A robust Correlation Filter based tracker with rich representation and a relocation component

Menglei Jin[1], Weibin Liu[1*] and Weiwei Xing[2]
[1] Institute of Information Science, Beijing Jiaotong University
Beijing 100044, China
[2] School of Software Engineering, Beijing Jiaotong University
Beijing 100044, China
*Corresponding author: Weibin Liu, wbliu@bjtu.edu.cn

Received December 16, 2018; revised March 24, 2019; accepted April 19, 2019; published October 31, 2019

Abstract

Correlation Filter was recently demonstrated to have good characteristics in the field of video object tracking. The advantages of Correlation Filter based trackers are reflected in the high accuracy and robustness it provides while maintaining a high speed. However, there are still some necessary improvements that should be made. First, most trackers cannot handle multi-scale problems. To solve this problem, our algorithm combines position estimation with scale estimation. The difference from the traditional method in regard to the scale estimation is that, the proposed method can track the scale of the object more quickly and effective. Additionally, in the feature extraction module, the feature representation of traditional algorithms is relatively simple, and furthermore, the tracking performance is easily affected in complex scenarios. In this paper, we design a novel and powerful feature that can significantly improve the tracking performance. Finally, traditional trackers often suffer from model drift, which is caused by occlusion and other complex scenarios. We introduce a relocation component to detect object at other locations such as the secondary peak of the response map. It partly alleviates the model drift problem.

Keywords: Visual tracking, Kernelized Correlation Filter, feature representation, scale estimation, relocation component.

1. Introduction

Video object tracking is a very important area in computer vision. Because of its widespread applications in intelligent transportation, military and human-computer interaction, it has already appealed to several countries around in the world. For example, for highways and transportation, effective detection and real-time tracking of moving vehicles are prerequisites for analysis and identification of vehicle behavior. In a vehicle violation capture system, vehicle tracking is also critical. Additionally, in human-computer interactions, manpower detection and tracking results are generally used as the basis for human-computer interaction. As more and more people have recently become interested in visual tracking, the tracking field has made great strides. However, tracking still needs to overcome many challenges, such as Deformation, Illumination Variation, Fast Motion, Background Clutter, Scale Variation, Occlusion and Out-of-View [1].

In this paper, we focus on real-time, model-free and single-object tracking. What is model-free tracking? This term refers to tracking with a model that has no clear appearance or shape. We only use the bounding box information in the first frame to predict the state of the object in each subsequent frame. Currently, there are mainly two kinds of methods using in video object tracking: the generative approach and discriminative approach. The generative approach is mainly used to model the object in the current frame and to find the area that is most similar to the current model in the next frame, such as the MEEM (Multiple Experts using Entropy Minimization) [2] and the ASMS (Scale-adaptive mean-shift) tracker [3]. The discriminative approach, which is also called tracking-by-detection, utilizes various features of the image and machine learning algorithms to distinguish the foreground from the background. The latest development for the discriminative approach is based on Correlation Filter and deep learning. In 2010, the MOSSE (Minimum Output Sum of Squared Error) tracker [4] first applies Correlation Filter to the field of video object tracking, and it has become the most promising tracking algorithm because of its high speed and accuracy. Later on, more improved Correlation Filter based trackers [5]-[11] were proposed, and they have become some of the most popular algorithms in video object tracking. Correlation measures the similarity of two functions at a certain time. The simplest idea of applying Correlation Filters to video object tracking is to design a filter that can get the maximum response value when it acts on the object. In particular, the KCF (Kernelized Correlation Filter) tracker [5], which was proposed in 2014, has achieved impressive tracking results. The fundamental idea which the Correlation Filter based tracking algorithm has successful improvement is that it tactfully introduces cyclic shift to solve the problem of insufficient training samples. In addition, the use of the characteristics of the cyclic matrix converts dense matrix operations into point multiplication, which reduces time complexity and greatly improves tracking efficiency.

However, the Correlation Filter based tracker should be improved acclimatize itself in some complicated scenarios. (1) The multi-scale problem, which refers to trackers cannot handle scale changes well during the tracking process. The size of the tracking box in KCF/DCF is determined by the scale of the object in the first frame, and then the size of the tracking box remains unchanged throughout the tracking process. In fact, as the distance between the object and the lens changes, the scale of the object will follow this kind of change. (2) The feature representation problem, which refers to the extracted image features not being powerful enough. This problem affects the accuracy and robustness of the tracker. (3) The tracking drift

problem, which is very likely to occur when the object is occluded or moves rapidly during the tracking process. Conventional Correlation Filter based trackers default to estimating the correct position for each frame. This position estimation method may cause model drift or tracking loss problems.

2. Related Work

In this section, we will briefly introduce several tracking algorithms as the basis of our work. The Struck (Structured Output with Kernels) tracker [12] uses the online structure output SVM to solve the tracking problem, which is the best tracking algorithm before the occurrence of the Correlation Filter based trackers. The MOSSE tracker [4] first introduced the Correlation Filter into the field of tracking, and showed the potential of the Correlation Filter. In 2012, Henriques proposed the CSK (Circulant Structure with Kernels) tracker [9], which constructs training samples through cyclic shift, and fully solves the problem of insufficient training samples in video object tracking. This algorithm uses ridge regression, which is one of the simplest classification algorithms in machine learning, to train samples to build a classifier. Although it has a decrease in speed compared with the MOSSE tracker [4], it has immensely improved the tracking performance. In 2014, the KCF/DCF tracker extended the feature representation in the CSK tracker [9] from raw pixels to multi-channel Hog features. Similarly, the ACNT (Adaptive Color Name Tracker) [8] extended the feature representation in the CSK tracker from raw pixels to Color Names. Both algorithms utilize different features of the image to improve tracking performance, yet the multi-scale problem has still existed. To resolve it, the DSST (Discriminative Scale Space Tracker) [6] designed two independent Correlation Filters: the translation filter and the scale filter. In the tracking process, it first estimates the position of the object with the translation filter and then constructs the image pyramid at the estimated position to obtain response by the scale filter to ultimately estimates the scale of the object. The SAMF (Scale Adaptive Filter with Multiple Feature Integration) tracker [13] uses another way to estimate the scale, which is similar to the multi-scale detection method commonly used in object detection algorithms. It estimates the scale of the object by detecting seven scaled image blocks. Therefore, the algorithm can simultaneously detect the position change and scale change of the object.

In addition to the multi-scale problem in the framework of the Correlation Filter tracking, there is a boundary effect problem that is caused by the cyclic shift. Based on DCF, the SRDCF (Spatially Regularized Discriminative Correlation Filters) tracker [14] utilize a larger detection area and regularization term to penalize the filter coefficients in the boundary area. Another algorithm based on MOSSE, the CFLB (Correlation Filters with limited boundaries) tracker [15] uses larger size detection blocks and smaller size filters to increase the proportion of real samples. Both of these algorithms increase the tracking performance at the expense of tracking speed because they all have no closed solution in the DCF tracker. In order to adapt to the object deformation and fast motion scenes, the Staple tracker [16] combines the DSST and DAT (Distractor-aware tracker) [17]. In the tracking process, the tracker uses a relatively simple linear weight strategy to obtain the final response map. To solve the model drift problem, another paper [18] proposed an optimized tracking framework. There are also some papers [19]-[20] combined traditional algorithms with algorithms based on Correlation Filter. They have achieved good tracking results obviously.

Recently, due to the rise of deep learning, a group of deep-learning-based tracking algorithms have also shown good performance, including HCF [21], SiamFC [22], EAST: [23], CFNet [24], C-COT [25], ECO [26], MDNET [27], FCNT [28] and so on.

3. The Kernelized Correlation Filter

In this section, we briefly introduce the baseline work: the kernelized Correlation Filter. In Section 3.1, we introduce the Discriminative Correlation Filter in detail. In Section 3.2, we extend the linear case in DCF to nonlinear case.

3.1 Discriminative Correlation Filter

The DCF/KCF tracker uses the Ridge Regression algorithm to train a classifier. It is the simplest regression algorithm in in machine learning. To address the lack of training samples, the algorithm applies a cyclic shift operation to build training samples. One of the most important reasons for using ridge regression is that it can obtain a simple closed solution without requiring complex iterations for any input. To simplify the problem, we would give a one-dimensional example. The object image block $\boldsymbol{x}$ is a column vector of $\boldsymbol{n}$ dimensions. Then, we use the matrix $\boldsymbol{p}$ to shift $\boldsymbol{x}$ in a chain, where $\boldsymbol{p}$ is the permutation matrix. When $\boldsymbol{u}$ is a negative number, $\boldsymbol{x}$ moves in the opposite direction. Due to the cyclic property, the shifted signal can be obtained with the following formula.

$$\{\boldsymbol{p^u x}|\boldsymbol{u} = \boldsymbol{0}, \boldsymbol{1}, \boldsymbol{2}, \ldots \boldsymbol{n-1}\} \tag{1}$$

Where $\boldsymbol{x_i}$ satisfies $\boldsymbol{x_i} = \boldsymbol{p^i x}$ and $\boldsymbol{i}$ represent the number of shifts of the cyclic shift. Therefore, the objective function can be defined as:

$$\boldsymbol{min_w} \sum_{\boldsymbol{i}} (\boldsymbol{f}(\boldsymbol{x_i}) - \boldsymbol{y_i})^2 + \boldsymbol{\lambda}\|\boldsymbol{w}\|^2 \tag{2}$$

The first part of equation (2) is the least-squares regression, and the second part is the regularization term employed to prevent overfitting. $\boldsymbol{f}$ is the function that we are looking for to minimize the square error between the actual output of the sample $\boldsymbol{x_i}$ and the regression target $\boldsymbol{y_i}$. This function satisfies $\boldsymbol{f}(\boldsymbol{x_i}) = \boldsymbol{w^T x_i}$. We rewrite formula (1) in the form of a matrix.

$$\boldsymbol{min_w}\|\boldsymbol{Xw} - \boldsymbol{y}\|^2 + \boldsymbol{\lambda}\|\boldsymbol{w}\|^2 \tag{3}$$

Here, $\boldsymbol{X}$ is a matrix, and each row of it represents a training sample. $\boldsymbol{w}$ indicates the weight coefficient. $\boldsymbol{y}$ is a column vector, and each of its columns is a label. By using the property that a circulant matrix can be diagonalized by a Fourier transform matrix, we can get:

$$\hat{\boldsymbol{w}} = \frac{\hat{\boldsymbol{x}}^* \odot \hat{\boldsymbol{y}}}{\hat{\boldsymbol{x}}^* \odot \hat{\boldsymbol{x}} + \boldsymbol{\lambda}} \tag{4}$$

Where $\hat{\boldsymbol{y}}$ and $\hat{\boldsymbol{x}}$ are the Fourier transforms of $\boldsymbol{y}$ and $\boldsymbol{x}$ respectively, and $\hat{\boldsymbol{x}}^*$ represents the conjugate transpose of $\hat{\boldsymbol{x}}$.

3.2 The Kernelized Correlation Filter

In the non-linear case, the objective function can be expressed as,

$$\boldsymbol{w} = \min_{\boldsymbol{w}}\|\boldsymbol{\phi}(\boldsymbol{X})\boldsymbol{w} - \boldsymbol{y}\|^2 + \boldsymbol{\lambda}\|\boldsymbol{w}\|^2 \tag{5}$$

Where $\boldsymbol{\phi}$ denotes a nonlinear mapping function that can make the mapped sample linearly separable in the new space. $\boldsymbol{w}$ is a vector in the space that is spanned by the vector group $[\boldsymbol{\phi}(\boldsymbol{x_1}), \boldsymbol{\phi}(\boldsymbol{x_2})\boldsymbol{\phi}(\boldsymbol{x_3}), \ldots \boldsymbol{\phi}(\boldsymbol{x_n})]$. Therefore, we can obtain the following formula.

$$\boldsymbol{w} = \sum_{\boldsymbol{i}} \boldsymbol{\alpha_i}\boldsymbol{\phi}(\boldsymbol{x_i}) \tag{6}$$

Here, $\boldsymbol{\alpha_i}$ is the coefficient in the linear representation. Using equation (6), equation (5) can be converted to:

$$\boldsymbol{\alpha} = \min_{\boldsymbol{\alpha}} \left\| \boldsymbol{\phi}(\boldsymbol{X})\boldsymbol{\phi}(\boldsymbol{X})^{\boldsymbol{T}}\boldsymbol{\alpha} - \boldsymbol{y} \right\|^2 + \boldsymbol{\lambda}\|\boldsymbol{w}\|^2 \tag{7}$$

After deduction, it can be simplified as:

$$\boldsymbol{\alpha} = (\boldsymbol{k} + \boldsymbol{\lambda}\boldsymbol{I})^{-1}\boldsymbol{y} \tag{8}$$

$\boldsymbol{I}$ in the equation (8) represents the unit matrix. $\boldsymbol{k}$ is a nuclear matrix of nuclear space. It satisfies $\boldsymbol{k} = \boldsymbol{\phi}(\boldsymbol{X})\boldsymbol{\phi}(\boldsymbol{X})^{\boldsymbol{T}}$. Each element in $\boldsymbol{k}$ is,

$$\boldsymbol{k_{i,j}} = \boldsymbol{\phi}^{\boldsymbol{T}}(\boldsymbol{x_i})\boldsymbol{\phi}(\boldsymbol{x_j}) = \boldsymbol{k}(\boldsymbol{x_i}, \boldsymbol{x_j}) \tag{9}$$

Where $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ satisfy $\boldsymbol{x_i} = \boldsymbol{p^i x}$ and $\boldsymbol{x_j} = \boldsymbol{p^j x}$, respectively. We can prove that $\boldsymbol{k}$ is a circulant matrix. Once again, by using the properties of the circulant matrix, we can obtain the formula,

$$\boldsymbol{\alpha} = \boldsymbol{F^{-1}}\left(\frac{\widehat{\boldsymbol{y}}}{\widehat{\boldsymbol{k}}^{\boldsymbol{xx}} + \boldsymbol{\lambda}}\right) \tag{10}$$

Where $\boldsymbol{k^{xx}}$ is the first row of the nuclear matrix $\boldsymbol{k}$. $\boldsymbol{F^{-1}}$ represents the inverse Fourier transform. $\widehat{\boldsymbol{k}}^{\boldsymbol{xx}}$ and $\widehat{\boldsymbol{y}}$ represent the Fourier transforms of $\boldsymbol{k^{xx}}$ and $\boldsymbol{y}$, respectively.

After the filter is trained, we can detect the object from the next frame of image. To improve the computational efficiency, the candidate samples are constructed using the cyclic shift method, which is the same as the method for constructing the training samples. We get the subwindow image in the current frame, where the position is the same as the object in the previous frame. We assume that the base sample is $\boldsymbol{z}$. We can obtain a candidate image patch $\boldsymbol{z_j}$ with the following formula.

$$\boldsymbol{z_j} = \boldsymbol{p^j z} \tag{11}$$

Where $\boldsymbol{j}$ represents the number of shifts of the cyclic shift. The response obtained on each sample $\boldsymbol{z_j}$ can be calculated using the following formula.

$$f(z_j) = w^T \phi(z_j) = \sum_{i=1}^{n} \alpha_i \, \phi^T(x_i) \phi(z_j) \tag{12}$$

Then, we can quickly calculate the response values of all the candidate image patches by the following formula.

$$f(z) = (k^z)^T \alpha = \left(C(k^{xz})\right)^T \alpha \tag{13}$$

Where k^z represents the kernel matrix between all training sample image blocks and all candidate image blocks. Because k^z can be proved as a cyclic matrix, it satisfies $k^z = C(k^{xz})$. Through Fourier transform, equation 13 can be converted to:

$$f = F^{-1}\left(\hat{k}^{xz} \odot \hat{\alpha}\right) \tag{14}$$

Where $\hat{k}^{xz}$ represents the Fourier transform of k^{xz}. The solution to k^{xz} has different forms for different kernel functions. When using the Gaussian kernel function, k^{xz} can be expressed as:

$$k^{xz} = exp(-\frac{1}{\sigma^2}(\|x\|^2 + \|z\|^2 - 2F^{-1}(\hat{x}^* \odot \hat{z}))) \tag{15}$$

Here, σ represents the standard deviation of the Gaussian distribution. $\hat{z}$ represents the Fourier transforms of z.

4. Proposed Method

In this section, we will describe the method we have proposed in detail. In section 4.1, we introduce a scale Correlation Filter that can quickly estimate the scale of the object. In both sections 4.2 and 4.3, we will describe the two important components of the proposed method in detail, which are the most noteworthy parts of this paper.

As shown in **Fig. 1**, we clearly show the entire tracking process for the algorithm. From **Fig. 1**, we can see that our tracking framework is mainly divided into three parts: position estimation, re-location, and scale estimation. In the first frame of the video sequence, we train the required translation filter and scale filter. In each of the following frames, the position of the object is estimated by the translation filter. Then, we use the obtained response map to detect the tracking confidence. When the tracking confidence is high, it goes directly to the next step. When the tracking confidence does not reach the standard, it starts the relocation component. Finally, we use scale Correlation Filter to determine the exact scale of the object.

4.1 Fast Scale Correlation Filter

It is clear that the KCF/DCF tracker does not consider the scale change of the object during the tracking process. To solve the multi-scale problem, we incorporate the scale Correlation Filter in DSST into the KCF tracking framework. Before estimating the scale of the object, we primarily determine the position of the object through a Kernelized Correlation Filter. Then, we extract a series of samples of different scales in the current position to train a one-dimensional scale filter. Its objective function is:

$$\varepsilon = \left\| \sum_{l=1}^{d} \boldsymbol{h}^l * \boldsymbol{f}^l - \boldsymbol{g} \right\|^2 + \lambda \sum_{l=1}^{d} \left\| \boldsymbol{h}^l \right\|^2 \tag{16}$$

Where $\boldsymbol{l} = \boldsymbol{1}, \boldsymbol{2}, \ldots \boldsymbol{d}$ is the dimension of the feature, $\boldsymbol{h}$ is the scale filter we want to train, $\boldsymbol{f}$ is the input image patch, $\boldsymbol{g}$ is the expected output of the Gaussian distribution, and $\boldsymbol{\lambda}$ is the regularization coefficient that eliminates the zero-frequency component of the spectrum.

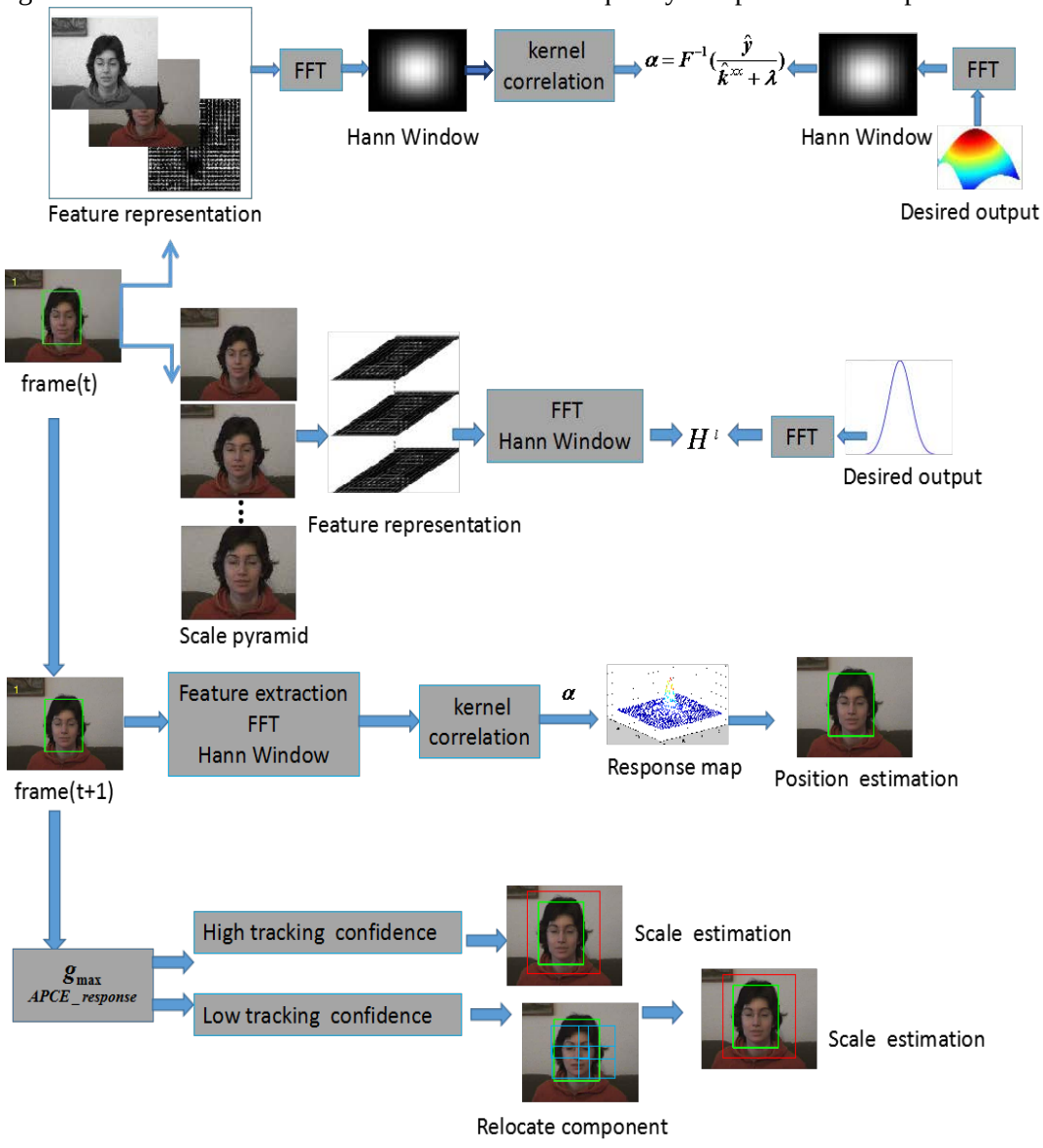

Fig. 1. The pipeline of our proposed method

By simplifying, we can find:

$$\boldsymbol{H}^l = \frac{\overline{\boldsymbol{G}}\boldsymbol{F}^l}{\sum_{k=1}^{d} \overline{\boldsymbol{F}}^k \boldsymbol{F}^k + \lambda} = \frac{\boldsymbol{A}_t^l}{\boldsymbol{B}_t} \tag{17}$$

By convention, variables in the time domain are generally represented by lowercase letters, while variables in the frequency domain are generally represented by uppercase letters. In particular, $\overline{\boldsymbol{G}}$ and $\overline{\boldsymbol{F}}^{\boldsymbol{k}}$ represent the conjugate complex numbers of $\boldsymbol{G}$ and $\boldsymbol{F}^{\boldsymbol{k}}$, respectively. Therefore, the response can be obtained through the following formula.

$$\boldsymbol{y} = \boldsymbol{F}^{-\mathbf{1}}\left\{\frac{\sum_{l=1}^{d}\overline{\boldsymbol{A}}^{l}\boldsymbol{Z}^{l}}{\boldsymbol{B}+\boldsymbol{\lambda}}\right\} \tag{18}$$

Here, $\boldsymbol{Z}$ is the Fourier transform of a feature map at the predicted location.

It is worth noting that before the feature was extracted, we performed a double-threshold judgment for the tracking object. When the size of the object is larger than $\boldsymbol{threshold}_{\boldsymbol{max}}$, we reduce its resolution. On the one hand, we do not require so many details, but on the other hand, the details increase the computation time and affects the real-time tracking. When the tracking object is smaller than $\boldsymbol{threshold}_{\boldsymbol{min}}$, we upsample the object image block to obtain more texture information for the image, which improves the tracking accuracy [26].

In the KCF tracker, the expansion coefficients in width and height of the object are both set to 2.5. However, by analyzing the video sequences, the high object may move in a fixed direction. For example, pedestrian scenes basically only contain motion in the left or right direction. Therefore, when the height and width ratio exceed $\boldsymbol{threshold}_{\boldsymbol{sz(1)/sz(2)}}$, we increase the search range in the width direction. Specifically, through experimental verification, the expansion coefficient in height of the object is set to 2.1, and the expansion coefficient in width of the object is set to 2.5 [21].

As we all know, training an independent filter requires a certain number of samples. Therefore, the number of scales cannot be set too small. However, If the number of scales is set too large, it will seriously affect the speed of the tracker. Additionally, the number of scales must be set to an odd number. Because it is possible that the scale of the object will not change, and the scaling of the scale appears in pairs, such as 0.9 and 1.1. Finally, by tuning the parameters on the OTB-2013 dataset, we reduced the number of scales from 33 to 17 in our work. The size of the response map is 1×17. Then, we used the triangle interpolation algorithm to interpolate the number of scales from 17 to 33 for more advanced accurate scale positioning [7].

4.2 Rich Image Feature Representation

Recently, more people have paid more attention to the observation model in tracking without considering the image feature representation module. We can draw a conclusion from a previous paper [29] that feature extraction is the most important part of the tracking process. Choosing a different observation model when the feature is good enough is not very important for tracking results. From recent research, we found that the FHOG (Fast Histogram of Oriented Gradients) feature is very popular for non-deep learning algorithms. The FHOG feature is a classic dense trait descriptor that has achieved good results in tracking. However, it is not enough to utilize a single feature in complex scenes. In our work, we come up with more powerful feature representation for video object tracking. We found that the nuclear correlation only needs to be modified as shown in formula (19) when KCF deals with multi-channel problems. Therefore, it does not provide more complicated calculations.

$$\boldsymbol{k}_{\boldsymbol{xx'}} = \boldsymbol{exp}\left(-\frac{\mathbf{1}}{\boldsymbol{\sigma}^{\mathbf{2}}}\left(\|\boldsymbol{x}\|^{\mathbf{2}} + \|\boldsymbol{x'}\|^{\mathbf{2}} - \mathbf{2}\boldsymbol{F}^{-\mathbf{1}}\left(\sum_{\boldsymbol{c}}\hat{\boldsymbol{x}}_{\boldsymbol{c}}^{\ *} \odot \hat{\boldsymbol{x}}_{\boldsymbol{c}}\right)\right)\right) \tag{19}$$

Where $\boldsymbol{x'}$ can represent both $\boldsymbol{x}$ and $\boldsymbol{z}$. Therefore, this formula can be used to solve both $\boldsymbol{k}_{\boldsymbol{xx}}$ and $\boldsymbol{k}_{\boldsymbol{xz}}$. Here, $\boldsymbol{c}$ represents the number of connected channels, i.e., $\boldsymbol{x} = [\boldsymbol{x}_{\mathbf{1}}, \boldsymbol{x}_{\mathbf{2}}, \cdots \boldsymbol{x}_{\boldsymbol{c}}]$.

In the process of designing feature representation, we fuse FHOG features with the Color Names. The Color Names are the human language tags of color. We often make use of them to describe the world around us. Prior papers [30]-[31] detail the process of learning 11 color from the images in Google, using PLSA (Probabilistic Latent Semantic Analysis). Like the ACNT tracker, we can convert RGB values into Color Names features through mapping matrices.

In addition, we consider that the gradient intensity of the target image generally does not vary drastically during the tracking process. Therefore, we also consider the gradient value of the image, as well as the processing and combination of image gradients. Inspired by the CNN features, we first use the Sobel operator to calculate image gradients in horizontal, vertical, 45-degree and 135-degree planes. Let us consider horizontal direction gradient processing as an example. We assume that the size of the horizontal direction gradient value matrix $\boldsymbol{D_x}$ is $\boldsymbol{m} * \boldsymbol{n}$. When the element in $\boldsymbol{D_x}$ is smaller than $-\boldsymbol{thrGrad}$, the corresponding element is set to 1. When the element in $\boldsymbol{D_x}$ is larger than $\boldsymbol{thrGrad}$, the corresponding element is set to 2. The value of the remaining position is 0. Then, we divide the matrix into $\boldsymbol{cellSize} * \boldsymbol{cellSize}$ cells. Next, we cycle through the pixel values in the cell and record the number of pixel values for a specific value. We will calculate the integral image as $\boldsymbol{feat}$. Finally, we use the following formula to normalize the integral image:

$$\boldsymbol{feat_{normalized}} = \boldsymbol{feat}./(\boldsymbol{cellSize} * \boldsymbol{cellSize}) \tag{20}$$

Finally, we can obtain an $(\boldsymbol{m/cellSize}) * (\boldsymbol{n/cellSize}) * \boldsymbol{3}$ matrix. Using the same approach, we can cope with gradients in other directions. So, we can obtain a 12-dimensional feature.

4.3 Relocation Component

Usually, we utilize a trained Correlation Filter and candidate image patches to calculate the response map during the detection process. We identify the location of the object by finding the location of the maximum in the response map. However, the LMCF (Large Margin with Circulant Feature Maps) tracker [32] propose that the response map is not always distributed in a Gaussian in complex scenes, especially when the tracking object is occluded and moving fast. At this point, the location of the object may not be at the position of the peak of the response map. Once we adopt the detection results directly, there is a great possibility that there would be tracking drift and even more tracking loss. After being enlightened by LCT (Long-term Correlation Tracker) [33], we introduced a relocation component. Compared with LCT, it is not necessary to train an additional Correlation Filter and a detector to reflect the tracking confidence by using our method, which will enhance the tracking efficiency. During the tracking process, we utilize the maximum response value and $\boldsymbol{APCE\_response}$ [32] as the tracking confidence indicator. In our work, it is defined as:

$$\boldsymbol{APCE\_response} = \frac{|\boldsymbol{g_{max}} - \boldsymbol{g_{min}}|^2}{\boldsymbol{mean}\left(\sum_{m,n}(\boldsymbol{g_{m,n}} - \boldsymbol{g_{min}})^2\right)} \tag{21}$$

Where $\boldsymbol{g_{max}}$ and $\boldsymbol{g_{min}}$ represent the maximum and minimum values of the response map, respectively. m and n represent the length and width of the response map, respectively. When these two tracking confidence indicators are less than the historical mean within a specified range, we think the tracking result is inaccurate, at the same time the relocation component will work. Considering the expensive calculation cost, we did not adopt the method of global search in the process of relocation. We believe there are several locations where tracking

object is most likely to occur. First, when the object moves fast, it is not in the vicinity of the position of the preceding frame. Through the paper [32], we found that the location of the object may be at the second peak or other peaks. Additionally, when the object does not move quickly, it should probably appear near the location of the position estimation in the previous frame. We use the following formula to calculate the length of the search range.

$$\boldsymbol{l} = \sqrt{\frac{\boldsymbol{c_1}}{\boldsymbol{g_{max}} * (\boldsymbol{target\_sz}(\mathbf{1}) + \boldsymbol{target\_sz}(\mathbf{2}))}} \tag{22}$$

Here, $\boldsymbol{c_1}$ is a constant. We set a step size in the rectangle search area to determine the position of the object. We use the Correlation Filter model to traverse the estimated positions of these two cases and compare the resulting response plots. The maximum position of $\boldsymbol{g_{max}}$ is the final estimated position of the object.

5. Experiments and Analysis

In this section, we mainly introduce the experimental results of the algorithm. Before presenting the experimental results, we will introduce the algorithm's experimental setup in section 5.1. In Sections 5.2 and 5.3, we will test and analyze the performance of the algorithms on the OTB-2013 [1] and OTB-2015 [34] datasets in detail to fully illustrate the effectiveness of the algorithm.

5.1 Experimental Setup

In this section, we mainly introduce the Experimental Setup for this paper. In section 5.1.1, we show the parameter settings for our experiments. Then, in Sections 5.1.2 and 5.1.3, we separately describe the datasets used in this paper, the evaluation indicators and evaluation methods.

5.1.1 Parameters

In this paper, we implemented the algorithm that we proposed using MATLAB language. It should be mentioned that the choice of parameters is very important when testing on a dataset. Even if there are slight differences, they are very likely to affect the performance of the trace. **Table 1** below shows some of the parameters selected in the paper.

Table 1. Main parameters used in our method

Parameter	Value
$threshold_{max}$	150*150
$threshold_{min}$	80*80
$thrGrad$	8
c_1	20.5
cellSize	4
$threshold_{sz(1)/sz(2)}$	2.50
orientation	9
appearance model update rate	0.012

5.1.2 Datasets

To test the performance of the algorithm, we adopt the OTB-2013 [1] and OTB-2015 [34] datasets, which are widely accepted in the video object tracking field. Prior to the advent of

OTB-2013 [1], without an accepted database, we could not compare the performance of the algorithm well. Therefore, the significance of this database is pretty valuable and it facilitates the development of tracking algorithms. This dataset contains of 50 video sequences. OTB-2015 [34] extends it from 50 video sequences to 100 video sequences. The video sequence includes 11 complex tracking scenes: Illumination Variation, Scale Variation, Occlusion, Deformation, Motion Blur, Fast Motion, In-Plane Rotation, Out-of-Plane Rotation, Out-of-View, Background Clutters, Low Resolution. Of these 100 sequences, there are 26 gray sequences and 74 colorful sequences. The tracking targets appearing in this database cover 36 bodies, 26 faces or heads, and a total quantity of 58897 frame images. The lengths of the video sequence include both short-term and long-term lengths.

5.1.3 Evaluation indicators and evaluation methods

The database provides two kinds of evaluation indicators: accuracy and success rate. Accuracy is an evaluation indicator that is used to measure the center location error. Specifically, it calculates the accuracy of the tracker through the difference between the estimated position of the algorithm and the ground truth. In the precision plot, since different thresholds correspond to different accuracy rates, we can obtain a central pixel position error curve. To some extent, this indicator can be used to measure the accuracy of tracker. However, this indicator has an obvious disadvantage: it does not reflect the changes in the scale of the tracking object. The success rate indicator solves this problem. It indicates the overlap score. The area overlap rate is measured using the ratio of overlapping areas and can be calculated by the following formula:

$$\emptyset = \frac{\boldsymbol{S_t} \cap \boldsymbol{S_g}}{\boldsymbol{S_t} \cup \boldsymbol{S_g}} \tag{23}$$

In this formula, $\boldsymbol{S_t}$ denotes the area estimated by the proposed algorithm, and $\boldsymbol{S_g}$ denotes the actual tracking target area. We combine this indicator with a precision plot to evaluate the algorithm tracking performance and show that the combination is more reliable and fairer for the performance between the algorithms.

Additionally, the dataset provides three evaluation methods: one-pass evaluation (OPE), temporal robustness evaluation (TRE), and spatial robustness evaluation (SRE). The difference in the conventional OPE, the TRE starts tracking from any frame with adding time interference. The SRE performs the tracking test by adding interference to the ground truth of the first frame of the image. In this paper, we only focus on OPE because OPE is basically the same as the other two evaluation methods.

5.2 Comparison with the-state-of-the-art trackers

In this section, we compare the proposed algorithm with the current 20 state-of-art trackers on the OTB-2013 and OTB-2015 datasets. The algorithms used for comparison are classic. For example, the most cited algorithm us DCF/KCF in the Correlation Filtering field. The comparison algorithm includes not only the original algorithm but also some improved algorithm based on the original tracking algorithm, including the DCF_CA (Context-Aware Discriminative Correlation Filter) [35], and the KCF_AT, DCF_AT (Target response Adaptation Correlation Filter) [36] tracker. We also compare the algorithms DSST and SAMF that can solve multi-scale problems as well as their improvements: FDSST (Fast Discriminative Scale Space Tracker) [7], SAMF_CA [35], and SAMF_AT [36]. Additionally, we compare some trackers based on deep learning such as DLSSVM (Dual Linear Structured

SVM and Explicit Feature Map) [37] and SiamFC. Currently, deep learning is very popular because of its good tracking performance. LCT, MEEM, SRDCF, Staple, Staple_CA, and RPT (Reliable Patch Tracker) [38] are also taken into account. In the related work, we have introduced these algorithms in detail.

We tested the performance of these algorithms on the OTB-2013 and OTB-2015 datasets. **Fig. 2** shows the experimental results in detail. In the upper right corner of the image, we show the top ten trackers that track performance. From **Fig. 2**, we can easily find that our proposed tracker shows good tracking performance for both evaluation indicators. On the OTB-2013 dataset, the accuracy is 86.1% and the success rate is 64.9%. Compared with KCF, these two indicators increased by 12.1% and 13.5% respectively. On the OTB-2015 dataset, the accuracy is 81.3% and the success rate is 60.3%. Compared with the original KCF, these two indicators increased by 11.7% and 12.6%, respectively. Additionally, we can easily find that LCT, SRDCF, and SAMF_CA have also achieved good tracking results for these two datasets.

On the OTB-2013 dataset, LCT ranks second in both evaluation indicators. LCT is a long-term object tracker. It uses three Correlation Filters and a re-detection module in its tracking framework. Similar to our algorithm, the re-detection module mitigates the effects of occlusion and model drift. On the OTB-2015 datasets, the second-place algorithms are SRDCF and SAMF_CA, respectively. The similarities between these two algorithms are that they use different methods to solve the boundary effect problems, which have caused by cyclic shifts. Therefore, these methods of mitigating border effects are effective and worthy being referenced.

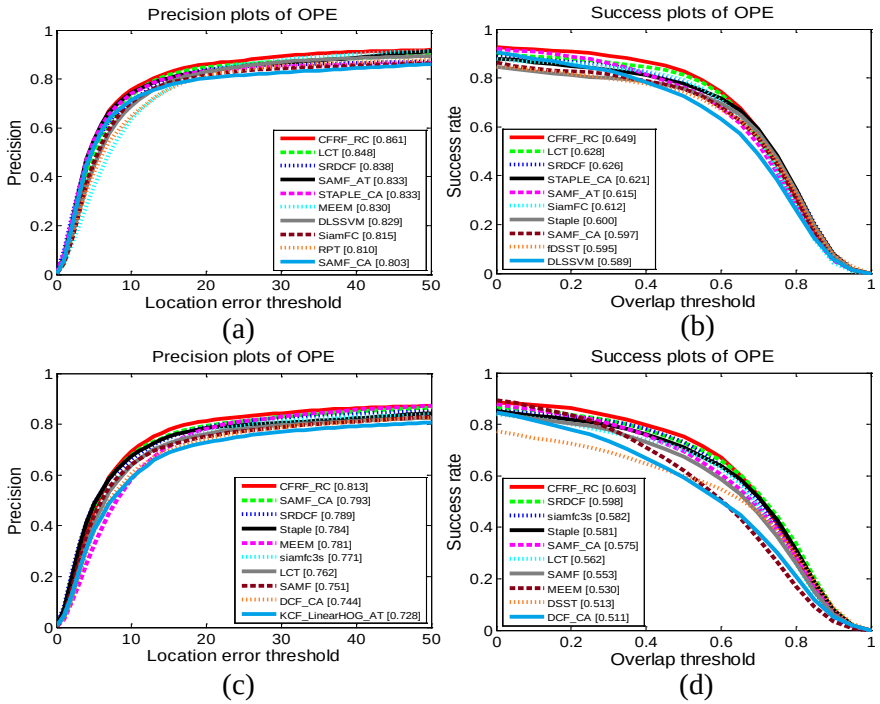

Fig. 2. Experimental results of the state-of-the-art trackers on the OTB-2013 and OTB-2015 datasets. (a) Precision plot of OPE on OTB-2013. (b) Success plot of OPE on OTB-2013. (c) Precision plot of OPE on OTB-2015. (d) Success plot of OPE on OTB-2015.

As we all know, tracking is usually very complicated in practical scenarios. Tracking algorithms may encounter various challenges during the tracking process. In our work, we

have tested not only the overall performance of our proposed algorithm on the datasets but also the tracking performance for the tracker for all kinds of attributes. **Fig. 3** shows the partial test results for the trackers on OTB-2013 and OTB-2015 datasets. In this section, we only use the AUC evaluation indicator. From the test results, whatever it is on OTB-2013 or on OTB-2015, our proposed algorithm is considered the best when the target is under Out-of-Plane Rotation, Occlusion or In-Plane Rotation. Under Scale Variation and Deformation attribute, our proposed algorithm ranks second compared with other algorithms.

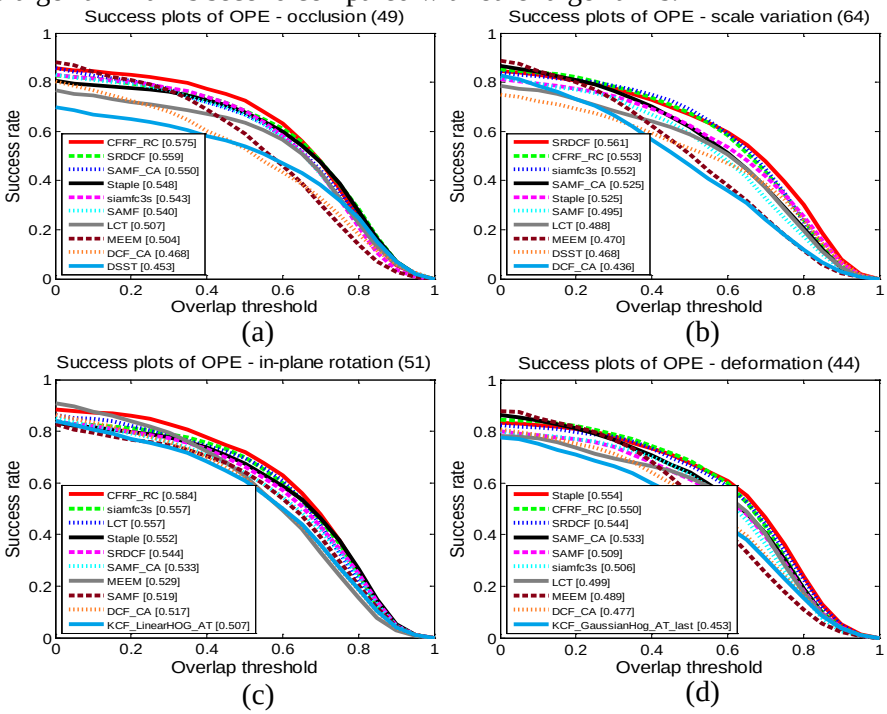

Fig. 3. Experimental results of the state-of-the-art trackers on the OTB-2015 dataset. (a) Success plots of OPE under occlusion. (b) Success plots of OPE under scale variation. (c) Success plots of OPE under in-plane rotation. (d) Success plots of OPE under deformation.

Next, we conducted a detailed attribute analysis. On the OTB-2015 dataset, the success rate of our method is 55.3% under Scale Variation. We found that the algorithm can still obtain good tracking results with Scale Variation because we combined position estimation with scale estimation. During the tracking process, we did not directly adopt the results of the position estimation of the target. We constructed an image pyramid at the estimated position to estimate the scale of the target. Additionally, our scale Correlation Filter is constantly updated linearly. Here, we applied the compressed scale Correlation Filter to the KCF tracker and called it KCF_Scale. The specific implementation details are in Section 4.1. We then compared the KCF_Scale tracker with the SAMF and DSST multi-scale trackers on the OTB-2013 and OTB-2015 datasets. The experimental results are shown in **Table 2**. We can see that the KCF_Scale tracker has achieved a great speed increase while maintaining a high precision. On the one hand, the reduction in the number of scales increases the speed of tracking. On the other hand, we use another trick in the algorithm implementation to improve the tracking speed. We utilize the Hermitian symmetry of the Fourier coefficient of a real function to reduce the computational complexity and memory consumption by half. It can greatly improve tracking speed without losing any performance.

Table 2. Experimental results on the OTB-2013 and OTB-2015 datasets.

Tracker	Precision (2013)	AUC (2013)	Speed (fps)	Precision (2015)	AUC (2015)	Speed (fps)
KCF_Scale	0.793	0.581	86.5	0.762	0.561	85.4
SAMF	0.785	0.579	18.5	0.751	0.553	17.9
DSST	0.740	0.554	28.9	0.680	0.513	21.5

Additionally, our method's success rate is 57.5% under occlusion. The introduction of the relocation component enhances the algorithm's ability to resist occlusion. For other attributes, our proposed method also shows good performance on the benchmark. To show the tracking results of different trackers in the video sequence more intuitively, we show the comparisons between our method and other trackers in several sequences in **Fig. 4**.

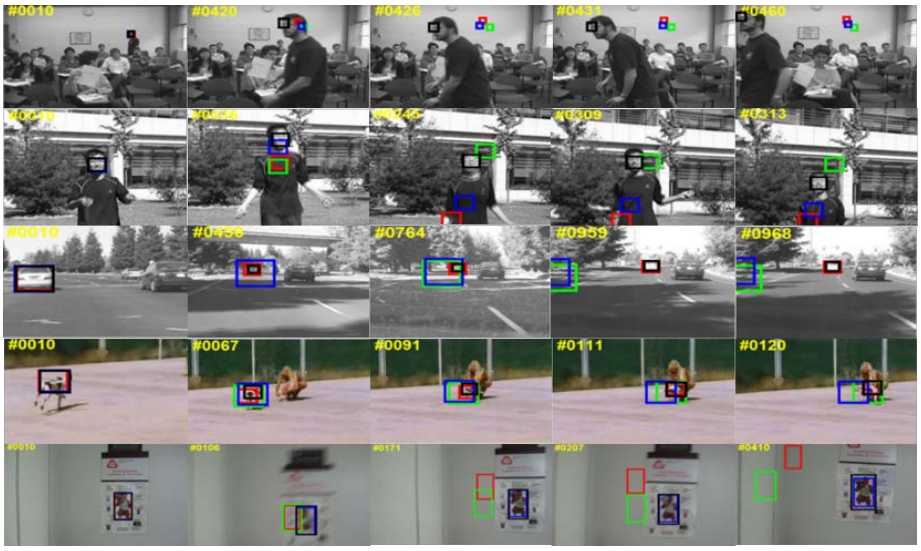

Fig. 4. Comparison of the tracking bounding boxes generated by different trackers for several challenging sequences. (From top to bottom: Freeman3, Jumping, Car, Dog, BlurOwl). The black, red, green, and blue tracking boxes are generated by our method, DCF_CA, DSST, and KCF, respectively.

5.3 Self-contrast Test

In this section, we verified the effectiveness of our work by experiments. We still use the OTB-2013 and OTB-2015 datasets. We applied different components on the base trackers separately. For example, in the KCF_RF tracker, we fused different features. The KCF_RC tracker adds the relocation component to the KCF. The same is true for other algorithms, such as the DCF_RF, DCF_RC, SAMF_RF, and SAMF_RC tracker. Here, we take the KCF tracker as an example. The accuracy of the KCF_RF tracker is 79.6%, and the success rate is 54.7%. Compared with its base tracker, its accuracy is increased by 5.6% and the success rate is increased by 3.3%. The accuracy of the KCF_RC tracker is 77.9%, and the success rate is 54.0%. Compared with its base tracker, its accuracy is increased by 3.9% and the success rate is increased by 2.6%. It is not difficult to determine from **Fig. 5** that the innovations we propose are all valid.

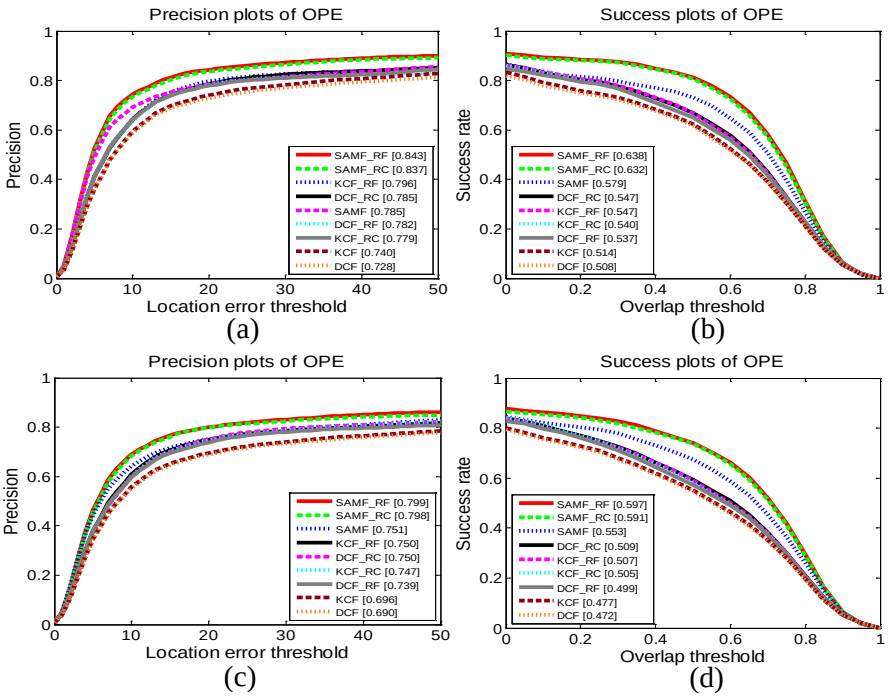

Fig. 5. Experimental results of various combination trackers on OTB-2013 and OTB-2015 datasets.

6. Conclusion

In this paper, we propose a novel tracking algorithm. The introduction of the double threshold judgment and the triangle interpolation algorithm makes the scale estimation more quickly and accurately. In the feature extraction module, we develop a novel and powerful feature that can significantly improve the tracking performance. In addition to the traditional FHOG and CN features, we also consider the gradient value of the image, as well as the processing and combination of image gradients. Furthermore, the position of the tracking object may not be in the right position when the response value is maximized. To solve this problem, we introduce a relocation component. It is a real-time tracker.

Acknowledgments

This research is partially supported by National Natural Science Foundation of China (No.61 876018), Program for New Century Excellent Talents in University (NCET-13-0659).

References

[1] Wu, Yi, Jongwoo Lim, and Minghsuan Yang, "Online Object Tracking: A Benchmark," in *Proc. of 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2411-2418, 2013. Article (CrossRef Link)

[2] Zhang, Jianming, Shugao Ma, and Stan Sclaroff, "MEEM: Robust Tracking via Multiple Experts Using Entropy Minimization," in *Proc. of european conference on computer vision*, vol. 8694, pp. 188-203, September 06-12, 2014. Article (CrossRef Link)

[3] Vojir, Tomas, Jana Noskova, and Jiri Matas, "Robust scale-adaptive mean-shift for tracking," *Pattern Recognition Letters*, vol. 49, pp. 250-258, 2014. Article (CrossRef Link)
[4] Bolme D S, Beveridge J R, Draper B A, et al, "Visual object tracking using adaptive correlation filters," in *Proc. of 2010 IEEE Computer Society Conference on computer vision and pattern recognition*, pp. 2544-2550, June 13-18, 2010. Article (CrossRef Link)
[5] Henriques J F, Caseiro R, Martins P, et al, "High-Speed Tracking with Kernelized Correlation Filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583-596, March 1, 2015. Article (CrossRef Link)
[6] Danelljan M, Hager G, Khan F S, et al, "Accurate Scale Estimation for Robust Visual Tracking," in *Proc. of british machine vision conference*, 2014. Article (CrossRef Link)
[7] Danelljan M, Hager G, Khan F S, et al, "Discriminative Scale Space Tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 8, pp. 1561-1575, Aug. 1, 2017. Article (CrossRef Link)
[8] Danelljan M, Khan F S, Felsberg M, et al, "Adaptive Color Attributes for Real-Time Visual Tracking," in *Proc. of computer vision and pattern recognition*, pp. 1090-1097, June 23-28, 2014. Article (CrossRef Link)
[9] Henriques J F, Caseiro R, Martins P, et al, "Exploiting the circulant structure of tracking-by-detection with kernels," in *Proc. of european conference on computer vision*, vol. 7575, pp. 702-715, 2012. Article (CrossRef Link)
[10] Lukezic A, Vojir T, Zajc L C, et al, "Discriminative Correlation Filter with Channel and Spatial Reliability," in *Proc. of 2017 IEEE Conference on computer vision and pattern recognition*, pp. 4847-4856, July 21-26, 2017. Article (CrossRef Link)
[11] Galoogahi, Hamed Kiani, Ashton Fagg, and Simon Lucey, "Learning Background-Aware Correlation Filters for Visual Tracking," in *Proc. of international conference on computer vision*, pp. 1144-1152, October 22-29, 2017. Article (CrossRef Link)
[12] Hare S, Golodetz S, Saffari A, et al, "Struck: Structured Output Tracking with Kernels," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 10, pp. 2096-2109, 2016. Article (CrossRef Link)
[13] Li, Yang, and Jianke Zhu, "A Scale Adaptive Kernel Correlation Filter Tracker with Feature Integration," in *Proc. of european conference on computer vision*, vol. 8926, pp. 254-265, September 06-12, 2014. Article (CrossRef Link)
[14] Danelljan M, Hager G, Khan F S, et al, "Learning Spatially Regularized Correlation Filters for Vi sual Tracking," in *Proc. of international conference on computer vision*, pp. 4310-4318, Decemb er 11-18, 2015. Article (CrossRef Link)
[15] Galoogahi, Hamed Kiani, Terence Sim, and Simon Lucey, "Correlation filters with limited bound aries," in *Proc. of computer vision and pattern recognition*, pp. 4630-4638, June 7-12, 2015. Article (CrossRef Link)
[16] Bertinetto L, Valmadre J, Golodetz S, et al, "Staple: Complementary Learners for Real-Time Tracking," in *Proc. of computer vision and pattern recognition*, pp. 1401-1409, June 27-30, 2016. Article (CrossRef Link)
[17] Possegger, Horst, Thomas Mauthner, and Horst Bischof, "In defense of color-based model-free tracking," in *Proc. of computer vision and pattern recognition*, pp. 2113-2120, June 7-12, 2015. Article (CrossRef Link)
[18] Ma C, Huang J, Yang X, et al, "Adaptive Correlation Filters with Long-Term and Short-Term Memory for Object Tracking," *International Journal of Computer Vision*, vol. 126, no. 8, pp. 771-796, 2018. Article (CrossRef Link)
[19] Wang J, Liu W, Xing W, et al, "Visual object tracking with multi-scale superpixels and color-feat ure guided kernelized correlation filters," *Signal Processing-image Communication*, vol. 63, pp. 4 4-62, 2018. Article (CrossRef Link)
[20] Wang J, Liu W, Xing W, et al, "Two-level superpixel and feedback based visual object tracking," *Neurocomputing*, vol. 267, pp. 581-596, 2017. Article (CrossRef Link)

[21] Ma C, Huang J, Yang X, et al, "Hierarchical Convolutional Features for Visual Tracking," in *Proc. of international conference on computer vision*, pp. 3074-3082, December 7-13, 2015. Article (CrossRef Link)
[22] Bertinetto L, Valmadre J, Henriques J F, et al, "Fully-Convolutional Siamese Networks for Object Tracking," in *Proc. of european conference on computer vision*, pp. 850-865, Jun 30, 2016. Article (CrossRef Link)
[23] Huang, Chen, Simon Lucey, and Deva Ramanan, "Learning Policies for Adaptive Tracking with Deep Feature Cascades," in *Proc. of international conference on computer vision*, pp. 105-114, October 22-29, 2017. Article (CrossRef Link)
[24] Valmadre J, Bertinetto L, Henriques J F, et al, "End-to-End Representation Learning for Correlation Filter Based Tracking," in *Proc. of computer vision and pattern recognition*, pp. 5000-5008, July 21-26, 2017. Article (CrossRef Link)
[25] Danelljan M, Robinson A, Khan F S, et al, "Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking," in *Proc. of european conference on computer vision*, pp. 472-488, October 08-16, 2016. Article (CrossRef Link)
[26] Danelljan M, Bhat G, Khan F S, et al, "ECO: Efficient Convolution Operators for Tracking," in *Proc. of computer vision and pattern recognition*, pp. 6931-6939, July 21-26, 2017. Article (CrossRef Link)
[27] Nam, Hyeonseob, and Bohyung Han, "Learning Multi-domain Convolutional Neural Networks for Visual Tracking," in *Proc. of computer vision and pattern recognition*, pp. 4293-4302, June 27-30, 2016. Article (CrossRef Link)
[28] Wang L, Ouyang W, Wang X, et al, "Visual Tracking with Fully Convolutional Networks," in *Proc. of international conference on computer vision*, pp. 3119-3127, December 7-13, 2015. Article (CrossRef Link)
[29] Wang N, Shi J, Yeung D Y, et al, "Understanding and Diagnosing Visual Tracking Systems," in *Proc. of international conference on computer vision*, pp. 3101-3109, December 7-13, 2015. Article (CrossRef Link)
[30] De Weijer, J. Van, Cordelia Schmid, and Jakob J. Verbeek, "Learning Color Names from Real-World Images," in *Proc. of computer vision and pattern recognition*, pp. 1-8, June 17-22, 2007. Article (CrossRef Link)
[31] De Weijer J V, Schmid C, Verbeek J J, et al, "Learning Color Names for Real-World Application s," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512-1523, 2009. Article (CrossRef Link)
[32] Wang, Mengmeng, Yong Liu, and Zeyi Huang, "Large Margin Object Tracking with Circulant Feature Maps," in *Proc. of computer vision and pattern recognition*, pp. 4800-4808, July 21-26, 2017. Article (CrossRef Link)
[33] Ma C, Yang X, Zhang C, et al, "Long-term correlation tracking," in *Proc. of computer vision and pattern recognition*, pp. 5388-5396, June 7-12, 2015. Article (CrossRef Link)
[34] Wu, Yi, Jongwoo Lim, and Minghsuan Yang, "Object Tracking Benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834-1848, September 1, 2015. Article (CrossRef Link)
[35] Mueller, Matthias, Neil Smith, and Bernard Ghanem, "Context-Aware Correlation Filter Tracking," in *Proc. of computer vision and pattern recognition*, pp. 1387-1395, July 21-26, 2017. Article (CrossRef Link)
[36] Bibi, Adel, Matthias Mueller, and Bernard Ghanem, "Target Response Adaptation for Correlation Filter Tracking," in *Proc. of european conference on computer vision*, vol. 9910, pp. 419-433, October 08-16, 2016. Article (CrossRef Link)
[37] Ning J, Yang J, Jiang S, et al, "Object Tracking via Dual Linear Structured SVM and Explicit Feature Map," in *Proc. of computer vision and pattern recognition*, pp. 4266-4274, June 27-30, 2016. Article (CrossRef Link)
[38] Li, Yang, Jianke Zhu, and Steven C H Hoi, "Reliable Patch Trackers: Robust visual tracking by exploiting reliable patches," in *Proc. of computer vision and pattern recognition*, pp. 353-361, June 7-12, 2015. Article (CrossRef Link)

Menglei Jin received the B.S. degree in Electronic and Information Engineering at Hebei University in 2016. He is pursuing his M.S. degree in Signal and Information Processing at Beijing Jiaotong University, Beijing. His research interests mainly include video object tracking, image processing.

Weibin Liu received the Ph.D. degree in Signal and Information Processing from Institute of Information Science at Beijing Jiaotong University, China, in 2001. During 2001-2005, he was a researcher in Information Technology Division at Fujitsu Research and Development Center Co., LTD. Since 2005, he has been with the Institute of Information Science at Beijing Jiaotong University, where currently he is a professor in Digital Media Research Group. He was also a visiting researcher in Center for Human Modeling and Simulation at University of Pennsylvania, PA, USA during 2009-2010. His research interests include computer vision, computer graphics, image processing, virtual human and virtual environment, and pattern recognition. He is a member of the IEEE, ACM, IEICE and CCF.

Weiwei Xing received her B.S. degree in Computer Science and Technology and Ph.D. degree in Signal and Information Processing from Beijing Jiaotong University, in 2001 and 2006 respectively. During 2011-2012, she was a visiting scholar at University of Pennsylvania. Currently, she is a professor at School of Software Engineering, Beijing Jiaotong University. Her research interests mainly include intelligent information processing and artificial intelligence.