remote sensing

MDPI

Article

SiamRhic: Improved Cross-Correlation and Ranking Head-Based Siamese Network for Object Tracking in Remote Sensing Videos

Afeng Yang [1], Zhuolin Yang [1,*] and Wenqing Feng [2]

[1] School of Communication Engineering, Hangzhou Dianzi University, Hangzhou 310018, China
[2] School of Computer Science, Hangzhou Dianzi University, Hangzhou 310018, China; wq_feng@hdu.edu.cn
* Correspondence: andrewlin@hdu.edu.cn

Abstract: Object tracking in remote sensing videos is a challenging task in computer vision. Recent advances in deep learning have sparked significant interest in tracking algorithms based on Siamese neural networks. However, many existing algorithms fail to deliver satisfactory performance in complex scenarios due to challenging conditions and limited computational resources. Thus, enhancing tracking efficiency and improving algorithm responsiveness in complex scenarios are crucial. To address tracking drift caused by similar objects and background interference in remote sensing image tracking, we propose an enhanced Siamese network based on the SiamRhic architecture, incorporating a cross-correlation and ranking head for improved object tracking. We first use convolutional neural networks for feature extraction and integrate the CBAM (Convolutional Block Attention Module) to enhance the tracker's representational capacity, allowing it to focus more effectively on the objects. Additionally, we replace the original depth-wise cross-correlation operation with asymmetric convolution, enhancing both speed and performance. We also introduce a ranking loss to reduce the classification confidence of interference objects, addressing the mismatch between classification and regression. We validate the proposed algorithm through experiments on the OTB100, UAV123, and OOTB remote sensing datasets. Specifically, SiamRhic achieves success, normalized precision, and precision rates of 0.533, 0.786, and 0.812, respectively, on the OOTB benchmark. The OTB100 benchmark achieves a success rate of 0.670 and a precision rate of 0.892. Similarly, in the UAV123 benchmark, SiamRhic achieves a success rate of 0.621 and a precision rate of 0.823. These results demonstrate the algorithm's high precision and success rates, highlighting its practical value.

Keywords: object tracking; remote sensing images; Siamese network; attention mechanism

check for updates

Citation: Yang, A.; Yang, Z.; Feng, W. SiamRhic: Improved Cross-Correlation and Ranking Head-Based Siamese Network for Object Tracking in Remote Sensing Videos. *Remote Sens.* **2024**, *16*, 4549. https://doi.org/10.3390/rs16234549

Academic Editor: Pedro Melo-Pinto

Received: 21 October 2024
Revised: 29 November 2024
Accepted: 2 December 2024
Published: 4 December 2024

Copyright: © 2024 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 (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

1.1. Object Tracking in Traditional Scenarios

A key aspect of object tracking is accurately identifying the object in the first frame of a video sequence. The tracker must accurately locate the object in subsequent frames and assess its motion, providing crucial data for various applications.

In recent decades, object tracking has evolved from classical methods like mean shift, particle filters, and Kalman filters to modern techniques such as correlation filters and deep learning. Correlation filter-based methods [1–6] are recognized for their fast processing speeds, thus enabling real-time tracking. However, these methods still lack robustness in complex scenarios, such as variations in object aspect ratios, low resolution, and occlusions. Conversely, with advancements in information technology and computational performance, deep learning techniques have gained widespread application in object tracking tasks. However, existing algorithms are not specifically tailored for remote sensing images, leading to suboptimal tracking performance in complex scenarios.

Deep learning-based object tracking methods have gained prominence due to their strong feature representation capabilities [7–18]. The ability to learn features offline is crucial for deep learning models, enabling them to capture complex relationships from

large labeled datasets. Siamese network-based methods are particularly favored for object tracking due to their high precision and the advantages of end-to-end training. Siamese networks regard object tracking as a similarity learning problem, enabling end-to-end offline training to assess the similarity between the object and its search area. As a result, many tracking algorithms leveraging Siamese networks have been developed.

1.2. Object Tracking in Remote Sensing Videos

Object tracking in optical remote sensing videos is a major challenge in computer vision and is crucial for analyzing satellite and aerial imagery. The overhead perspective of remote sensing satellites captures extensive scene information, which is valuable for applications such as large-scale traffic monitoring, national security surveillance, and military operational guidance [19]. However, challenges such as small object sizes, severe occlusion, complex backgrounds, and low object resolution must be addressed. Thus, developing a precise and robust object tracking algorithm that meets the real-time requirements of remote sensing video tracking remains a critical challenge. Standard object tracking algorithms often struggle with remote sensing videos due to complex backgrounds. Accurately and swiftly identifying objects in remote sensing images is challenging due to the prevalence of small objects and varied backgrounds, which differ significantly from natural images. Developing a robust tracking model that can handle intricate backgrounds in remote sensing videos is essential.

To address these challenges, we propose an enhanced Siamese network for object tracking in remote sensing videos, incorporating a cross-correlation and ranking head based on the SiamRhic architecture. By incorporating the CBAM (Convolutional Block Attention Module) attention mechanism, the tracker can focus more on the object and effectively ignore irrelevant backgrounds, thereby enhancing the representational power of the convolutional features. The use of asymmetric convolution effectively mitigates interference from distracting objects when applying the depth-wise cross-correlation algorithm in the SiamRPN++ [18] network for remote sensing images. Finally, to address high confidence in distracting items and the mismatch between classification and regression, we introduce a ranking loss integrated with the original loss function to optimize network training. These improvements enhance the network's performance without introducing significant resource overhead.

The main contributions of this paper can be summarized as follows:

1. We use an enhanced version of the ResNet50 architecture as the backbone network. Additionally, we incorporate the CBAM attention mechanism between the third, fourth, and fifth convolutional layers of the feature extraction network. Experiments show that this enhances the representational capacity of the convolutional features effectively.
2. We implement asymmetric convolution to replace the original depth-wise cross-correlation algorithm, decomposing the large convolution process into two separate convolutions. This approach eliminates the need for sliding window operations and feature map concatenation during each iteration, improving both speed and performance.
3. A ranking loss is introduced to extend the original loss function of the Siamese network. The classification ranking loss ensures that positive samples are prioritized over difficult negative samples, helping to reduce classification confidence scores for distracting objects. The IoU ranking loss is introduced to address the discrepancies between classification and regression.

2. Related Works

In recent years, extensive research on object tracking technology has led to the development of numerous algorithms, primarily categorized into correlation filter-based methods and deep learning-based approaches.

2.1. Correlation Filter-Based Algorithms

In 2010, correlation filtering was applied to tracking tasks, exemplified by the Minimum Output Sum of Squared Error (MOSSE) [1] filter, which utilizes the Fast Fourier Transform (FFT). This method replaces convolution operations in the time domain with simple operations in the frequency domain, achieving an impressive speed of 600 Frames Per Second (FPS) while maintaining high tracking precision. The Circulant Structure Kernel (CSK) [2] uses matrix circular shifts for dense sampling during training, achieving high performance while leveraging diagonalization to enhance speed. Henriques et al. extended the concept of circular matrix sampling to propose the Kernelized Correlation Filter (KCF) [3]. KCF uses a multi-channel Histogram of Oriented Gradient (HOG) [4] features and introduces a kernel function, further enhancing tracking performance. The DSST [5] algorithm addresses object scale variation by employing multi-scale predictions, using filters of different scales to regress multi-scale bounding boxes and selecting the most suitable one for output. The LCT [6] algorithm adopts a detection-based strategy, separating the task into motion and scale estimation, effectively addressing challenges such as occlusion and object variation in long-term tracking. However, these methods require fine-tuning the network models for different tracking tasks, complicating the achievement of both precision and generalization.

2.2. Deep Learning-Based Algorithms

2.2.1. Anchor-Based Algorithms

In recent years, the rapid development of convolutional neural networks and their powerful feature extraction capabilities have greatly advanced object tracking. SiamFC [7] was the first algorithm to use Siamese networks for object tracking, framing the tracking challenge as a similarity computation problem. Subsequently, many tracking algorithms based on Siamese networks have been developed, building upon the foundation laid by SiamFC. To enable the network to generate bounding boxes closer to the object, SiamRPN [8] integrates the Region Proposal Network (RPN) [9], commonly used in object detection, into the Siamese network tracking framework. SiamRPN features both a classification branch and a regression branch. The classification branch uses anchor boxes of various scales to fit objects of different sizes, while the regression branch refines the positions of the object bounding boxes. This approach enables the tracking network to localize the object more accurately. The Disturbance-Aware Siamese Region Proposal Network (DaSiamRPN) [10] addresses data imbalance by incorporating widely used object detection datasets during training to augment positive samples, thereby improving the tracker's generalization performance.

2.2.2. Anchor-Free Algorithms

Despite achieving promising results with algorithms like SiamFC and SiamRPN, anchor-based trackers, while effectively addressing object scale variation, still require pre-defined hyperparameters for anchor boxes. These anchors must be manually fine-tuned based on the dataset characteristics, limiting their flexibility, particularly in scenarios with significant variations in object sizes. This rigidity can lead to poor object fitting and increased susceptibility to surrounding objects, causing tracking box drift or even complete tracking failure. Inspired by single-stage fully convolutional object tracking algorithms like FCOS [11], Xu et al. proposed SiamFC++ [12], incorporating the anchor-free approach into object tracking. SiamFC++ is the first algorithm to apply the anchor-free approach to object tracking. SiamFC++ draws inspiration from the FCOS architecture and introduces a quality assessment branch, enhancing its design. Another class of anchor-free object tracking algorithms is based on Zhou et al.'s CenterNet [13], which reformulates object detection as a key-point prediction problem. SiamKPN [14] integrates cascading with the CenterNet algorithm for object tracking. A cascading heatmap strategy with fixed variance progressively focuses on the object, while a strategy with reduced variance enhances object-distractor discrimination, demonstrating robust anti-interference capabilities.

Deep learning methods have recently achieved significant breakthroughs in traditional object tracking tasks in natural scenes. However, progress in applying these methods to remote sensing video tracking has been slower. In 2020, Hu et al. [15] introduced deep learning to remote sensing video tracking, combining it with optical flow [16] methods to process remote sensing image features. They use a pre-trained feature extraction network to separately extract appearance features from optical images and motion features from optical flow images. Motion features obtained through optical flow effectively distinguish the object from similar objects in complex backgrounds, improving tracking performance. In 2021, Feng et al. [17] applied the classic SiamRPN++ [18] algorithm from Siamese networks to remote sensing video tracking, combining it with a clustering-based frame difference method. They used a clustering method to segment the differential images between adjacent frames. Based on the clustering results, they refined the differential images and fused them with the original frame. This fusion served as input to the SiamRPN++ algorithm, enhancing the tracking network's ability to identify objects in remote sensing images. Although these algorithms have been applied to remote sensing video tracking, they are not specifically adapted to the unique characteristics of remote sensing images. As a result, their performance in handling complex scenarios in remote sensing images still needs improvement. The key distinction of this method is that traditional Siamese networks use depth-wise cross-correlation to combine features from the two branches, ensuring computational independence. In contrast, we use efficient parameter learning to strengthen the correlation and better explore the interrelation between the classification and regression branches.

3. Methods

This section outlines the components of SiamRhic, including CBAM attention, asymmetric convolution, and ranking loss. The SiamRhic framework is illustrated in Figure 1.

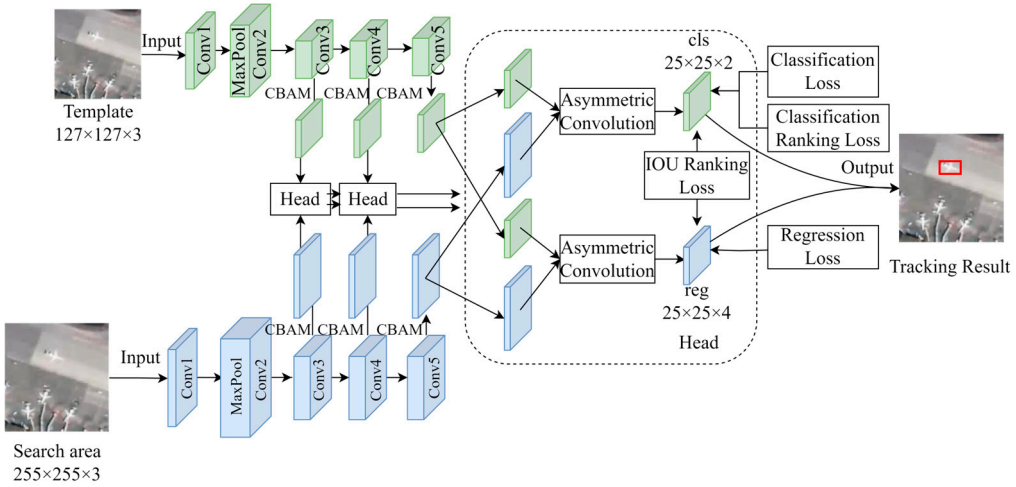


Figure 1. The network architecture starts by taking a template image and a search image. It then extracts deep features using an enhanced ResNet50 network with a weighted attention mechanism. The CBAM attention mechanism is incorporated between the third, fourth, and fifth convolutional layers of the feature extraction network. These features are then input into an adaptive head network for cross-correlation and multi-layer feature fusion. Finally, ranking loss is applied to suppress the classification confidence scores of interfering items and reduce the mismatch between classification and regression.

3.1. Overview

During model construction, an enhanced ResNet50 was used as the backbone, with down-sampling operations removed from the last two convolutional blocks to improve feature resolution. To enlarge the receptive field, dilated convolutions with dilation rates of

2 and 4 were applied to the fourth and fifth convolutional blocks, respectively. The template and search images are initially input, with the template image sized at 127 × 127 × 3 and the search image at 255 × 255 × 3. These images undergo feature extraction using the enhanced ResNet50 network, integrated with the CBAM attention mechanism. This module effectively leverages channel and spatial information from feature maps, highlighting informative channels and spatial features to enhance the overall representation capability.

Next, features from the third, fourth, and fifth convolutional blocks are extracted for subsequent multi-level feature aggregation. To minimize computational demand, the output feature maps from the third, fourth, and fifth convolutional blocks are resized by a neck network before being fed into the head network. The head network processes feature maps from both the template and search branches, using asymmetric convolutions to generate classification and regression maps. The classification feature map consists of two channels, representing the probability of foreground versus background for each pixel. In contrast, the regression feature map consists of four channels, representing the position coefficients of the object bounding box relative to the object's center. The classification and regression processes are performed three times, and applied to the feature maps from the three distinct convolutional blocks.

Finally, through multi-level feature aggregation, the classification and regression results are weighted and combined, producing the weighted sums for both classification and regression for each predicted box. The bounding box with the highest score is chosen as the final prediction for the frame.

3.2. Attention Mechanisms

The attention mechanism selectively focuses on important information while disregarding irrelevant data, aiming to extract key features. This technology is widely applied in deep learning domains such as computer vision, natural language processing, and speech recognition, driving the development of efficient algorithms. This study adopts the CBAM [20] attention mechanism, which considers both spatial information and learned correlations between channels to enhance feature extraction. This attention mechanism generates attention feature maps sequentially across both channel and spatial dimensions. The CBAM attention mechanism consists of spatial and channel attention components, as shown in Figure 2.

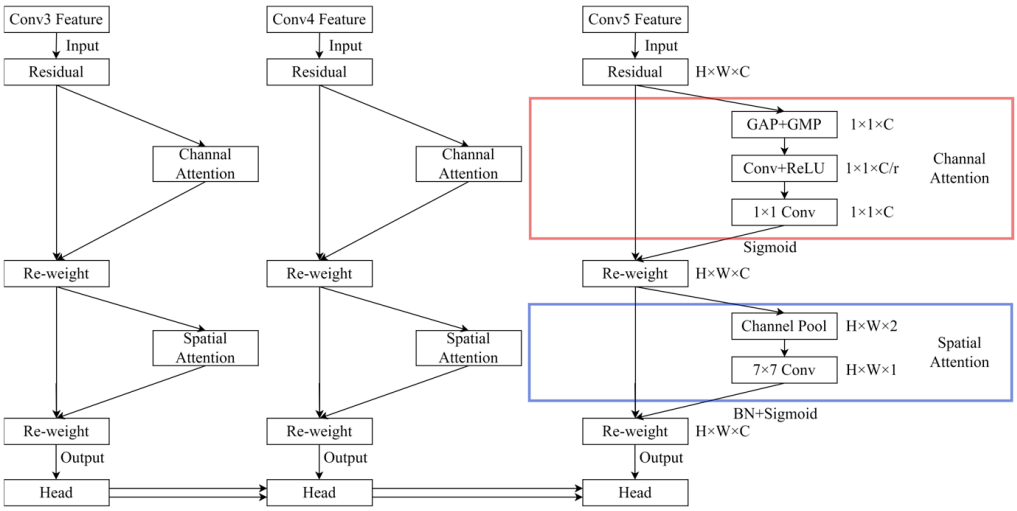


Figure 2. Attention mechanism. Feature maps from the third, fourth, and fifth convolutional blocks are processed through both channel and spatial attention mechanisms before being sent to the head network. The red box represents the channel attention mechanism, while the blue box represents the spatial attention mechanism.

The channel attention mechanism is initially applied to the input feature map. As shown in Figure 3, the feature map undergoes global average pooling (GAP) and global max pooling (GMP) operations, considering its width and height. Next, channel attention weights are computed using a multilayer perceptron (MLP) and normalized with the Sigmoid function. The formula is shown below.

$$\begin{aligned} M_c(F) &= \sigma(MLP(AvgPool(F)) + MLP(MaxPool(F))) \\ &= \sigma\left(W_1\left(W_0\left(F^c_{avg}\right)\right) + W_1(W_0(F^c_{\max}))\right), \end{aligned} \tag{1}$$

where F denotes the input feature map, σ denotes the Sigmoid function, MLP denotes the fully connected layer, and $AvgPool$ and $MaxPool$ denote the average pooling and maximum pooling operations, respectively, W_0 and W_1 denote the fully connected layer weights, F^c_{avg} denotes the average pooled features, and $F^c_{\max}$ denotes the maximum pooled features.

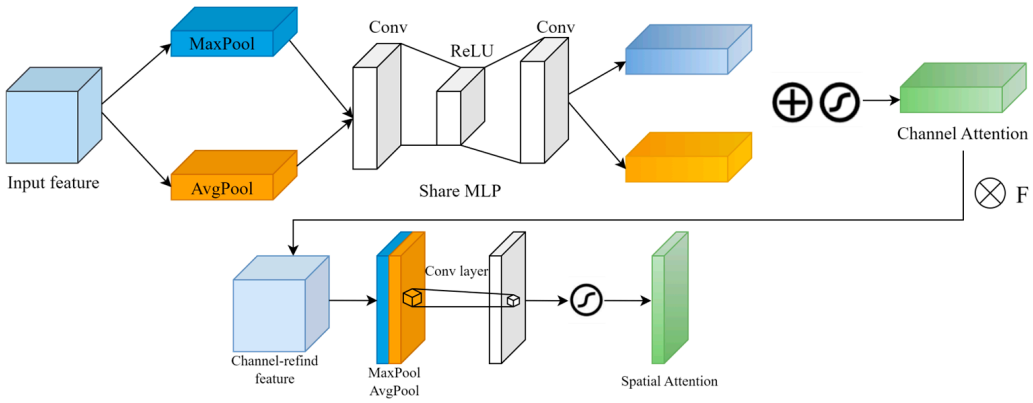


Figure 3. Channel attention module (CAM) and spatial attention module (SAM).

Similarly, to apply the attention mechanism in the spatial dimension, as shown in Figure 3, the output feature map after channel attention undergoes global average pooling and global max pooling, reducing its width and height to 1×1. A 7×7 convolutional kernel with ReLU activation is then applied to further reduce the dimensionality of the feature map. The feature map dimensions are then restored to their original size through a convolution operation. Finally, the feature map is normalized using the Sigmoid activation function. The formula is as follows:

$$\begin{aligned} M_c(F) &= \sigma\left(f^{7\times7}([AvgPool(F); MaxPool(F)])\right) \\ &= \sigma\left(f^{7\times7}\left(\left[F^c_{avg}; F^c_{\max}\right]\right)\right). \end{aligned} \tag{2}$$

where F denotes the input feature map, σ denotes the Sigmoid function, $f^{7\times7}$ denotes the convolution operation with a filter size of 7×7, $AvgPool$ and $MaxPool$ denote the average pooling and maximum pooling operations, respectively, F^c_{avg} denotes the average pooled features, and $F^c_{\max}$ denotes the maximum pooled features. Ultimately, the attention weights are applied to the original input feature maps through channel-wise multiplication, enabling re-calibration of channel attention within the feature maps. The feature maps from spatial and channel attention are combined and recalibrated on the original feature map by applying weights in both dimensions, yielding attention features in the spatial dimension.

3.3. Asymmetric Convolution

We replace the original depth-wise cross-correlation in the Siamese network with asymmetric convolutions. The depth-wise cross-correlation is a parameter-less, manually crafted module that cannot fully exploit the benefits of large-scale offline learning. Additionally, the original depth-wise cross-correlation is susceptible to interference from distracting objects, activates fewer channels, and suppresses diverse correlated feature

channels, leading to weaker object boundary resolution. To address the issue of consistent feature map sizes for concatenation, asymmetric convolutions decompose the convolution operation on concatenated feature maps into two equivalent operations.

The depth-wise cross-correlation divides the search image features into n sub-regions, each matching the size of the template features, using a sliding window approach. These sub-regions are then concatenated with the template features along the channel dimension, and each concatenated feature is merged into V_i through a convolution operation. However, this method incurs high computational complexity. To simplify the calculations, asymmetric convolutions are introduced as an equivalent alternative, transforming the process into two independent convolution operations instead of the original convolution, as shown in Figure 4:

$$\begin{aligned} v_i &= [\theta_z; \theta_x] * \begin{bmatrix} \overline{Z} \\ \overline{X}_i \end{bmatrix} = \theta_z * \overline{Z} + \theta_x * \overline{X}_i, \\ \overline{X}_i &\in \Re^{C \times \eta \times \omega}, \theta_z, \theta_x \in \Re^{P \times C \times \eta \times \omega}, v_i \in \Re^{P \times 1 \times 1}, \end{aligned} \tag{3}$$

where v_i denotes the fused features, θ_z and θ_x denote the convolution kernels acting on $\overline{Z}$ and $\overline{X}_i$, respectively, $\overline{Z}$ denotes the template frame feature map, $\overline{X}_i$ denotes the search frame feature map, C, η, ω denotes the number of channels, width, and height, respectively, and V_i is in the shape of $P * 1 * 1$.

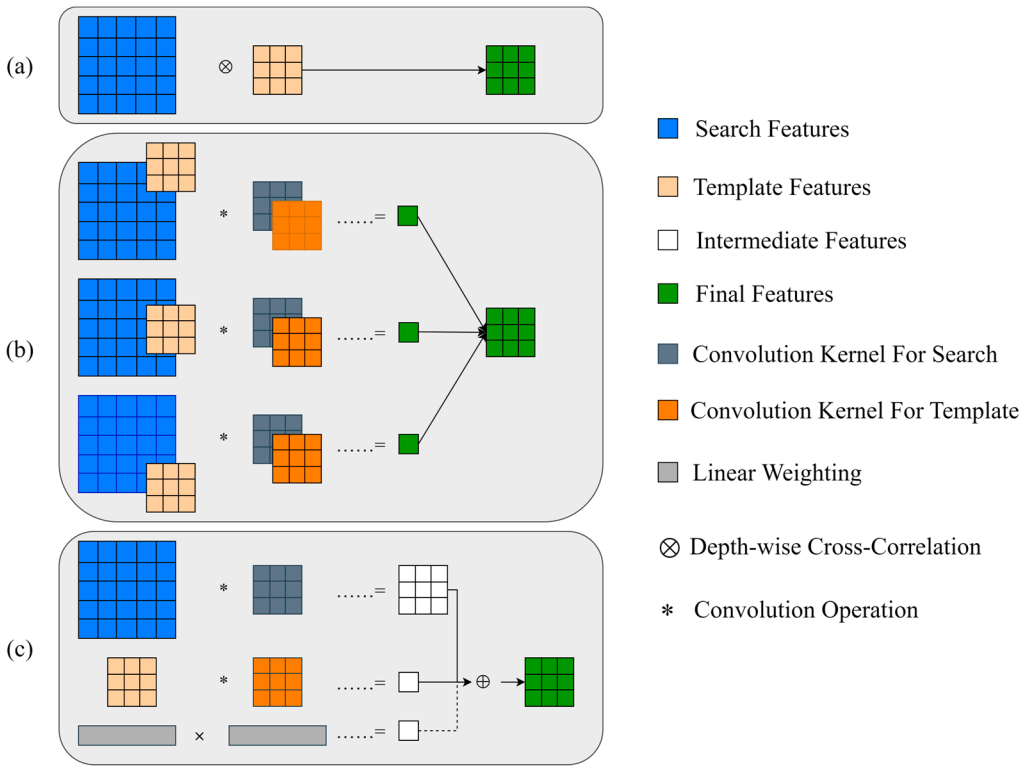


Figure 4. Asymmetric convolution. (**a**) DW-Xcorr. (**b**) A naive approach for fusing feature maps of varying sizes. (**c**) Symmetric convolution.

The depth-wise cross-correlation concatenates the template features Z with the features of the search sub-region X_i before performing the convolution operation. In contrast, asymmetric convolutions perform an equivalent operation, where the concatenated features are separately convolved with Z and X_i, and the results are then summed. The advantage of this approach is that it eliminates the need for sliding window concatenation and convolution operations, treating all sub-region features of the search image as a whole to simplify the calculation process. To fully leverage the benefits of offline learning, a

learnable parameter b is introduced, serving as a weight determined through continuous offline learning to obtain its optimal value. Subsequently, the formula becomes:

$$v_i = \theta_z * \overline{Z} + b\theta_x * \overline{X}_i. \tag{4}$$

where v_i denotes the fused features, θ_z and θ_x denote the convolution kernels acting on $\overline{Z}$ and $\overline{X}_i$, respectively, $\overline{Z}$ denotes the template frame feature map, $\overline{X}_i$ denotes the search frame feature map, and b denotes the learnable weight parameters.

3.4. Ranking Loss

The primary challenge of background interference in remote sensing images arises from the similarity between the local background surrounding the object and the object itself. Although this background information belongs to different categories, it shares similarities with the object in features such as color, shape, geometry, and texture, and is often in close proximity to the object. When similar backgrounds are near the object, they can be mistakenly identified as part of the object during feature extraction, leading to tracking drift. To address this issue, we propose a classification ranking loss that reformulates the classification task as a ranking problem. This constraint ensures that the confidence of positive samples in classification surpasses that of challenging negative samples. By modeling the relationship between foreground and background samples, the ability to distinguish positive and difficult negative samples is enhanced, helping to prevent the tracker from being misled by distracting objects. The original loss function L_{rpn} of the Siamese network is defined as follows:

$$L_{rpn} = \frac{1}{N_{pos}} \sum_{i \in A_{pos}} L_{cls}(A^i_{cls}, Y^i_{cls}) + L_{loc}(A^i_{loc}, Y^i_{loc}) + \frac{1}{N_{neg}} \sum_{i \in A_{neg}} L_{cls}(A^i_{cls}, Y^i_{cls}), \tag{5}$$

where N_{pos} and N_{neg} are the number of positive sample sets A_{pos} and negative sample sets A_{neg}, respectively. A_{cls} denotes the classification mapping, which aims at recognizing the foreground from the background. A_{loc} denotes the regression mapping, which aims to regress the object bounding box. Y_{cls} and Y_{loc} denote the classification and regression labels, respectively. L_{cls} is usually the cross-entropy loss, and L_{loc} is the commonly used IoU loss.

A classifier is first trained using supervised learning with a cross-entropy loss function. Negative samples are then ranked based on the object's predicted confidence score. Samples with scores below a certain threshold (e.g., 0.5) are discarded, leaving only the difficult negative samples. Positive samples are retained unchanged. These challenging negative samples closely resemble the object, creating a classification dilemma. To improve the tracker's ability to differentiate hard-negative samples, a classification ranking loss is introduced, prioritizing the expectations of training samples.

$$L_{rank-cls} = \frac{1}{\beta} \log(1 + \exp(\beta \cdot (P_- - P_+ + \alpha)), \tag{6}$$

where β controls the loss value, α is the ranking edge, and P_- and P_+ are the expectation values of difficult negative and positive samples. To address the mismatch between classification and regression, an IoU ranking loss is introduced, as shown in Figure 5. The positive sample with the highest classification confidence and IoU is selected as the final result, aiming to improve tracking precision. This loss links the classification and regression branches, ensuring that classification scores accurately reflect class identification

and object frame prediction. This coordination optimizes both branches simultaneously. The IoU ranking loss is defined as shown in the following equation.

$$L_{rank-iou} = \frac{1}{N_{pos}} \sum_{i,j \in A_{pos}, v_i^{iou} > v_j^{iou}} \exp(-\gamma \cdot (p_i - p_j)) + \frac{1}{N_{pos}} \sum_{i,j \in A_{pos}, p_i > p_j} \exp(-\gamma \cdot (v_i^{iou} - v_j^{iou})). \quad (7)$$

where $\gamma > 0$ is the hyperparameter that controls the loss value. p_i and p_j denote the prospective confidence scores of the positive samples i and j, respectively, and v_i and v_j denote the predicted IoU values corresponding to the samples i, j. Aligning classification scores with IoU reduces the IoU ranking loss by bridging the gap between the classification and regression branches, allowing high classification confidence scores to better reflect accurate localization predictions.

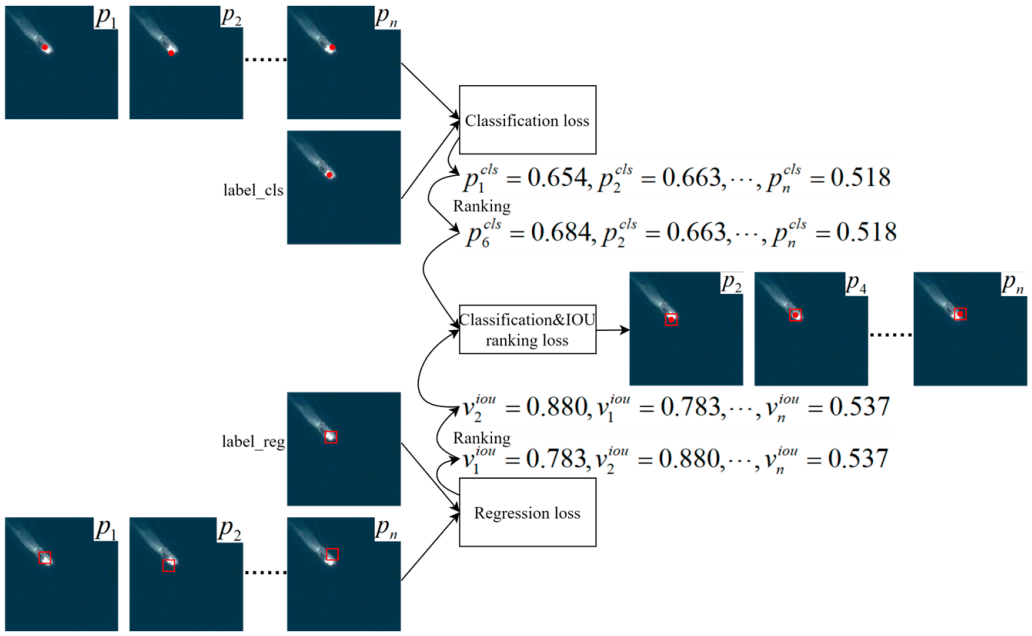


Figure 5. Ranking loss. We focus on samples with high classification confidence and increased IoU to achieve higher rankings, leveraging the relationship between the classification and regression branches. The red points represent the center point of the object obtained by classification, and the red boxes represent the bounding box of the object obtained by regression.

4. Experiments

4.1. Implementation Details

The algorithm presented in this paper is implemented using the PyTorch 1.3.0 framework with CUDA 10.0 and Python 3.7 and utilizes GPU acceleration. The GPU configuration used four NVIDIA GTX 2080Ti cards.

During training, we pre-trained the network on the GOT-10K [21], Lasot [22], and ILSVRC2015_VID [23] datasets, which contain common categories in remote sensing videos, such as airplanes, cars, ships, and trains. We trained the model for 20 epochs, starting with a learning rate of 0.001, which increased to 0.005 during the first five epochs. This was followed by an exponential decay from 0.005 to 0.00005 over the subsequent 15 epochs. In the first 10 epochs, we trained only the head network. In the following 10 epochs, we fine-tuned the backbone network using one-tenth of the current learning rate. We set the weight decay to 0.0001 and the momentum to 0.9. The two proposed ranking losses are optimized alongside the original loss function of the Siamese network, combined in a ratio of 1:0.5:0.25. The margin for the ranking loss α is set to 0.5. To ensure stable training, β and

γ are set to 4 and 3, respectively, for all experiments. During the testing phase, we evaluated the trained network using performance metrics such as success rate and precision on the object tracking benchmark datasets OTB100, UAV123, and the remote sensing benchmark dataset OOTB.

4.2. Evaluation Index

The precision map tracks the percentage of frames where the center location error (CLE) is below a defined threshold. d_c is set to denote the CLE, defined as

$$d_c = \sqrt{(x - X)^2 + (y - Y)^2}, \tag{8}$$

In addition to the precision plot, a normalized precision plot is introduced to evaluate the tracker. The normalized precision plot shows the percentage of frames where the normalized CLE is below a defined threshold, ranging from 0 to 1. Let d_n denote the normalized CLE, defined as:

$$d_n = \sqrt{((x - X)/W)^2 + ((y - Y)/H)^2}, \tag{9}$$

where W and H are the width and height of the ground truth.

In the success rate plot, the Success Rate (SR) is calculated as the percentage of successful frames with overlap exceeding a defined threshold, ranging from 0 to 1. Given the predicted bounding box r_p and ground truth r_g, the overlap fraction s is given by:

$$s = |r_p \cap r_g| / |r_p \cup r_g|. \tag{10}$$

where $\cap$ and $\cup$ denote intersection and concatenation, respectively, and $|\bullet|$ denotes the number of pixels in a given region.

4.3. Experiments on the OTB Benchmark

OTB100 [24] is the most widely used evaluation dataset in object tracking, consisting of 100 video sequences and a total of 102 tracking objects. The specific scenarios depicted in the videos include Background Clutter (BC), Illumination Variation (IV), Scale Variation (SV), Occlusion (OCC), Deformation (DEF), Motion Blur (MB), Fast Motion (FM), In-Plane Rotation (IPR), Out-of-Plane Rotation (OPR), Out-of-View (OV), and Low Resolution (LR), making it a comprehensive benchmark for evaluating tracking algorithms. In this test, our algorithm is compared with nine mainstream object tracking algorithms, including Ocean [25], ATOM [26], DaSiamRPN [10], GradNet [27], SiamRPN [8], SiamDW [28], SRDCF [29], CFNet [30], and SiamFC [7].

Overall Evaluation: Figure 6 shows the precision and success rate curves for all algorithms at various thresholds. The area under the curve (AUC) of the success rate plot is used to rank the tracking algorithms. Table 1 presents the comparative results for each algorithm. As shown in Figure 6, the success rate of the algorithm introduced in this paper is 0.670, while the precision rate is 0.892. Our algorithm outperforms most existing methods in terms of success rate and precision on the OTB100 dataset. The results indicate that the enhanced tracking algorithm presented in this paper exhibits competitive performance.

Attribute-Based Evaluation: To further demonstrate the effectiveness of our algorithm in addressing common challenges in object tracking, we conducted independent experiments on 11 annotated attributes of object objects from the OTB100 dataset. The 11 attributes include background clutter, deformation, fast motion, in-plane rotation, out-of-view, illumination variation, low resolution, motion blur, occlusion, scale variation, and out-of-plane rotation. The success rate curve is shown in Figure 7, while the precision rate curve is displayed in Figure 8. It is clear that our algorithm yields the best results for the attributes of out-of-view, occlusion, motion blur, and background clutter.

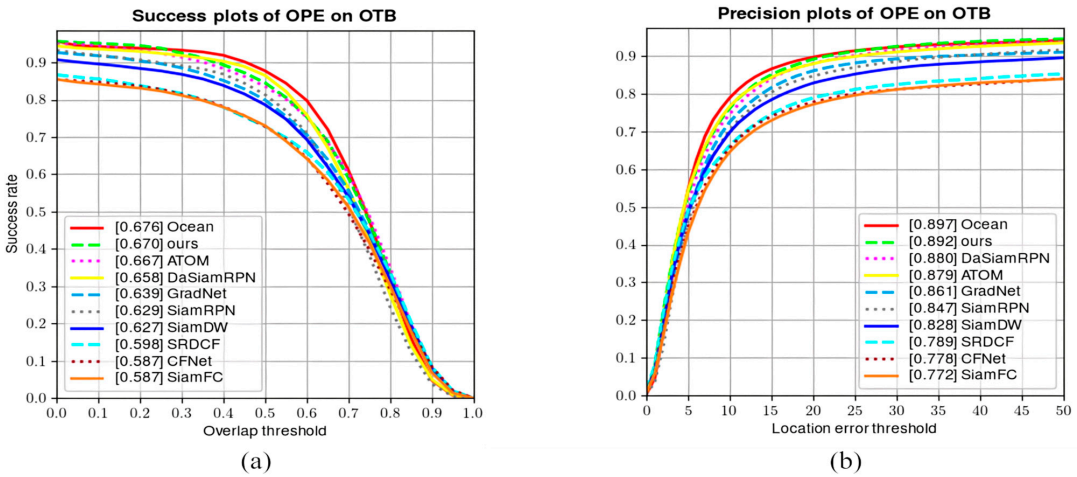


Figure 6. The precision and success rates of our tracker compared to other trackers on the OTB100 dataset. (**a**) Success plots; (**b**) Precision plots.

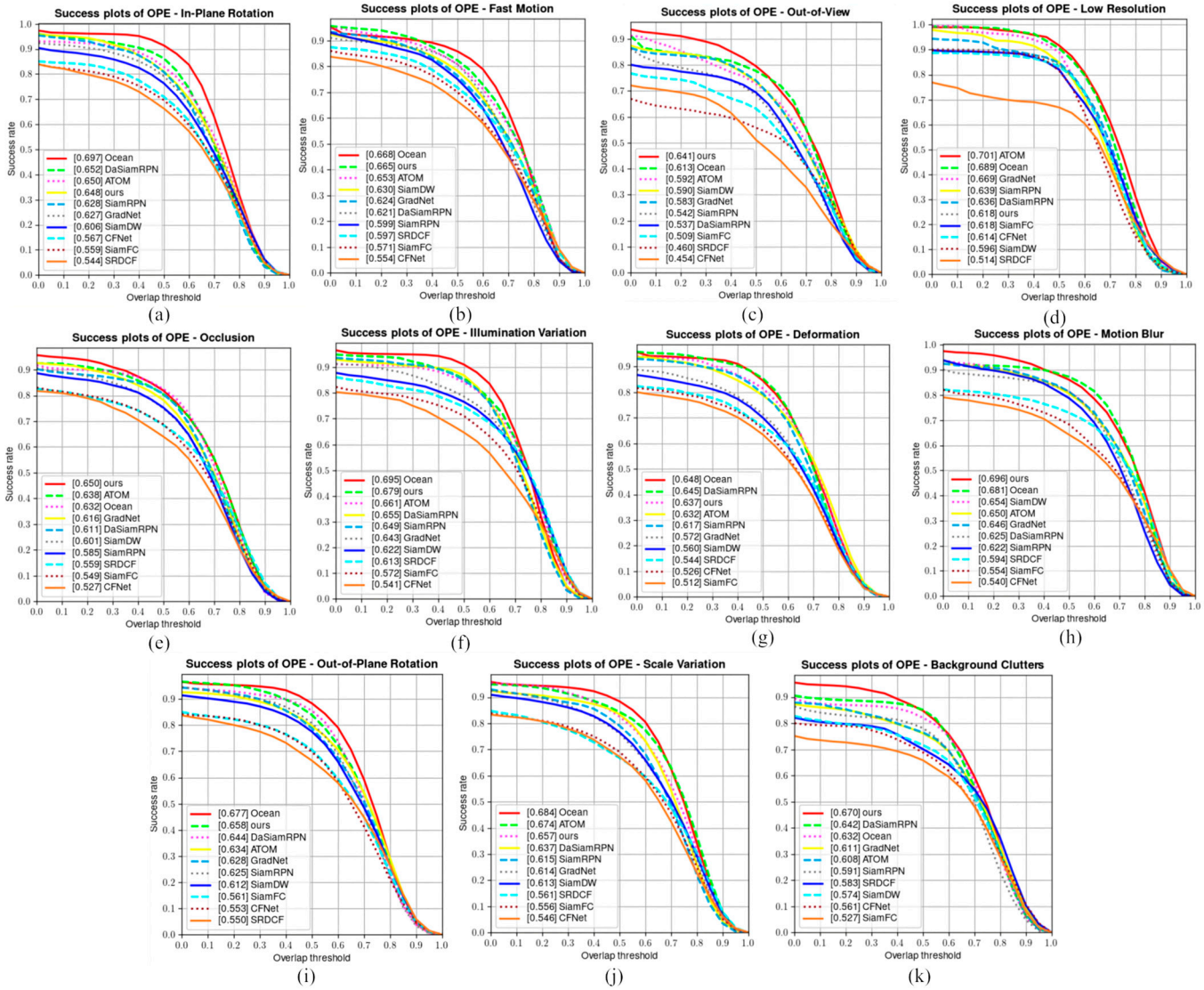


Figure 7. The success rate of our tracker compared to other trackers across the 11 challenges of the OTB100 dataset. (**a**) In-plane Rotation; (**b**) Fast Motion; (**c**) Out-of-view; (**d**) Low Resolution; (**e**) Occlusion; (**f**) Illumination Variation; (**g**) Deformation; (**h**) Motion Blur; (**i**) Out-of-plane Rotation; (**j**) Scale Variation; (**k**) Background Clutter.

Table 1. Experimental results of each algorithm on the OTB100 dataset.

Arithmetic	Success	Precision
GradNet	0.639	0.864
CFNet	0.587	0.778
SiamFC	0.587	0.772
SiamRPN	0.629	0.847
ATOM	0.667	0.879
DaSiamRPN	0.658	0.880
SiamDW	0.627	0.828
Ocean	0.676	0.897
SRDCF	0.598	0.789
Ours	0.670	0.892

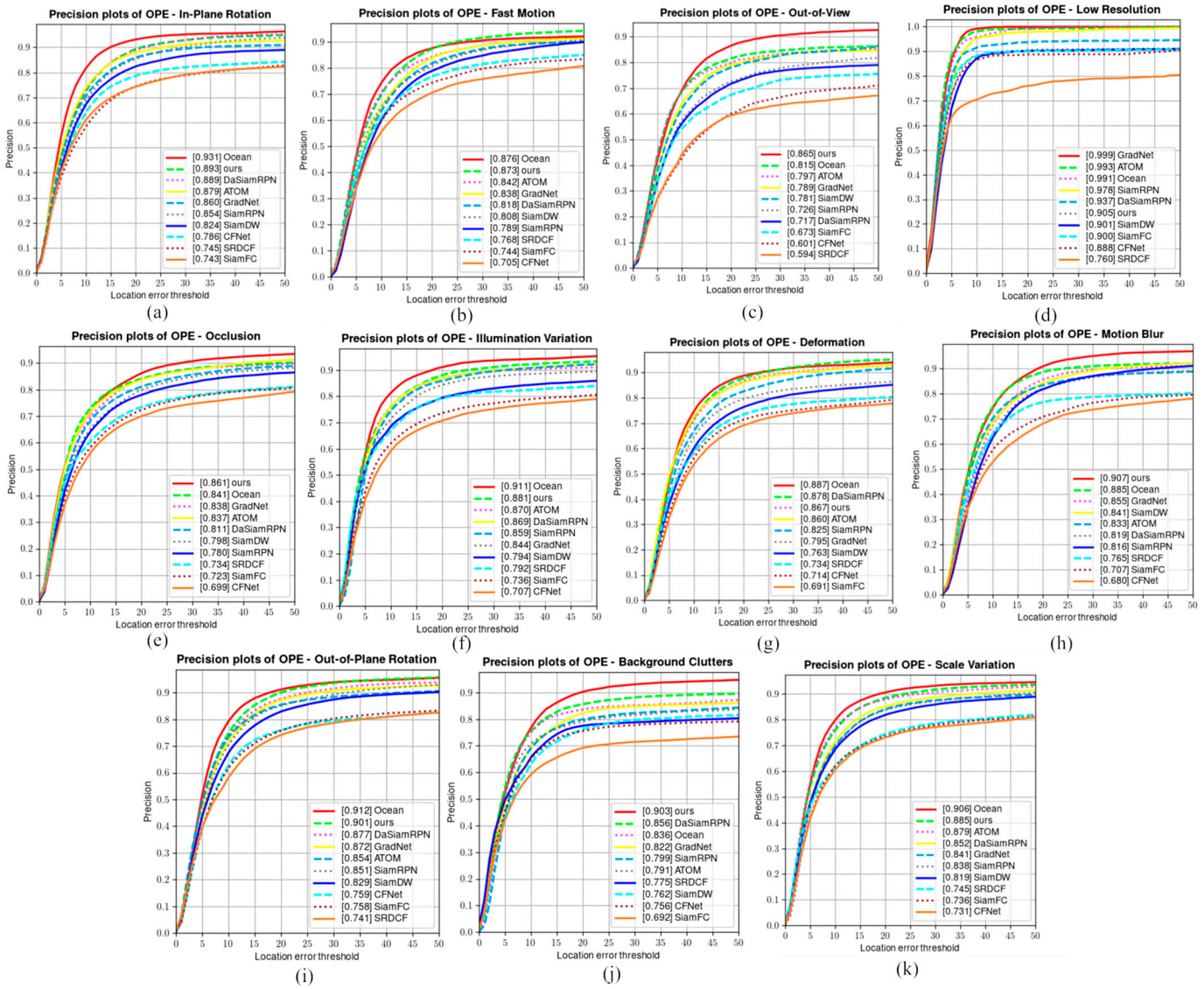


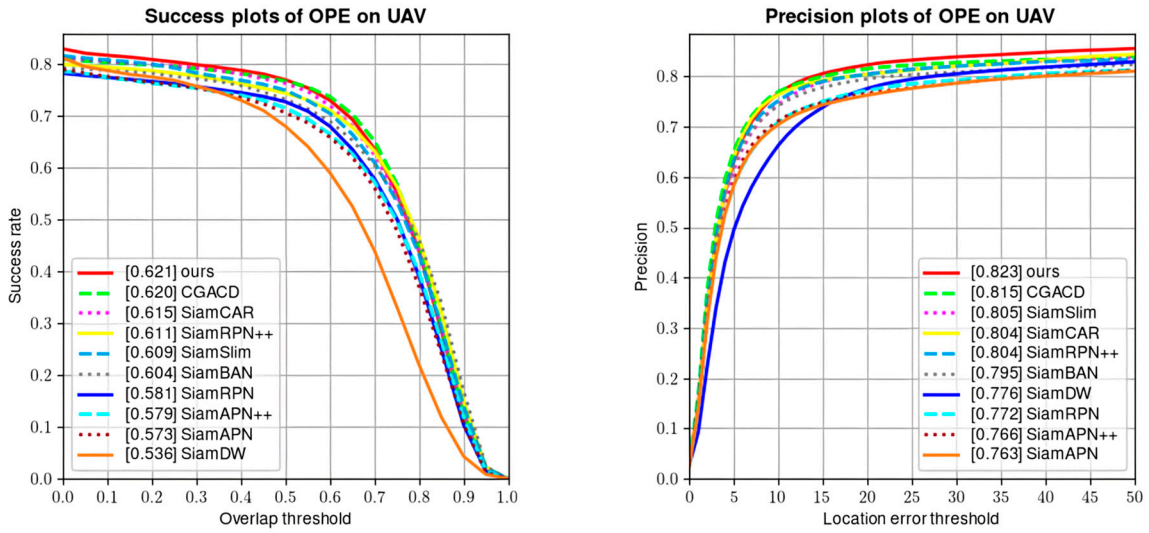
Figure 8. The precision of our tracker in comparison to other trackers across the 11 challenges of the OTB100 dataset. (**a**) In-plane Rotation; (**b**) Fast Motion; (**c**) Out-of-view; (**d**) Low Resolution; (**e**) Occlusion; (**f**) Il-lumination Variation; (**g**) Deformation; (**h**) Motion Blur; (**i**) Out-of-plane Rotation; (**j**) Background Clutter; (**k**) Scale Variation.

4.4. Experiments on the UAV123 Benchmark

We evaluate the algorithm through experimental comparisons on the UAV123 dataset. The UAV123 dataset [31] contains 123 manually annotated drone sequences, totaling over

110,000 frames. The UAV123 dataset includes diverse tracking objects, such as humans, birds, and cars. The UAV123 dataset presents complex filming environments, introducing a variety of challenges for object tracking. The evaluation metrics include success rate and precision. The dataset includes 12 challenge attributes: aspect ratio change, background clutter, camera motion, fast motion, full occlusion, illumination variation, low resolution, out-of-view, partial occlusion, similar objects, scale variation, and viewpoint change. During the evaluation process, we compared our algorithm to the leading algorithms in the field, including CGACD [32], SiamBAN [33], SiamCAR [34], SiamDW [28], SiamGAT [35], SiamRPN [8], SiamRPN++ [18], SiamAPN [36], SiamAPN++ [36], and SiamSlim [37].

Overall Evaluation: Figure 9 shows that the algorithm achieves a success rate of 0.621 and a precision of 0.823. Table 2 presents the comparative results for each algorithm. These values represent improvements of 0.006 and 0.019, respectively, compared to the SiamCAR algorithm.



(a) (b)

Figure 9. The precision and success rates of our tracker, along with those of the comparison trackers, are evaluated on the UAV123 dataset. (**a**) Success plots; (**b**) Precision plots.

Table 2. Experimental results of each algorithm on the UAV123 dataset.

Arithmetic	Success	Precision
SiamAPN	0.573	0.763
SiamAPN++	0.579	0.766
SiamSlim	0.609	0.805
CGACD	0.620	0.815
SiamCAR	0.615	0.804
SiamRPN++	0.611	0.804
SiamBAN	0.604	0.795
SiamRPN	0.581	0.772
SiamDW	0.536	0.776
Ours	0.621	0.823

Attribute-Based Evaluation: We also perform independent experiments on 12 labeled attributes from the UAV123 dataset. The 12 attributes are as follows: aspect ratio variation, background clutter, camera motion, fast motion, complete occlusion, illumination variation, low resolution, out-of-view, partial occlusion, similar objects, scale variation, and viewpoint variation. Figures 10 and 11 show the success rate and precision of our algorithm, along with nine other algorithms with comparable performance, evaluated across twelve challenging

attributes. Our algorithm performs particularly well in the attributes of full occlusion, aspect ratio change, scale variation, partial occlusion, and low resolution.

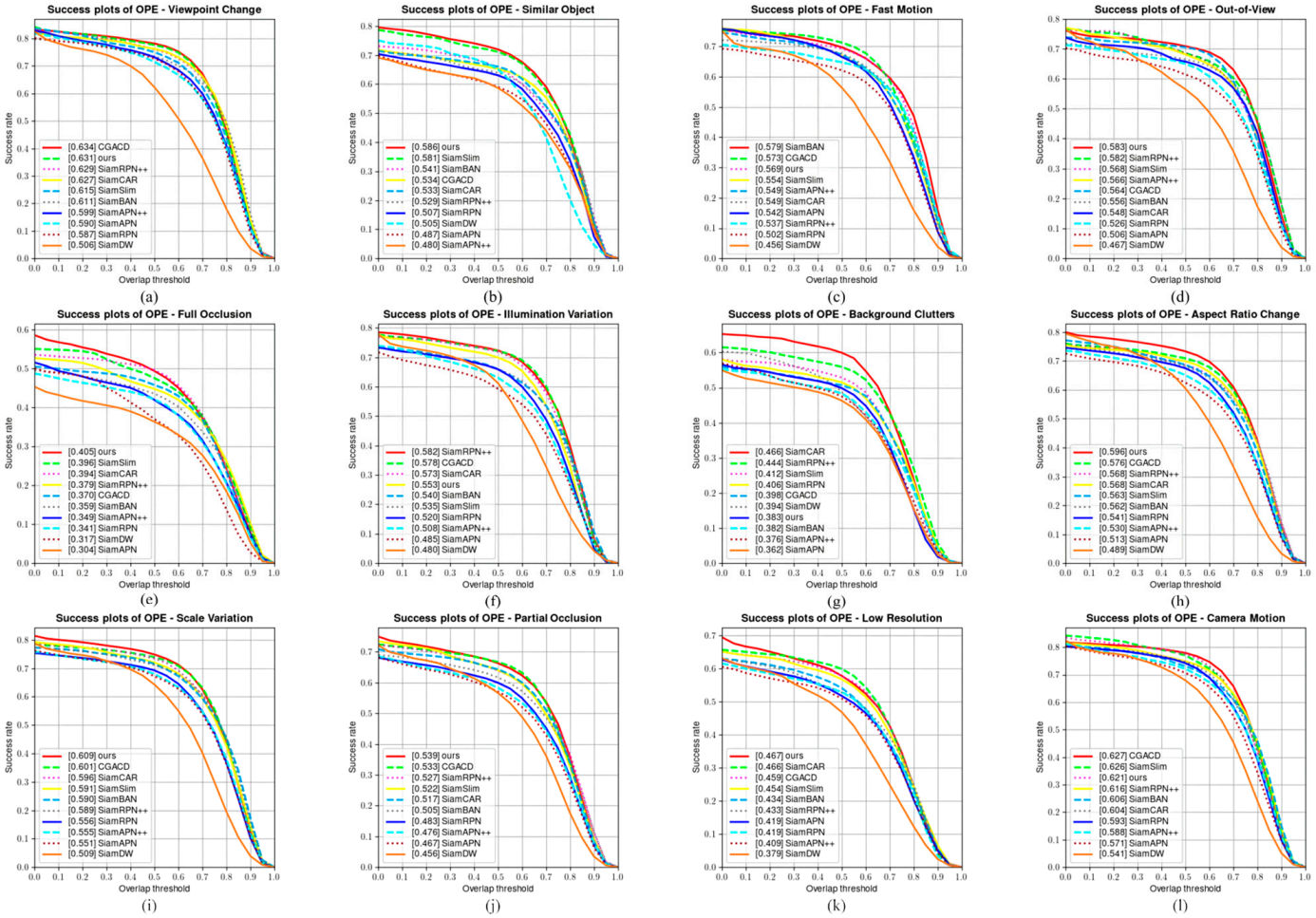


Figure 10. The success rates of our tracker, along with those of the comparison trackers, are assessed across the twelve challenges of the UAV123 dataset. (**a**) Viewpoint Change; (**b**) Similar Object; (**c**) Fast Motion; (**d**) Out-of-view; (**e**) Full Occlusion; (**f**) Illumination Variation; (**g**) Background Clutter; (**h**) Aspect Ratio Variation; (**i**) Scale Variation; (**j**) Partial Occlusion; (**k**) Low Resolution; (**l**) Camera Motion.

4.5. Experiments on the OOTB Benchmark

The OOTB dataset [38], a recent contribution to remote sensing, contains 29,890 frames from 110 video sequences. It includes common satellite video object categories, such as cars, ships, airplanes, and trains. On average, each video contains 271.7 frames, with the shortest sequence having 90 frames and the longest containing 750 frames. Cars and trains present the greatest challenges, while ships and airplanes are relatively easier to track. This is due to cars being smaller and often operating against complex backgrounds, while trains have a larger aspect ratio and are prone to non-rigid deformations. Consequently, the OOTB dataset contains more cars, while trains are less common due to their infrequent appearances in typical scenes. Each frame is manually annotated with oriented bounding boxes, and each sequence is labeled with 12 detailed attributes. The dataset contains a total of 12 challenge attributes, including non-rigid deformation (DEF), in-plane rotation (IPR), partial occlusion (PO), and illumination variation (IV). In this test, our algorithm was compared with the remaining eight object tracking algorithms, including SiamRPN [8], SiamBAN [33], DROL [39], CLNet [40], SiamAPN [36], SiamAPN++ [36], SiamKPN [14], and SiamSA [41].

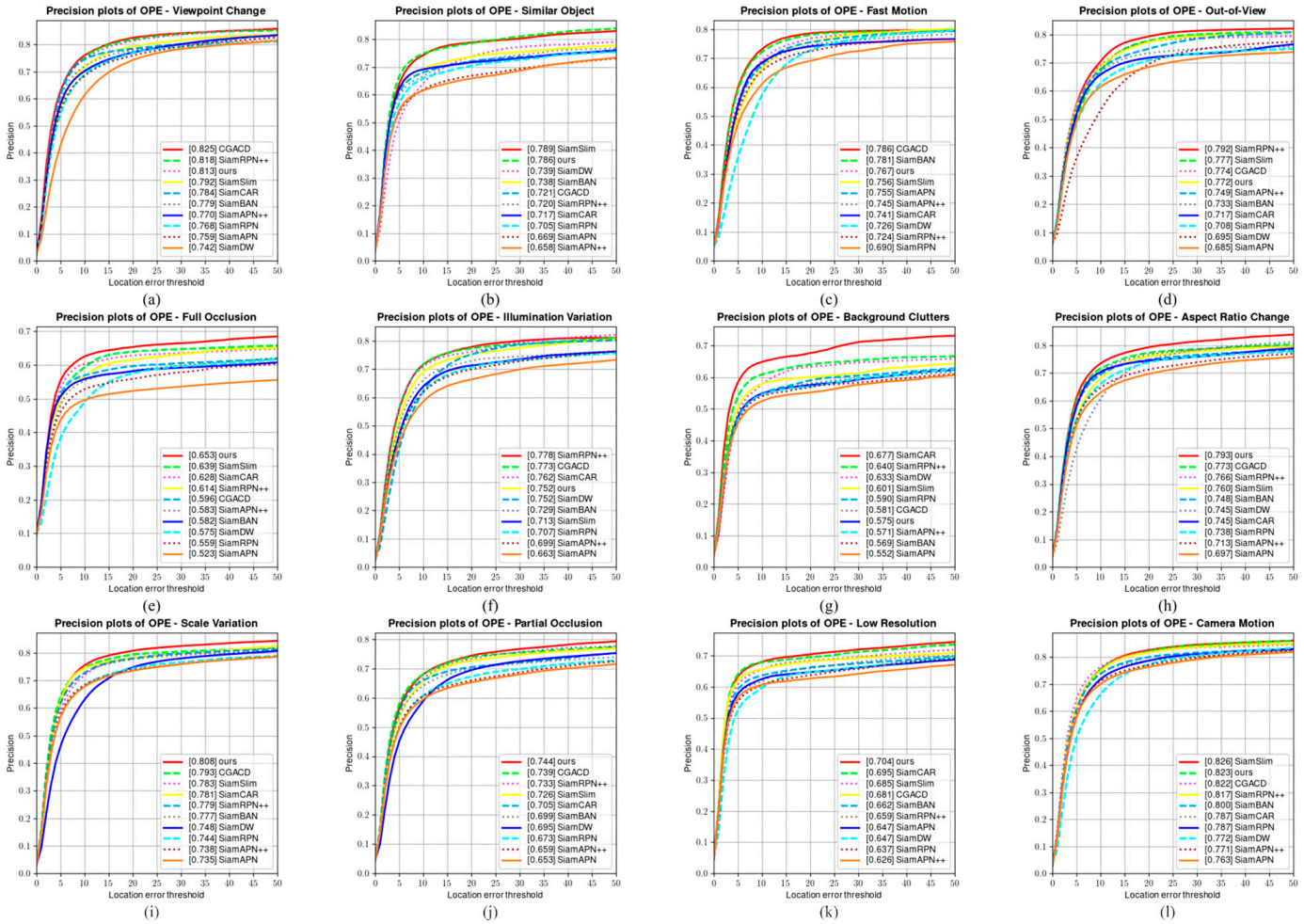


Figure 11. The precision of our tracker, as well as that of the comparison trackers, is evaluated across the twelve challenges presented in the UAV123 dataset. (**a**) Viewpoint Change; (**b**) Similar Object; (**c**) Fast Motion; (**d**) Out-of-view; (**e**) Full Occlusion; (**f**) Illumination Variation; (**g**) Background Clutter; (**h**) Aspect Ratio Variation; (**i**) Scale Variation; (**j**) Partial Occlusion; (**k**) Low Resolution; (**l**) Camera Motion.

Overall Evaluation: Figure 12 shows the precision, normalized precision, and success rate curves for all algorithms at various thresholds. Table 3 presents the comparative results for each algorithm. Our algorithm achieves overall precision, normalized precision, and success rates of 0.812, 0.786, and 0.533, respectively, on the OOTB remote sensing dataset, outperforming the other tracking algorithms.

Table 3. Experimental results of each algorithm on the OOTB dataset.

Algorithm	Success	Normalize	Precision
SiamBAN	0.495	0.684	0.709
SiamRPN	0.490	0.744	0.747
DROL	0.236	0.377	0.283
CLNet	0.514	0.774	0.796
SiamAPN++	0.476	0.752	0.767
SiamSA	0.500	0.736	0.752
SiamKPN	0.518	0.733	0.735
SiamAPN	0.444	0.693	0.687
Ours	0.533	0.786	0.812

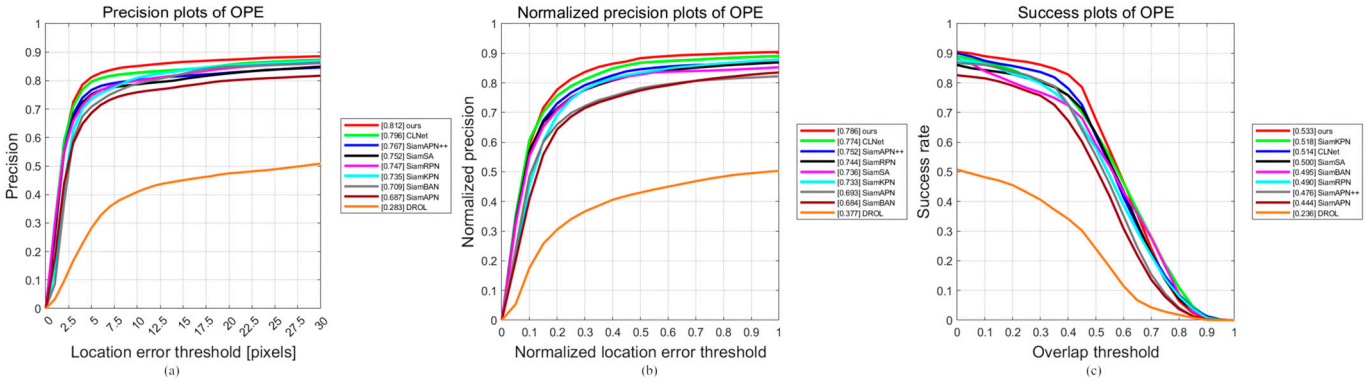


Figure 12. The precision, normalized precision, and success rates of both our tracker and the comparison trackers are assessed on the OOTB dataset. (**a**) Precision plot; (**b**) Normalized precision plots; (**c**) Success plots.

Attribute-Based Evaluation: We perform independent experiments on the four object categories and 12 annotated attributes from the OOTB dataset. The four object categories include cars, airplanes, ships, and trains. The 12 attributes are non-rigid deformation, in-plane rotation, partial occlusion, full occlusion, illumination variation, motion blur, background Clutter, Abnormal, similar appearance, less textures, isotropic motion, and anisotropic motion. Tables 4–6 present the experimental results for success rate, normalized precision, and precision of our algorithm, along with eight other high-performing comparison algorithms, across the 12 challenging attributes. Our algorithm performs strongly across eight attributes: background clutter, deformation, isotropic motion, in-plane rotation, illumination variation, low texture, motion blur, and similar appearance.

Table 4. The precision of our tracker and comparison trackers on the 12 challenges of the OOTB.

	AM	BC	DEF	FO	IM	IPR	IV	LT	MB	OON	PO	SA
SiamBAN	0.719	0.688	0.502	0.502	0.816	0.721	0.670	0.518	0.716	0.505	0.699	0.622
SiamRPN	0.874	0.763	0.449	0.396	0.807	0.706	0.717	0.667	0.720	0.433	0.759	0.673
DROL	0.146	0.274	0.205	0.051	0.229	0.289	0.307	0.178	0.242	0.210	0.118	0.189
CLNet	0.884	0.808	0.483	0.585	0.836	0.766	0.785	0.714	0.794	0.516	0.794	0.701
SiamAPN++	0.650	0.775	0.470	0.652	0.805	0.697	0.776	0.691	0.780	0.516	0.614	0.603
SiamSA	0.735	0.758	0.466	0.571	0.777	0.714	0.764	0.682	0.776	0.477	0.653	0.631
SiamKPN	0.812	0.740	0.479	0.535	0.821	0.712	0.703	0.606	0.747	0.503	0.722	0.656
SiamAPN	0.676	0.713	0.420	0.487	0.765	0.613	0.697	0.597	0.704	0.416	0.592	0.577
Ours	0.847	0.826	0.517	0.576	0.843	0.784	0.791	0.715	0.816	0.533	0.774	0.714

(AM) Anisotropic Motion; (BC) Background Clutter; (DEF) Deformation; (FO) Full Occlusion; (IM Isotropic Motion; (IPR) In-plane Rotation; (IV) Illumination Variation; (LT) Less Textures; (MB) Motion Blur; (OON) Abnormal; (PO) Partial Occlusion; (SA) Similar Appearance.

Table 5. The normalized precision of our tracker and comparison trackers on the 12 challenges of the OOTB.

	AM	BC	DEF	FO	IM	IPR	IV	LT	MB	OON	PO	SA
SiamBAN	0.598	0.655	0.599	0.522	0.715	0.690	0.668	0.488	0.704	0.564	0.614	0.591
SiamRPN	0.808	0.729	0.658	0.401	0.747	0.742	0.730	0.630	0.729	0.583	0.730	0.671
DROL	0.165	0.352	0.377	0.059	0.215	0.382	0.425	0.221	0.357	0.336	0.137	0.268
CLNet	0.784	0.763	0.697	0.541	0.756	0.781	0.758	0.668	0.749	0.655	0.752	0.687
SiamAPN++	0.579	0.735	0.669	0.603	0.737	0.711	0.771	0.652	0.767	0.672	0.597	0.607
SiamSA	0.684	0.724	0.667	0.533	0.717	0.732	0.746	0.648	0.757	0.668	0.624	0.640
SiamKPN	0.693	0.722	0.700	0.535	0.716	0.720	0.722	0.591	0.737	0.647	0.669	0.652
SiamAPN	0.601	0.693	0.551	0.450	0.696	0.635	0.724	0.560	0.728	0.532	0.537	0.576
Ours	0.749	0.780	0.715	0.539	0.751	0.776	0.777	0.670	0.777	0.663	0.725	0.697

(AM) Anisotropic Motion; (BC) Background Clutter; (DEF) Deformation; (FO) Full Occlusion; (IM Isotropic Motion; (IPR) In-plane Rotation; (IV) Illumination Variation; (LT) Less Textures; (MB) Motion Blur; (OON) Abnormal; (PO) Partial Occlusion; (SA) Similar Appearance.

Table 6. The success rates of our tracker and comparison trackers on the 12 challenges of the OOTB.

	AM	BC	DEF	FO	IM	IPR	IV	LT	MB	OON	PO	SA
SiamBAN	0.450	0.491	0.398	0.344	0.537	0.495	0.470	0.387	0.520	0.378	0.461	0.432
SiamRPN	0.555	0.493	0.386	0.250	0.514	0.466	0.478	0.441	0.494	0.360	0.511	0.458
DROL	0.106	0.224	0.208	0.036	0.140	0.235	0.258	0.143	0.227	0.198	0.095	0.170
CLNet	0.546	0.518	0.409	0.350	0.518	0.491	0.497	0.468	0.517	0.398	0.531	0.474
SiamAPN++	0.359	0.476	0.381	0.392	0.499	0.425	0.487	0.433	0.481	0.400	0.398	0.388
SiamSA	0.465	0.502	0.418	0.381	0.503	0.481	0.504	0.459	0.520	0.437	0.450	0.441
SiamKPN	0.527	0.531	0.405	0.373	0.527	0.500	0.498	0.442	0.522	0.401	0.495	0.463
SiamAPN	0.401	0.453	0.328	0.299	0.482	0.399	0.457	0.383	0.473	0.337	0.368	0.385
Ours	0.538	0.542	0.420	0.369	0.541	0.513	0.519	0.483	0.542	0.421	0.517	0.492

(AM) Anisotropic Motion; (BC) Background Clutter; (DEF) Deformation; (FO) Full Occlusion; (IM Isotropic Motion; (IPR) In-plane Rotation; (IV) Illumination Variation; (LT) Less Textures; (MB) Motion Blur; (OON) Abnormal; (PO) Partial Occlusion; (SA) Similar Appearance.

4.6. Experiments on the LaSOT Benchmark

The LaSOT [22] dataset is designed for large-scale, long-term single-object tracking with high-quality, dense annotations. LaSOT contains 1400 videos across 70 categories, with 20 sequences per category, totaling over 3.52 million manually annotated frames. The dataset features high-quality, dense annotations, with every frame manually labeled. The dataset emphasizes long-term tracking, with an average video length of 2512 frames. The shortest sequence contains 1000 frames, while the longest has 11,397 frames. Furthermore, the dataset includes natural language descriptions, establishing a connection between visual appearance and textual context. LaSOT was introduced to overcome limitations of existing datasets, such as small size, poor annotation quality, and class imbalance, offering a large-scale, specialized benchmark for evaluating deep trackers.

Overall Evaluation: Table 7 and Figure 13 shows the precision and success rate curves for all algorithms at various thresholds. Our algorithm achieves an overall precision and success rate of 0.539 and 0.532, respectively, on the LaSOT dataset, outperforming the other tracking algorithms.

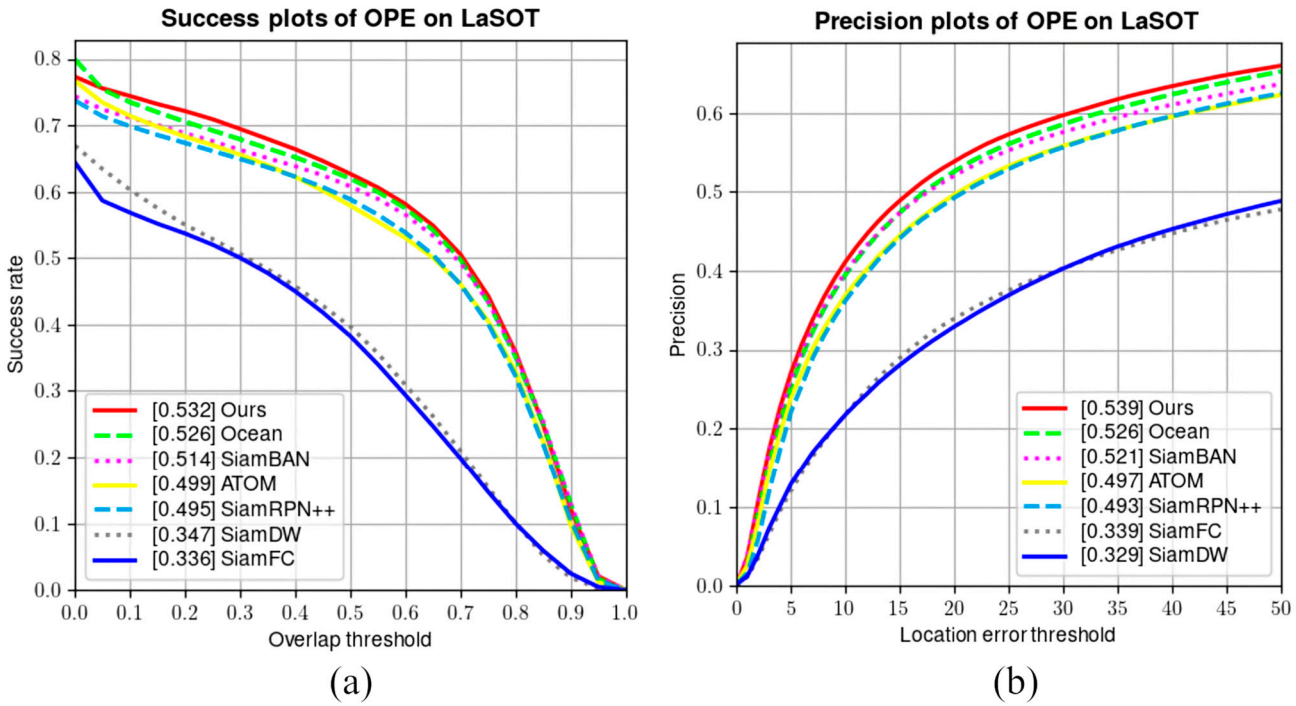


Figure 13. The precision and success rates of our tracker compared to other trackers on the LaSOT dataset. (**a**) Success plots; (**b**) Precision plots.

Table 7. Experimental results of each algorithm on the LaSOT dataset.

Algorithm	Success	Precision
SiamFC	0.336	0.339
SiamDW	0.347	0.329
SiamRPN++	0.495	0.493
SiamBAN	0.514	0.521
ATOM	0.499	0.497
Ocean	0.526	0.526
Ours	0.532	0.539

4.7. Qualitative Evaluation

4.7.1. Qualitative Evaluation on the OOTB Benchmark

To illustrate tracking performance, we present results for common object categories in remote sensing scenes, such as cars, airplanes, ships, and trains, as shown in Figure 14. The green box represents the tracking results of our algorithm, the blue box denotes the results of the SiamRPN algorithm, and the red box shows the outcomes of the SiamBAN algorithm.



Figure 14. Visualization of the tracking results for our tracker and the comparative trackers across four video sequences from the OOTB dataset. The tracking results, displayed from left to right and top to bottom, correspond to the videos car_11_1, plane_1_1, ship_12_1, and train_1_1.

As shown in Figure 14, when tracking the airplane object, the red box loses track due to background interference at frame 150. In frame 190, although the object remains within the blue box, a significant portion of the irrelevant background is included, while the green box tightly fits the object. When tracking the airplane object, the red box at frame 500 captures only part of the airplane due to deformation during its movement. When tracking ship objects, the ship's wake can cause the tracker to mistakenly identify it as part of the ship. At frame 300, both the red and blue boxes show poor tracking results due to the interference of the ship's wake.

4.7.2. Qualitative Evaluation on the OTB Benchmark

To evaluate tracking performance, we visualized and selected representative results, as shown in Figure 15. The green dashed box denotes the tracking results of our algorithm, the blue dashed box represents the SiamRPN algorithm, and the red box corresponds to the SiamBAN algorithm.

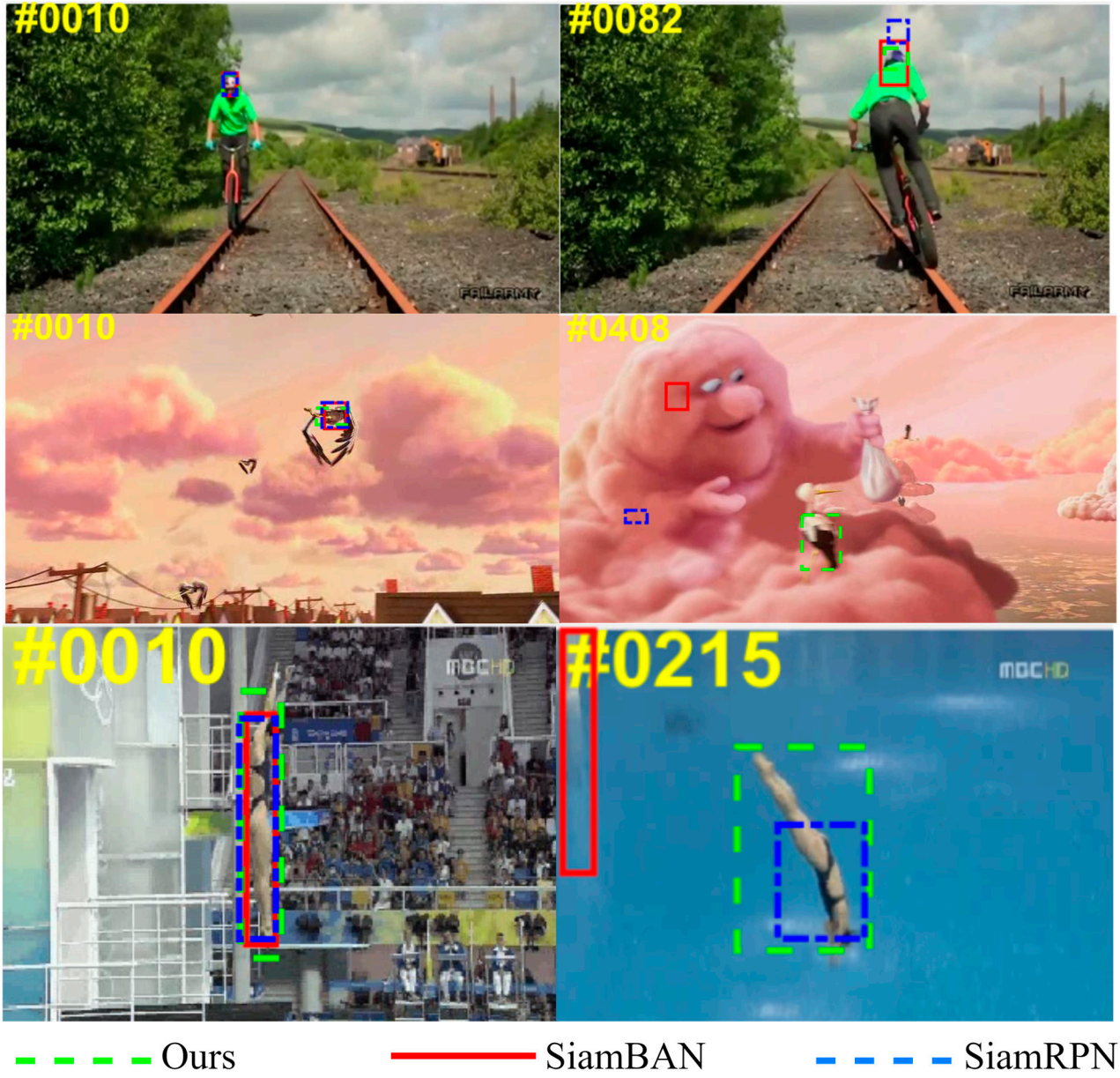


Figure 15. Visualization of the tracking results for our tracker and the comparative trackers across four video sequences from the OTB dataset.

Figure 15 illustrates that in the first set of images when the cyclist transitions from facing the camera to facing away, the blue box loses track. While the red box successfully tracks the cyclist, it includes a significant amount of irrelevant background information. In the second set of images, as the eagle undergoes continuous shape changes during movement, only the green box successfully tracks the object by frame 408. In the third set of images, at frame 215, the red box loses track of the diver, while the blue box maintains tracking but fails to fully encompass the object.

4.7.3. Qualitative Evaluation on the UAV123 Benchmark

To aid in the evaluation of tracking performance, we visualized and highlighted representative results, as shown in Figure 16. The green box indicates the tracking results of our algorithm, the blue box corresponds to the SiamRPN algorithm, and the red box represents the SiamBAN algorithm.

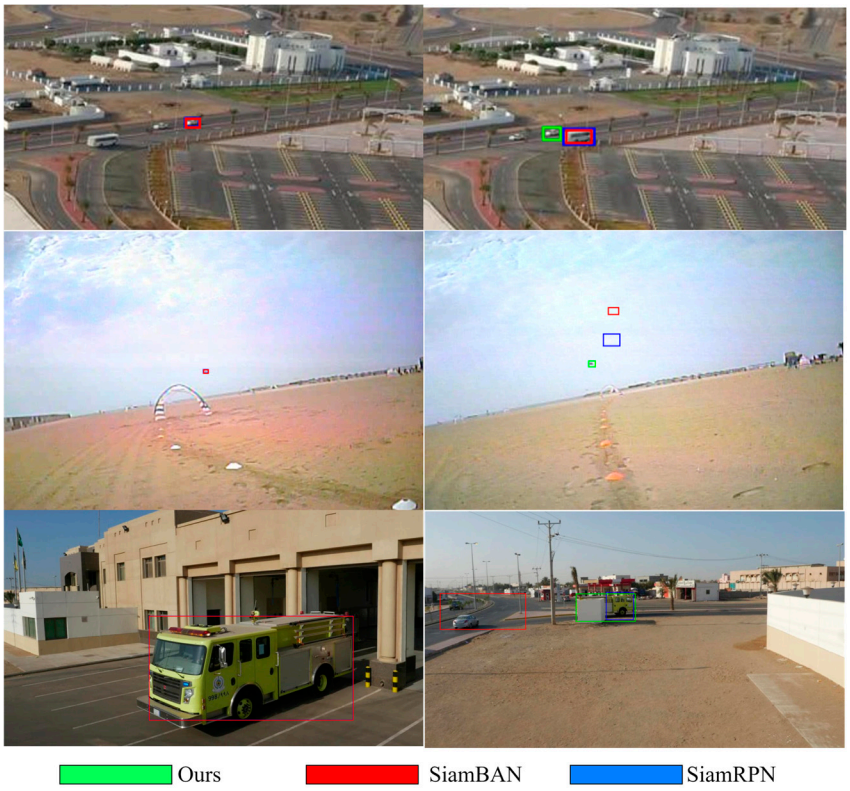


Figure 16. Visualization of the tracking results for our tracker and the comparative trackers across four video sequences from the UAV123 dataset.

Figure 16 shows that when tracking a car if another car with a similar color and appearance obstructs the object, only the green box successfully maintains tracking, while the blue and red boxes fail due to the obstruction. When tracking the drone object, due to its small size and relatively simple background, only the green box successfully maintains tracking. When tracking the truck object, interference from similar objects and partial occlusion causes the red box to fail, while the blue box fails to encompass the entire object.

4.8. Ablation Study

In the ablation study, we conducted several comparative experiments on the OOTB dataset to assess the impact of different modules on the overall performance. The algorithm's performance was evaluated using three key metrics: precision, normalized precision, and success rate. In the modified network, we incorporated the CBAM attention mechanism, asymmetric convolutions, and ranking loss. As shown in Table 8, when individual modules were added, the incorporation of ranking loss yielded the best results,

improving the success rate, normalized precision, and precision by 0.010, 0.017, and 0.022, respectively, compared to the original network. Additionally, the inclusion of the other two modules also resulted in performance improvements. When two modules were added, the combination of the CBAM attention mechanism and ranking loss resulted in the best performance, with improvements of 0.021, 0.034, and 0.040 in success rate, normalized precision, and precision, respectively, compared to the original algorithm. Furthermore, the results slightly decreased when both the CBAM attention mechanism and asymmetric convolutions were added, compared to using the CBAM attention mechanism alone. Finally, when all three modules, i.e., CBAM attention mechanism, asymmetric convolutions, and ranking loss, were added simultaneously, the algorithm achieved the highest success rate, normalized precision, and precision, with improvements of 0.027, 0.042, and 0.051, respectively, over the original algorithm. Therefore, the final precision rate in this scenario was used to compare with mainstream algorithms in subsequent experiments, validating the effectiveness of our approach.

Table 8. Experimental results for each module on the OOTB dataset. We compared the performance with some modules disabled, where √ and × indicate whether a module is in use or not, respectively.

CBAM	Asymmetric Convolution	Ranking Loss	Success	Normalize	Precision
×	×	×	0.506	0.744	0.761
√	×	×	0.512	0.755	0.772
×	√	×	0.509	0.749	0.767
×	×	√	0.516	0.761	0.783
√	√	×	0.510	0.751	0.771
√	×	√	0.527	0.778	0.801
×	√	√	0.523	0.770	0.790
√	√	√	0.533	0.786	0.812

5. Discussion

5.1. Discussion on the Performance of Algorithms in Complex Scenarios

We performed independent experiments on various annotation attributes across different datasets, as shown in Figures 7, 8, 10 and 11, and Tables 4–6. The results show that our method effectively handles complex scenes, including occlusions and complex backgrounds. Furthermore, our qualitative evaluation across various datasets, as shown in Figures 14–16, confirms that the algorithm performs competitively in challenging scenarios, such as lighting changes and occlusions, while accurately distinguishing between objects and backgrounds.

5.2. Discussion of Model Generalization Performance

This algorithm uses separate datasets for training and testing. The algorithm's visualization results on the OTB100, UAV123, and OOTB datasets, shown in Figures 14–16, demonstrate satisfactory tracking performance. It effectively distinguishes objects from backgrounds while maintaining strong generalization across different datasets. Experimental results demonstrate that the algorithm exhibits strong generalization performance.

5.3. Discussion on the Contribution of Algorithms

Analyzing the strengths and weaknesses of the proposed method is essential for identifying the most suitable visual tracking model. In Section 4, the proposed method is compared to state-of-the-art approaches across three datasets, providing comprehensive results. Compared to other Siamese network-based trackers, the results in Tables 1–3 show that our method achieves competitive performance across the three criteria discussed in Section 4.2. Specifically, on the OOTB remote sensing dataset, our method improved the success rate by 1.5%, normalized precision by 4.3%, and precision by 7.7% compared to SiamKPN. Additionally, we conducted independent experiments on various annotation attributes across

different datasets. The results demonstrate that our method effectively handles complex scenarios, including occlusion and complex backgrounds. Qualitative evaluations across various datasets further confirm that our method reliably handles tracking tasks in diverse scenarios. While our proposed method outperforms other state-of-the-art trackers, it still has certain limitations. The proposed SiamRhic method addresses the mismatch between classification and regression by leveraging attention mechanisms to enhance discriminative features and using ranking loss to reduce the classification confidence of interference terms. Although asymmetric convolution simplifies the calculation process and improves speed and performance, it also increases the tracker's computational complexity. In the future, we aim to enhance the proposed method by reducing computational complexity while maintaining precision and robustness in remote sensing video object tracking.

6. Conclusions

In this work, we apply deep learning methods to remote sensing video tracking and propose an improved Siamese network based on the SiamRhic architecture, which incorporates cross-correlation and ranking heads. This method addresses the challenges of similar objects and background interference in remote sensing videos. By reconstructing the network, we integrate the CBAM attention mechanism and asymmetric convolution into the Siamese architecture, and employ a ranking loss function during training to address issues such as vulnerability to distracting elements and mismatches between classification and regression. We conducted extensive experiments on various benchmarks, along with ablation studies, demonstrating that our algorithm effectively addresses tracking challenges in complex environments, resulting in a significant improvement in object tracking success rates. However, the experiments also reveal that the algorithm exhibits lower overall success rates when tracking train objects. Therefore, improving the algorithm's success rate in tracking train objects will be a key focus of future research.

This method is primarily designed for object tracking in remote sensing images and has demonstrated strong performance in this domain. In the future, we aim to further refine this method, explore its potential applications across various fields, enhance its versatility and practicality, and support a broader range of application scenarios. For instance, we plan to extend the method to video analysis, focusing on tracking low-resolution or distant objects, to evaluate its effectiveness in this domain. In autonomous driving, we aim to adapt our method to complex, dynamic traffic environments to enhance object tracking precision.

Author Contributions: Conceptualization, A.Y. and Z.Y.; methodology, A.Y. and Z.Y.; software, Z.Y.; validation, A.Y. and W.F.; formal analysis, A.Y. and W.F.; investigation, Z.Y.; resources, A.Y.; data curation, Z.Y.; writing—original draft preparation, A.Y. and Z.Y.; writing—review and editing, A.Y., Z.Y. and W.F.; visualization, Z.Y.; supervision, A.Y. and W.F.; project administration, A.Y.; funding acquisition, A.Y. All authors have read and agreed to the published version of the manuscript.

Funding: Supported by the Fundamental Research Funds for the Provincial Universities of Zhejiang (GK249909299001-006).

Data Availability Statement: The data used to support the findings of this study are available from the corresponding author upon request.

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

References

1. Bolme, D.S.; Beveridge, J.R.; Draper, B.A.; Lui, Y.M. Visual object tracking using adaptive correlation filters. In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, San Francisco, CA, USA, 13–18 June 2010; pp. 2544–2550.
2. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. Exploiting the circulant structure of tracking-by-detection with kernels. In Proceedings of the Computer Vision-ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, 7–13 October 2012; Springer: Berlin/Heidelberg, Germany, 2012; Part IV 12, pp. 702–715.
3. Henriques, J.F.; Caseiro, R.; Martins, P.; Batista, J. High-speed tracking with kernelized correlation filters. *IEEE Trans. Pattern Anal. Mach. Intell.* **2014**, *37*, 583–596. [CrossRef] [PubMed]

4. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'OS), San Diego, CA, USA, 20–25 June 2005; pp. 886–893.
5. Danelljan, M.; Hager, G.; Khan, F.S.; Felsberg, M. Discriminative scale space tracking. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1561–1575. [CrossRef] [PubMed]
6. Ma, C.; Yang, X.; Zhang, C.; Yang, M.H. Long-term correlation tracking. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 5388–5396.
7. Bertinetto, L.; Valmadre, J.; Henriques, J.F.; Vedaldi, A.; Torr, P. Fully-Convolutional Siamese Networks for Object Tracking. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 850–865.
8. Bo, L.; Yan, J.; Wei, W.; Zheng, Z.; Hu, X. High Performance Visual Tracking with Siamese Region Proposal Network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 8971–8980.
9. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149. [CrossRef] [PubMed]
10. Zhu, Z.; Wang, Q.; Li, B.; Wu, W.; Yan, J.; Hu, W. Distractor-aware Siamese Networks for Visual Object Trackings. In Proceedings of the European Conference on Computer Vision, Munich, Germany, 8–14 September 2018; pp. 101–117.
11. Tian, Z.; Shen, C.; Chen, H.; He, T. FCOS: Fully Convolutional One-Stage Object Detection. In Proceedings of the International Conference on Computer Vision, Seul, Republic of Korea, 27 October–2 November 2019; pp. 9626–9635.
12. Xu, Y.; Wang, Z.; Li, Z.; Yuan, Y.; Yu, G. SiamFC++: Towards Robust and Accurate Visual Tracking with Target Estimation Guidelines. In Proceedings of the AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 12549–12556.
13. Zhou, X.; Wang, D.; Krahenbuhl, P. Objects as Points. *arXiv* **2019**, arXiv:1904.07850.
14. Li, Q.; Qin, Z.; Zhang, W.; Zheng, W. Siamese Keypoint Prediction Network for Visual Object Tracking. *arXiv* **2020**, arXiv:2006.04078.
15. Hu, Z.; Yang, D.; Zhang, K.; Chen, Z. Object tracking in satellite videos based on convolutional regression network with appearance and motion features. *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.* **2020**, *13*, 783–793. [CrossRef]
16. Li, Z.; Yuan, L.; Nevada, R. Global data association for multi-object tracking using network flows. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Anchorage, AK, USA, 23–28 June 2008; pp. 1–8.
17. Feng, J.; Hui, B.; Liang, Y.; Yao, Q.; Zhang, X. Improved SiamRPN++ with Clustering-Based Frame Differencing for Object Tracking of Remote Sensing Videos. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium IGARSS, Brussels, Belgium, 11–16 July 2021; pp. 4163–4166.
18. Li, B.; Wu, W.; Wang, Q.; Zhang, F.; Xing, J.; Yan, J. SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4282–4291.
19. Yang, J.; Pan, Z.; Liu, Y.; Niu, B.; Lei, B. Single Object Tracking in Satellite Videos Based on Feature Enhancement and Multi-Level Matching Strategy. *Remote Sens.* **2023**, *15*, 4351. [CrossRef]
20. Woo, S.; Park, J.; Lee, J.Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In *Computer Vision—ECCV 2018, Proceedings of the 15th European Conference on Computer Vision, Munich, Germany, 8–14 September 2018*; Springer: Cham, Switzerland, 2018. [CrossRef]
21. 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.* **2021**, *43*, 1562–1577. [CrossRef] [PubMed]
22. 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 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 15–20 June 2019; pp. 5369–5378. [CrossRef]
23. Krizhevsky, A.; Sutskever, I.; Hinton, G. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
24. Wu, Y.; Lim, J.; Yang, M.-H. Online object tracking: A benchmark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Portland, OR, USA, 23–28 June 2013; pp. 2411–2418.
25. Zhang, Z.; Peng, H.; Fu, J.; Li, B.; Hu, W. Ocean: object-aware anchor-free tracking. In Proceedings of the Computer Vision-ECCV 2020: 16th European Conference, Glasgow, UK, 23–28 August 2020; Springer International Publishing: Berlin/Heidelberg, Germany, 2020; Part XXI 16, pp. 771–787.
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, 15–20 June 2019; pp. 4660–4669.
27. Li, P.; Chen, B.; Ouyang, W.; Wang, D.; Yang, X.; Lu, H. GradNet: Gradient-guided network for visual object tracking. In Proceedings of the International Conference on Computer Vision, Seul, Republic of Korea, 27 October–2 November 2019; pp. 6162–6171.
28. Zhang, Z.; Peng, H. Deeper and Wider Siamese Networks for Real-Time Visual Tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 4591–4600.
29. Danelljan, M.; Hager, G.; Shahbaz Khan, F.; Felsberg, M. Learning spatially regularized correlation filters for visual tracking. In Proceedings of the International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 4310–4318.

30. Zhang, G.; Li, Z.; Li, J.; Hu, X. CFNet: Cascade Fusion Network for Dense Prediction. *arXiv* **2023**, arXiv:2302.06052.
31. Mueller, M.; Smith, N.; Ghanem, B. A benchmark and simulator for uav tracking. In Proceedings of the European Conference on Computer Vision, Amsterdam, The Netherlands, 11–14 October 2016; pp. 445–461.
32. Fei, D. Research on Visual Target Tracking Method Based on Attention Mechanism. Master's Thesis, Harbin Institute of Technology, Harbin, China, 2021. [CrossRef]
33. Chen, Z.; Zhong, B.; Li, G.; Zhang, S.; Ji, R. Siamese box adaptive network for visual tracking. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 6668–6677.
34. Guo, D.; Wang, J.; Cui, Y.; Wang, Z.; Chen, S. SiamCAR: Siamese fully convolutional classification and regression for visual tracking. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 6268–6276.
35. Guo, D.Y.; Shao, Y.Y.; Cui, Y.; Wang, Z.; Shen, C. Graph attention tracking. In Proceedings of the 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 9538–9547.
36. Cao, Z.; Fu, C.; Ye, J.; Li, B.; Li, Y. SiamAPN++: Siamese Attentional Aggregation Network for Real-Time UAV Tracking. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021; pp. 3086–3092.
37. Shen, H.; Lin, D.; Song, T. A real-time siamese tracker deployed on UAVs. *J. Real-Time Image Process.* **2022**, *19*, 463–473. [CrossRef]
38. Chen, Y.; Tang, Y.; Xiao, Y.; Yuan, Q.; Zhang, Y.; Liu, F.; He, J.; Zhang, L. Satellite video single object tracking: A systematic review and an oriented object tracking benchmark. *ISPRS J. Photogramm. Remote Sens.* **2024**, *210*, 212–240. [CrossRef]
39. Zhou, J.; Wang, P.; Sun, H. Discriminative and Robust Online Learning for Siamese Visual Tracking. In Proceedings of the 34th AAAI Conference on Artificial Intelligence, New York, NY, USA, 7–12 February 2020; pp. 13017–13024.
40. Dong, X.; Shen, J.; Shao, L.; Porikli, F. CLNet: A Compact Latent Network for Fast Adjusting Siamese Trackers. In Proceedings of the Computer Vision—ECCV 2020, 16th European Conference, Glasgow, UK, 23–28 August 2020. [CrossRef]
41. Zheng, G.; Fu, C.; Ye, J.; Li, B.; Lu, G.; Pan, J. Scale-Aware Siamese Object Tracking for Vision-Based UAM Approaching. *IEEE Trans. Ind. Inform.* **2023**, *19*, 9349–9360. [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.