Heliyon 11 (2025) e41612

Contents lists available at ScienceDirect

Heliyon

journal homepage: www.cell.com/heliyon

Research article

Three-stage cascade architecture-based siamese sliding window network algorithm for object tracking [☆]

Zheng Yang [a], Kaiwen Liu [b], Quanlong Li [b], Yandong Hou [b], Zhiyu Yan [a,*]

[a] *School of Electrical Engineering, Yellow River Conservancy Technical Institute, Dongjing street, Kaifeng, 475004, Henan, China*
[b] *School of Artificial Intelligence, Henan University, Mingli street, Zhengzhou, 450000, Henan, China*

A R T I C L E   I N F O

Keywords:
Single object tracking
Deep learning
Siamese networks
Sliding window
Three-level cascade

A B S T R A C T

To enhance the correlation of feature information and enrich the pattern of cross-correlation metrics, we propose the Siam ST algorithm, which is based on a three-stage cascade (TSC) architecture. The sliding window is introduced in the last three layers of convolution blocks, which can obtain the global information of images and fully capture the target feature. The TSC structure is developed by using the regional proposal network. It makes the features of the current frame interact with the previous frame. As a result, our method has a high effect of robustness and association features extraction. Therefore, our ablation experiments are conducted on the VOT2016 dataset, and comparison experiments are conducted on four datasets, VOT2018, LaSOT, Tracking Net, and UAV123. Our proposed algorithm demonstrates a significant improvement compared to SiamRPN++ across four datasets.

1. Introduction

In recent years, with the development of deep learning, single object tracking tasks have become a major branch of computer vision [1]. It has a very wide range of applications in the fields of human-computer interaction, unmanned driving, medical imaging, intelligent security, etc [2–4]. The primary task of single object tracking is to learn target features across different frames to obtain the target's motion trajectory throughout the video data. However, this task presents several challenges, such as decreased accuracy due to appearance variations caused by fast motion, occlusion, and complex backgrounds, as well as reduced tracking precision from insufficient feature extraction. Additionally, there is the issue of balancing computational efficiency and accuracy. Many research teams have proposed methods to address these challenges, and it can be divided into a generative model and a discriminative model based on different observation models.

The generative single object tracking algorithm constructs an appearance model of the tracking target by using the posterior probability, and then searches for the target region that best matches the known model based on the reconstruction error minimization in subsequent frames. Representative generative algorithms include principal component analysis [5], incremental learning [6], etc. Wang [7] reconstructed the target template using an inverse sparse representation formula and locally weighted distance measurement, allowing the tracker to complete the tracking task by solving only a single particle optimization problem. However, the model

☆ This work was supported in part by the National Natural Science Foundation of China (No. 61374134) and in part by the Natural Science Foundation of Henan Province (No. 232300421149).

* Corresponding author.
E-mail addresses: hhsykfyz@163.com (Z. Yang), messi@henu.edu.cn (K. Liu), 1750550897@qq.com (Q. Li), hydong@henu.edu.cn (Y. Hou), yan_yu123@126.com (Z. Yan).

https://doi.org/10.1016/j.heliyon.2024.e41612
Received 10 July 2024; Received in revised form 30 December 2024; Accepted 31 December 2024

Available online 6 January 2025

exhibits poor adaptability to appearance changes, and its performance significantly degrades when there are substantial background variations or interference. Zhang [8] proposed a method that decomposes the target into multiple sub-targets, using an optimization function based on the relative positions between sub-targets to correct and accurately update the target's size and movement direction, thereby enhancing the algorithm's robustness. Nevertheless, the accuracy of the algorithm decreases during rapid target motion or occlusion, and the method's computational efficiency is low due to the need to track multiple sub-targets, making it unsuitable for real-time tracking applications. In addition, due to the limitations of these algorithms in feature extraction, especially in cases where feature correlation is weak, generative methods often fail to fully capture the deep features of the target, leading to a decline in tracking performance.

With the development and application of artificial intelligence, deep learning-based single-object tracking algorithms have gradually become a mainstream method for tracking tasks. As a discriminative model, the Siamese network has emerged as a key research focus in deep learning for object tracking. The Siamese Network is a type of neural network architecture that typically consists of two or more subnetworks sharing the same weights. Compared to traditional generative methods, the Siamese network can more efficiently accomplish tracking tasks by constructing similarity measures to directly compare the target features between the initial frame and the current frame. Luca [9] proposed a siamese network structure—based Siam FC for target tracking tasks. It has a two-branch structure with shared parameters to obtain the feature information of the initial frame target and the current frame target, respectively. Their cross-correlation is calculated by the distance function. However, this model is unable to capture the deep features of the target, which results in poor performance and limitations when dealing with significant background changes and occlusion. In paper [10], the Siam RPN algorithm including region proposal network (RPN) was proposed, which includes two branches: classification and regression. The 2k classification results and 4k regression results are obtained by convolving the initial frame target feature with the current frame target feature to achieve the target tracking task. However, this model fails to capture multi-scale and multi-level features, and is unable to combine deep and shallow features in complex scenes, resulting in insufficient feature correlation. Consequently, despite the significant improvement in target localization accuracy achieved by incorporating the RPN module, Siam RPN still exhibits considerable limitations in handling complex backgrounds and extracting feature correlations. Li et al. [11] proposed the Siam RPN++ algorithm with the goal of improving the performance of Siam RPN in situations where the target is easily lost in complex backgrounds. By introducing a spatially aware sampling strategy, the algorithm enhances the translation invariance of target positioning, thereby improving its performance in scenarios involving complex backgrounds, multi-object interference, and rapid target motion. However, it still has certain limitations in terms of computational complexity, reliance on training data, and handling significant variations in target appearance.

The above siamese network-based target tracking algorithm has made great progress, but the following problems still exist: 1) Although the residual branches is added to the Residual Neural Network (Resnet) structure (backbone of siamRPN++), the lack of correlation between image features will lead to low efficiency of target feature extraction. 2) The initial frame feature template is used as a convolution kernel to convolution with the current frame feature template in RPN. However, this algorithm-limits the richness and reliability of target matching.

To enhance the performance of single-object tracking, this paper proposes a model named Siam ST. Specifically, the innovations of this paper are as follows: (1) A Siam ST algorithm based on a three-stage cascade (TSC) architecture: This algorithm is designed to enhance feature correlation and improve tracking accuracy, particularly in scenarios with complex backgrounds and fast-moving targets, significantly improving the model's robustness and tracking performance. (2) Introduction of a sliding window mechanism to enhance feature extraction capability: A sliding window mechanism is incorporated into the last three layers of the convolutional network, effectively capturing global image information and strengthening the correlation between different regions, thereby significantly improving the model's feature extraction capability, especially in complex scenes. (3) Interaction between current and previous frame features through the TSC structure: The TSC structure enables efficient interaction between the current frame and previous frame features, allowing the model to capture multi-scale and deep features more effectively, thus improving its ability to handle occlusion and complex backgrounds.

2. Basic theory approach

2.1. Residual network

Compared to traditional machine learning algorithms, deep learning algorithms offer superior flexibility in end-to-end model building. Many scholars have developed excellent models using deep learning techniques. The residual network [12] model is particularly popular, as it utilizes residual learning instead of unreferenced function learning, which effectively addresses the problem of vanishing gradient when deepening the network structure and mining features.

As shown in Fig. 1, the residual convolutional layer differs from the regular convolutional layer in that it has an additional residual branch on the main trunk. The residual branch adds the input features, which have undergone dimension transformation, to the processed features of the convolutional layer.

The convolutional block of ResNet is composed of a BTNK1 module and several BTNK2 modules. As shown in parts (b) and (c) above, the BTNK1 module differs from the BTNK2 module in that it has a convolutional layer for the residual branch. This ensures consistency in the number of features channels between the backbone and branch. This structure leads to stable and efficient deep feature mining.

In extensive studies, ResNet34 and ResNet50 are constructed by layered stacking of multiple residual convolution blocks. In addition, ResNet101 has stronger feature mining capability with the support of high performance computers.

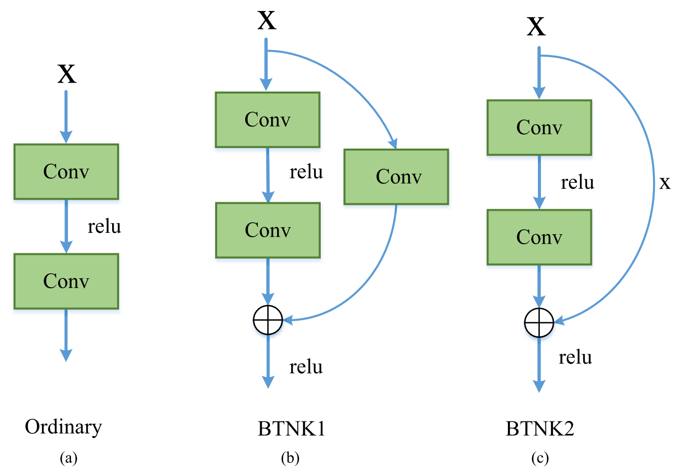

Fig. 1. Residual branch structure diagram.

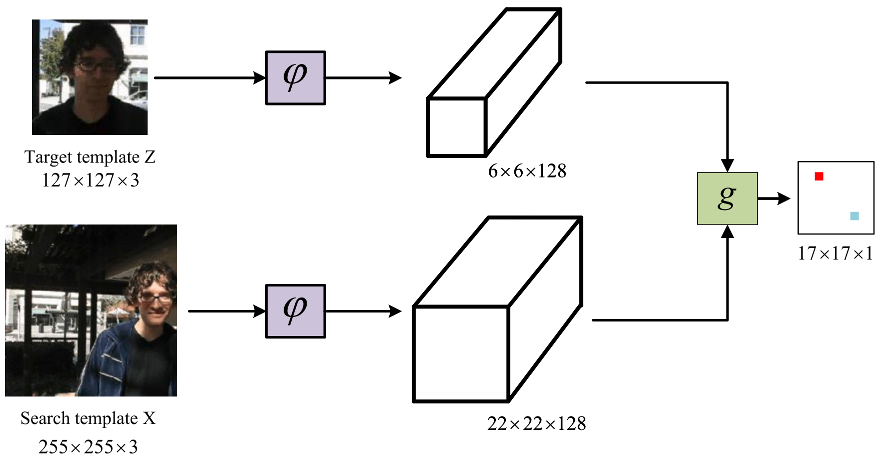

Fig. 2. Siam Network architecture diagram.

2.2. Siamese network

Because online learning limits the richness of model learning capabilities to a certain extent, and seriously affects the real-time speed of the system. Therefore, the end-to-end Siam FC algorithm based on the initial offline stage has been unanimously recognized by the majority of scholars. The Siam FC consists of two neural networks with the same structure that share weights. Its essence is to map the input to a new space and form a new representation, and then calculate the degree of similarity between the two inputs through the loss value. The basic framework of the Siam FC is shown in Fig. 2. From the above figure, the target template z of size $127 \times 127 \times 3$ and the search template x of size $255 \times 255 \times 3$ are used to generate the image features of size $6 \times 6 \times 128$ and size $22 \times 22 \times 128$ respectively by the feature extraction network function φ, and then the response map of size $17 \times 17 \times 1$ calculated by the mutual correlation function g.

The Siamese Network similarity learning is shown in Equation (1):

$$\mathrm{f}(z) = g(\varphi(z), \varphi(x)) \tag{1}$$

In the above equation, function f representation similarity learning method, z denotes the initial frame target template of sample image frame, x indicates that the search image target of the current frame, function φ represents the feature extraction of the image by the backbone network, function g represents the inter correlation measure function between the template features and the current frame features. Specifically, When the function g is a simple distance or similarity metric, the function f can be considered as an embedding function.

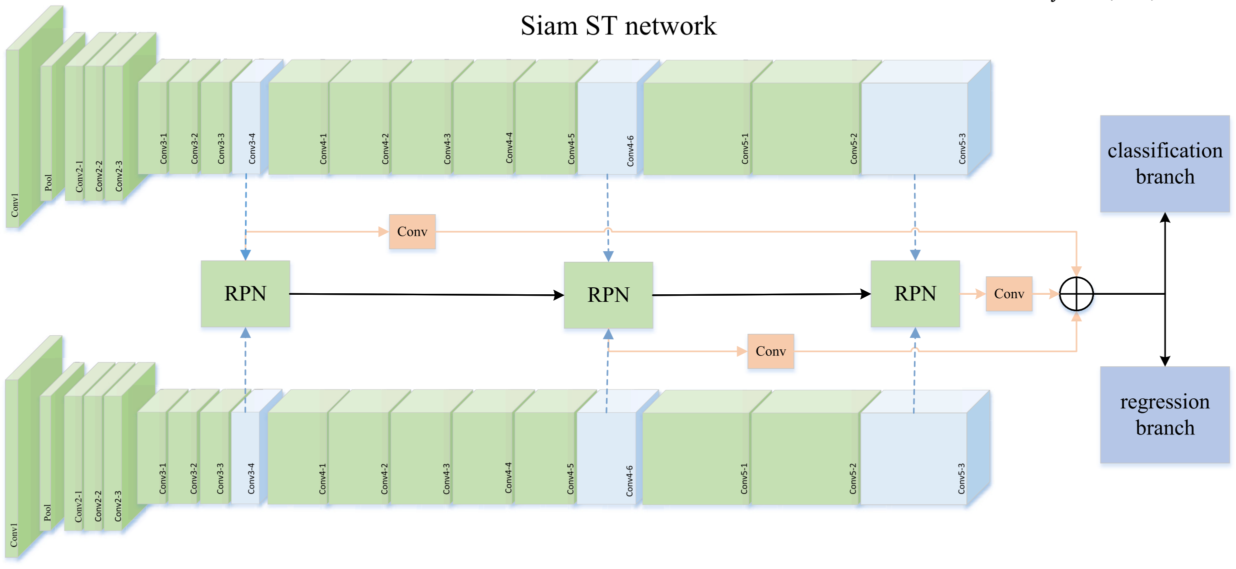

Fig. 3. Siam ST structure diagram.

3. Siam ST Net

To address the lack of information interaction between different regions of the image, and the singularity of target matching in the region suggestion network. We proposed a sliding window twin network tracking algorithm based on TSC architecture. Our model first leverages a sliding window mechanism embedded within the backbone network, utilizing local windows and global cyclic shifting operations to extract multi-scale low-dimensional, mid-dimensional, and high-dimensional features from the input current and adjacent frames. This approach significantly enhances the global receptive field of feature extraction and improves the correlation between different regions. Subsequently, the extracted features are aggregated through the Three-Stage Cascade (TSC) mechanism, integrating cross-frame contextual information to further strengthen feature correlation. Additionally, a structural re-parameterization strategy is introduced to optimize the computational efficiency of multi-branch fusion, effectively reducing hardware costs. Finally, the fused features are input into the Region Proposal Network (RPN) for classification and regression tasks. The specific framework is shown in Fig. 3.

Firstly, the residual network with added sliding window is used as the main feature extraction network of the siamese architecture in this paper, so as to extract the target for each subsequent frame separately. Because it follows the siamese network structure, these two feature extraction networks share parameters. To cope with the different strategies, we call the last three modules of the backbone low-dimensional features, mid-latitude features and high-dimensional features. Second, the target features extracted from the three different dimensional dimensions are used to generate diverse suggestion results through TSC and RPN, respectively. The results of each branch are superimposed and aggregated to achieve the final determination, and generate classification and regression outputs.

3.1. Residual networks with sliding windows

In this paper, structural changes are made to ResNet50, and the specific structure is shown in Fig. 4. Part (a) shows the basic network framework of Resnet50, which is mainly divided into 4-layer convolutional structures, each layer of convolution contains 3, 4, 6, 3 convolutional blocks; part (b) shows the residual structure of the original convolutional residual block, which mainly contains forward propagation branch and residual branch; Part (c) shows an improvement on the residual network, which consists mainly of forward propagation branches, external residual branches, and internal residual branches.

In the forward propagation branch, the sliding window module is added to the backbone network to better obtain the global information of the image, which is conducive to capturing the information correlation between different regions of the image itself. An internal residual branch is added between the first and third layers of convolution to facilitate weakening the negative impact of similar targets on the sliding window module. To avoid parameter explosion, we only modified the three convolutional blocks of the backbone. Experiments show that the sliding window convolution residual structure has a more ideal effect than the original structure. The proposed feature extraction method can be seen from Algorithm 1.

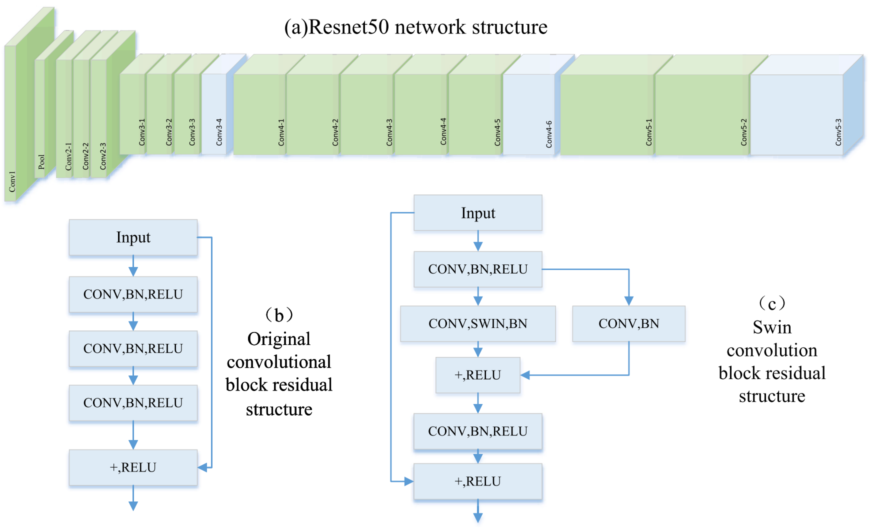

Fig. 4. Slide window residual convolution structure diagram.

Algorithm 1 Swin module with residual connection for feature extraction.

```
Input: Initial feature maps $F(Z)$ and $F(X)$, window size $M$, shifting step size $S$
Output: Enhanced feature map $F'(X)$

1: for each feature map $F(X)$ do
    2: Store the original feature map as $F_{\text{residual}}(X)$
    3: Convolution operation: Calculate local features using
        $y_i^l = W_i * x_i^{l-1} + b_i$
    where $W_i$ and $b_i$ are the weight and bias of the convolution layer
    4: Apply sliding window of size $M \times M$ to extract local features
    5: Normalization: Perform Batch Normalization
        $\hat{y}_i^l = \frac{y_i^l - E(y_i^l)}{\sqrt{Var(y_i^l)}}$
    6: Hierarchical merging: Combine patches within each window to form merged features
    7: Apply cyclic shifting with step size $S$ to capture cross-window interactions
    8: Masked multi-head self-attention: Calculate self-attention using
        $Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d}} + B\right)V$
    9: Aggregate features to form the intermediate feature map $\hat{F}(X)$
    10: Add residual connection:
        $F'(X) = \hat{F}(X) + F_{\text{residual}}(X)$
11: end for
```

In order to effectively capture the local area features of the image, the Swin module realizes the hierarchical feature by merging small-size image. The small size images converge the information from different windows in the same window by means of cyclic shift. Finally, the information correlation between different windows is obtained by self-attention. The hierarchical representation of the feature image is shown in Fig. 5.

First, the image features are split by the split module into N patches of equal height and width, and each patch has a side length of P. As shown in Part (a), the feature is split into 16 patch blocks, and each of block represents a valid sequence length, which is determined by the patch edge length and the number of image channels. The product of patch size and channels number is called a patch token. Secondly, the tensor represented by each patch is projected into any dimension through the linear embedding layer to obtain a new linear embedding result, which is delivered to the Swin module with self-attention mechanism.

As shown in Figure (b) above, the merge layer in this module merges the number of adjacent patches per group 2×2 into one, and the number of tokens is reduced to $\frac{1}{4}$, but the dimension of the tokens rises by 4 times. Because the feature maps are not of the same size in different backbone network layers, the patch tensor received by the Swin module is also different. The hierarchical representation strategy of windows builds a hierarchical representation by gradually merging adjacent window patches, and expands

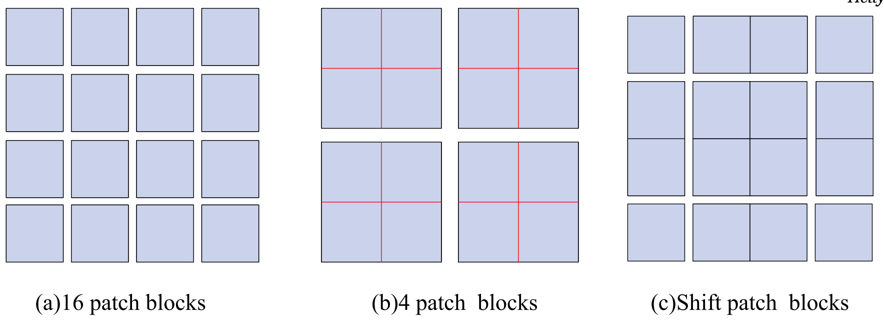

Fig. 5. Window change trend graph.

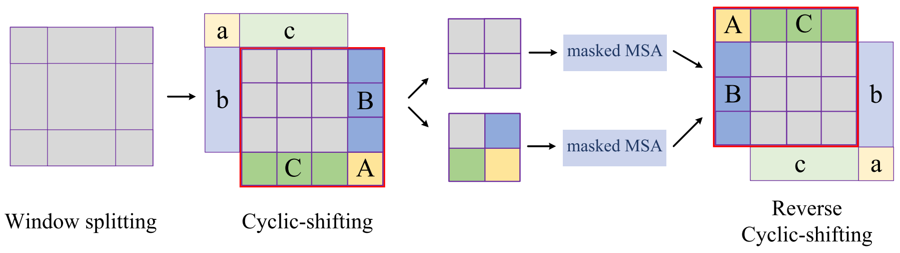

Fig. 6. Cyclic-shifting diagram.

the originally limited receptive field to the global level. The hierarchical representation of feature maps enables dense prediction of anchor boxes in the split window. This method has good gain for image feature capture without increasing the heavy computational burden.

After a hierarchical representation, image features undergo a cyclic-shifting strategy to fully capture the correlation between their neighboring windows as shown in Figure (c) above. In the process of cyclic-shifting, it is easy to have a window size less than $M \times M$ and the number of additional windows is increased, so we proposed an approach by cyclic-shifting toward the top-left direction to solve this problem. The sliding window change rule is shown in Fig. 6.

As shown in the figure above, after the image feature is segmented, it begins to change in the way of cyclic shift to the upper left, and the sub windows a, b, and c are moved to the positions corresponding to A, B, and C. The batch window in the lower right corner area can be composed of four non-adjacent sub-windows in the feature map, which can reflect the advantages of cross-window correlation calculation of this strategy. Its feature region covers most of the original image global position, which provides reliable insights for subsequent target correlation calculations in a larger receptive field. Through the cyclic-shifting strategy, the number of batch windows is still consistent with the number of windows in the rule partition (the number of windows divided by the rule remains unchanged after cyclic shift to the upper left corner, as shown in A, B, and C in the figure above). Because cyclic shifting causes the patch window to contain features from different windows. When using the masked MSA mechanism, we calculate the self-attention normally, and then fill the unwanted attention value to 0 through the mask operation. It can limit self-attention calculations to sub windows. In the case of global calculation self-attention, it will greatly increase the cost of operation. However, window-based self-attention has a certain degree of scalability, which can effectively avoid this problem and achieve simple and efficient quasi-global calculation.

The shift window-based Swin module consists of an MSA module (window multi-head self-attention), a SW-MSA module (shift window multi-head self-attention), and a two-layer MLP structure sandwiched with GeLU nonlinearity. The LayerNorm (LN) is set between each MSA module and each MLP module. In addition to that, residual connections are applied after each module. The specific flow of the Swin module is shown in Fig. 7 (split modules and merge layers are not additionally represented in the diagram).

The windows that divide the image evenly without overlapping help us reduce the computational burden, and self-attention calculations in the local window will also help us model efficiently. Regarding the complexity of the W-MSA module calculation as shown in Equation (2).

$$\Omega(W - MSA) = 4hwC^2 + 2M^2hwC \tag{2}$$

In the above equation, M represents the size of each window; h and w represent the width and height dimensions of the whole feature map; C represents the number of channels of the feature.

Compared with W-MSA, SW-MSA can better introduce cross-window interaction while maintaining the computational efficiency of non-overlapping windows. As shown in Fig. 5 part (b) and part (c), it uses the regular window division strategy to evenly divide the feature map into 4 windows of consistent size starting from the feature area in the upper left corner (the local window size is set to M

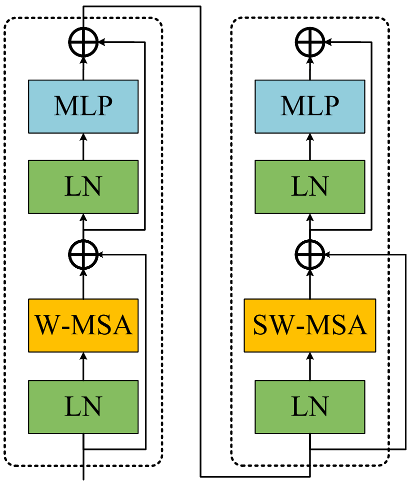

Fig. 7. Swin module.

$= 4$ in this paper). Next, the following module utilizes a window configuration that is shifted from the previous layer. Specifically, the regular division window is cyclically shifted by $\left(\left\lfloor\frac{M}{2}\right\rfloor, \left\lfloor\frac{M}{2}\right\rfloor\right)$ pixels up to the left.

Using the shift window division strategy, Fig. 7 is calculated as shown in Equation (3):

$$\begin{aligned} \hat{z}^{l} &= W-MSA(LN(\hat{z}^{l-1})) + z^{l-1} \\ z^{l} &= MLP(LN(\hat{z}^{l})) + \hat{z}^{l} \\ \hat{z}^{l+1} &= SW-MSA(LN(\hat{z}^{l})) + z^{l} \\ z^{l+1} &= MLP(LN(\hat{z}^{l+1})) + \hat{z}^{l+1} \end{aligned} \tag{3}$$

where $\hat{z}^{l}$ and z^{l} represent the (S)W-MSA and MLP output characteristics in block l.

When calculating self-attention, the relative position deviation of each head needs to be measured according to Equation (4):

$$Attention(Q,K,V) = SoftMax(QK^{T}/\sqrt{d} + B)V \tag{4}$$

In the above equation, $Q,K,V \in R^{M^2 \times d}$ Represents the query matrix, key matrix, and value matrix, M^2 Represents the total number of patches in a window, and $B \in R^{M^2 \times M^2}$ represents the relative position deviation to indicate the relative position between patches. In order to reduce the amount of our parameters, by defining $\hat{B} \in R^{(2M-1)\times(2M-1)}$, a certain amount of memory storage space can be effectively freed. This is because the patches in each window have a total of $2M-1$ fetched values, and using 2D relative position encoding, the relative position can be made $(2M-1)\times(2M-1)$.

3.2. *Three-stage cascade in the RPN*

In the Siam RPN++ algorithm model, when the backbone network has fully mined the features, the extracted image features are fed to the regional propose network. First, the input features are upscaled by convolution to the dimensions of $2k$ and $4k$ (k representing the number of anchor boxes set for each anchor position). Secondly, the convolution feature is realized by the sliding window method (The sliding window here differs from the previous section in that it only has simple window movement operations, no cyclic displacement and embedding of the Swin module), and the template feature is used as the convolution kernel to achieve cross-convolution with the search area feature. Finally, the $2k$ classification results and $4k$ regression results are obtained.

In our TSC structure, the mutual convolution between the current frame feature and the adjacent frame feature is realized by aggregating branches. It makes the output results are superimposed and aggregated with the RPN output to achieve the association of contextual information. In the aggregation branch, the structure of three branches in different states is transformed into an 3×3 convolutional branch by structural re-parameterize. This strategy allows for efficient inter-off tasks without incurring high costs in terms of hardware infrastructure. The specific process of the TSC regional recommendation network is shown in Fig. 8.

As can be seen from the above figure, the region proposal network consists of three regional proposal subnetworks, and the black arrows indicate the weight calculation and output strategy of the traditional RPN module. The blue arrows indicate the cascade branches through which each submodule passes. Under the action of the convolutional layer, the input features in the RPN are transformed to the same size of output features, and anchor classification results and regression results are produced.

In order to avoid the situation that gaps between frames from causing significant changes in target characteristics and background interference. We perform multiple sequence tests of the same target at each cascade branch using the feature template of the target at frame $i-1$ as the proposed convolution kernel. Unlike the traditional RPN, after going through the region proposal network, the current frame target feature template not only has high similarity to the initial target, but also has high similarity to the neighboring frame targets. A high response value of a target feature often indicates its accuracy over the entire video sequence.

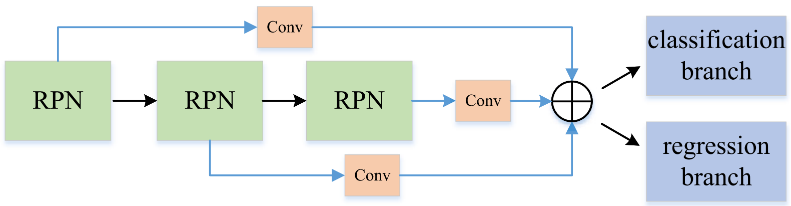

Fig. 8. Three-stage cascade regional proposal network.

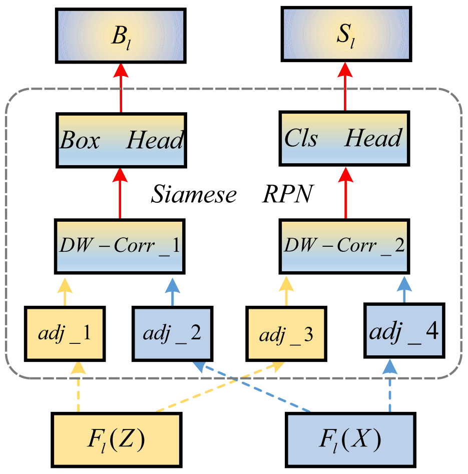

Fig. 9. Regional proposal network diagram.

In the RPN module, we still follow the cross-correlation strategy of Siam RPN++, and its internal specific framework is shown in Fig. 9.

In the picture above, B_l and S_l represent anchor frame regression results and classification results; $F_l(Z)$ and $F_l(X)$ represent real template features and search image features, respectively; $DW - Corr$ represents two non-identical deep convolution kernels; adj represents four sets of high-dimensional features, which are used for classification and regression after quadratic convolution.

In order to compare similarities more effectively, we will perform hierarchical aggregation of multi-level features extracted from the last three residual blocks of the sliding window feature extraction network. Such a strategy can effectively compare the similarity between the template target and the tracked target. Since the output sizes of the three RPN modules have the same spatial resolution, a weighted sum is applied directly to the RPN outputs. The specific calculation is shown in Equation (5):

$$\begin{aligned} S_{all} &= \sum_{l=3}^{5} \alpha_i * S_l \\ B_{all} &= \sum_{l=3}^{5} \beta_i * B_l \end{aligned} \tag{5}$$

where α and β represent different weight matrices. Since we will use a TSC structure instead of a single-branch progressive method to process correlation operations, we will largely avoid situations such as negative decision amplification and similarity metric simplification. However, considering that multiple branches often bring a heavy computational burden and increase the cost of running memory. Therefore, we use the structural re-parameterization strategy to solve the above situation and verify the effectiveness of this strategy through experiments.

Through the structural re-parameterization strategy, the network can have higher performance while taking into account the efficient inference speed. When training, try to use multi-branch structure to improve network performance, and use structural re-parameterization method to change it into a single-channel structure during inference. It can make the memory use less and the inference speed faster. The structure of each stage is shown in Fig. 10.

We construct the training time network using itself and the 1×1 convolutional branch and remove the branches by structural re-parameterization. Because the constant branch can be considered as a degenerate 1×1 convolution, and the latter can be further considered as a degenerate 3×3 convolution. From this, we can construct a single 3×3 kernel using the training parameters of the original 1×1 kernel, the constant and 3×3 branches, and the batch normalization (BN) layer.

In this paper, we use a convenient branch to model the information flow as $y = f(x) + x$, and change it to $y = g(x) + f(x)$ when x and $f(x)$ cannot reach unity in dimensionality, where $g(x)$ is implemented by convolution.

In particular, to fuse the convolutional layer with the batch normalization (BN) layer, the convolution formula (Equation (6)) and the BN layer formula (Equation (7)) are combined to yield Equation (8).

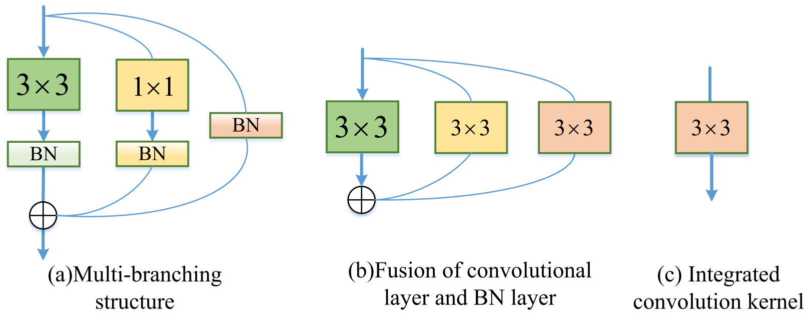
3×3
1×1
BN
BN
BN
(a)Multi-branching structure
3×3
3×3
3×3
(b)Fusion of convolutional layer and BN layer
3×3
(c) Integrated convolution kernel

Fig. 10. Schematic diagram of structural re-parameterization.

$$Conv(x) = W(x) + b \tag{6}$$

$$BN(x) = \gamma * \frac{x - mean}{\sqrt{var}} + \beta \tag{7}$$

$$BN(Conv(x)) = \gamma * \frac{W(x) + -mean}{\sqrt{var}} + \beta \tag{8}$$

Further simplified as Equation (9).

$$BN(Conv(x)) = \frac{\gamma * W(x)}{\sqrt{var}} + (\frac{\gamma * (b - mean)}{\sqrt{var}} + \beta) \tag{9}$$

This is actually still a convolutional layer, except that the weights take into account the parameters of BN, so that the fused weights and biases are computed as shown in Equation (10) and Equation (11):

$$W_{fused} = \frac{\gamma * W(x)}{\sqrt{var}} \tag{10}$$

$$B_{fused} = (\frac{\gamma * (b - mean)}{\sqrt{var}} + \beta) \tag{11}$$

The final fused result is obtained as shown in Equation (12).

$$BN(Conv(x)) = W_{fused} + B_{fused} \tag{12}$$

To obtain the integrated convolution kernel, this paper implements the flow from part (a) to part (c) in Fig. 2-7 through Equation (13):

$$\begin{aligned} M^{(2)} = {} & \mathrm{bn}(\mathrm{M}^{(1)} * \mathrm{W}^{(3)}, \mu^{(3)}, \delta^{(3)}, \gamma^{(3)}, \beta^{(3)}) \\ & + \mathrm{bn}(\mathrm{M}^{(1)} * \mathrm{W}^{(1)}, \mu^{(1)}, \delta^{(1)}, \gamma^{(1)}, \beta^{(1)}) \\ & + \mathrm{bn}(\mathrm{M}^{(1)}, \mu^{(0)}, \delta^{(0)}, \gamma^{(0)}, \beta^{(0)}) \end{aligned} \tag{13}$$

where $W^{(3)} \in R^{C_2 \times C_1 \times 3 \times 3}$ is used to denote the 3×3 convolution kernel with C_1 input channel and C_2 output channel; $W^{(1)} \in {}^{C_2 \times C_1}$ is used to denote the 1×1 branch convolution kernel; $\mu, \delta, \gamma, \beta$ are used to denote the cumulative mean, standard deviation value, learning scaling factor and deviation of the BN layer, respectively; and $*$ is used to denote the convolution operator. The algorithm proposed in this module is shown in Algorithm 2 below.

4. Experiment

4.1. Experimental facilities and setup

The experiments were all run on a desktop computer with a GPU of Nvidia GTX 2080ti, an operating system of 64-bit Ubuntu 18.04, a processor of Intel(R) Core(TM) i5-10400F with a main frequency of 2.9GHz, 32G of RAM, and a programming environment of Python 3.7.

During the training process, we experimented with different hyperparameter settings such as batch size, learning rate, and weight decay to evaluate their impact on model performance. For batch size, we tested values of 64, 128, and 256. From these experiments, we observed that a larger batch size (256) helped improve the convergence speed and yielded greater stability and accuracy across multiple datasets. For the learning rate, we experimented with values of 0.01, 0.001, and 0.0001. The results showed that a lower learning rate (ranging from 0.1 to 0.001) effectively prevented the model from getting stuck in local minima during early training, ensuring greater robustness. In terms of weight decay, we tested values between 0.0001 and 0.01. The results indicated that a smaller weight decay helped control the model's complexity and prevented overfitting. We evaluated the performance of different hyperparameter combinations on both the validation and test sets. Ultimately, we selected a batch size of 256, an initial learning

Algorithm 2 Three-stage Cascade (TSC) in the RPN.

```
Input: Current feature $F(X)$, adjacent feature $F_{adj}(X)$, template $F(Z)$, weights $\alpha, \beta$
Output: Bounding box parameters $B_{all}$

1: Initialize RPN with features.
2: Upscale features using convolution to dimensions $2k$ and $4k$.
3: Cross-correlate $F(Z)$ with $F(X)$.
4: Stage 1: Extract features from $F(X)$.
    4.1: Fuse with BN:
        $W_{\text{fused}} = \frac{\gamma \cdot W(x)}{\sqrt{\text{var}}}$
    4.2: Calculate fused bias:
        $B_{\text{fused}} = \left( \frac{\gamma(b - \text{mean})}{\sqrt{\text{var}}} + \beta \right)$
5: Stage 2: Aggregate current frame features $F(X)$ with adjacent frame features $F_{adj}(X)$.
6: Stage 3: Fuse features into a single branch using structural re-parameterization.
7: Return bounding box parameters $B_{all}$.
```

rate of 0.1, a cosine annealing schedule with 30 cycles, a momentum factor of 0.9, and a weight decay of 10^{-4}, using the standard SGD optimizer. These hyperparameters were fine-tuned through multiple experiments and were confirmed to significantly improve model performance. For the residual network incorporated into the Swin module, we experimented with different window sizes (set to $M = 7$, $M = 4$, and $M = 2$) based on the feature map size. For all experiments, the query dimension of each head was set to $d = 16$, and the expansion factor for each MLP was set to $\alpha = 4$. Additionally, the initial learning rate was set to 0.001, with a weight decay of 0.01. During the fine-tuning process, the model was further trained for 30 epochs with a learning rate of 10^{-5} and a weight decay of 10^{-8}.

4.2. Data set and evaluation indicators

Five datasets, VOT2016 [13], VOT2018 [14], LaSOT [15], Tracking Net [16], UAV123 [17] were used in this experiment. The VOT series datasets are characterized by rich color information and high resolution, providing more semantic details in the video images. VOT2016 is a commonly used dataset in the field of visual object tracking, featuring abundant color information and high-resolution video frames. The semantic richness in these frames presents multidimensional challenges for object tracking tasks. VOT2016 encompasses a wide variety of scenarios, object types, and background variations, making it highly effective for testing the robustness of tracking algorithms in complex environments. Therefore, this dataset offers a comprehensive performance evaluation for the tracking algorithms in this study, particularly in how they handle high-resolution and complex visual information. VOT2018 maintains 60 video sequences but “has refined lower-quality sequences and optimized video annotations, resulting in a more rigorous evaluation benchmark compared to VOT2016, which helps assess the performance of algorithms in real-world application scenarios.

LaSOT is a large-scale, high-quality video tracking dataset. It comprises 1,400 video sequences, averaging 2,512 frames per sequence, providing sufficient samples for models across various scenarios. This dataset includes a total of 70 object categories, each with 20 sequences, encompassing a wide range of real-world challenges, which facilitates testing an algorithm’s generalization capability and robustness across multi-category scenarios. Moreover, every frame in this dataset has been manually annotated and rigorously verified, ensuring labeling accuracy and consistency, providing a high-quality benchmark for large-scale object tracking. Twenty percent of this dataset has been designated as our test set, which includes a total of 280 video sequences.

Tracking Net is a collection of 30,643 video clips, specifically selected and re-annotated for object tracking from a large object detection dataset. This dataset provides a wide variety of objects and scene types. The test set consists of 511 videos and 70 object categories. “Manual annotations are not provided for the test set of this dataset, and evaluations can only be performed through an online server, ensuring a fair comparison. Due to its scale and diversity of scene types, TrackingNet is a significant benchmark dataset in the field of object tracking, capable of effectively testing an algorithm’s generalization ability in large-scale and real-world scenarios.

UAV123 dataset primarily consists of ground videos captured by UAVs in 720P resolution at 30 FPS. It includes 123 video sequences, totaling 112,578 frames, with the shortest sequence containing 109 frames and the longest sequence containing 3,085 frames. The unique feature of this dataset lies in its diverse shooting angles, significant background variations, and drastic changes in object appearance. As a result, this dataset is well-suited for testing models’ performance in scenarios with rapid changes in viewpoint and background, effectively reviewing the learning rate of algorithm models to a certain extent.

The VOT2016 and VOT2018 datasets discriminate the results of the experiments related to this paper by the performance metrics Accuracy (A), Robustness (R), and Expect average overlap rate (EAO). The LaSOT dataset uses the One-pass Evaluation (OPE) strategy with Precision (P), Normalized Precision (NP), and Success (S) metrics, and the UAV123 dataset measures the distance between the prediction results and the true results by Precision and Success metrics, respectively. The UAV123 dataset measures the distance between the predicted results and the true results, and the intersection ratio between the two, respectively.

Table 1
Ablation experimental results based on VOT2016 dataset.

Methods	Baseline	Modle1	Modle2-1	Modle2-2	Modle3
Baseline	✓	✓	✓	✓	✓
Swin		✓			✓
TSC				✓	✓
TSC-T			✓		
A	0.640	0.658	0.824	0.643	0.669
R	0.196	0.198	0.108	0.158	0.146
EAO	0.463	0.472	0.783	0.476	0.501

Table 2
Comparative test results based on the Tracking Net dataset.

Methods	S%	P%	NP%
SiamFC	57.1	53.3	66.3
ATOM	70.3	64.8	77.1
MDNet	60.6	56.5	70.5
DiMP	74.0	68.7	80.1
SiamRPN++	73.3	69.4	80.0
SiamFC++	75.4	70.5	80.0
STARK-S50	80.3	77.6	85.1
Ours	80.7	78.2	85.4

4.3. Ablation experiments

In order to fully validate the effectiveness of our methods in this paper, we compare the previous variant experiments in the ablation experiment section as well. The ablation experiments are chosen to be conducted on the VOT2016 dataset, and the performance metrics A, R, and EAO are used as evaluation criteria to demonstrate the effect of each module. The specific results of the ablation experiments are shown in Table 1.

As can be seen from Table 1, the base algorithm Siam RPN++ with the VOT2016 dataset, the metrics A, R, and EAO arrived at 0.640, 0.196, and 0.463, respectively; in Model1 with the addition of the Swin module, the accuracy metrics were relatively improved by 2.8%, the robustness metrics remained basically the same, and the EAO was relatively improved by 2%, which shows that Swin module is effective for the overall algorithm.

Modle2-1, as the initiating experiment for the TSC architecture in this paper, still uses the structural re-parameterization strategy to fuse convolutional layers with BN layers and then combine multiple branches into one. However, Modle2-1 differs in that it delivers the result of aggregating the initial frame truth template with the output of the backbone network as a positive sample to the RPN module. In a subsequent study, it was found that such an approach was not reasonable, since the RPN is a classification and regression network implemented by convolving the two with each other, so it is not possible to use an "$(A + B) \cap A$"-like strategy to determine the similarity of the targets.

Inspired by the previous experiment Modle2-1, this paper designs its variant experimental TSC architecture Modle2-2. Modle2-2 obtains the mutual correlation coefficients of the target and the initial truth template through the RPN branch; the mutual correlation coefficients of the target and the adjacent frame template through the TSC branch, and finally generates classification results and regression results by weighted evaluation. On VOT2016, the A, R and EAO metrics reached 0.643, 0.158 and 0.476, respectively, and their metrics R and EAO improved by 19.3% and 2.8%, respectively, compared to Baseline. Modle3 uses the Swin module and the three-stage cascade (TSC) architecture together in Siam RPN++, and the results are remarkable. The evaluation indexes A, R, and EAO reached 0.669, 0.146, and 0.507, respectively, which were 4.5%, 25.5%, and 8.2% higher compared to baseline.

In summary, the Siam ST algorithm model has a stronger tracking capability and higher tracking accuracy with robust overall results.

4.4. Comparison test

In order to more fully demonstrate the excellent performance of the algorithm model in this paper, this paper compares with some of the current mainstream algorithms on Tracking Net dataset and UAV123 dataset, and uses precision (P), normalized precision (NP) and success rate (S) as the judging criteria. Among them, the mainstream algorithms include SiamFC [9], ATOM [18], MDNet [19], DiMP [20], SiamFC++ [21], SiamRPN++ [11], STARK-S50 [22]. These algorithms include twin network algorithms based on deep learning, as well as target tracking algorithms based on the transformer architecture, and thanks to their pioneering contributions, target tracking tasks are rapidly evolving. The specific results are shown in Table 2.

According to Table 2, it can be seen that the algorithm in this paper achieves the highest standards in accuracy, normalized accuracy and success rate indices, 80.7, 78.2 and 85.4, respectively. Each index outperforms the second place (STARK-S50) by 0.4 (S), 0.6 (P) and 0.3 (NP), respectively, with an average contrast gain of about 0.5%. Each index exceeds the baseline (SiamRPN ++) by 7.4 (S), 8.8

Table 3
Comparative experimental results based on UAV123 dataset.

Methods	TrTr	TrSiam	SiamFC	TrDiMP	Siam RPN++	STARK-S50	Ours
AUC	65.2	67.4	57.1	67.5	59.3	68.4	68.8

Table 4
Detailed information of the three benchmark data sets.

Methods	A	R	LN	EAO
Siam Mask	0.609	0.276	60	0.380
Siam DW	0.538	0.398	85	0.270
Update Net	0.587	0.276	59	0.393
Siam R-CNN	0.612	0.220	47	0.406
Ta-Siam RPN++	0.593	0.272	58	0.360
Da Siam RPN	0.586	0.276	59	0.383
ATOM	0.590	0.204	44	0.401
SiamRPN++	0.600	0.235	50	0.414
Ours	0.615	0.217	45	0.408

Table 5
Comparative experimental results based on LaSOT dataset.

Methods	S	P
TrDiMP	0.640	0.666
TrSiam	0.624	0.645
STARK-S50	0.658	0.697
SiamRPN++	0.495	0.493
Ours	0.667	0.716

(P), and 5.4 (NP), respectively, with an average contrast gain of approximately 9.58%. SiamFC, SiamFC++, and DiMP demonstrate decent tracking performance but lack the ability to effectively model complex feature relationships in challenging scenarios. In contrast, ATOM and MDNet employ online learning strategies to enhance target tracking capabilities, but they still struggle to match the accuracy and robustness of our model when handling rapid target movements and appearance changes. Similarly, SiamRPN++, with its relatively fixed feature extraction process, does not achieve the overall performance level demonstrated by our model.

The normalized accuracy (NP) of the algorithm in this paper can reach 85.4% on the Tracking Net dataset. It is proved that the Siam ST proposed in this paper can effectively use the global information of images and video contextual feature templates for mutual correlation calculation, which makes the algorithm have good target recognition estimation capability. It can accurately capture the target position and accurately predict the target's bounding box parameters under various difficult scenarios such as target occlusion, deformation and target loss (partial loss). This experiment also demonstrates that the self-attention mechanism in Transformer is fused and nested in Siamese network is effective. The global dependencies modeled by the self-attention mechanism provide excellent feature representation for the tracking task.

In the UAV123 dataset, this paper also did a rich comparison experiment to demonstrate the validity and excellence of our experiment by the area under the line (AUC) parameter of the success rate curve. The specific results are shown in Table 3.

According to Table 3, the result in this paper achieves 68.8, which is ahead of other algorithms, 9.5 higher than Siam RPN++, with a comparative improvement of about 16%, and 1.4 higher than TrSiam, with a comparative improvement of about 2.1%. In addition, this paper has a faster running speed compared to Siam RPN++.

On the VOT2018 dataset, we added five more siamese network-based algorithmic models, Siam Mask [23], Siam DW [24], Update Net [25], Siam R-CNN [26], Ta-Siam RPN++ [27], and Da Siam RPN [28], for conducting comparison experiments. The specific experimental results are shown in Table 4.

As can be seen from Table 4, the algorithm model in this paper achieves the highest EAO of 0.408, which is a 0.5% improvement compared to the second place Siam R-CNN. The EAO of ATOM reached 0.401, but the method performs poorly when handling fast-moving targets and occlusions. Update Net and Ta-Siam RPN++ showed significant disadvantages in terms of EAO, with scores of only 0.393 and 0.360, respectively, indicating their inefficiency in dealing with complex tracking tasks. Similarly, Siam Mask and Da Siam RPN lacked the capability to effectively model target appearance variations, resulting in lower performance in both EAO and accuracy compared to our method.

Like the Tracking Net dataset, the LaSOT dataset also uses a one-time evaluation strategy, and the metrics also use precision (P), normalized precision (NP), and success rate (S) as the judging criteria. Despite the differences in the context and parameters of the images in the dataset, this paper achieves the same excellent results on LaSOT.

As shown in Table 5, the performance of our algorithm on the LaSOT dataset is demonstrated. Specifically, the algorithm proposed in this paper achieves a success rate of 0.667 and an accuracy of 0.716, outperforming all the compared models. The STARK-S50 model falls short in both success rate and accuracy due to its relatively weak ability to handle target variations, particularly in complex scenarios involving occlusion, target deformation, and fast-moving targets. While TrDiMP demonstrates more stable performance, it is less effective than our model in managing fast-moving targets and complex backgrounds. TrSiam has difficulty with target appearance changes and occlusion, leading to a noticeable decline in tracking performance. Among all the models, SiamRPN++ performs the worst, as its limited feature extraction capability hampers its adaptability to complex environments.

Through the experiments, we have validated the adaptability and robustness of the model. The selected datasets cover a variety of scenarios, target types, and complex background variations, effectively testing the model's performance in diverse environments. The experimental results demonstrate that Siam ST performs exceptionally well across these datasets, significantly enhancing the model's generalization capability. However, the complexity introduced by the multi-stage architecture and sliding window mechanism could potentially increase the risk of overfitting in certain specific scenarios. To address this potential issue, we incorporated a Three-Stage Cascade (TSC) structure into the model design. This structure facilitates progressive feature aggregation and multi-level interaction, enhancing the model's adaptability to complex backgrounds and fast-moving targets. By capturing target features from multiple perspectives, the model reduces its dependency on specific scenes or background conditions, thereby indirectly mitigating the risk of overfitting.

5. Conclusion

The Siam ST algorithm model proposed in this paper solves, to a certain extent, the problem of lack of consideration of global location correlation when extracting image features from the backbone network; alleviates the singularity of the mutual correlation metric when matching tracking targets; and enhances the overall robustness of the model. The sliding window module (Swin) and the TSC module are used to improve the backbone network and the region suggestion network to a certain extent. In this paper, we demonstrate the validity and reliability of our method through extensive experiments. Our algorithm is also well-suited for practical applications. In the field of autonomous driving, where the external environment often contains significant interference and variability, our model enhances the perception of global information about the target, ensuring stability and accuracy even in complex scenarios, thereby contributing to safer driving. Furthermore, our algorithm has valuable applications in medical imaging, where it can track dynamic changes associated with specific diseases within the body, providing useful information for medical diagnosis. Additionally, target tracking algorithms are essential in various practical contexts, such as intelligent security systems and agricultural production. Our algorithm can undoubtedly be applied to these practical scenarios, but further optimization is needed. In future research, we will focus on adapting and refining the algorithm for different application environments.

CRediT authorship contribution statement

Zheng Yang: Writing – original draft, Methodology, Investigation, Formal analysis, Conceptualization. **Kaiwen Liu:** Writing – review & editing, Validation, Methodology. **Quanlong Li:** Writing – review & editing, Methodology. **Yandong Hou:** Writing – review & editing, Investigation, Funding acquisition, Conceptualization. **Zhiyu Yan:** Writing – review & editing, Software, Methodology, Investigation, Formal analysis.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Yandong Hou reports financial support was provided by National Natural Science Foundation of China. Yandong Hou reports financial support was provided by Natural Science Foundation of Henan Province. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability statement

Research data collected for this study will be made available if requested by contacting the corresponding author.

References

[1] X. Lu, J. Li, Z. He, W. Liu, L. You, Visual object tracking via collaborative correlation filters, Signal Image Video Process. 14 (1) (2020) 177–185.
[2] K. Zhao, H. Zhao, Z. Wang, J. Peng, Z. Hu, Object-preserving Siamese network for single-object tracking on point clouds, IEEE Trans. Multimed. (2023).
[3] Z. Li, Y. Lin, Y. Cui, S. Li, Z. Fang, Motion-to-matching: a mixed paradigm for 3d single object tracking, IEEE Robot. Autom. Lett. (2023).
[4] S. Hu, X. Zhao, K. Huang, Sotverse: a user-defined task space of single object tracking, Int. J. Comput. Vis. 132 (3) (2024) 872–930.
[5] Z.Y. Xiang, T.Y. Cao, P. Zhang, T. Zhu, J.F. Pan, Object tracking using probabilistic principal component analysis based on particle filtering framework, Adv. Mater. Res. 341 (2012) 790–797.
[6] Y. Zhang, T. Wang, K. Liu, B. Zhang, L. Chen, Recent advances of single-object tracking methods: a brief survey, Neurocomputing 455 (2021) 1–11.
[7] D. Wang, H. Lu, Z. Xiao, M.-H. Yang, Inverse sparse tracker with a locally weighted distance metric, IEEE Trans. Image Process. 24 (9) (2015) 2646–2657.
[8] C. Xiu, S. Wei, R. Wan, Y. Cheng, J. Luo, H. Tian, et al., Camshift tracking method based on target decomposition, Math. Probl. Eng. 2015 (2015).
[9] L. Bertinetto, J. Valmadre, J.F. Henriques, A. Vedaldi, P.H. Torr, Fully-convolutional Siamese networks for object tracking, in: Computer Vision–ECCV 2016 Workshops, Proceedings, Part II 14, Amsterdam, the Netherlands, October 8-10 and 15-16, 2016, Springer, 2016, pp. 850–865.
[10] B. Li, J. Yan, W. Wu, Z. Zhu, X. Hu, High performance visual tracking with Siamese region proposal network, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8971–8980.

[11] B. Li, W. Wu, Q. Wang, F. Zhang, J. Xing, J. Yan, Siamrpn++: evolution of Siamese visual tracking with very deep networks, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4282–4291.

[12] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 770–778.

[13] G. Roffo, S. Melzi, et al., The visual object tracking vot2016 challenge results, in: Computer Vision–ECCV 2016 Workshops, Proceedings, Part II, Amsterdam, the Netherlands, October 8-10 and 15-16, 2016, Springer International Publishing, 2016, pp. 777–823.

[14] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Čehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey, et al., The sixth visual object tracking vot2018 challenge results, in: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 2018.

[15] H. Fan, L. Lin, F. Yang, P. Chu, G. Deng, S. Yu, H. Bai, Y. Xu, C. Liao, H. Ling, Lasot: a high-quality benchmark for large-scale single object tracking, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 5374–5383.

[16] M. Muller, A. Bibi, S. Giancola, S. Alsubaihi, B. Ghanem, Trackingnet: a large-scale dataset and benchmark for object tracking in the wild, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 300–317.

[17] M. Mueller, N. Smith, B. Ghanem, A benchmark and simulator for uav tracking, in: Computer Vision–ECCV 2016: 14th European Conference, Proceedings, Part I 14, Amsterdam, the Netherlands, October 11–14, 2016, Springer, 2016, pp. 445–461.

[18] M. Danelljan, G. Bhat, F.S. Khan, M. Felsberg, Atom: accurate tracking by overlap maximization, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4660–4669.

[19] H. Nam, B. Han, Learning multi-domain convolutional neural networks for visual tracking, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 4293–4302.

[20] G. Bhat, M. Danelljan, L.V. Gool, R. Timofte, Learning discriminative model prediction for tracking, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 6182–6191.

[21] Y. Xu, Z. Wang, Z. Li, Y. Yuan, G. Yu, Siamfc++: towards robust and accurate visual tracking with target estimation guidelines, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 12549–12556.

[22] B. Yan, H. Peng, J. Fu, D. Wang, H. Lu, Learning spatio-temporal transformer for visual tracking, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 10448–10457.

[23] Q. Wang, L. Zhang, L. Bertinetto, W. Hu, P.H. Torr, Fast online object tracking and segmentation: a unifying approach, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 1328–1338.

[24] Z. Zhang, H. Peng, Deeper and wider Siamese networks for real-time visual tracking, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 4591–4600.

[25] L. Zhang, A. Gonzalez-Garcia, J.V.D. Weijer, M. Danelljan, F.S. Khan, Learning the model update for Siamese trackers, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 4010–4019.

[26] P. Voigtlaender, J. Luiten, P.H. Torr, B. Leibe, Siam r-cnn: visual tracking by re-detection, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 6578–6588.

[27] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Čehovin Zajc, T. Vojir, G. Bhat, A. Lukezic, A. Eldesokey, et al., The sixth visual object tracking vot2018 challenge results, in: Proceedings of the European Conference on Computer Vision (ECCV) Workshops, 2018.

[28] Z. Zhu, Q. Wang, B. Li, W. Wu, J. Yan, W. Hu, Distractor-aware Siamese networks for visual object tracking, in: Proceedings of the European Conference on Computer Vision (ECCV), 2018, pp. 101–117.