Improved ECO Algorithm Based on Residual Neural Network

Huifang Qi¹*, ChangMeng Wang²,³ and Bo Li²,³

¹School of Information Science and Engineering, Yanshan University, Qinhuangdao, China
²School of Electrical Engineering, Yanshan University, Qinhuangdao, China
³Key Laboratory of industrial computer control engineering in Hebei Province, Yanshan University, Qinhuangdao, China

*Corresponding author e-mail: qhftaoyao@163.com

Abstract. Video target tracking is one of the hot fields of computer vision, and its application is also very extensive. However, due to the complexity and variability of tracking environment, which brings some challenges to the research of target tracking. ECO (Efficient Convolution Operators) Algorithm is proposed based on convolutional neural network in three aspects. Firstly, residual neural network ResNet50 is adopted instead of convolutional neural network to extract the appearance features of target, and deeper residual neural network is applied to obtain more abundant target semantic information, so as to improve the tracking effect of tracking algorithm. Secondly, sample space classification strategy is improved. Different weights are assigned to shallow feature and deep feature, which make the deep feature play a more important role and improve the effect of target tracking. Finally, the method of scale estimation is improved so that better bounding boxes can be estimated with the scale changing. Experimental results show that the distance accuracy and success rate of the algorithm.

1. Introduction

Target tracking is a basic and key part in the field of computer vision. It plays an important role in national defense technology, unmanned driving, biomedical and other fields. The process of target tracking is as follows: firstly, given a video sequence, we select a target in the initial frame of the video sequence, and then calculate each subsequent frame to infer the target's trajectory and its state [1]. But in real life, the target is often in a complex and changeable environment, which greatly reduces the success rate of tracking target. Therefore, the target tracking algorithm designed can not only meet the requirements of stability and speed, but also adapt to the changing environment.

Early target tracking methods include optical flow [2], Mean shift tracking algorithm [3], Kalman Filter [4] and classical algorithm. In 2010, Bolme and others [5] used knowledge of correlation filter in target tracking. This is the first attempt of correlation filter in this field, which is Mosse (minimum output sum of squared error) algorithm. They use Fourier transform to transform correlation operation to frequency domain, and put forward the concept of minimum output square error. Using correlation filter to track target, the operation speed of tracking algorithm is obviously improved. In 2012, Henriques proposed the CSK (circular structure with kernels) algorithm [6], which used the detected core loop structure to achieve target tracking; in 2013, Zhang [7] proposed STC (spatial temporal) algorithm. This algorithm was based on the Bayesian tracking framework. He built the model...
according to the spatiotemporal relationship formed by tracking target and surrounding area. In 2014, Henry [8] further improved the CSK algorithm and proposed KCF (kernized correlation filters) target tracking algorithm, which used the idea of kernel function. Xu Fang and others [9] improved the KCF algorithm and proposed an adaptive tracking algorithm based on kernel correlation filter. Danelljan M and others [10] used the depth network to extract features of the target. Then, they proposed a cubic difference method to transform the feature map into a continuous spatial domain. This is the C-COT algorithm. The algorithm transformed single resolution feature mapping into multi-resolution feature mapping, and fused different resolution convolution layers. After that, Danelljan M and others simplified the C-COT algorithm in feature extraction and proposed ECO algorithm to reduce the complexity of the model. Operation efficiency was improved obviously. In recent years, more and more researchers apply deep learning network to target tracking algorithm, which improves the tracking accuracy. DLT [12] (deep learning tracker) algorithm introduced deep learning into the field of moving target tracking for the first time. Subsequently, scholars began to study the combination of target tracking and other deep neural networks, such as convolutional neural networks (CNN) [13], recurrent neural networks (RNN) [14], etc. In 2015, Wang Naiyan [15] improved DLT algorithm and proposed SO-DLT algorithm, which was a successful application of large scale CNN network in target tracking. In 2016, H Nami [16] proposed MDnet algorithm based on Multi-Domain training idea. The network structure of the algorithm is divided into sharing layer and domain-specific layer, which could deal with different kinds of targets. On the basis of HCF algorithm, Ma [17] used three-layer network to train correlation filter respectively, which greatly improved the tracking accuracy. DeepLMCF was improved on the basis of LMCF [18] (large margin object tracking with circular feature maps) algorithm. Based on the original algorithm, he introduced the depth feature, and used the feature of depth feature to express the object better. The accuracy and success rate of target tracking were further improved.

Based on ECO algorithm, this paper improves it from three aspects. Firstly, a deeper residual network is used to replace VGGNet to extract deeper target features, which can fully mine the semantic information of the target, and greatly improve the effect of target tracking; secondly, the shallow and deep features are assigned different weights, and then the final target is fused to predict the final position; finally, The original target scale estimation method (DSST algorithm) is further optimized. Simulation results show that the proposed algorithm greatly improves the speed and accuracy of target tracking.

2. ECO Algorithm Introduction

2.1. Continuous convolution operator

C-COT introduces a continuous convolution operator based on training samples to integrate multichannel features with different resolutions. The linear convolution operator $S_f$ is introduced, which maps the sample $x$ to an objective confidence function $s(t) = S_f \{x\}(t)$ defined in a continuous interval $[0, T)$. Here $s(t)$ is the confidence score of the target at the position $t \in [0, T)$. In the continuous domain, the operator $S_f$ is constructed by a series of convolution filters

$$f = (f^1, ..., f^D) \in L^2(T)^D.$$ The convolution operator can be defined as:

$$S_f \{x\} = \sum_{d=1}^{D} f^d \ast J_d \{x^d\}$$

(1)

Each feature channel is interpolated firstly, then the convolution response of each feature channel is calculated by formula (1), then all convolution responses are summed to obtain the confidence degree of final positioning.
In the continuous domain learning framework, each training sample $x_j$ is calibrated by the expected output $y_j \in L^2(T)$ of the convolution operator, we can get the loss function formula of the correlation filter as follows:

$$E(f) = \sum_{j=1}^{M} \alpha_j \|S_f \{x_j\} - y_j\|_2^2 + \sum_{d=1}^{D} \|wf^d\|_2^2$$ (2)

In formula (2), $\alpha_j \geq 0$ represents the weight of each training sample, $w$ is the space regularization penalty term, and $M$ is the number of training samples. The filter $f$ is trained by minimizing the function (2).

2.2. Feature Dimension Reduction

The C-COT framework model has too many parameters, and it is easy to cause over-fitting. The ECO algorithm is optimized for these problems, and convolution operation is performed efficiently, which greatly improves the performance and speed of tracker during operation.

Before performing the convolution operation, it is necessary to perform dimensionality reduction processing on the extracted features to reduce the number of parameters model. In training process of continuous filters, filter base $f^1, ..., f^C$ with the number $C$ ($C < D$) is used as their main component. The linear combination of learning coefficient $p_{d,c}$ is denoted as $\sum_{c=1}^{C} p_{d,c} f^c$, which together with filter $f^c$ forms a filter of feature layer $d$. This learning coefficient can be expressed as a matrix $P = (p_{d,c})$ of $D \times C$, so that the vector product $Pf$ of the matrix can be expressed as a new multi-channel filter. Thus, the factorized convolution operation formula is:

$$S_{pf} \{x\} = Pf \ast J \{x\} = f \ast P^T J \{x\} = \sum_{c,d} p_{d,c} f^c \ast J_d \{x^d\}$$ (3)

We analyzes Equation (3) from another angle. First, multiply the interpolated feature map by $P^T$ and the reduced-dimensional feature map is $C$-dimensional, and then convolve with the corresponding filter $f$. The factorization convolution operation further reduces the dimensionality of the feature map, greatly reduces the calculation amount of the entire algorithm, and improves the calculation speed. In addition, the weight matrix $P$ only needs to be learned in the first frame and remains unchanged in subsequent frames, which also reduces the amount of calculation to a certain extent.

2.3. Generate sample space

In ECO algorithm, Guassian Mixture Model is used to train the sample set, which can eliminate redundancy and strengthen the diversity between samples. To further simplify the model, we use an online update algorithm. Given a new sample $x_j$ in the sample space, first we use $\pi_m = \gamma_x$, $u_m = x_j$, to initialize the new group $m$, if the number of Gaussian components exceeds the set value, the model is simplified. If the weight $\pi$ of a component is lower than the threshold group weight, the group is discarded; otherwise, the groups $k$ and $l$ with the closest spatial distance are combined into one component $n$ according to formula (4).

$$\pi_n = \pi_k + \pi_l, u_n = \frac{\pi_k u_k + \pi_l u_l}{\pi_k + \pi_l}$$ (4)

Then the expected loss function formula (2) can be further expressed as:

$$E(f) = \sum_{j=1}^{L} \pi \|S_f \{u_j\} - y_j\|_2^2 + \sum_{d=1}^{D} \|wf^d\|_2^2$$ (5)
In formula (5), \( L \) is the number of Gaussian groups, and \( u_i \) is the expectation. Observing the above formula, we can see that Gaussian expectation \( u_i \) and weight \( \pi_i \) directly replace \( x_j \) and \( \alpha_j \) in formula (2), thus combining the probability distribution and the loss function. The sample generation model reduces the number of sample spaces, reduces the complexity of the algorithm, and does not reduce the difference between sample features.

2.4. Model update strategy

In terms of model update, previous trackers are updated frame by frame, and this update method often consumes a lot of system memory. In ECO algorithm, a sparse update strategy is adopted, which is updated every \( N_s \) frame, which has a significant effect on reducing the amount of calculation, and \( N_s \) does not affect the update of the spatial model. It is obtained through experiments that when \( N_s \approx 5 \) we can be obtained Ideal tracking results.

3. Improved methods

3.1. Deep feature extraction network ResNet

The residual neural network was first proposed by four scholars from Microsoft Research. ResNet added residual blocks to the ordinary convolutional neural network to train deeper networks, and added one to the forward neural network Shortcut connection, we finally get the ResNet model by continuously superimposing the residual structure. As shown in Figure 1:

![Residual structure](image)

In the residual structure of Figure 1, \( x \) represents input, \( y(x) \) represents expected output, and \( h(x) \) represents output of the network before the second layer activation function. The residual unit contains two kinds of mappings, the mapping obtained by the input itself and the residual mapping after the network operation. This structure of ResNet can directly bypass the input information to the output, which not only better protects the integrity of the information, but also the entire network only needs to learn the difference between the input and output, which solves the problem of CNN information loss to a certain extent, which simplifies the learning objectives and difficulty.

ResNet has different network layers. The more commonly used ones are 50-layer, 101-layer, and 152-layer. Considering the calculation amount of various structures, feature representation and other factors, this article uses the ResNet50 structure. The network configuration is shown in Table 1.
### Table 1. Network configuration of ResNet-50

| layer name | conv 1 | conv 2_x | conv 3_x | conv 4_x | conv 5_x |
|------------|--------|----------|----------|----------|----------|
| Output size | 112 × 112 | 56 × 56 | 28 × 28 | 14 × 14 | 7 × 7 |
| 50-layer | 7 × 7, 64, stride 2 | 1 × 1, 64 | 3 × 3, 64, stride 3 | 1 × 1, 128 | 1 × 1, 256 |
| | | 3 × 3, 128, stride 4 | 3 × 3, 256, stride 6 | 3 × 3, 12, stride 3 |

#### 3.2. Feature extraction and feature fusion

In order to extract deeper features, the residual network is used instead of VGGNet. Here, ResNet50 is used to extract the target feature, the output of the second residual block is extracted as the shallow feature, and the output of the fourth residual block is used as the depth feature. Figure 2 shows the overall algorithm flow chart.

![Figure 2. Overall algorithm flow chart](image)

It can be seen from Figure 2 that when using ResNet to extract features, the output size of each residual block is 112 × 112, 56 × 56, 28 × 28, 14 × 14, 7 × 7, respectively. The size of the preprocessed original input image is 224 × 224. The size of the feature map output by the fourth residual block is 14 × 14, which is 0.125 times of the input image of the first layer, which leads to the loss of some information. We know that in feature extraction, deep features can better express target semantic information than shallow features. Therefore, deep features play an important role in feature extraction of targets. Therefore, in order to obtain a better tracking effect, in the final convolution response summation, a greater weight is assigned to the deep features.

When calculating the convolution response, the final response graph can be obtained by adding up the responses corresponding to all filters. According to the above analysis, we need to assign greater weight to deep features. The confidence function is as follows:

\[
S_f(x) = W_1 \sum_{a=1}^{C_1} f^a \ast J_a \{x^a\} + W_2 \sum_{b=1}^{C_2} f^b \ast J_b \{x^b\}
\]  

(6)
In equation (6), \( C_s \) represents shallow feature dimension, \( C_d \) represents deep feature dimension, \( W_1 \) and \( W_2 \) respectively represent the coefficient of each feature layer convolution response, that is, the weight of the corresponding feature layer. The extracted feature map is first interpolated, and then convolved with the corresponding filter. According to formula (6), different weights are assigned to different feature layers to solve the convolution response. Finally, the target position is obtained. When the filter is retrained, the loss function is as follows:

\[
E(f) = \sum_{j=1}^{m} \alpha_j \left\| W_1 \sum_{a=1}^{C_s} f^a \ast J_1 \{x_j^a\} + W_2 \sum_{b=1}^{C_d} f^b \ast J_2 \{x_j^b\} - y_j \right\|^2 + \sum_{c=1}^{C} w^c d^c
\]

(7)

In formula (7), \( \alpha_j \geq 0 \) represents the weight of each training sample, \( w \) is the spatial regularization penalty, \( m \) represents the number of training samples, and \( C = C_s + C_d \). It can be seen from Section 1.3 that the model of formula (7) can be further optimized, and the optimized loss function is expressed as:

\[
E(f) = \sum_{j=1}^{l} \pi_j \left\| S_j \{u_j\} - y_j \right\|^2 + \sum_{d=1}^{D} w^d d^d
\]

(8)

In this way, the model is updated.

3.3. Scale adaptive mechanisms

In order to improve the robustness of target scale changes and increase the efficiency of calculation, we designed a new scale adaptive method inspired by the mechanism of the DSST algorithm, which increases the tracking effect of the relevant filter. First, we construct a feature pyramid in the rectangular frame around the target, using \( P \times R \) to represent the scale of the previous frame of the target. \( S \) represents scale layer. \( n \in \{[S - 1], \ldots, [S - 1/2]\} \) The algorithm extracts several scales with scale size of \( a^nP \times a^nR \) in the center of the target image, \( a \) is scale factor between the feature layers, \( \log \left( \frac{H}{W} \right) \) represents the width and height of the original image. Through experiments, it shows that when \( S = 10 \), that is, a 10-layer scale design is adopted, the model training and target scale capture and tracking effect perform best.

Considering the reasonable change of the target scale, the smallest scale factor equation (9) and the largest scale factor equation (10) are respectively expressed as:

\[
\delta_{\text{min}} = a^{\left\lfloor \log_{\frac{I}{W}} \left( \frac{1}{2} \right) \right\rfloor}
\]

(9)

\[
\delta_{\text{max}} = a^{\left\lceil \log_{\frac{H}{W}} \left( \frac{1}{2} \right) \rceil}
\]

(10)

In formulas (9)-(10), \( I \) represents the size of the input image supported by the network; \( a \) represents the scale factor between feature layers; \( W \) and \( H \) represent the width and height of the original image. The optimal target scale must be between \( \delta_{\text{min}} \) and \( \delta_{\text{max}} \) as the scale factor. This limitation makes the algorithm more in line with the scale changes in the real scene, which can improve the computational efficiency.

4. Experimental Results and Analysis

4.1. Overall experimental results and analysis

We conduct experiments on the OTB100 test set to test the effectiveness of our proposed method. We select the fourth residual block of ResNet50 for deep feature extraction. When the feature is fused to obtain the response, the weights assigned to deep features and shallow features are 0.6 and 0.4, respectively. The scale layer \( S \) is 10 and the scale factor \( a \) is 1.03.

First, we carry out overall experiment. We use a one-pass evaluation method based on the accuracy map and the success rate map to evaluate the performance of the tracking algorithm. Here we...
also compare with nine common target tracking methods. The experimental results are shown in Figure 3.

![Figure 3](image)

**Figure 3.** Comparison chart of success rate and accuracy

It can be seen from Figure 3 that the distance accuracy of the method in this paper is 0.902, which is 2.15% higher than ECO algorithm, and the coincidence success rate is 0.862, which is 5.38% higher than the ECO algorithm. In terms of accuracy and success rate, the algorithm in this paper is superior to the ECO algorithm and ranks first in the tracking algorithm. It can be seen that the algorithm in this paper can significantly improve the tracking effect.

![Figure 4](image)

**Figure 4.** Comparison chart of success rate and accuracy under scale attribute

Figure 4 shows the test results for the target scale. The results of the graph show that the tracking accuracy of the algorithm proposed in this paper is 87.3%, and the success rate is 85.0%. Compared with the ECO algorithm, the accuracy and success rate are increased by 3.1% and respectively. 7.1%. It shows that the algorithm proposed in this paper can better deal with the change of target scale.

Next, we test the algorithm in 11 different environments. Table 2 shows the tracking accuracy and success rate of this article and the ECO algorithm.

| Environmental attributes | Fast | Mobile | Backgound | Compl-exity | Sports | Fuzzy | Defor-mation | Light | Inplane rotation | Low score Resol-ution | Occlu-sion | Outside the plane Rotation | Out of sight | Scale Change | Over all |
|--------------------------|------|--------|------------|-------------|--------|-------|-------------|-------|-----------------|------------------------|------------|---------------------------|-------------|-------------|---------|
| Accu                     | Ours | 0.906  | 0.923      | 0.851       | 0.897  | 0.958 | 0.874       | 0.953 | 0.893          | 0.897                  | 0.969      | 0.873                     | 0.902       |
|                           | ECO  | 0.876  | 0.856      | 0.812       | 0.897  | 0.871 | 0.844       | 0.844 | 0.883          | 0.882                  | 0.883      | 0.842                     | 0.878       |
| Succ                     | Ours | 0.839  | 0.897      | 0.833       | 0.904  | 0.927 | 0.831       | 0.87  | 0.817          | 0.851                  | 0.926      | 0.850                     | 0.862       |
|                           | ECO  | 0.780  | 0.768      | 0.785       | 0.879  | 0.825 | 0.766       | 0.740 | 0.796          | 0.802                  | 0.788      | 0.779                     | 0.812       |
It can be seen from Table 2 that, except for the deformation attribute whose distance accuracy is the same as ECO, under any other attributes, both the distance accuracy and the success rate are superior to the ECO algorithm, which also shows the effectiveness of the algorithm in this paper.

4.2. Different Weight Distribution Experiment

In target tracking, when we apply deep convolutional network for feature extraction, we can extract different features from shallow layer and deep layer. These features represent different meanings and play different roles in tracking. The shallow features contain a lot of complex visual information, which is conducive to the accurate positioning of the target; while the deep features contain semantic information, which is used to express the image and can also improve the robustness of the tracker. Therefore, the weight of each layer should be allocated reasonably when performing feature fusion. From a theoretical point of view, giving a large weight to deep features can achieve better tracking results to a certain extent, but this will cause the weight of shallow features to be too low, which will reduce the performance of shallow features.

In order to get most suitable weight distribution. We need to do experiments to test. Here we assign multiple sets of weights to the deep and shallow layers on OTB100 for testing, and select the most appropriate weight distribution based on the test results. Here are five sets of experiments to determine the distribution ratio. Table 3 shows the distance accuracy under different weights, and Table 4 shows the success rate under different weights.

Table 3. Distance accuracy comparison of deep feature with different weights

| Deep Weight | 0.5  | 0.6  | 0.7  | 0.8  | 0.9  |
|-------------|------|------|------|------|------|
| Distance    | Ours | 0.894| 0.902| 0.897| 0.901| 0.870|
|             | ECO  | 0.883| 0.883| 0.883| 0.883| 0.883|

Table 4. Comparison of success rate of deep features with different weights

| Deep Weight | 0.5  | 0.6  | 0.7  | 0.8  | 0.9  |
|-------------|------|------|------|------|------|
| Success rate| Ours | 0.855| 0.862| 0.851| 0.82 | 0.78 |
|             | ECO  | 0.818| 0.818| 0.818| 0.818| 0.818|

It can be seen from Table 3 that in the five experiments conducted, we can find that it is not that the higher the deep weight, the higher the distance accuracy. When the weight of the depth feature is 0.6, the distance accuracy is the highest at this time, reaching 0.902. When the weight of deep layer increases, the distance accuracy will not increase, but has a downward trend. It can be seen from Table 4 that when the depth weight is at 0.6, good results are also obtained, but when the depth weight is increased again, the success rate will decrease instead, and we can see that when the depth weight is assigned to 0.9, the success rate is lower than the ECO algorithm.

From the data given above, it can be concluded that when a higher weight is assigned to the depth feature, the tracking effect can be improved, but if the weight is too high, the performance ability of the shallow feature will also be reduced, making the tracking distance accuracy and The success rate has dropped. Therefore, when performing feature fusion, it is necessary to balance the weights of deep features and shallow features to achieve the best tracking effect. According to the above experiments, in this algorithm, when the depth feature weight is 0.6 and the shallow feature is 0.4, a better tracking effect can be achieved.
5. Conclusion
In this paper, the ECO algorithm based on convolutional neural network is improved, mainly from three aspects. Firstly, residual neural network ResNet50 is used to replace convolutional neural network (CNN) to extract the appearance features of the target and obtain richer semantic information of the target by applying a deeper network, thereby improving the tracking effect of the tracking algorithm. Secondly, when obtaining the characteristic response, the weight distribution of each layer affect the training sample. Therefore, the shallow features and the deep features are combined with different weights, and the deep features are assigned greater weights, so that the deep features play a more important role and improve the effect of target tracking. Finally, in order to make the scale estimation more efficient, further improvements have been made on the basis of the original algorithm, so that the algorithm in this paper can estimate a better bounding box as the scale changes. For the algorithm in this article, We tested it on the OTB100 test set, and the results obtained showed the effectiveness of the algorithm. It has achieved a great improvement in distance accuracy and success rate compared to the ECO algorithm.

6. Acknowledgments
This work was financially supported by National Natural Science Foundation of China fund.

References
[1] Yacoob Y, Davis L S. Recognizing human facial expressions from long image sequences using optical flow[J]. IEEE Transactions on pattern analysis and machine intelligence, 1996, 18(6):636-642.
[2] Comaniciu D, Ramesh V, Meer P. Real-time tracking of nonrigid objects using mean shift[C]// Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662). IEEE, 2000, 2:142-149.
[3] Ristic B, Arulampalam S, Gordon N. Beyond the Kalman Filter Particle Filters for Tracking Applications[J]. IEEE Trans of Aerospace &Electronic Systems, 2003, 19(7):37-38.
[4] Hess R, Fern A. Discriminatively trained particle filters for complex multi-object tracking[A]. Proceeding of IE-EE Conference on Computer Vision and Pattern Recognition[C]. Miami, FL, USA: IEEE, 2009: 240-247.
[5] Bolme D S, Beveridge J R, Draper B A, et al. Visual object tracking using adaptive correlation filters[C]// Computer Vision & Pattern Recognition. CVPR. 2010. 5539960.
[6] Henriques J F, Caseiro R, Martins P, et al. Exploiting the circulant structure of tracking-by-detection with kernels[C]. European Conference On Computer Vision, 2012: 702-715.
[7] ZHANG K, LEI Z, YANG M H, et al. Fast Tracking via Spatio-Temporal Context Learning[J]. Computer Science, 2013.
[8] Henriques J F, Rui C, Martins P, et al. High-speed tracking with kernelized correlation filters[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015, 37(3): 583-596.
[9] Danelljan M, Robinson A, Khan F S, et al. Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking[C]// European Conference on Computer Vision. Springer, Cham, 2016: 472-488.
[10] Danelljan M, Bhat G, Khan F S, et al. ECO: Efficient Convolution Operators for Tracking[C]. Proceedings of the IEEE conference on computer vision and pattern recognition, 2017: 6638-6646.
[11] Wang N, Yeung D Y. Learning a deep compact image representation for visual tracking[A]. International Conference on Neural Information Processing Systems[C]. Neural information processing systems foundation, 2013: 809-817.
[12] LeCun Y L, Bottou L, Bengio Y, et al. Gradient-Based Learning Applied to Document Recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.
[13] Jozefowicz R, Zaremba W, Sutskever I. An Empirical Exploration of Recurrent Network Architectures[C]. International Conference on International Conference on Machine Learning. 2015: 2342-2350.
[14] Bromley J, Bentz J W, Bottou, Leon, et al. Signature Verification Using A "Siamese" Time Delay Neural Network [C]. International Conference on Neural Information Processing Systems. 1993: 669-688.

[15] Wang N, Li S, Gupta A, et al. Transferring Rich Feature Hierarchies for Robust Visual Tracking [J]. Computer Science, 2015.

[16] Nam H, Han B. Learning Multi-domain Convolutional Neural Networks for Visual Tracking [C]. Computer Vision And Pattern Recognition, 2016: 4293-4302.

[17] Ma C, Huang J B, Yang X, et al. Hierarchical Convolutional Features for Visual Tracking [C]. IEEE International Conference on Computer Vision (ICCV). IEEE Computer Society, 2015.

[18] Wang M M, Liu Y, Huang Z Y. Large margin object tracking with circulant feature maps. In: Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, HI, USA: IEEE, Wang N, 4800-4808.