; Kanakis, M.; Danelljan, M.; Van Gool, L. Efficient visual tracking with exemplar transformers. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, Waikoloa, HI, USA, 3–7 January 2023; pp. 1571–1581. [CrossRef]
31. Kang, B.; Chen, X.; Wang, D.; Peng, H.; Lu, H. Exploring lightweight hierarchical vision transformers for efficient visual tracking. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 4–6 October 2023; pp. 9612–9621. [CrossRef]
32. Lin, L.; Fan, H.; Zhang, Z.; Xu, Y.; Ling, H. Swintrack: A simple and strong baseline for transformer tracking. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 16743–16754. [CrossRef]
33. Kugarajeevan, J.; Kokul, T.; Ramanan, A.; Fernando, S. Optimized Information Flow for Transformer Tracking. *arXiv* **2024**, arXiv:2402.08195. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.