# 物体の状況を考慮した適応制御を備えた相関フィルターに基づく移動物体追跡

明治大学大学院先端数理科学研究科

2021 年度

博士学位請求論文

# Visual Tracking based on Correlation Filter with Adaptive Control Considering the Situation of Objects

学位請求者　先端メディアサイエンス専攻

唐　兆前

# Abstract

Computer vision plays a significant role in various areas, and it is a trendy research direction at present. In some high-level applications of computer vision, visual tracking is an essential component. After discriminative correlation filters are widely used in visual tracking, visual tracking has made great progress. Discriminative correlation filter trackers achieve a natural balance between excellent performance and real-time, making visual tracking technology easier to apply in actual life. However, there are still many challenges in visual tracking, such as deformation, illumination variation, occlusion, scale variation, *etc*. This thesis proposed three improved trackers based on a discriminative correlation filter. The first one is based on the most straightforward CF framework, which solves some common problems, such as scale variation, object rotation, etc. The second CF framework has better robustness and has been further improved in many challenges. The last algorithm is aimed at UAV, a popular research direction, and can meet video tracking with both low frame rate and high frame rate. The relationship among the three is an extension, that is to say, the latter is further improved based on the idea of the former.

The framework of the first tracker is the classic CF framework KCF. KCF is one of the representative algorithms to introduce multiple channels feature (HOG feature) to object tracking. The cell size of the HOG feature is set to 4, and this leads to a stride of the training and detection samples being greater than one pixel. The performance of KCF is seriously influenced and the precision of the object location decreases. The proposed method, named as CFCA, combines the four highest response scores with the corresponding luminance histogram similarity, enhancing the detection accuracy. Besides, CFCA improves the scale estimation method to solve scale variation. For occlusion, CFCA relocates the object by the judgment of the object state. However, the relocation method only mitigates the negative influence from occlusion and does not solve it completely.

The second tracker (CFASE) adopts a more robust DCF framework. In terms of the framework, CFASE increases the precision of temporal regularization. The framework of CFASE can learn the correlation filter model information from more frames than the previous frame to improve the robustness of the DCF model. For scale variation, CFASE only adopts the HOG feature to estimate the object's scale instead of hand-crafted features, and then the obtained scale is used to locate the object. This method increases the precision of the scale estimation. CFASE analyzes the object state to enhance the precision of the scale estimation and the location.

The last tracker (BASTR) is proposed to improve the universality of the tracker. The framework of BASTR is complex. For all kinds of challenges, adaptive spatial regularization and temporal regularization are introduced to increase the robustness of the tracker. For videos with high frame rate, scale pool technology can obtain better performance. In the contrast, DSST is better for videos with low frame rate. The generalize of the tracker is enhanced by selecting the scale estimation method accurately.

The evaluation experiments are conducted on different tracking benchmark databases. The result of the experiments verifies the validity of the proposed trackers. The advantages and disadvantages of each algorithm are also described in detail through experimental results.

# Acknowledgments

First, I would like to acknowledge my supervisor, professor Kaoru Arakawa. From selecting the topic to the completion of this thesis, Professor Arakawa put forward many valuable opinions. Professor Arakawa's rigorous academic attitude and profound knowledge, unpretentious and approachable personality profoundly impact me, and I would like to express my highest respect and gratitude to my mentor!

Finally, I would like to thank my wife, parents, and friends who accompany me, thank them for all the helpful suggestions and comments, and thank them for their support and understanding of my life!

## Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Introduction and motivation

Eyes are an important way for human beings to know and perceive the world. Human cognition of the world is also a process of visual learning [87]. Through continuous learning of new things, human cognition of objects is enriched. Today, if the computer learns to understand and automate tasks as the human visual system, human beings will gain great convenience in many aspects. Computer vision [48, 81] is the research about the eye of the computer, an interdisciplinary scientific field that tries to mimic the human visual system. Computer vision involves acquiring, processing, analyzing, and comprehending digital images and extracting high-dimensional data from the real world to generate numerical or symbolic information. In other words, computer vision transforms visual images (the retina's input) into descriptions of the world and can produce appropriate action. In this processing, some scientific fields (physics, geometry, and learning theory) achieve useful information from digit image data. Computer vision tasks contain sub-domains such as visual tracking [7, 11, 16, 26], object detection [44, 84], image restoration [15, 42], image segmentation [74, 80], 3D scene modeling [23], motion estimation [14], *etc*.

It is an arduous task in computer vision systems that accurately recognizes and tracks objects as human vision systems. Therefore, visual tracking is one of the interesting researches in the field of computer vision. Computer vision aims to perform higher-level tasks, visual tracking is an essential component. It involves knowledge of machine learning, image processing, pattern recognition, artificial intelligence, *etc*. With the continuous development of target tracking technology, visual tracking has been applied in many everyday scenarios as an intelligent surveillance system [62, 69], UAV [95], Intelligent transportation [58], self-driving [22, 24], and human-computer [21, 76, 77]. Visual tracking is divided into single-object tracking (SOT) [46, 47] and multi-object tracking (MOT) [1,18, 86]. The research content of this paper belongs to the field of single object tracking. Single object tracking is regarded as a challenging task in visual tracking, especially in complex scenarios. Given the initial information (generally position and size) of the temporally changing object in the first frame of a video sequence, generate the tracking model to locate the target object in the remaining image sequences. In other words, visual tracking aims to automatically identify objects in a video or an image sequence and high accurately estimate the future position of an object tracked. As shown in Fig 1.1, the position and size of the object are given, the new location and scale are estimated from the search region (yellow box) in the current frame. In general, the target is surrounded by a square (red box) to show the user where the object is on the screen [49]. The image data of visual tracking can have many forms, such as image sequence or video sequences. In the research, video sequences are habitually converted into image sequences for processing.

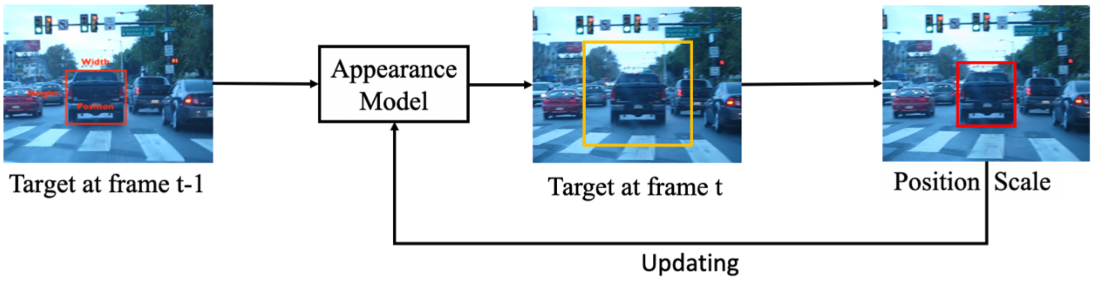


Figure 1.1 The process of tracking.

In general, each computer vision application employs a range of computer vision tasks [22, 58, 62]. As a mature technology, visual tracking has been applied to some everyday scenarios by combining object detection, image segmentation, and it is an essential part of many applications. Visual tracking is often applied in surveillance systems that single background, immobile camera, *etc*. When a target object is a rigid object with a simple background, most tracking algorithms [26, 41, 45, 46, 47] can achieve outstanding performance, and even some simple algorithms perform better than complex ones. However, for better application in more complex scenarios (UAV [95], Self-driving [22, 24]), researchers want to get more robust tracking algorithms, so specific objects and simple backgrounds are not limited in the study of visual tracking. Visual tracking algorithms are supposed to accurately localize objects of interest and do so in the least amount of time possible. A real-time object tracking model must enhance tracking speed. Since many challenges make it difficult for visual tracking models to perform detection and tracking effectively, it is difficult for trackers to balance outperformance and real-time. The main challenges are as follows:

In the process of moving, the non-rigid object will inevitably be deformed. Object deformation not only includes deformation of appearance but also the target rotates in the image plane. In general, slight deformation will not seriously affect the training of the feature model, even if the feature does not have deformation invariance. Therefore, most trackers can get a significant performance on special occasions. However, with the extension of the evaluation benchmark databases, the target object with the intense deformation or rotation becomes more and more. On this occasion, the feature model of the target object is hard to keep up with the change of the object appearance, thus directly affecting the reliability of the feature model. It is difficult for many trackers to maintain excellent experiment results on different benchmarks. Many effective methods are adopted to solve object deformation, such as the target being divided into several sub-modules for tracking or features with deformation invariance being introduced. For the CF framework, object deformation is also a significant challenge.

On many benchmark databases [68, 93], the videos' scenarios are varied, which leads to more challenges in the tracking process. Illumination variation can decrease the robustness of the color features [53, 54, 55]. Background clutter makes that the trained model overly learns background information, causing over-drifted. Fast motion results in the target object not being in the search region, and tracking fails. These challenges affect the robustness and accuracy of the visual tracking algorithms. For solving these challenges, many outstanding trackers are proposed. However, it remains a great challenge to solve all problems from external environments distractions at once.

In the process of object tracking, the target's location is changing, but also the scale of the target is changing. Many trackers ignore scale variation and achieve significant performance on a video with high frame rate. However, it is difficult for these trackers to obtain outperformance on a video with low frame rate. The main reason is that the range of the scale variation is too wide to be ignored for the video with low frame rate [17] [19]. It is difficult for a tracker to keep the robustness of the model without accurate scale estimation. The scale misestimation seriously impact tracking.

In addition, some factors influence the performance of trackers, such as out of view, and low resolution [93, 94].

Visual tracking algorithms are applied in an increasingly wide range of applications, but there are still unresolved problems, as mentioned above. Recently, deep learning algorithms have been introduced to gain attention for their excellent performance. However, these trackers have no way to obtain the real-time performance that relies on a single CPU. For many applications, the algorithms not only need to perform excellently but also maintain real-time performance. Otherwise, there is no way to achieve practical purposes. This paper mainly proposes improvement approaches based on discriminative correlation filter (DCF) for many of these reasons.

## 1.2 Earlier research

In recent ten years, the research of visual tracking has achieved outstanding progress. According to the development process, object tracking algorithms can be divided into three aspects. Firstly, object trackers are based on support vector machines (SVM) [89, 90]. SVM is used in the tracking-by-detection method to learn an online classifier to distinguish a target from its local background. Followed by object trackers based on a discriminative correlation filter (DCF) [26, 27 ,31, 47]. DCF trackers break the limitation of running speed, achieve the balance between the outperformance and real-time. Finally, deep learning is popular in some research fields [16, 56]. Object trackers based on deep learning also obtain excellent performance.

### 1.2.1 Visual tracking based on SVM

The normal understanding of target tracking is to distinguish the target object from the background. That is, to detect the target from the background. Therefore, tracking-by-detection [82, 88] is a particularly popular approach to tracking before 2013. This method treats the object tracking as a detection task. trackers based on support vector machines (SVM) becomes the dominant algorithms in the field of tracking-by-detection [82]. An excellent SVM tracker aims to maintain a high robust classifier trained online to distinguish the object from the background. These trackers treat the tracking problem as a classification task and use an online learning method to update the object model [35, 83]. They need to train a classifier with high robustness to distinguish the target object from its surrounding background. The process of trackers is separated into two distinct phases: Tracking and Update. During tracking, many samples are generated in a local region around the position from the previous frame using a sliding window approach, and an obtained classifier estimates new object location by searching for the maximum classification score. Using the estimated object position, trackers generate a set of training samples to update the classifier online [8, 51, 96].

Before 2013, there were no excellent tracking benchmark databases [5, 25, 30, 43] to evaluate and analyze the strengths and weaknesses of the algorithm. Most of the traditional object trackers [4, 33, 41, 52] are based on a specific task to propose a specific solution algorithm so that the generalization is not strong. Besides, traditional tracking algorithms cannot fully use high dimension features because of the high complexity of optimization and detection processes. High dimension features are critical to tracking performance because object appearance with high dimensional features is conducive to generating robustness model the object better than low dimensional ones. However, traditional tracking algorithms also provided a solid foundation for object tracking development, and many outstanding ideas are still used today.

### 1.2.2 Visual tracking based on discriminative correlation filter

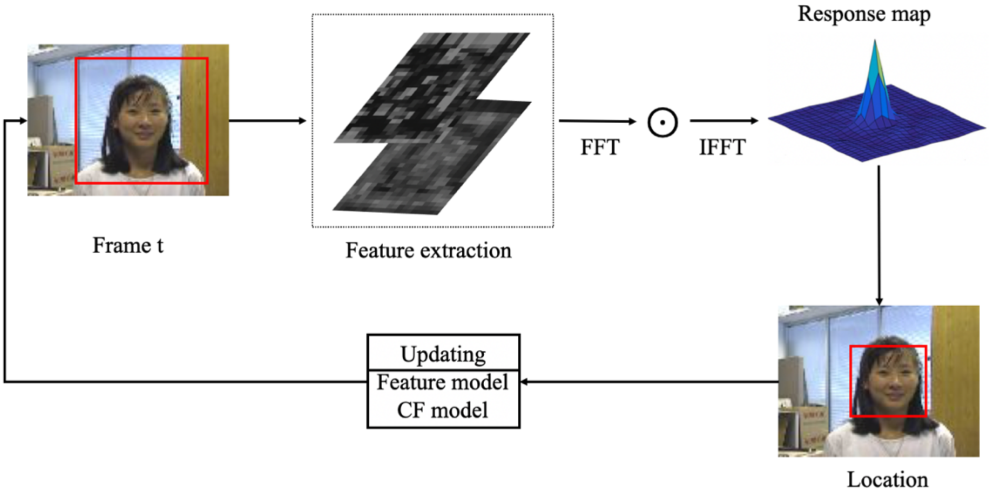


Figure 1.2 The process of discriminative correlation filter (DCF).

In 2010, a simple tracking approach was proposed, called the Minimum Output Sum of Squared Error (MOSSE) filter [26]. MOSSE is the first algorithm to introduce correlation filtering into visual tracking. Although this tracker does not achieve the excellent experiment result, the running speed of the tracker was surprisingly fast (operating at 669 frames per second), which also indicates that the discriminative correlation filter has a bright future in visual tracking. In the 1980's and 1990's, many correlation filters [3, 9, 10] were applied in the image process. For target objects with varying appearances and enforced hard constraints, these filters always produce peaks of the same height. These researches make a foundation for the application of correlation filtering in visual tracking.

Running speed of several hundred FPS is the highlight of the CF trackers [26, 46, 47], which directly opens a new research direction in visual tracking. Followed by trackers obtaining significant performance by solving different problems, such as scale variation [63, 65], boundary effect [38, 66], multi-channel features [36, 47], etc. As shown in Fig 1.2, the tracking process of trackers based on CF is the same as traditional tracking algorithms, including detection and updating. CF trackers extract the features from the object surrounding region. For calculation operation, CF trackers convert from the spatial domain to the frequency domain with FFT and then achieve a confidence map by the obtained CF model and the feature model. The peak score of the confidence map corresponds to the object's location. Finally, the CF model and the feature model are updated online. Trackers based on DCF regard the samples (As a hypothesis rather than an actual sample) generated by a sliding window as a cyclic matrix and use the characteristics of the cyclic matrix to transform the correlation calculation from the spatial domain to the frequency domain, thus achieving a significant increase in speed. Indeed, the excellent performance of the DCF trackers depends not only on the outstanding framework but also on the object feature. The robustness of the object feature can affect the performance of trackers.

### 1.2.3 Visual tracking based on Deep learning

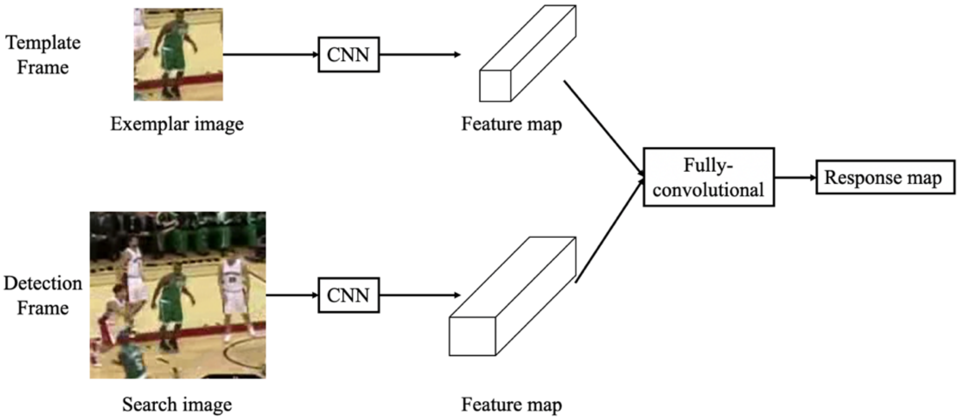


Figure 1.3 The structure of Siamese trackers

In recent, with the massive advancements in deep learning, deep neural networks [2] have achieved impressive accomplishments in some fields, such as object detection [84], object segmentation [61, 74, 80], and object tracking [16]. In the early stages, trackers based on deep learning introduce CNN features into the DCF framework [27, 50, 92]. The used feature from the deeper the network, the better the experimental result for target tracking. While the high dimensional feature improves the performance of trackers [60], the running speed of DCF trackers is affected by a dozens-fold decrease. DCF trackers also lost their original speed advantages.

At present, most of the object trackers based on deep learning are proposed using an offline trained fully-convolutional Siamese network architecture [16, 56, 78]. The architecture is shown in Fig 1.3, Siamese architecture is fully convolutional concerning the search image patch. The similarity function is used to compute for the feature model and all translated sub-windows within the search image patch, generating a scalar-valued score map (The dimension depends on the size of the search image). To improve the running speed of trackers based on deep learning, the fully connected layer of some trackers adopts the same method as DCF trackers to calculate the score map.

Trackers based on deep learning are becoming more sophisticated than just object tracking. By combining object detection, object segmentation, and object tracking, the performance of trackers become more excellent [16, 78]. However, it is difficult for trackers based on deep learning to achieve real-time with a single CPU.

## 1.3 Structure of this thesis

In this thesis, I propose three trackers based on the discriminative correlation filter. These trackers balance real-time and outperformance in tracking. Firstly, an improved tracker is proposed based on the simplest CF framework, which solves some common challenges, such as scale variation, object rotation, etc. Based on the improved method of the first algorithm, the second CF framework has better robustness and has been further improved in many challenges. The performance of the tracker is significantly improved. For a popular search direction UAV recently, an algorithm is proposed to meet video tracking with both high frame rate and low frame rate. The relationship between the three is an extension. The latter is further improved based on the idea of the former.

The main structure of this thesis is as follows.

In chapter 2, the classic algorithms based on discriminative correlation filters are reviewed. I will illustrate the theory of DCF trackers, and there are still problems.

In chapter 3, to improve the performance of KCF, Correlation Filter tracker using Confidence map and Adaptive model (CFCA) is proposed. Firstly, an improved scale estimation is used to deal with scale variation. Besides, KCF locates the object position by the maximum score of the confidence map. When the object is occluded, or the object deformation occurs, the reliability of the confidence map will decrease. The position corresponding to the maximum score is not necessarily the object's location. The proposed tracker combines the multiple high scores of the confidence map and the similarity of the luminance histogram to improve the precision of detection. And then, CFCA adaptively adjusts the learning rate of the model training by estimating the object's state. In the worst case, CFCA tries to relocate the target object with the peak of the confidence map.

In chapter 4, Correlation Filter tracker using a spatial-temporal regularized with Advanced Scale Estimation (CFASE) is proposed. In temporal regularization, CFASE trains the correlation filter model more precisely using temporal prediction from the previous two filters, the robustness of the DCF model is enhanced. CFASE trains the scale estimation model by the HOG feature and the localization model by the hand-crafted feature to solve scale variation. In tracking, the obtained scale is used to locate the object. This method increases the precision of the scale estimation. Average peak-to-correlation energy (*APCE*) [72] is introduced to evaluate the accuracy of scale estimation and object location.

In chapter 5, background-aware correlation filter tracker with spatial-temporal regularization (BASTR) is proposed. The framework of the proposed tracker (BASTR) is complex. Adaptive spatial regularization and temporal regularization are proposed to solve all kinds of challenges. For videos with high frame rate, scale pool technology can obtain better performance. In contrast, DSST is better for videos with low frame rate. The generalize of the tracker is enhanced by selecting the scale estimation method accurately.

Chapter 6 demonstrates the result of the evaluation experiments. I analyze the advantage and the relation of the proposed trackers. The validity of the trackers is further illustrated.

In the final chapter, Chapter 7 summarizes this paper and analyzes future work.

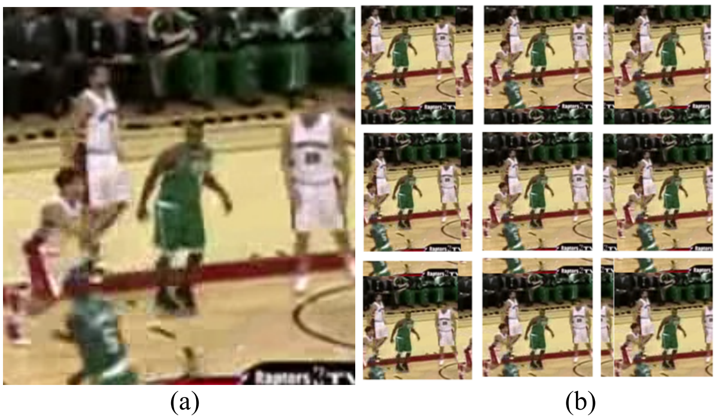
# Chapter 2

# Related work

This section briefly reviews the principle of classic DCF-based algorithms and the problems that remain to be solved [47]. Discriminative Correlation Filter is a hot research orientation in visual tracking because of its high speed and excellent performance. Many excellent DCF-based trackers have been proposed, and I demonstrate three significant trackers from these trackers which kernelized correlation filter (KCF), background aware correlation filters (BACF) [38], and spatial-temporal regularized correlation filter (STRCF) [31]. KCF is the most basic CF framework. Because of such a simple framework, KCF can run many times or even tens of times faster than other tracking algorithms. However, the periodic assumption of KCF produces the boundary effect, which limits the development of CF algorithms. BACF optimal the CF framework based on KCF. It does not entirely discard the background information and conduct a crop operation on each sample. The main contribution of BACF is that increase the number of training samples and improve the quality of training samples. STRCF is the latest of the three trackers. STRCF introduces temporal regularization into the CF framework, and this improves the robustness of the detection model.

## 2.1 Kernelized Correlation Filter

João F. Henriques proposed a kernelized correlation filter (KCF) [47] tracker in 2014, which is the original form of our proposed CF-based tracker [63, 65, 97, 98]. While KCF is not the first CF-based tracker, it is the most influential tracker in CF-based trackers because KCF makes a new research direction in object tracking. KCF is the baseline tracker of what most CF-based trackers are based on today. In prior, João F. Henriques has proposed CSK based on the grayscale feature. Compared to CSK [46], the main contribution of KCF is introducing multiple channel features (HOG features) [32, 73] to train the CF model. Compared to the single-channel feature (Grayscale feature), KCF obtains a significant improvement in performance by the



(a) (b)

Figure 2.1 (a) Base sample (b) Training samples

multiple channel feature. Notwithstanding multiple channels features [40] bringing the computation burden, the running speed of KCF is still faster than most trackers.

The task of most visual tracking algorithms is distinguishing between the object and the background [41, 45, 83]. These classifiers are trained with translated and scaled sample patches. The extreme challenges for these classifiers are the number of negative samples. If the classifier obtains many samples from an image, this operation takes a burden of computation and makes samples full of redundancy. Therefore, some classifiers select only a few samples from each frame. In contrast, KCF regards the tracking problem as a linear regression task. Based on object position from the previous frame, select the image patch to sample and train a linear regression model. This regression model can calculate the response of a small window sampling. The sample with the strongest response is taken as the new object position. KCF adopts an online method to update the regression model.

It is a great challenge for traditional trackers to produce as many samples as possible and keep the computational burden low. The success of KCF is that it solves this problem. As shown in Fig 2.1, KCF makes training samples by all possible cyclic shifts of a base sample (vertical and horizontal). From a 2D image point of view, all possible cyclic shifts of a base sample form a circulant matrix. This method generates a large number of positive samples that benefit from improving the appearance model's robustness. All circulant matrices are made diagonal by the Discrete Fourier Transform (DFT). The circulant matrix multiplication vector is equivalent to the inverse order of the generated vector and the vector convolution, which can be further transformed into the Fourier transform multiplication (the convolution itself includes the reverse order operation). The convolution of two signals in the time domain corresponds to the dot product of the Fourier transform of two signals in the frequency domain. This is the important reason for the high speed of KCF.

Since Ridge Regression admits a simple closed-form solution and can achieve outperformance, KCF focuses on it. The final goal of KCF is to learn a correlation filter model $\boldsymbol{f}$. In other words, find a function $f(X) = \boldsymbol{f}^T X$ ($X$ denotes the image patch, $X$ includes as $x_1 \dots x_i \dots x_K$, $x_i$ is the sample of each cyclic shift) that minimizes the squared error over samples $x_i$ and their regression targets $y_i$ as follows.

$$\min_{\boldsymbol{f}} \sum_i^K \|x_i * \boldsymbol{f} - y_i\|^2 + \lambda \|\boldsymbol{f}\|^2 \tag{2.1}$$

Here, $K$ is the number of cyclic shifts. $*$ denotes circular convolution. $\lambda$ is a regularization parameter.

As mentioned above, the minimizer has a closed-form solution, which is given by

$$\boldsymbol{f} = (X^T X + \lambda I)^{-1} X^T y \tag{2.2}$$

Where, $I$ denote an identity matrix. $y$ is the desired output, following a Gaussian function.

Although Equation 2.2 is neat, KCF is not satisfied and makes further optimal. The kernel trick [12, 20, 79] is used to make the non-linear regression function more robust. The astonishing factor is that the optimization progress is still linear. KCF adopts the kernel trick to optimize Equation 2.2, and the finalize the kernelized version of Ridge Regression is given by

$$\alpha = (K + \lambda I)^{-1} y \tag{2.3}$$

Where $K$ is the kernel matrix and $\alpha$ is the vector that represents the solution in the dual space.

At this point, KCF has optimized the formula to the simplest, which significantly reduces the calculation burden.

Since the object has some changes or is influenced by the background and some natural variation, KCF adopts online learning to update the feature and CF models. These can be express as

$$FM_{t+1} = (1 - \eta) * FM_{t-1} + \eta * FM_t \tag{2.4}$$

$$\alpha_{t+1} = (1 - \eta) * \alpha_{t-1} + \eta * \alpha_t \tag{2.5}$$

Where, $FM_t$ is the feature model in the *t-th* frame, $\alpha_t$ is the correlation filter model in the *t-th* frame. $\eta$ is the learning rate.

The other reason for the outperformance of KCF is that it extends from a single-channel feature (Grayscale feature) to a multiple-channel feature (Histogram of Oriented Gradients, HOG) [73] by simply summing over them in the Fourier domain. High dimensions feature will increase the computation. However, high dimensions features have excellent robustness for many complex scenarios.

Although KCF obtains outperformance in visual tracking, it still has some problems. KCF updates the feature and CF models online to mitigate the negative influence from occlusion or background clutter. However, when long-term occlusion [37] or deformation occurs, the online update method causes the model to not keep up with changes in the object's appearance. In addition, KCF ignores the influence of scale variation. As mentioned above, the periodic assumption of KCF produces the boundary effect, which limits the development of CF algorithms.

## 2.2 Background Aware Correlation Filter

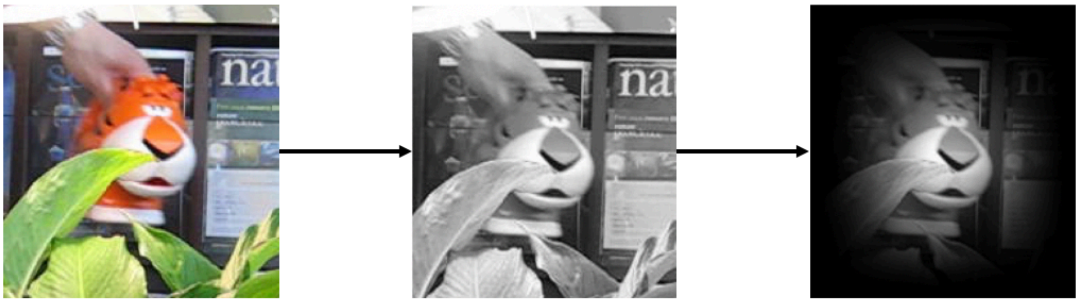


Figure 2.2 The impact of the window function

The detection and training operation of KCF is converted to the frequency domain by FFT for high-speed calculation. However, FFT will cyclically stitch the image signal, resulting in these signals not being continuous at the stitching place (it can also be considered that these signals are not real). This is the boundary effect. The boundary effect has a severe adverse effect on the tracking performance. To mitigate the boundary effect, KCF introduces a Gaussian window on the image patch.  As shown in Fig 2.2, the grayscale features (or other features) of the search region are extracted, and window functions are added to the grayscale features to keep only the central part of the image feature and smooth the surrounding background information. However, with the expansion of the search region, the impact of the window function also decreases. The search region of KCF is limited to 2.5 times of object size. If the search region is large, the boundary effect will destroy the performance of trackers. If the search region is small, it is

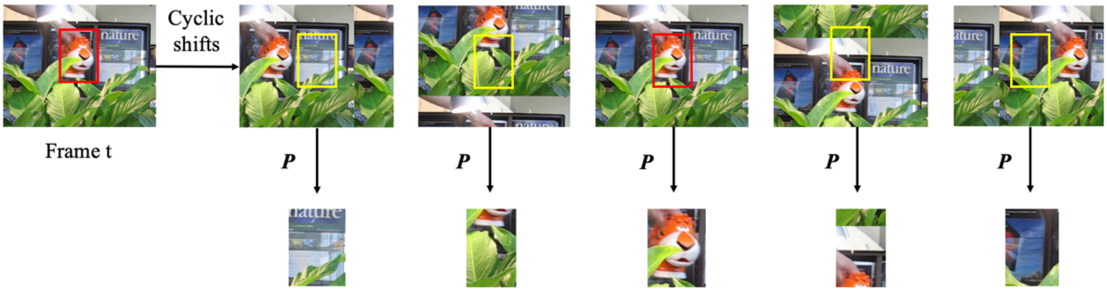


Figure 2.3 The crop operation of BACF

difficult for the trackers to detect the target object. Therefore, the performance of KCF is challenging to get significantly improved.

The windows function cannot solve the boundary effect. In subsequent research, two solved strategies became the mainstream in the field of tracking. BACF [38] is one of them. The main reason for the outstanding performance of BACF is to increase the number and quality of training samples. BACF improves the quality of the training samples by the crop operation. As shown in Fig 2.3, BACF obtains all possible positive and negative patches extracted from the search area, then uses a binary matrix P to intercept the central part of each training sample. Therefore, a high-quality training sample (positive sample contains little interference information) is obtained. A red box surrounds the positive sample, and a yellow box surrounds the negative sample. The positive sample contains the information of the target object and a small amount of information about the surroundings, and the negative sample contains the background information on the entire search area (the background information is directly discarded in the traditional DCF trackers). It is these high-quality samples that enable BACF to train highly robust models. Although BACF does not aim to solve the boundary effect, the crops operation does so indirectly. BACF can enlarge the size of the search region (5 times of object size) because of no boundary effect. A more extensive search region naturally leads to more training samples in training samples' quantity. For BACF, the thousands of samples used to train the correlation filter model have been increased to tens of thousands or even hundreds of thousands. The robustness of the trained correlation filter is undoubtedly much better than before because of lots of high-quality training samples. Because of the full use of the background information, the author named this method the background-aware correlation filter method.

To learns multi-channel background-aware correlation filters, BACF minimizes the following objective:

$$E(\boldsymbol{f}) = \frac{1}{2}\left\| y - \sum_{c=1}^{C} \boldsymbol{f}_c * (Px_c) \right\|^2 + \frac{\gamma}{2}\sum_{c=1}^{C} \|\boldsymbol{f}_c\|^2 \qquad (2.6)$$

Here $y$ is the Gaussian-distributed ground-truth. $P$ denotes a binary matrix ($T \times T$) which crops the mid of $x_c$ ($T$ is the length of $x$). The channel number $C$ and the spatial correlation operator $*$. $\gamma$ denotes the regularization parameter. BACF introduces the fast Fourier transform to cast the formula into the frequency domain and proposes an efficient Alternating Direction Method of Multipliers (ADMM) [85] optimization algorithm to convert the original problem into two sub-problems. The sub-problems can be closed-form solutions. Through intelligent optimization and simplification techniques, BACF's performance and speed have reached an astonishing level.

## 2.3 Spatial-Temporal Regularized Correlation Filter

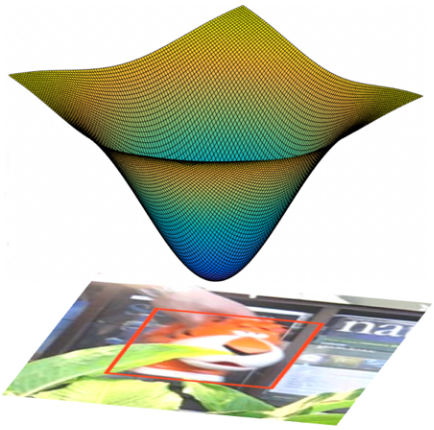


Figure 2.4 The visualization of spatial regularization

In addition to BACF, another algorithm is SRDCF [66] which was proposed to reduce the negative influence from the boundary effect. BACF introduces a binary matrix on the training samples. SRDCF adopts a spatial regularization that adds a regular coefficient matrix (w) to the correlation filter. The shape of w is shown in Fig 2.4. The purpose of spatial regularization is obvious. The center coefficient is lower at the target and the surrounding coefficient is higher at the background. The obtained correlation filter can pay more attention to the target object information, and the filter response results in the background are as low as possible. Spatial regularization can effectively suppress the response of the background area so that the search area can be enlarged, and better performance can be obtained in scenes such as the complex background. Since SRDCF adopts the Gauss-Seidel method to optimize the equations, SRDCF has lost its early real-time tracking capabilities based on DCF trackers. STRCF [31] adopts the same methods as SRDCF to solve the boundary effects. However, STRCF uses the alternating direction method of multipliers (ADMM) [85] to efficiently solve the introduction of spatial regularization.

As mentioned above, most of the traditional DCF trackers adopt the online method to update the feature model and the correlation filter model. The efficiency of the correlation filter relies on the robustness of the feature model. The fixed learning rate cannot meet the requirements of the change of scene. For example, occlusion or deformation can reduce the robustness of the feature model, so that no effective correlation filter. STRCF introduces temporal regularization into the correlation filtering framework to deal with special scenes. The detail of the framework is expressed as follow,

$$\underset{\boldsymbol{f}}{min}\frac{1}{2}\left\|\sum_{d=1}^{D} x_t^d * \boldsymbol{f}^d - y\right\|^2 + \frac{1}{2}\sum_{d=1}^{D}\left\|w \cdot \boldsymbol{f}^d\right\|^2 + \frac{\mu}{2}\|\boldsymbol{f} - \boldsymbol{f}_{t-1}\|^2 \qquad (2.7)$$

Here $\mu$ is a regularization parameter, $d$ is the channel of the features. $w$ is the spatial regularization weight, $\boldsymbol{f}_{t-1}$ denotes the correlation filter obtained in the *t-1-th* frame.

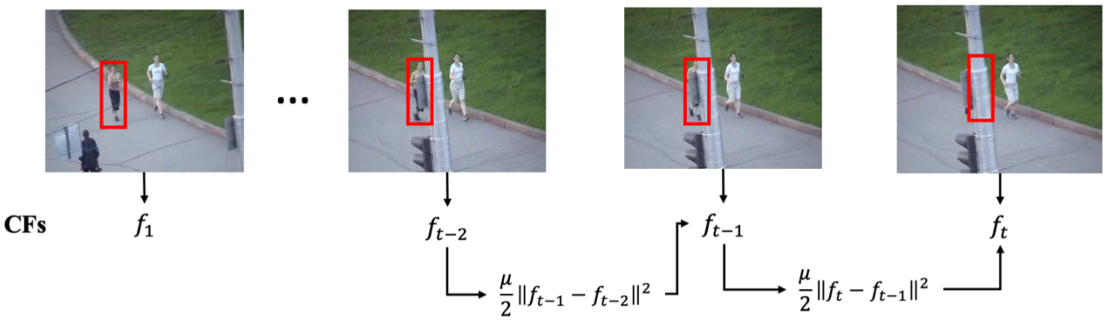

CFs
$f_1$
…
$f_{t-2}$
$f_{t-1}$
$f_t$
$\frac{\mu}{2}\|f_{t-1} - f_{t-2}\|^2$
$\frac{\mu}{2}\|f_t - f_{t-1}\|^2$

Figure 2.5 The process of temporal regularization

Fig 2.5 shows the process of temporal regularization. When training the correlation filter model $\boldsymbol{f}_t$ in the current frame, the correlation filter model $\boldsymbol{f}_{t-1}$ in the previous frame is needed. The role of temporal regularization is the same as the online update. The goal of DCF trackers is that to train an efficient correlation filter model. The traditional DCF trackers update the feature model and the correlation filter model with an online method, preventing the corruption of the feature model and the correlation filter. To mitigate the negative influence from each frame, the fixed learning rate is usually set to relatively small. Temporal regularization of STRCF also fulfills this aim and achieves better performance than the online method. The meaning of temporal regularization is that it ensures the obtained correlation filter tends to the filter in the previous frame. Therefore, STRCF can successfully track the target in the presence of occlusion and simultaneously well adapt to larger appearance changes. The parameter of temporal regularization is also fixed. If adaptively adjusts the parameter with the state of the object, STRCF can get better performance.

# Chapter 3

# Correlation Filter-Based Visual Tracking using Confidence Map and Adaptive Model

In this chapter, I proposed Correlation Filter tracker using Confidence map and Adaptive model (CFCA) based on the classic CF framework. As mentioned in section 2.1, KCF balances the outstanding performance and calculation requirement low. I introduce an improved scale pool method into KCF because KCF ignores the impact of scale variation. Since CF trackers [46, 47] locate the object's position by the maximum score of the confidence map, in general, this detection method is robust for common scenarios. However, when occlusion or deformation occurs, the confidence map fluctuates wildly. I combine the four high scores of the confidence map with the luminance histogram similarity to improve detection accuracy. Finally, judge whether to re-location by the object state. The performance of the tracker is excellent in out of view, scale variation, and occlusion.

## 3.1 Adaptive Scale Pool

Since KCF adopts a single scale in tracking, when the target object has deformation or occluded, the precision of the tracking will decrease. Scale pool is a widely accepted scale estimation method for CF trackers. In this section, we briefly review the principle of scale pool [63] and propose the adaptive scale pool.

### 3.1.1 The principle of Scale Pool

KCF does not adopt methods to deal with scale variation. In other words, a single scale is used to track the object from the initial frame to the end frame. The confidence map is obtained by calculating with the current search patch and trained CF model, and the maximum score of the confidence map corresponds to the new location. It is a one-pass process. If the object's size does not change in tracking, KCF can obtain significant performance. Otherwise, the model of

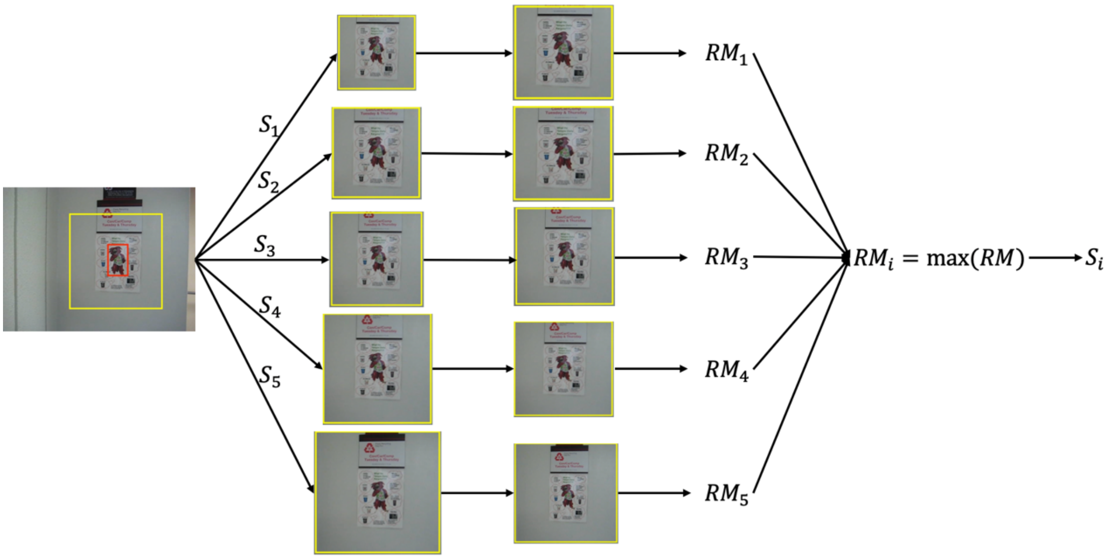


Figure 3.1 The process of scale pool.

KCF will learn too much background information or too little target information, which leads to the robustness of the CF model decreasing. It is effective for CF trackers to adopt scale pool technology to deal with scale variation.

As shown in Fig 3.1, tracking is not a one-pass process because of the scale pool. Suppose the scale pool has five scales. In the current frame, multiple scales $S = \{S_1, S_2, S_3, S_4, S_5\}$ are applied on the search region based on the previous position (target: red box, search region: yellow box). For the convenience of calculation, the multiple search regions will be resized to the same scale. The five response maps are obtained by using the previous CF model. Each response map has a maximum score $RM_i$, and the maximum response value corresponds to the object's new location. The target scale $S_i$ corresponds to the response map where the maximum score $RM_i$ is located. However, the number of the tracking process depends on the number of scales. The number of scales directly increases the burden of calculation.

### 3.1.2 The improved Scale Pool

With scale pool technology, CF trackers can locate the object's location meanwhile estimating the scale variation. Most CF trackers [31, 63, 66] adopt scale pool technology to solve scale variation because of the high precision tracking. However, some CF trackers have a boundary effect, so the search region is limited to 2.5 times object size. When a small object is tracked, the scale pool becomes valueless because cropped patches have the same scale. Since the scale gap of scale pool $S_{gap}$ is set to 0.01, the difference in the patch size is less than 1.0, which becomes zero in quantizing it to an integer. For the small object, an improved scale pool is proposed.

The number of scales is assumed to be five in Fig 3.1, and our method adopts three scales $S = \{S_1, S_2, S_3\}$ to estimate scale variation. On the assumption that the search patch $M_T = (H, W)$ (Here, $H$ and $W$ are the height and width of the image patch), the size of the obtained image is patches $M_{T1}$, $M_{T2}$, $M_{T3}$. To make significance of the scale pool for the small object, the cropped patches should have a different scale ($M_{T1} \neq M_{T2} \neq M_{T3}$). We need to resize the search area manually to ensure the difference of the obtained image region. If the minimum of $H \times S_{gap}$ and $W \times S_{gap}$ is less than 1.0, it is set to 1.0. For the other one, it changes with proportional ($Ratio$). The resize parameter $Ratio$ can be expressed as follows

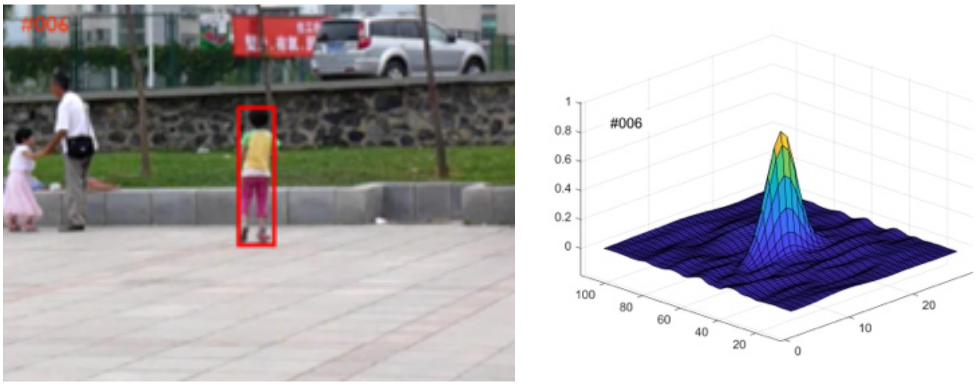
$$Ratio = \frac{\max(H,W)}{\min(H,W)} \tag{3.1}$$

The adaptive scale estimation can be express as follows

$$M_{Ti} = \begin{cases} H_i \pm 1, W_i \pm Ratio & if\ H = \min(H,\ W) \\ H_i \pm Ratio, W_i \pm 1 & if\ W = \min(H,\ W) \end{cases} \tag{3.2}$$

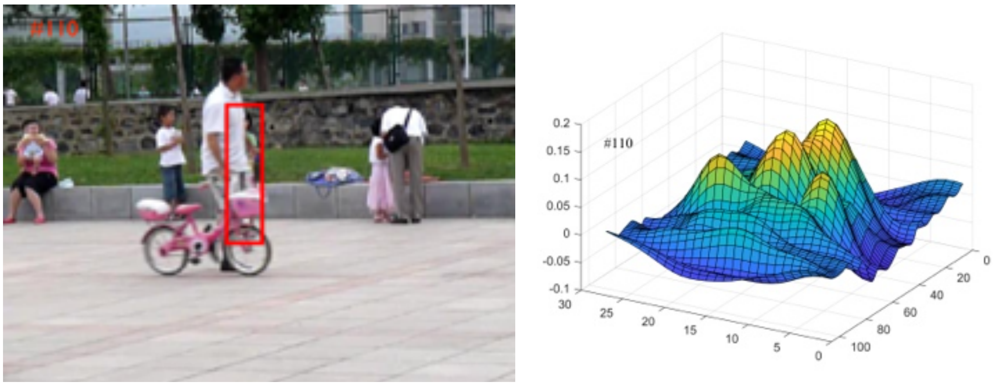
As shown in Equation 3.2, to estimate the scale of the small object accurately, the minimum of $(H, W)$ is need to be judged. If $H$ is less than $W$, the search patch sizes as {*H*-1.0, *W*-*Ratio*}, {*H*, *W*}, and {*H*+1.0, *W*+*Ratio*} are applied. If $W$ is less than $H$, the search patch sizes as {*H*-*Ratio*, *W*-*1.0*}, {*H*, *W*}, and {*H*+*Ratio*, *W*+*1*} are applied.

## 3.2 Utilization of Confidence Map



(a)

(b)



(c) (d)

Figure 3.2 (a)(b) The normal object tracking and a corresponding confidence map. (c)(d) The abnormal object tracking and a corresponding confidence map.

Fig 3.2 shows the standard tracking and corresponding confidence map and the abnormal tracking and corresponding confidence map. In Fig 3.2 (a), the target object is surrounded by a red box accurately. The corresponding confidence map (Fig 3.2 (b)) is smooth and has only one vertex. In contrast, when the target object is occluded, it is inaccurate for detection (Fig 3.2 (c)). The confidence map drastically fluctuates when there are multiple vertexes (Fig 3.2 (d)). Therefore, the fluctuation of the confidence map reflects the reliability of the tracking. The greater the confidence map fluctuation, the more unreliable the tracking performance.

When the confidence map drastically fluctuates, it is inaccurate for CF trackers to locate the object's location by the maximum score of the confidence map. In addition, the CF tracker adopts the HOG feature to train the model, a stride of the training and detection samples being greater than one pixel because the cell size of the HOG feature is set to 4. This leads to a decrease in the precision of the detection. We propose a novel method to increase the detection accuracy, combining the four high scores of the confidence map and the luminance histogram similarity.

Firstly, the HOG feature is a significant textural feature. It plays a vital role in the field of the image process. Most CF trackers obtain outstanding performance by using the HOG feature. However, the HOG feature has poor robustness in non-rigid object deformation. The confidence map of the CF tracker fluctuates dramatically for the non-rigid object. The luminance histogram describes the object's gray-level distribution, which can keep robust for the non-rigid object. Therefore, the confidence map and the luminance histogram are combined to locate the object accurately. The cosine similarity $LHS$ of the luminance histogram can be expressed as follows:

$$LHS = \frac{\sum_{i=1}^{n}(T_i \times N_i)}{\sqrt{\sum_{i=1}^{n}(T_i)^2} \times \sqrt{\sum_{i=1}^{n}(N_i)^2}} \tag{3.3}$$

where $T_i$ is the $i-th$ element in the luminance histogram vector of the initial object, and $N_i$ is *the i-th* element in the luminance histogram vector of the current object in the image patch.

As mentioned above, it is inaccurate for the CF trackers to locate the object's position by the highest score of the confidence map. We select the highest four scores $\{R_i | i \in \{1, 2, 3, 4\}\}$ from the confidence map. The four scores' corresponding location may be four adjacent values or four vertices. And then evaluate the value of the products of $R_i$ and the luminance histogram similarity $LHS_i$ around the four positions $i \in \{1, 2, 3, 4\}$ as $Y = \{(R_i \times LHS_i) | i \in \{1, 2, 3, 4\}\}$. The location of the object is estimated as position $i$, which takes the maximum value in $Y$. The proposed method increases the precision of the detection.

When the object's state is judged to drift, we adopt the relocation method to search the object's position. As shown in Fig 3.2 (c, d), there are multiple peaks in the confidence map. When the number of vertices increases, any of these vertices could be where the target is. Based on these vertices' locations, multiple research regions are cropped to relocate the object's location. Since the selection of the vertices will produce a computation burden, we only consider the confidence scores, which are higher than half of the largest score in the original confidence map.

## 3.3 Adaptive model update

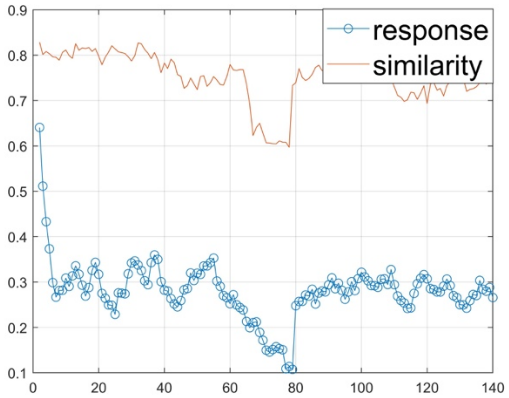


Figure 3.3 The maximum score and luminance-histogram similarity at each frame for the video "*jogging*".

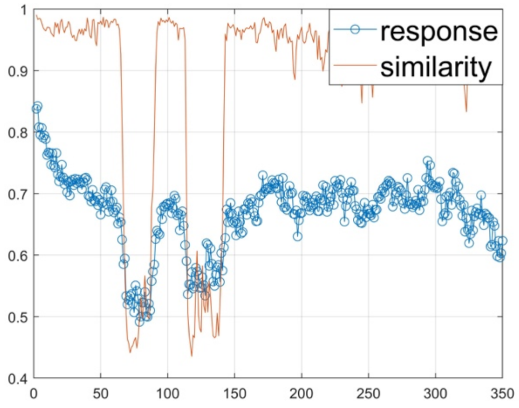


Figure 3.4 The maximum score and luminance-histogram similarity at each frame for the video "*car2*".

Most CF trackers update the feature model and the detection model online. Since the CF trackers aim to locate the target object accurately, trackers need to maintain the robustness of the detection model. In the process of tracking, the scenario of the object constantly changes. The fixed learning rate causes the model to learn too much error information in the current frame or not learn the change information of the target object. The novel method is proposed to increase the robustness of the feature model and the CF model by adaptive adjusting the learning rate.

The HOG feature's CF trackers are susceptible to an object's state change, such as occlusion, rotation, and deformation. The maximum score of the confidence map reflects the object's state to some extent. When abnormal tracking occurs, the confidence map's maximum score will change dramatically. As shown in Fig 3.3, the initial response score is the highest because of the most reliable model. However, with the change of the CF model, the tracking performance decreases until an equilibrium score is reached. The highest score decreases drastically around the 0-10'*th* frame in image sequence "*jogging*" because of deformation. When the CF model learns enough deformation information, the maximum score changes gently. However, in the same case, since the luminance histogram keeps robust to deformation, the similarity of the luminance histogram keeps steady. Combining the maximum score with the luminance histogram similarity increases the accuracy of the judgment about the object's state.

In our tracker, we divide the target object into four states. The highest score of the confidence map decreases dramatically, and the luminance histogram similarity keeps high. The object's state is to be recognized as deformation. In this scenario, we increase the learning rate to meet the requirement of the appearance change of the object. In Fig 3.3, both the maximum score of the confidence map and the luminance histogram similarity drop in the 65-80'th frame in "*jogging*", the object is judged to be drifted. The learning rate is adjusted to zero, preventing the feature model and the CF model contamination by the error information. Since the luminance-histogram cannot keep robust on illumination various, the luminance-histogram similarity decreases drastically in the 110-140'$th$ frames in image sequence "*car2*". At the same time, the highest score also becomes smaller. The learning rate of the model is reduced to mitigate the negative influence of illumination variation. Combining Fig 3.3 and Fig 3.4, the luminance histogram similarity of the object is greater than 0.7, the maximum score varies a little, the object's state is judged to be normal.

We are combining the maximum score and the luminance-histogram similarity to judge the object's state. Here, a judgment coefficient σ is introduced to adjust the learning rate, and the formulation can be expressed as follows.

$$\sigma = \begin{cases} 0 & \Delta R > a, \quad LHS < b \\ x & \Delta R \leq a, \quad LHS < b \\ 1 & \Delta R \leq a, \quad LHS \geq b \\ y & \Delta R > a, \quad LHS \geq b \end{cases} \tag{3.4}$$

Here, $\Delta R$ is the change of the maximum score in adjacent frame and $LHS$ is the value of the luminance histogram similarity. The parameters $a$, $b$, $x$ and $y$ are controlled to update the model adaptively, where $x$ takes a value from 0 to 1 and y greater than 1.

When $\sigma = 0$, the object is supposed to be drifted, the object's location should be relocated. The judgment coefficient $\sigma$ is applied to the feature model and the CF model update as follows

$$X_t = (1 - \sigma\eta)X_{t-1} + \sigma\eta X \tag{3.5}$$

$$w_t = (1 - \sigma\eta)w_{t-1} + \sigma\eta w \tag{3.6}$$

## 3.4 Experiments

In this section, the analysis experiments of the proposed tracker are conducted. Firstly, I introduce the evaluation database and analysis metric. Next, the comparison of nine CF trackers (KCF [47], CSK [46], DSST [65], SAMF [63], SRDCF [66], CFHA [98], KCFAMSR [97], Staple [57], CN [70]) to analyze the strength and weaknesses of the tracker.

### 3.4.1 OTB Benchmark Database

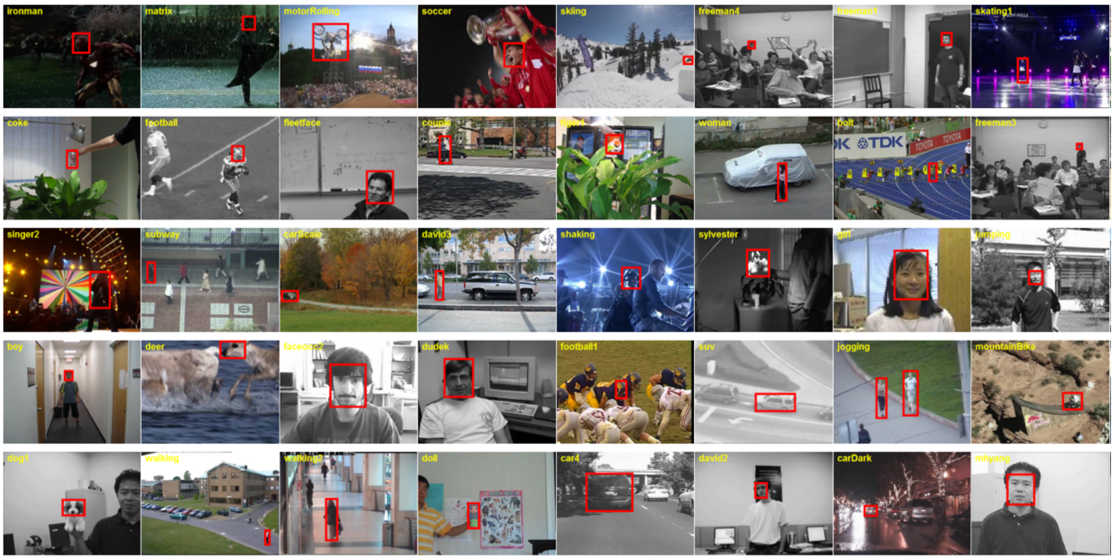


Figure 3.5 The partial image sequences of the OTB benchmark database.

We conduct the evaluation experiments on the OTB-2013 and OTB-2015 benchmark databases to evaluate the performance of the proposed CFCA tracker. OTB benchmark dataset [93, 94] is a widely recognized visual tracking database of a single target. As shown in Fig 3.5, it contains color image sequences and grayscale image sequences. OTB database can be divided into two parts as OTB-2013 and OTB-2015. OTB-2013 contains 51 annotated challenging image sequences, and OTB-2015 is the extended version of OTB-2013, which includes 100 annotated image sequences. There are no standard visual tracking benchmark databases before the OTB benchmark database. To better evaluate tracking algorithms and analyze the trackers' strengths and weaknesses, the OTB benchmark database categorizes the image sequences by annotating with the 11 attributes. An image sequence can have multiple attributes. The 11 attributes are shown as follows,

SV (Scale Variation): The ratio of the bounding boxes of the first frame and the current frame is out of the range.

MB (Motion Blur): The target region is blurred because of the motion of the target or camera.

OCC (Occlusion): The target object is partially or fully occluded.

IV (Illumination Variation): The illumination in the target region is significantly changed.

DEF (Deformation): The deformation of a non-rigid object.

FM (Fast Motion): The motion of the ground truth is larger than the threshold (20 pixels).

IPR (In-Plane Rotation): The target object rotates in the image plane.

OPR (Out-of-Plane Rotation): The target object rotates out of the image plane.

OV (Out-of-View): Some portion of the target object leaves the view.

BC (Background Clutters): The background near the target has a similar color or texture.
LR (Low Resolution): The number of pixels inside the ground-truth bounding box is less than a threshold (threshold=400).

OPE (One-Pass Evaluation) is the conventional method that evaluates the tracking algorithms. Initialize the first frame with the target's position in the ground-truth, and then run the tracking algorithm to get the average accuracy and success rate. The OTB benchmark database adopts the same method to analyze the trackers. Besides, we also analyze the running speed of the trackers to verify their feasibility of the trackers.

The meanings of the three-evaluation metrics are introduced as follows.

**Precision Plot**: The center point of the object position (bounding box) is estimated by the tracking algorithm and the center point of the artificially labeled (ground-truth) target, the distance between the two is less than the percentage of the video frame of the given threshold (Generally, the threshold is set to 20 pixels). Different thresholds result in different percentages, so a curve can be obtained. The disadvantage of this evaluation method is that it cannot reflect the changes in the size and scale of the target object.

**Success Plot**: First, define the overlap score (*OS*) , the bounding box obtained by the tracking algorithm (denoted as a), and the bounding box given by ground-truth (denoted as b), the overlap ratio is defined as: $OS = \frac{|a \cap b|}{|a \cup b|}$. When the *OS* of a frame is greater than the set threshold, the result of the frame is regarded as successful, and the percentage of the total successful frames in all frames is the success rate. The threshold value range of *OS* is 0~1(Generally, the threshold is set to 0.5), so a curve can be drawn.

**FPS (Frame Per Second)**: The tracking results of how many pictures are given by the tracking algorithm per second. In general, FPS>=25 means real-time performance. The higher the fps, the higher the efficiency.

Some trackers only pursue success rate [67], ignoring FPS. An excellent algorithm can keep the balance between success rate and real-time.

### 3.4.2 Parameter Setting

The parameter adjustment is essential for the algorithms. In this section, I introduce the tracker's parameters and analyze the influence of the parameters.

Firstly, the regular parameter of the tracker is introduced. All the evaluation experiments are conducted on the Matlab-R2020a platform and a PC machine with an Intel (R) Core (TM) i7-9700F CPU (3.00GHZ), 16GB memory. In traditional CF trackers, since an image sequence has many frames, the learning rate setting is small, the learning rate η is set to 0.015 in CFCA. The cell size of the HOG feature is set to 4. The search region is set to 2.5 times the target object's size because of the boundary effect. The scale gap $S_{gap}$ is set to 0.05.

Table 3.1 Success of CFCA with non-adaptive model update on the OTB-2013 with different $S_{gap}$.

| $S_{gap}$ | 0.025 | 0.05 | 0.075 | 0.1 |
|---|---|---|---|---|
| Success | 0.606 | 0.619 | 0.586 | 0.586 |

Table 3.2 Success of CFCA on the OTB-2013 with different a, b, x, y.

| a | 0.05 | | | 0.10 | | | 0.15 | | |
|---|---|---|---|---|---|---|---|---|---|
| b | 0.5 | 0.7 | 0.9 | 0.5 | 0.7 | 0.9 | 0.5 | 0.7 | 0.9 |
| x=0.25<br>y=1 | 0.60 | 0.60 | 0.58 | 0.61 | 0.59 | 0.60 | 0.61 | 0.59 | 0.59 |
| x=0.25<br>y=2 | 0.59 | 0.59 | 0.61 | 0.60 | 0.59 | 0.61 | 0.60 | 0.60 | 0.60 |
| x=0.25<br>y=3 | 0.58 | 0.58 | 0.58 | 0.59 | 0.59 | 0.60 | 0.60 | 0.59 | 0.60 |
| x=0.5<br>y=1 | 0.58 | 0.60 | 0.58 | 0.61 | 0.61 | 0.59 | 0.61 | 0.57 | 0.59 |
| x=0.5<br>y=2 | 0.59 | 0.59 | 0.58 | 0.60 | 0.63 | 0.60 | 0.59 | 0.57 | 0.59 |
| x=0.5<br>y=3 | 0.59 | 0.58 | 0.59 | 0.60 | 0.61 | 0.59 | 0.59 | 0.57 | 0.59 |
| x=0.75<br>y=1 | 0.61 | 0.61 | 0.59 | 0.62 | 0.61 | 0.62 | 0.62 | 0.61 | 0.61 |
| x=0.75<br>y=2 | 0.59 | 0.60 | 0.59 | 0.60 | 0.61 | 0.61 | 0.60 | 0.61 | 0.62 |
| x=0.75<br>y=3 | 0.59 | 0.59 | 0.59 | 0.62 | 0.60 | 0.62 | 0.61 | 0.60 | 0.62 |

Table 3.3 Success of CFCA with adaptive model update on the OTB-2013 with different $S_{gap}$.

| $S_{gap}$ | 0.025 | 0.05 | 0.075 | 0.1 |
|---|---|---|---|---|
| Success | 0.565 | 0.631 | 0.595 | 0.578 |

Secondly, we conduct the setting experiments of the parameters. From the result of the experiment, we select the most appropriate parameter. As mentioned in section 3.1, scale pool is used to estimate scale variation in CFCA. The scaling step is the main impact factor in scale pool. Generally, the scale step is set to 0.01 in the CF trackers. Since CFCA estimates scale in every two frames instead of each frame, the scale step is set to 0.05 in CFCA. Table 3.1 and Table 3.3 show the experiments that success of CFCA without an adaptive model update on the OTB-2013 with different $S_{gap}$, the success of CFCA with an adaptive model update on the OTB-2013 with different $S_{gap}$, respectively. The experiment results are obtained by adjusting δ as 0.025, 0.05, 0.075, and 0.1. The best result is shown in Table 3.1 and Table 3.3. When $S_{gap}$ is set to 0.05, CFCA obtains the best performance.

CFCA adaptively adjusts the learning rate with the object's state. Equation 3.4 shows the parameters a, b, x, and y. CFCA conducts the valuation experiments on OTB2013 to determine the parameters. As shown in Fig 3.3 and Fig 3.4, b is set around 0.7, because the threshold of the luminance histogram similarity is about 0.7, the value range of b is 0.5, 0.7, 0.9. The value a is also considered to be around 0.1, by analyzing the data in Fig 3.3, the value range of a is 0.05, 0.10, 0.15. Since the value x is between 0 and one, x is set to 0.25, 0.5, and 0.75 in this experiment. The value y should be greater than or equal to 1, so that y is set as y ≥1, the value range of y is 1, 2, 3. Table 3.2 shows the success score of CFCA on the OTB-2013 with different a, b, x, y. The best success scores are shown on red font. From the experiment results, when a=0.10, b=0.7, x=0.50, y=2, CFCA obtains the highest success score.

### 3.4.3 Analysis of OTB

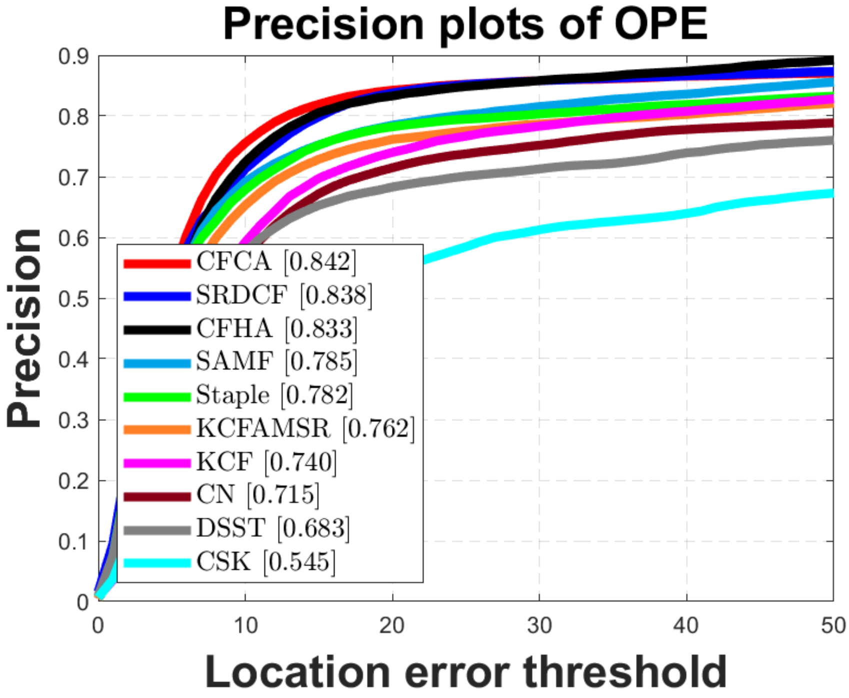


Figure 3.6 The precision plot of different CF trackers on OTB-2013.

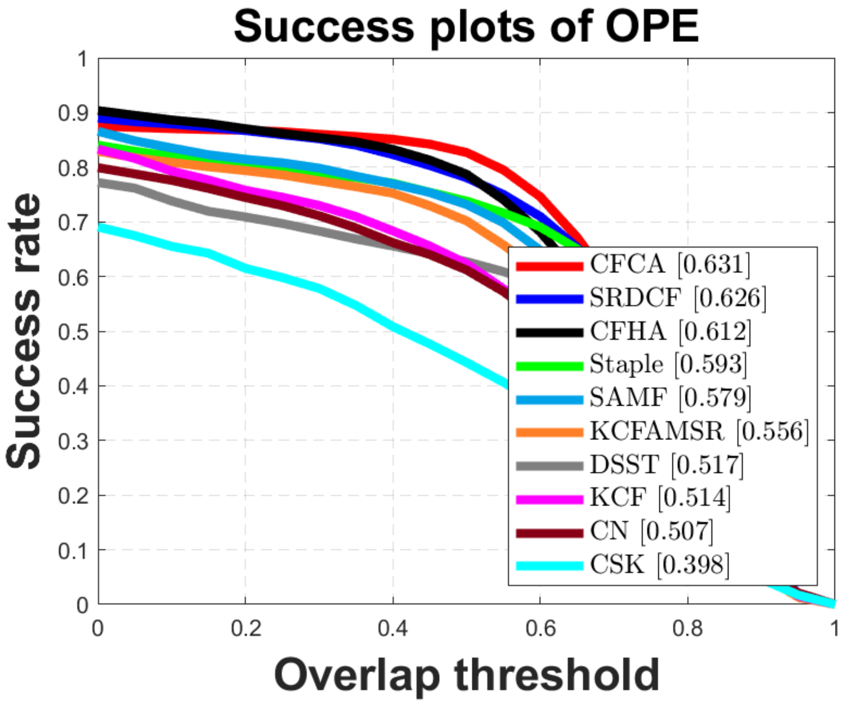


Figure 3.7 The success plot of different CF trackers on OTB-2013.

The evaluation experiments are performed on the OTB-2013 benchmark dataset. As shown in Fig 3.6 and Fig 3.7, CFCA achieves a precision score of 84.2% and a success score of 63.1%, respectively. Compared to the baseline tracker KCF, CFCA improves 14% and 23% on precision and success scores. The result of the experiment shows the validity of CFCA in tracking. SRDCF aims to mitigate the negative influence from the boundary effect and enlarge the search region to locate the target object. Since CFCA adopts reasonable methods to improve the performance in tracking, CFCA still obtains outperformance than SRDCF. SRDCF and SAMF adopt the scale pool to solve the scale variation and use the more robust feature to train the CF model. However, these trackers do not consider the state change of the object and the adaptive update model. Therefore, the performance of CFCA is better than these CF trackers. Moreover, the performances of CN and CSK are lower because they use the common robust feature to train the CF model and do not consider scale change, as well as a state change.

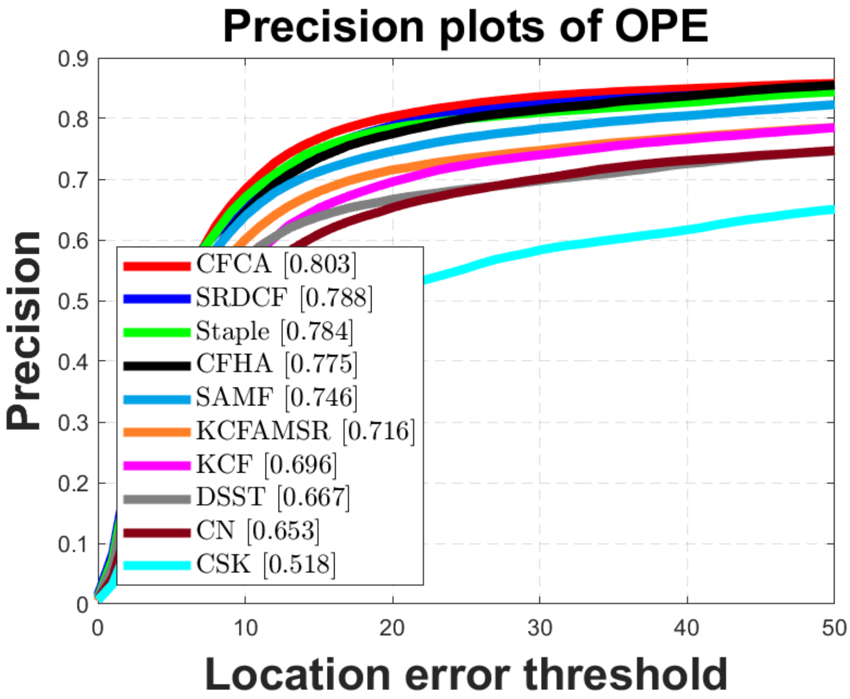


Figure 3.8 The precision plot of different CF trackers on OTB-2015.

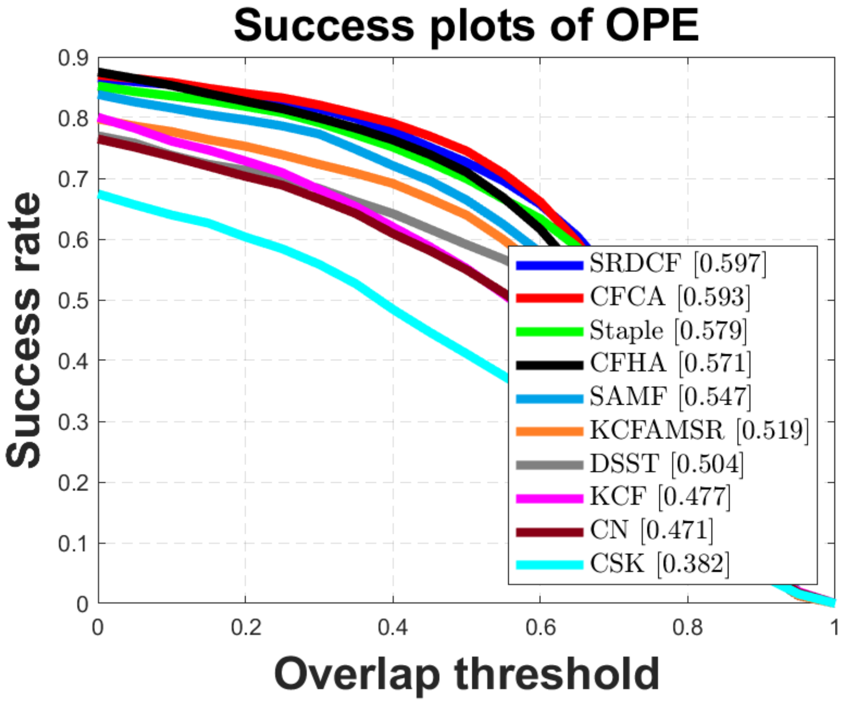


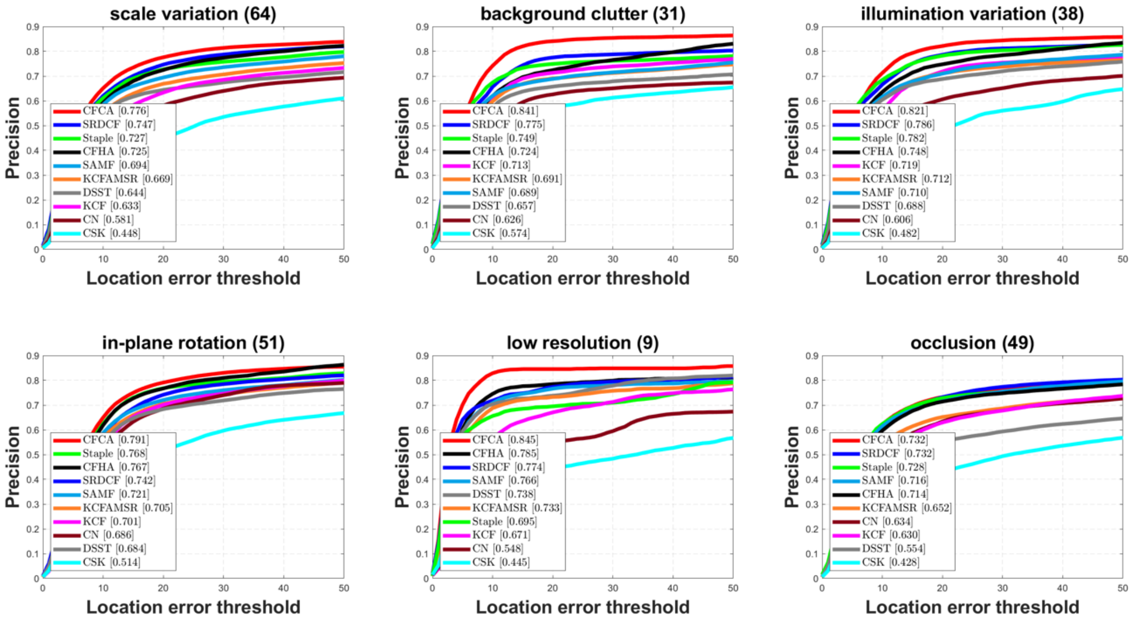
Figure 3.9 The success plot of different CF trackers on OTB-2015.

Fig 3.8 and Fig 3.9 show the tracking performance of CFCA on the OTB-2015 benchmark database. OTB-2015 extends OTB-2013 to have more image sequences. Since the parameters of CFCA are trained on OTB-2013, the experiment result of OTB-2015 has more persuasive. In OTB-2015, CFCA gets precision scores of 80.3% and success scores of 59.3%, respectively. In contrast to OTB2013, the success score of CFCA is slightly lower than SRDCF. Besides, CFCA achieves a gain of 10.7% and 11.6% on precision and success score than baseline KCF. Compared with other algorithms, CFCA obtains different degrees of improvement. The performance of CFCA on the OTB-2015 shows that the proposed method in CFCA can validity improve the performance of KCF in tracking.

Table 3.4 Precision and Success and speed of top-5 trackers on the OTB-2015. The best two results are shown in red and blue fonts, respectively.

| | SRDCF | Staple | CFHA | SAMF | CFCA |
|---|---|---|---|---|---|
| Precision | **0.788** | 0.784 | 0.776 | 0.746 | **0.803** |
| Success | **0.596** | 0.579 | 0.571 | 0.547 | **0.593** |
| FPS | 10.4 | 105.5 | **115.1** | 17.9 | **114.1** |

Table 3.4 shows the precision and success score and the running speed of the top-5 trackers on the OTB-2015 benchmark database. The best two results are shown in red and blue fonts, respectively. CFCA obtains the highest precision score, 80.3%. Although SRDCF achieves a better performance than CFCA in success score, the running speed of CFCA is 11 times faster than SRDCF. It is difficult for SRDCF to achieve real-time because of the low running speed. However, CFCA achieves the balance between performance and real-time.



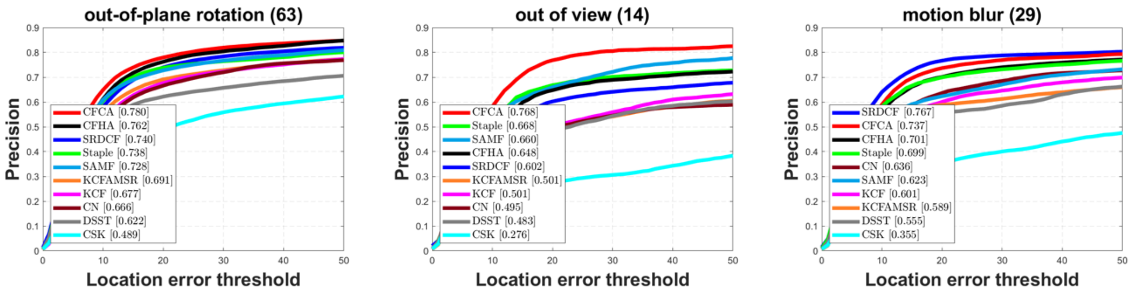


Figure 3.10 The precision plots are shown for nine attributes on the OTB-2015.

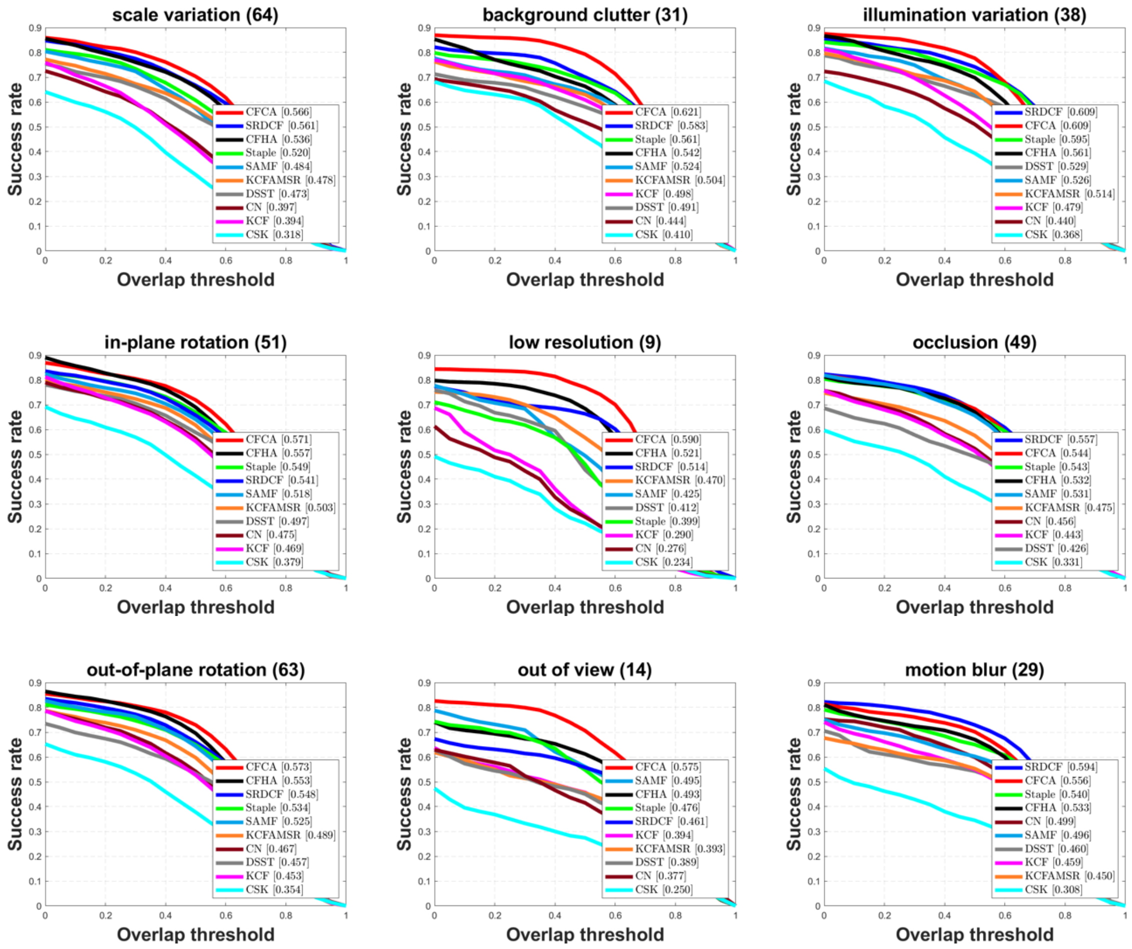
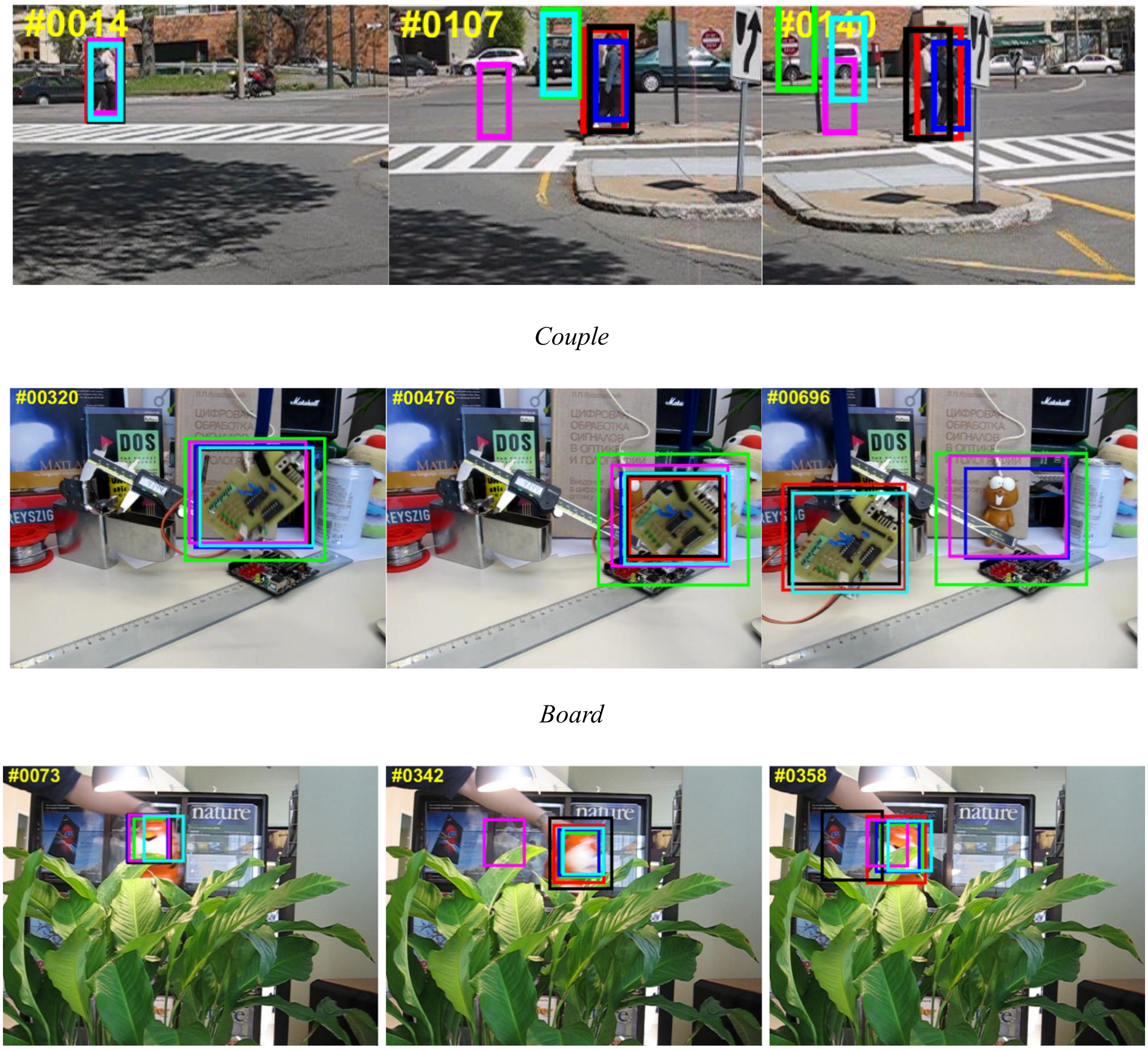


Figure 3.11 The success plots are shown for nine attributes on the OTB-2015.

Fig 3.10 and Fig 3.11 show the performance of CFCA on nine attributes on the OTB2015 benchmark database. Since CFCA accurately estimates the scale variation of a small object by improving the scale pool, CFCA gets the outperformance on the SV attribute, 77.6% and 56.6% on the precision score and the success score. In the LR attribute, CFCA obtains a precision score of 84.5% and a success score of 59.0%, respectively. Compared to KCF, it obtains a gain of 27.4% and 30.0% on precision and success, respectively. Moreover, CFCA obtains a significant performance on IPR and



*Couple*

*Board*

*Tiger1*

Figure 3.12 Qualitative evaluation on the three image sequences (*Couple*, *Board* and *Tiger1*).

OPR attributes because of the response map and luminance-histogram combination. The re-location method is used to improve the OCC and OV attributes. The best performance on other attributes shows the validity of CFCA.

We conduct a qualitative evaluation of different trackers on the three image sequences (*Couple*, *Board* and *Tiger1*). CFCA, SAMF, Staple, SRDCF, KCF and CFHA with different colors are showed, respectively.

In the initial frame, six trackers all keep significant performance in tracking. However, some trackers make some errors in the remaining sequences. For example, SAMF, Staple, and KCF lost the target object in *Couple*. In *Board,* Staple cannot accurately estimate the scale variation of the object and has drifted. CFCA achieves robust performance on the different image sequences.

# Chapter 4

# Visual Tracking via Adaptive Spatial-Temporal Regularized Correlation Filters

In chapter 3, the object's state estimation is used to adjust the learning rate of the model update. Based on the discriminative correlation filter, chapter 4 introduces advanced state estimates to increase the scale estimation and location accuracy. Correlation Filter tracker using a spatial-temporal regularized with Advanced Scale Estimation (CFASE) is proposed to achieve more significant tracking performance.

In terms of the DCF model, we propose a new method to estimate correlation filters more precisely using predictions from the previous two filters, considering the drift during the tracking process. Besides, we train two correlation filters models to obtain scale estimation and object location, respectively. The separated two correlation filter models help to reduce the adverse effects of scale changes on object location. Finally, our tracker introduces average peak-to-correlation energy (APCE) [72] to evaluate the accuracy of scale estimation and object location. The effectiveness of CFASE is verified with the experiment on different benchmark databases [75, 93, 94].

## 4.1 Adaptive Spatial-Temporal regularized

As mentioned in section 2.3, STRCF [31] mitigates the negative influence from the boundary effect by introducing spatial regularization. Moreover, STRCF introduces temporal regularization instead of the online update model method. STRCF adopts the same method as SRDCF to reduce the boundary effect, introducing a spatial regularization weight function to penalize the magnitude of the correlation filter coefficients $w$ in learning. The value of the weight depends on the spatial locations. The closer to the center, the lower the coefficient, and the closer to the surrounding, the higher the coefficient. However, STRCF uses alternating direction multipliers (ADMM) to optimize formulation instead of the iterative Gauss-Seidel method efficiently. This strategy ensures the real-time of STRCF. Furthermore, STRCF introduces temporal regularization to discard the online update model. The online update model method is over-reliant on the fixed learning rate and simultaneously has to learn both the feature model and the correlation filter model. These strategies limit the performance of tracking algorithms. The advantage of temporal regularization is that it uses the target object feature model of the current frame to train the correlation filter model. There is no need to consider too much about the adverse effects of the error feature model on training.

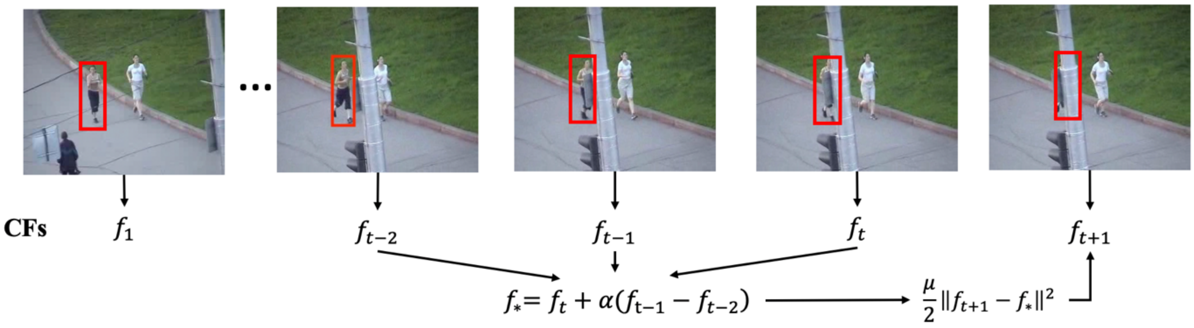

CFs
$f_1$
$f_{t-2}$
$f_{t-1}$
$f_t$
$f_{t+1}$
$f_*= f_t + \alpha(f_{t-1} - f_{t-2})$
$\frac{\mu}{2}\|f_{t+1} - f_*\|^2$

Figure 4.1 The process of CFASE's model update.

As mentioned above, temporal regularization makes sure that the obtained $\boldsymbol{f}$ is as similar as possible to the correlation filter in *th-1* frame, to prevent the corruption of the correlation filter, and it can also play a good role to against occlusion. However, STRCF only considers the correlation filter of two adjacent frames, the correlation filter of previous frame has excessive influence on the current correlation filter. When an error occurs during tracking, STRCF is difficult to keep the robustness of the correlation filter in learning. As shown in Fig 4.1, an adaptive spatial-temporal regularization is proposed to alleviate the negative influence of abnormal conditions during tracking. We make use of more discriminative correlation filter model information to train new DCF model $\boldsymbol{f}_t$. $\boldsymbol{f}_{t-1}$ denotes the correlation filter in *th-1* frame. We adopt $\boldsymbol{f}_*$ to replace $\boldsymbol{f}_{t-1}$. The expression of $\boldsymbol{f}_*$ as follows:

$$\begin{cases} \boldsymbol{f}_* = \boldsymbol{f}_t + \alpha(\boldsymbol{f}_{t-1} - \boldsymbol{f}_{t-2}) & if\ frame \geq 3 \\ \boldsymbol{f}_* = \boldsymbol{f}_t & else \end{cases} \tag{4.1}$$

CFASE aims to minimize the following formulation to obtain the optimum discriminative correlation filter model $\boldsymbol{f}$,

$$\min_{\boldsymbol{f}} \frac{1}{2}\left\|\sum_{d=1}^{D} x_t^d * \boldsymbol{f}^d - y\right\|^2 + \frac{1}{2}\sum_{d=1}^{D}\left\|w \cdot \boldsymbol{f}^d\right\|^2 + \frac{\mu}{2}\|\boldsymbol{f} - \boldsymbol{f}_*\|^2 \tag{4.2}$$

Here, $\alpha$ is a parameter. $\boldsymbol{f}_{t-1}$ denotes the correlation filter in *t-1 th* frame. $\mu$ is the parameter of temporal regularization.

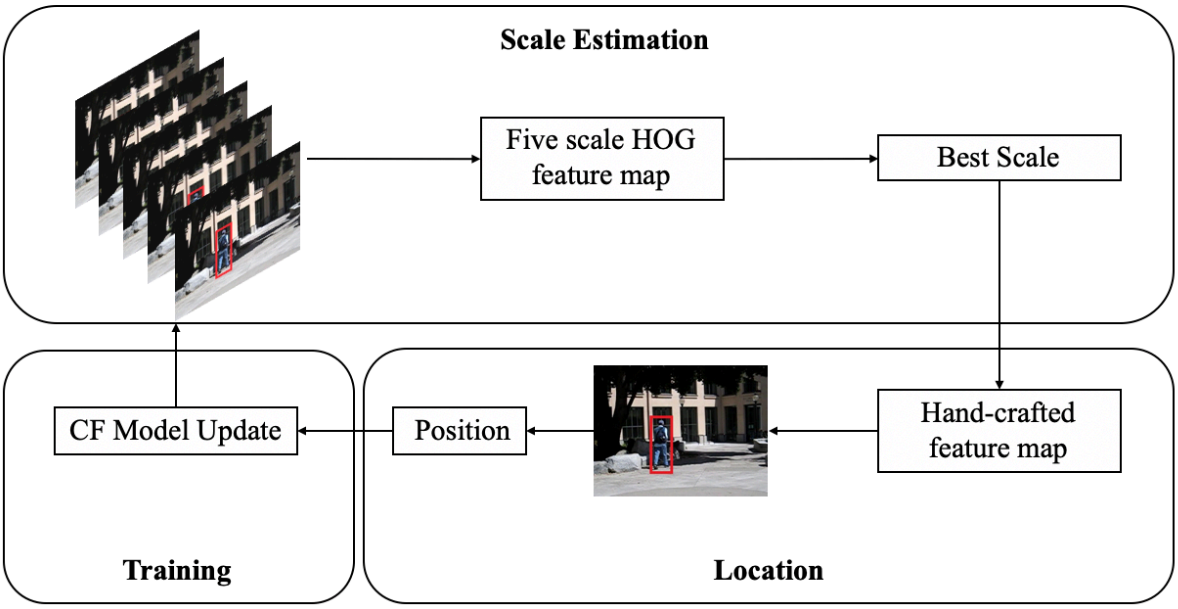


Figure 4.2 The process of the proposed method. The process of the tracker can be divided into three parts: Scale Estimation, Location, and Training.

The optimization function has not much changed, improving temporal regularization without calculation burden.

In general, the process of the tracking algorithms contains two parts which location and training. Location and scale estimation are one component in STRCF. In other words, the scale estimation is conducted meanwhile the position of the target object is obtained. As shown in Fig 4.2, the scale estimation and location are separated in the proposed method. CFASE adopts the HOG feature to estimate the scale of the target object. The best scale is directly used to locate the position of the object. The training model is conducted in the end. It looks very complicated to distinguish between the scale estimation model and the CF location model. It takes full advantage of the characteristics of the HOG feature. The high precision scale estimation makes for locating the position of the object.

## 4.2 State Estimation

The tracking algorithms aim to obtain the position of the target object accurately. Moreover, the scale is a significant factor affecting the performance of the tracking. However, the common DCF trackers [31, 38, 63] train the model directly after obtaining the object location and scale, completely ignoring the accuracy of the target location and scale. In CFASE, we judge the accuracy of the location and scale of the object.

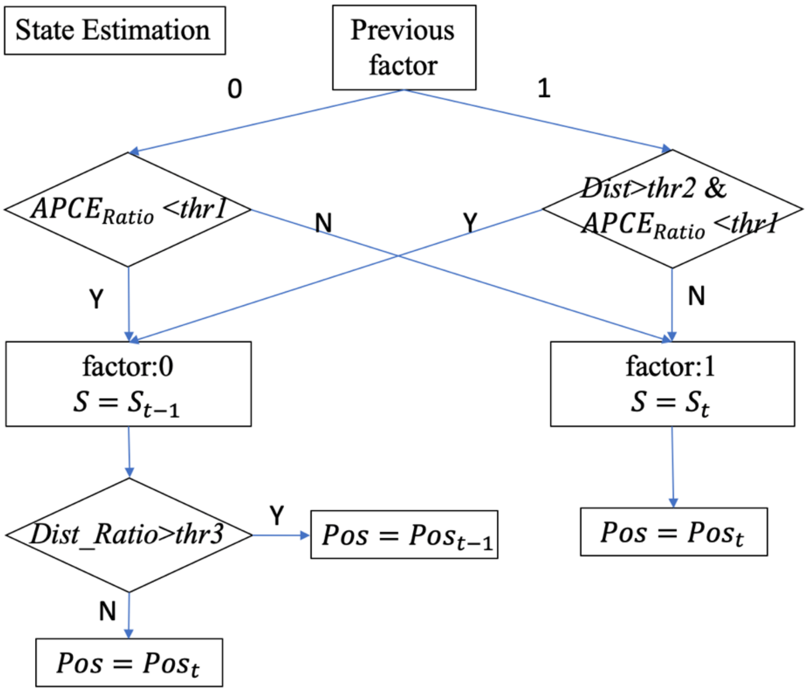


Figure 4.3 The process of judgment about scale estimation and location.

Fig 4.3 shows the process of judgment about scale estimation and location. $S$ and $S_t$ are the optimal scale and the obtained scale in the $t - th$ frame. $Pos$ and post are the final locations and the obtained location in the $t - th$ frame. Suppose the target object's state is normal. In that case, the obtained scale and location are updated ($S = S_t$, and $Pos = Pos_t$). Otherwise, the optimal scale and location are not updated, which assigns the value of the optimal scale and location in the previous frame ($S = S_{t-1}$, and $Pos = Pos_{t-1}$). In the process of judgment about scale estimation and location, the target object's state is the important factor.

As discussed in chapter 3, the confidence map of the DCF trackers reflects the performance of tracking. We introduce an evaluation metric (Average Peak-to Correlation Energy: *APCE*) [72], to enhance the accuracy of the state estimation. The fluctuation of the response map can reflect the confidence degree about the tracking performance. The ideal response map should be similar to the Gaussian distribution, have only one peak. In this case, the fluctuation of the response map is low. However, the response map will fluctuate when the detected target does not match the CF model. *APCE* takes full advantage of every value of the response map, reflects the fluctuated degree of response maps and the confidence level of the detected targets. The formulation of *APCE* can be expressed as follows:

$$APCE = \frac{\|R_{max} - R_{min}\|^2}{mean(\sum_{i,j}(R_{i,j} - R_{min})^2)} \tag{4.3}$$

Where $\boldsymbol{R_{max}}$, $\boldsymbol{R_{min}}$, and $\boldsymbol{R_{i,j}}$ denotes the maximum, minimum and the i row h column elements of response map.

To increase the accuracy of *APCE*'s judgment, we select the ratio between *APCE* in the current frame and the average of *APCE* to estimate the object's state. The judgment ratio can be expressed as follows,

$$APCE_{Ratio} = \frac{APCE_t}{mean(\sum_t APCE_t)} \tag{4.4}$$

A single metric cannot judge the target object's state accurately. As shown in Fig 4.2, the scale estimation and the object location are distinguished into two parts in CFASE. Scale pool is adopted to estimate the object's scale which calculates scale and location by producing the multiple research regions. CFASE can obtain the object scale $S_{HOG}$ and the object position $Pos_{HOG}$ in the scale estimation. $Pos_{HOG}$ is not the final object position in the current frame, and does not participate in the model update process. $S_{HOG}$ is used to the object location filter model to obtain the object position as $Pos_{Hand-Crafted}$. So, we can get two object positions with different models. Under normal conditions, $Pos_{HOG}$ and $Pos_{Hand-Crafted}$ should be similar. While obtaining the best object position, we can use the distance *Dist* between the two positions to judge the reliability of the scale [28]. The distance *Dist* can be defined as:

$$Dist = \sqrt{(Pos_{HOG} - Pos_{HOG+CN})^2} \tag{4.5}$$

Object tracking is a continuous process, and objects in two adjacent frames are more closely related. We take full use of the continuous of the object state. When the object state is normal in the previous frame, we use $APCE_{Ratio}$ and $Dist$ to estimate the object state because $APCE_{Ratio}$ alone cannot meet the requirements of the judgment. As shown in Fig 4.3, when $APCE_{Ratio}$ is small, and $Dist$ is large, the object's state is considered abnormal. It means that the estimated scale is not reliable. The scale $S_t$ should not be updated in the current frame. When the object state is abnormal in the previous frame, we adopt $APCE_{Ratio}$ to estimate the object's state because $Dist$ has already become unreliable. If $APCE_{Ratio}$ is small, the object's state is considered abnormal, S is not updated. In Fig 4.3, when the state is normal, the factor is defined as 1. When the state is abnormal, the factor is 0.

Moreover, once the change degree of object location in two adjacent frames is far greater than the historical position change, object location should not be updated and maintain the position of the previous frame $Pos_{t-1}$. The evaluation criterion (*Dist_Ratio*) is expressed as follows.

$$Dist\_Ratio = \frac{Dist(Pos_t, Pos_{t-1})}{Dist(Pos_{t-1}, Pos_{t-2})} \tag{4.6}$$

$Dist$ denotes the distance between two positions. *t* denotes the *t-th* frame, $Pos_t$ denotes the object position in the *t-th* frame.

## 4.3 Evaluation Experiments

The evaluation experiments of the proposed tracker are introduced in this section. Firstly, the evaluation benchmark database and analysis metric are introduced. Next, the comparison of different trackers (STRCF [31], SRDCF [66], BACF [38], CFHA [98], AutoTrack [95], KCF [47], SAMF [63], DSST [65], Staple [57], CSK [46]) to analyze the strength and weaknesses of the tracker.

### 4.3.1 Temple Color Benchmark Databases

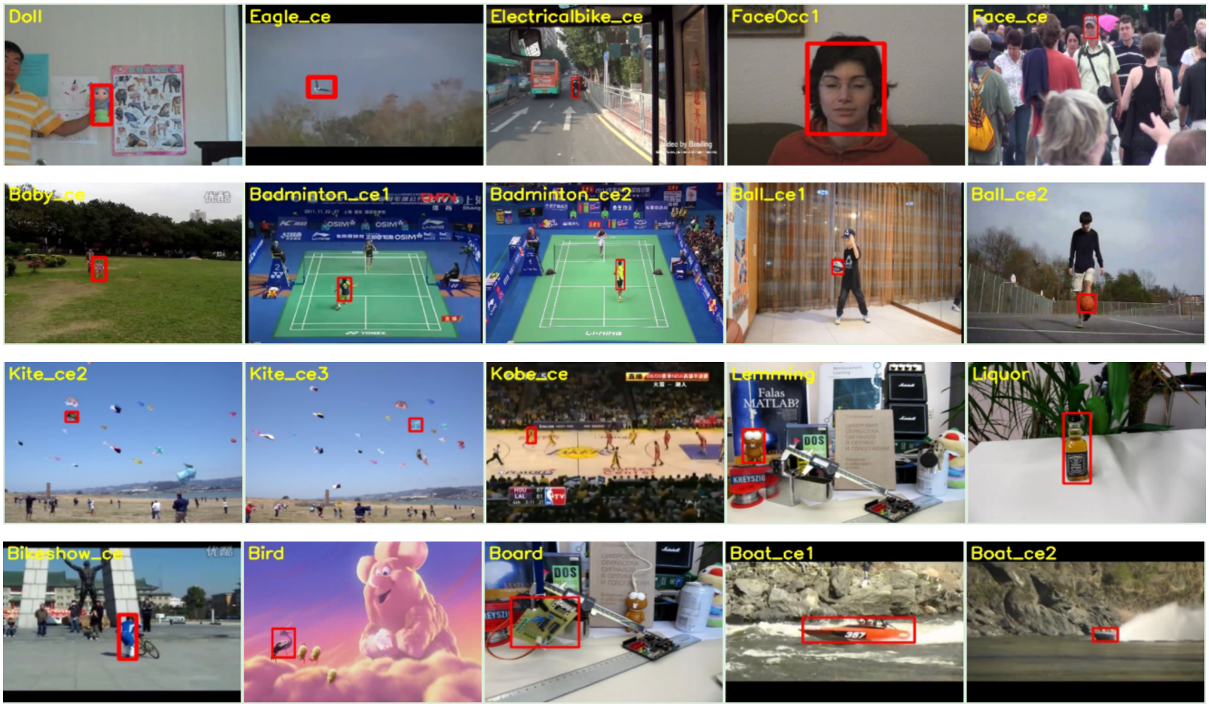


Figure 4.4 The partial image sequence of the Temple Color 128 benchmark database.

OTB benchmark database is the classic evaluation database for the tracking algorithms. The analysis experiments of CFASE are conducted on the OTB benchmark database. Since CFASE not only adopts the HOG feature but also uses the color name feature and grayscale feature to locate the object and train the CF model, the Temple Color 128 (TC-128) [75] benchmark database is used to evaluate the trackers (As shown in Fig 4.4). Temple color 128 benchmark database includes a large set of 128 color sequences that annotate with ground truth. Since color information can provide rich discriminative clues for visual tracking, the Temple color 128 database aims to thoroughly evaluate the trackers with the color feature. The trackers do not limit themselves to the grayscale image sequence.

The evaluation metrics still adopt precision scores, success scores, and FPS. The detailed introduction about these evaluation metrics is shown in section 3.4.1.

### 4.3.2 Parameter Setting

All the analysis experiments are conducted on the Matlab2019b platform and a PC machine with an Intel (R) Core (TM) i7-9700F CPU (3.00GHZ), 16GB memory. The shape of the search region is defined as square, the size = $\sqrt{5WH}$, ($W$ is the object's width, $H$ is the height of the object). The cell size of the HOG features is set to 4. The regularization parameter μ is set to 15 and 13 for the location filter and scale estimation filter, respectively. α is set to 0.5 and 0.2 for location filter and scale estimation filter, respectively. The scaling step is set to 1.01. The three parameters in Fig 4.3 are decided with experiments.

Table 4.1 Success score of CFASE on OTB-2013 with different *thr2*. *thr1* is set to 0.30. The best result is shown in red font.

| *thr2* | 5 | 10 | 15 | 20 | 25 | 30 |
|---|---|---|---|---|---|---|
| Success | 0.693 | 0.693 | 0.705 | 0.706 | 0.705 | 0.699 |

Table 4.2 Success score of CFASE on OTB2013 with different *thr1*. *thr2* is set to 20. The best result is shown in red font.

| *thr1* | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 |
|---|---|---|---|---|---|---|
| Success | 0.692 | 0.699 | 0.706 | 0.706 | 0.693 | 0.691 |

Table 4.3 Success score of CFASE on OTB2013 with different *thr3*. *thr1* is set to 0.25, *thr2* is set to 20. The best result is shown in red font.

| *thr3* | 1.4 | 1.7 | 2.0 | 2.3 | 2.6 | 3.0 |
|---|---|---|---|---|---|---|
| Success | 0.704 | 0.706 | 0.705 | 0.705 | 0.704 | 0.704 |

Table 4.4 Success score of CFASE on OTB2013 with different *thr3*. *thr1* is set to 0.3, *thr2* is set to 20. The best result is shown in red font.

| *thr3* | 1.4 | 1.7 | 2.0 | 2.3 | 2.6 | 3.0 |
|---|---|---|---|---|---|---|
| Success | 0.705 | 0.706 | 0.705 | 0.705 | 0.705 | 0.705 |

Table 4.1, Table 4.2, Table 4.3, and Table 4.4 show the success score with various values of the thresholds *thr1*, *thr2*, and *thr3* for $APCE_{Ratio}$, $Dist$, and $Dist\_Ratio$. It is careful for trackers to estimate the object's abnormal. Therefore, *thr1* and *thr2* are used simultaneously when the target transitions from normal to abnormal. Too large *thr1* or too small *thr2* denote that the tracking performance is not reliable. As shown in Table 4.1, the proposed tracker achieves the best performance when *thr2* is set to 20. However, the change of *thr2* has little influence on the tracking performance. In Table 4.2, when *thr1* is 0.30 or 0.25, the tracker performs the same result. The result of Table 4.2 also explains that our method is not so sensitive to the parameters. The parameters are relatively independent. For two *thr1*, we conduct more experiments to get the optimal threshold. In the results of Table 4.3 and Table 4.4, *thr3* is only effective under terrible conditions. The influence of *thr3* is little. Finally, *thr1*, *thr2*, and *thr3* are set to 0.30, 20, and 1.7 in our tracker in all the experiments. From the results, three thresholds have little effect on tracking performance.

Table 4.5 Success score of CFASE on OTB-2013 with the temporal regularization parameters $\mu1$ and $\mu2$. The best result is shown in red font.

| | $\mu1 = 12$ | $\mu1 = 13$ | $\mu1 = 14$ | $\mu1 = 15$ | $\mu1 = 16$ |
|---|---|---|---|---|---|
| $\mu2 = 12$ | 0.675 | 0.665 | 0.688 | 0.681 | 0.681 |
| $\mu2 = 13$ | 0.679 | 0.668 | 0.688 | 0.706 | 0.681 |
| $\mu2 = 14$ | 0.674 | 0.657 | 0.679 | 0.675 | 0.685 |
| $\mu2 = 15$ | 0.660 | 0.653 | 0.672 | 0.679 | 0.687 |
| $\mu2 = 16$ | 0.678 | 0.661 | 0.683 | 0.689 | 0.678 |

Table 4.6 Success score of CFASE on OTB-2013 with the parameters $\alpha1$ and $\alpha2$. The best result is shown in red font.

| | $\alpha1 = 0.1$ | $\alpha1 = 0.2$ | $\alpha1 = 0.3$ | $\alpha1 = 0.4$ | $\alpha1 = 0.5$ | $\alpha1 = 0.6$ |
|---|---|---|---|---|---|---|
| $\alpha2 = 0.1$ | 0.688 | 0.689 | 0.688 | 0.691 | 0.696 | 0.687 |
| $\alpha2 = 0.2$ | 0.688 | 0.689 | 0.689 | 0.691 | 0.706 | 0.694 |
| $\alpha2 = 0.3$ | 0.688 | 0.694 | 0.687 | 0.689 | 0.700 | 0.699 |
| $\alpha2 = 0.4$ | 0.687 | 0.687 | 0.687 | 0.687 | 0.700 | 0.701 |
| $\alpha2 = 0.5$ | 0.684 | 0.697 | 0.694 | 0.695 | 0.682 | 0.683 |
| $\alpha2 = 0.6$ | 0.684 | 0.698 | 0.696 | 0.684 | 0.670 | 0.669 |

As shown in Equation 4.1 and Equation 4.2, the learning rate $\alpha$ and the temporal regularization parameters $\mu$ affect the performance of the tracking model. Since CAFSE trains the location CF model and scale estimation CF model, there are four parameters (location CF model: $\mu1$, $\alpha1$; scale estimation CF model: $\mu2$, $\alpha2$) that need to train. Table 4.5 and Table 4.6 illustrate the success score of the tracker on the OTB-2013 benchmark with different parameters, respectively. Tracker is influenced by the temporal regularization parameters $\mu$. When $\mu1$ and $\mu2$ are set to 15 and 13, respectively, the tracker achieves the best performance. The performance of the tracker is less affected by parameter $\alpha$ . The performance of the tracker is not significantly degraded, even if the value of $\alpha$ is a little changed from the best value. When $\alpha1$ and $\alpha2$ are set to 0.5 and 0.2, respectively, the tracker achieves the best performance. In the experiment for the data OTB 2013, 2015 and TC-128, the values of all the parameters are set to the best ones presented in this section.

### 4.3.3 Analysis of OTB Benchmark Database

In this section, we offer comprehensive assessments to evaluate the performance of the proposed CFASE on the OTB-2013, OTB-2015 benchmark database. The compared state-of-the-art trackers including STRCF [31], SRDCF [66], BACF [38], CFHA [98], AutoTrack [95], KCF [47], SAMF [63], DSST [65], Staple [57], CSK [46] show that our tracker obtained the best performance.

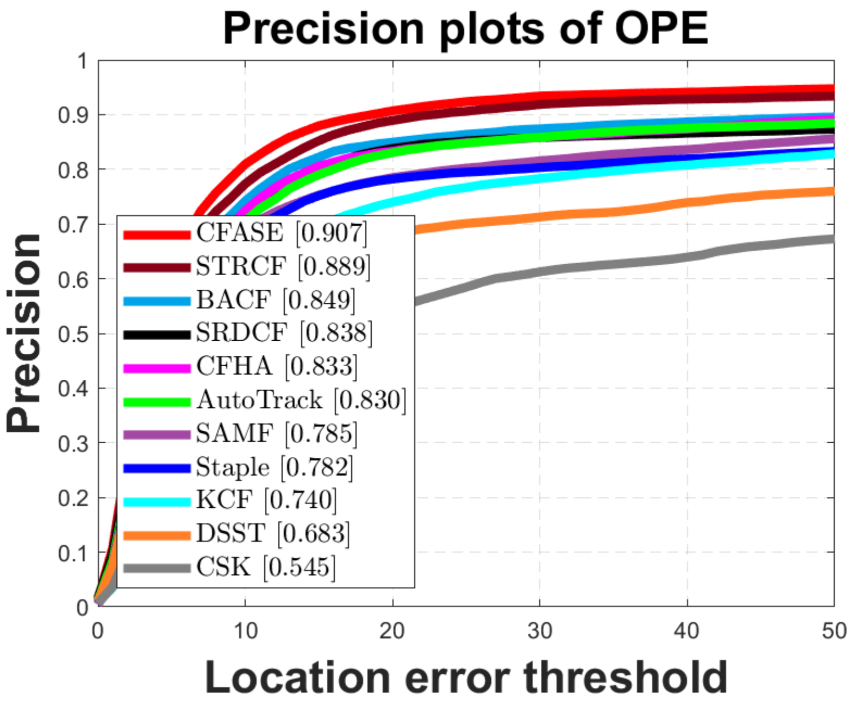


Figure 4.5 Comparisons with state-of-the-art DCF trackers on OTB-2013 benchmark in terms of precision plot.

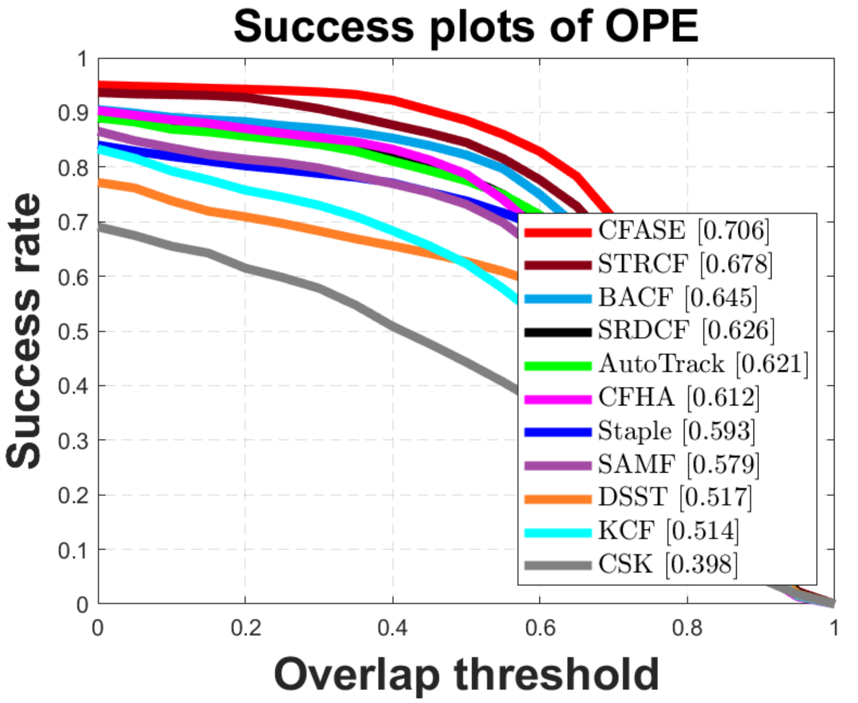


Figure 4.6 Comparisons with state-of-the-art DCF trackers on OTB-2013 benchmark in terms of success plot.

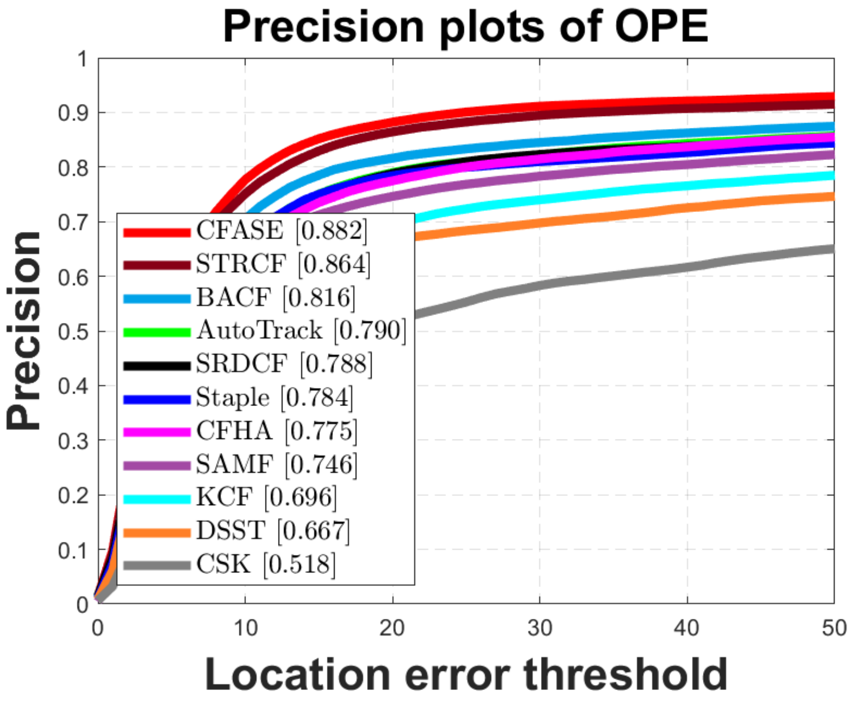


Figure 4.7 Comparisons with state-of-the-art DCF trackers on OTB-2015 benchmark in terms of precision plot.

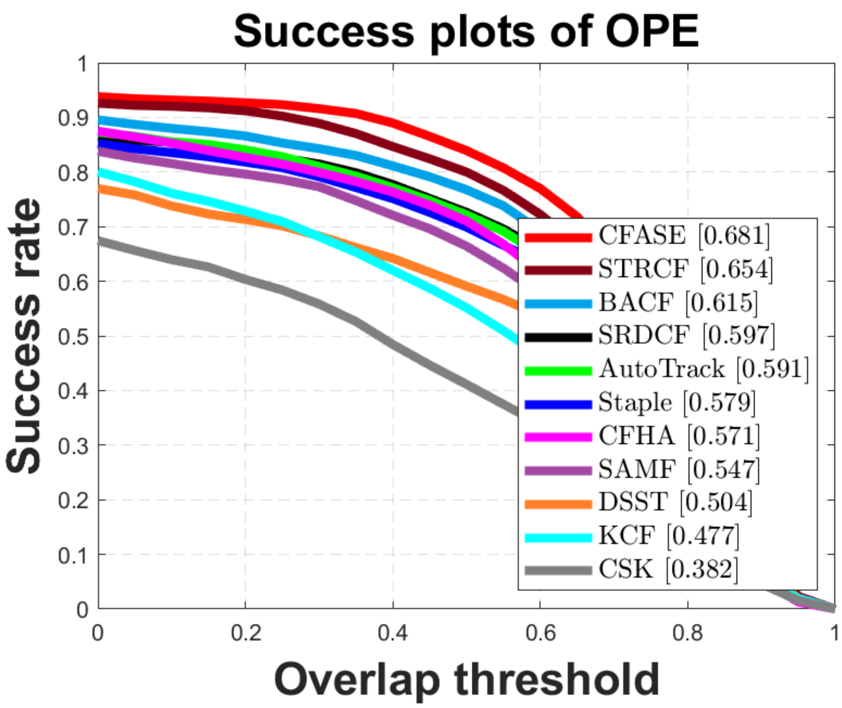


Figure 4.8 Comparisons with state-of-the-art DCF trackers on OTB-2015 benchmark in terms of success plot.

From the experiment result on OTB-2013 (Fig 4.5 and Fig 4.6), CFASE achieves the significant performance in tracking, obtains a precision score of 90.7% and a success score of 70.6%, respectively. Compared to the baseline STRCF, CFASE gains 1.8% and 2.8% on the precision and success scores, respectively. On OTB-2015 (Fig 4.7 and Fig 4.8), CFASE achieves a precision score of 88.2% and a success score of 68.1%, respectively. With the increase of the image sequences, the experimental data decreased. However, CFASE improvement to STRCF is unchanged, and it is still 1.8% and 2.7%.

Table 4.7 Success and speed of top-5 trackers on the OTB-2015. The best two results are shown in red and blue fonts, respectively.

| | SRDCF | BACF | AutoTrack | STRCF | CFASE |
|---|---|---|---|---|---|
| Success | 0.597 | 0.615 | 0.591 | **0.654** | **0.681** |
| FPS | 10.4 | **45.0** | **37.5** | 33.3 | 31.1 |

The DCF trackers with top-5 success scores on the OTB-2015 are selected to analyze the performance of the trackers. As shown in Table 4.7, although the running speed of CFASE is not the fastest, the performance of CFASE is best excellent on the OTB-2015.

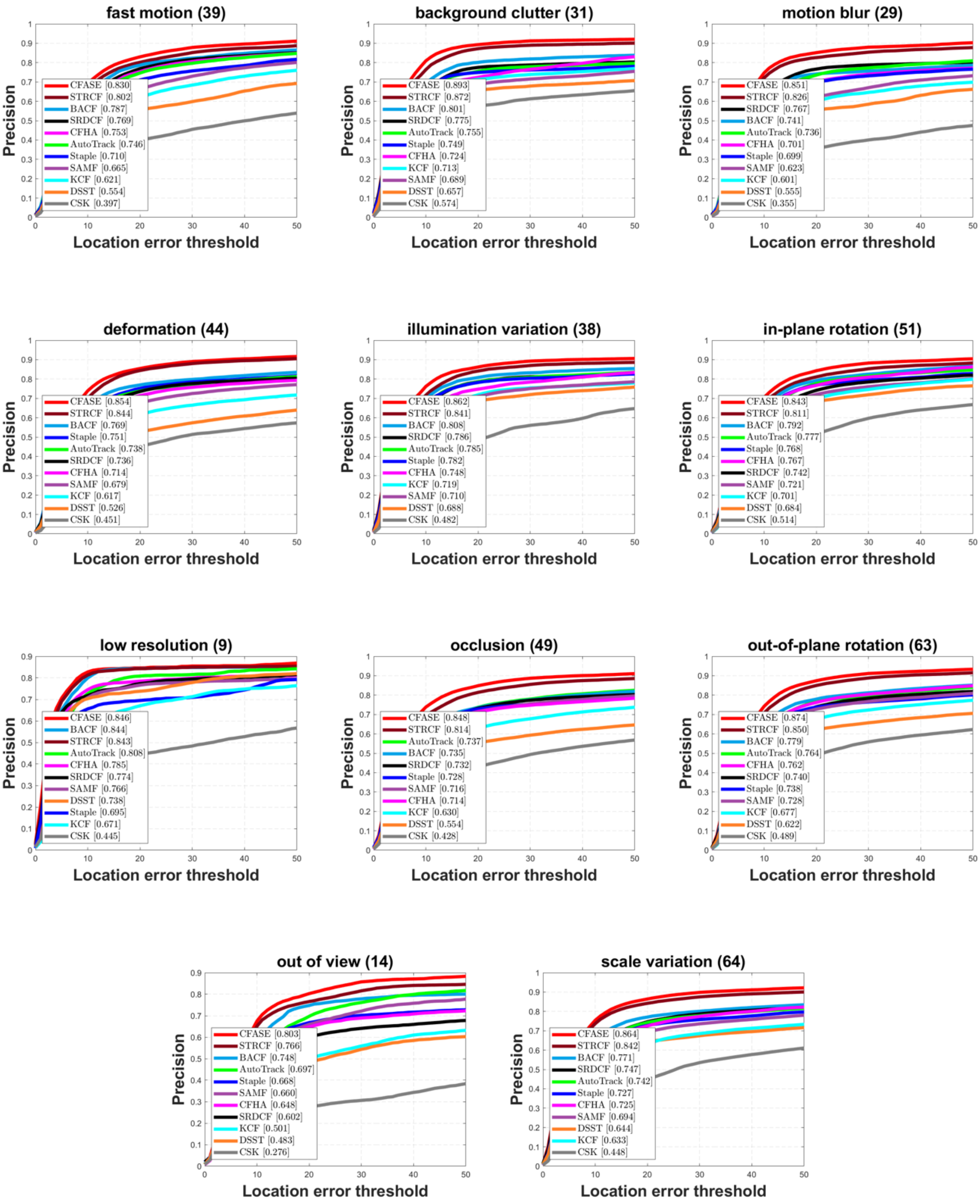


Figure 4.9 Attribute-based analysis of different trackers on the OTB-2015 dataset with 100 videos. The precision plots are shown for eleven attributes.

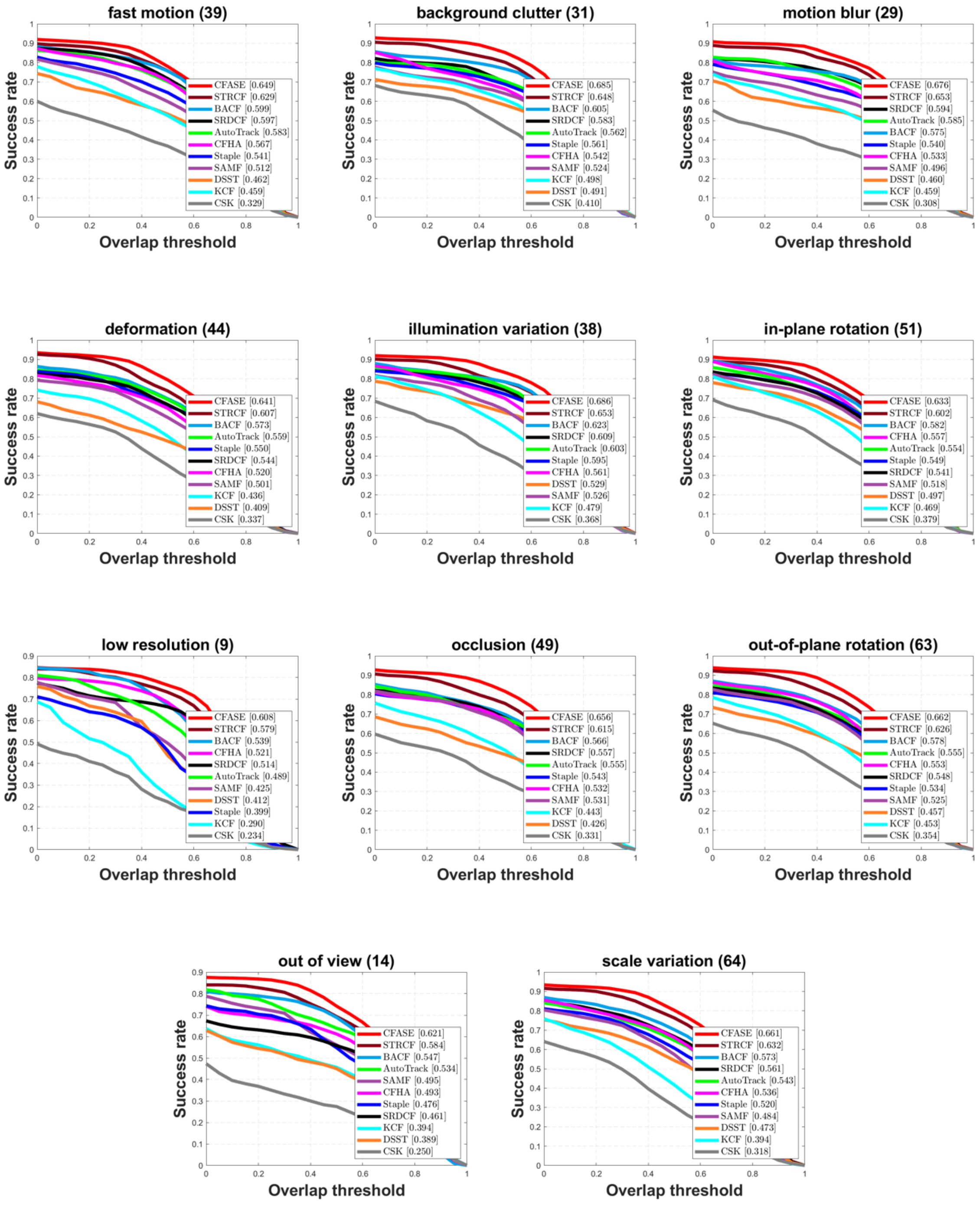


Figure 4.5 Attribute-based analysis of different trackers on the OTB-2015 dataset with 100 videos. The success plots are shown for eleven attributes.

The evaluation of the different DCF trackers by all attributes of OTB-2015. From the precision plot (as shown in Fig 4.9), compared to STRCF, CFASE obtains the improvement of 2.8%, 2.1%, 2.5%, 1.0%,2.1%, 3.2%, 0.3%, 3.4%, 2.4%, 3.7%, 2.2% on the 11 attributes (FM, BC, MB, DEF, ILL, IPR, LR, OCC, OPR, OV, and SV), respectively. Although CFASE does not significantly improve DEF and LR, it gets at least a 2.0% raise on the other attributes, especially CFASE obtains bigger than 3.0% on

IPR, OCC, and OV. This experiment results illustrate the validity of the accuracy estimation about the object position.

Fig 4.10 shows the success scores of the evaluated trackers on the 11 attributes. Compared to STRCF, CFASE obtains a gain of 2.0%, 3.7%, 2.3%, 3.4%, 3.3%, 3.1%, 2.9%, 4.1%, 3.6%, 3.7%, 2.9% on the 11 attributes (FM, BC, MB, DEF, ILL, IPR, LR, OCC, OPR, OV, and SV), respectively. CFASE gets the improvement is even more pronounced on success rates, even if DEF and LR. The improvement of CFASE gets a 3.0% or more increase on the most attributes. The proposed scale estimation is beneficial to increase the overlap ratio.

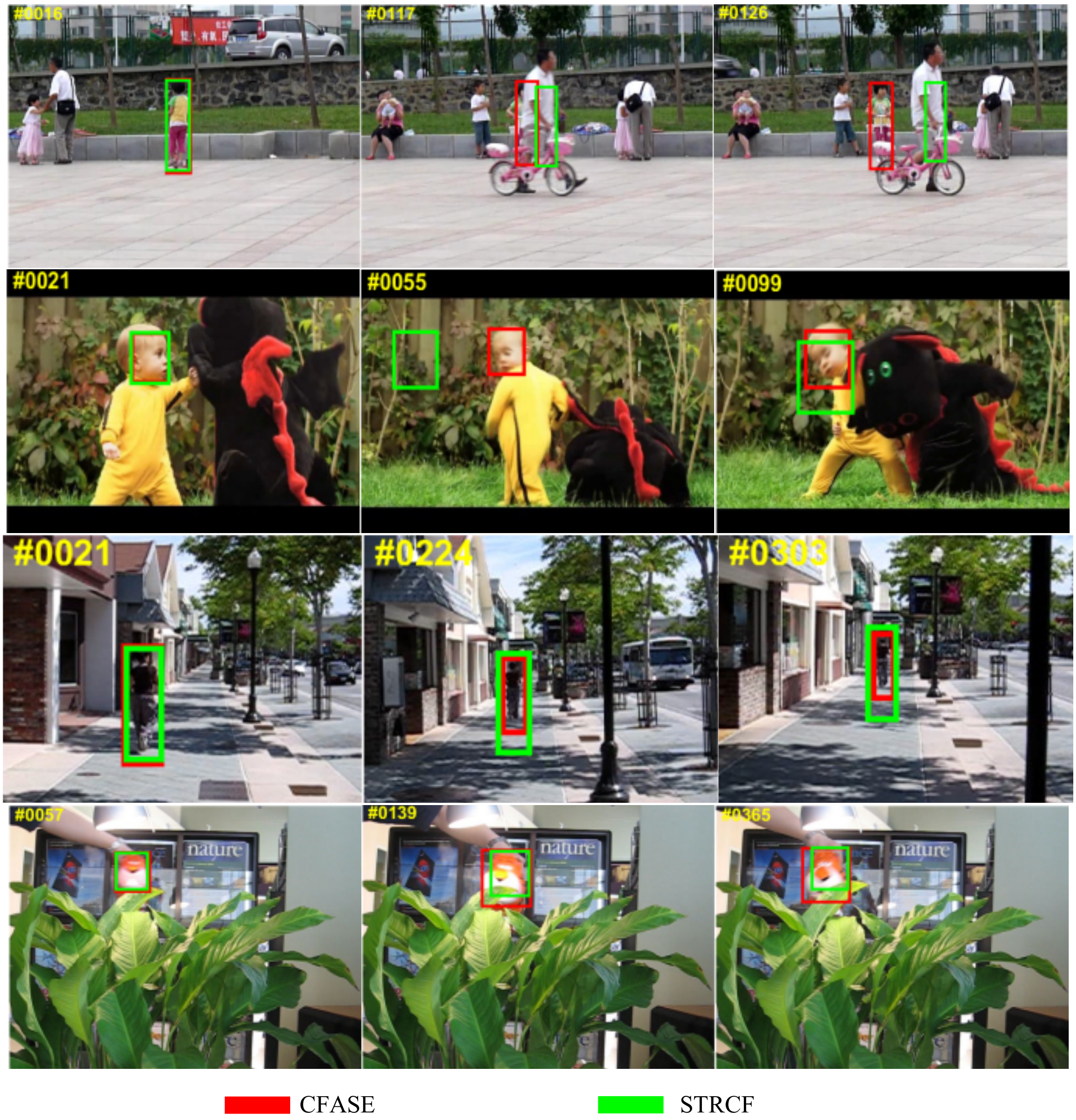


Figure 4.6 Qualitative evaluation of CFASE and STRCF on the four video sequences (*Girl2*, *DragonBaby*, *Human9*, *Tiger2*) with occlusion, fast motion, illumination variation, and deformation, respectively. CFASE obtains outperformance than STRCF on different conditions

In Fig 4.11, qualitative evaluation of CFASE and STRCF are shown on the four video sequences (*Girl2, DragonBaby, Human9, Tiger2*). From the tracking performance, CFASE is better than STRCF. As shown in *Girl2*, when occlusion occurs, CFASE is more accurate tracks the object without drifting. *DragonBaby*, with the attribute as fast motion, CFASE still locates the target object accurately. In *Human9*, *Tiger2*, CFASE more precision estimate scale than STRCF.

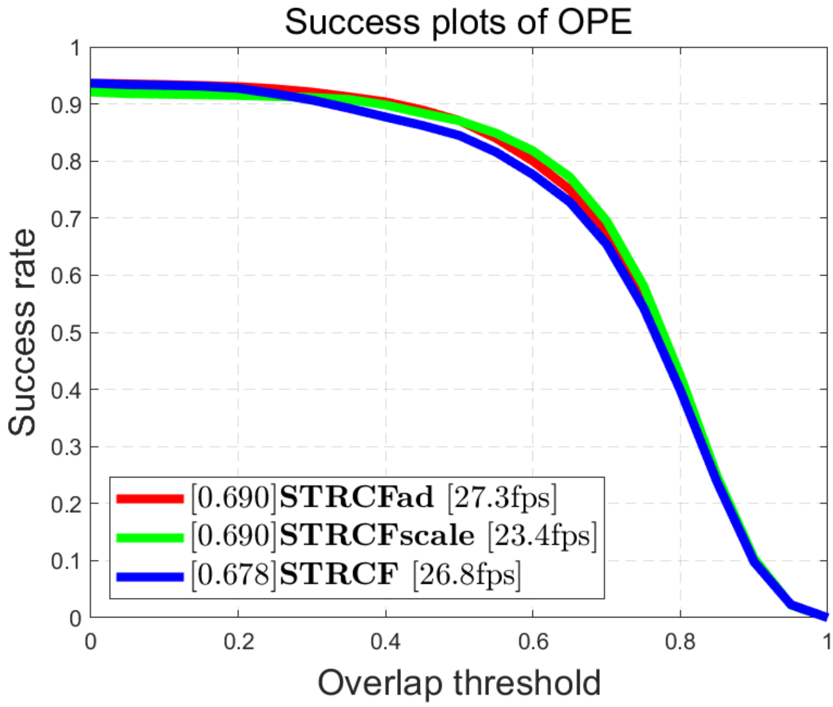


Figure 4.7 Success plots of STRCF, STRCF with adaptive model and Scale estimation filter on OTB-2013 dataset.

As shown in Fig 4.12, we illustrate the success scores of STRCF, STRCF with adaptive temporal regularization, and STRCF with scale estimation filter on OTB-2013. Adaptive temporal regularization improves, and scales estimation filters both get a gain of 1.2% than STRCF in success scores. The burden of calculation has not increased.

### 4.3.4 Analysis of Temple color 128

Most modern trackers employ color information to train the location model. OTB benchmarks contain some of the grayscale image sequences. It is not enough for some state-of-the-art trackers with color features to get the best evaluation, so we conduct our method on Temple color 128 benchmarks with STRCF [31], MEEM [45], Struck [83], ASLA [91], VTD [41], CN2 [70], DFT [59], CSK [46], KCF [47].

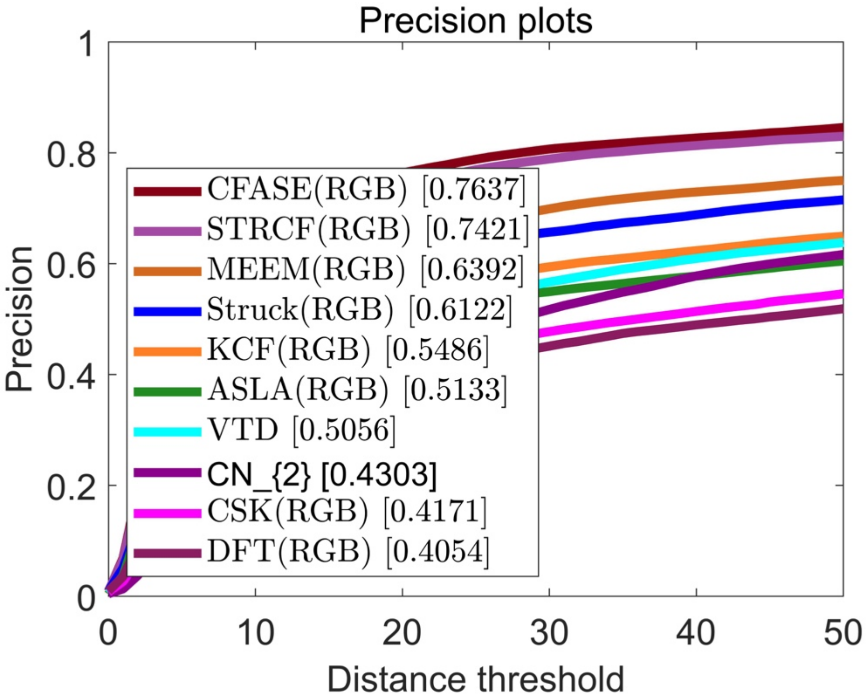


Figure 4.8 Comparisons with state-of-the-art trackers on TC-128 dataset in terms of precision plot.

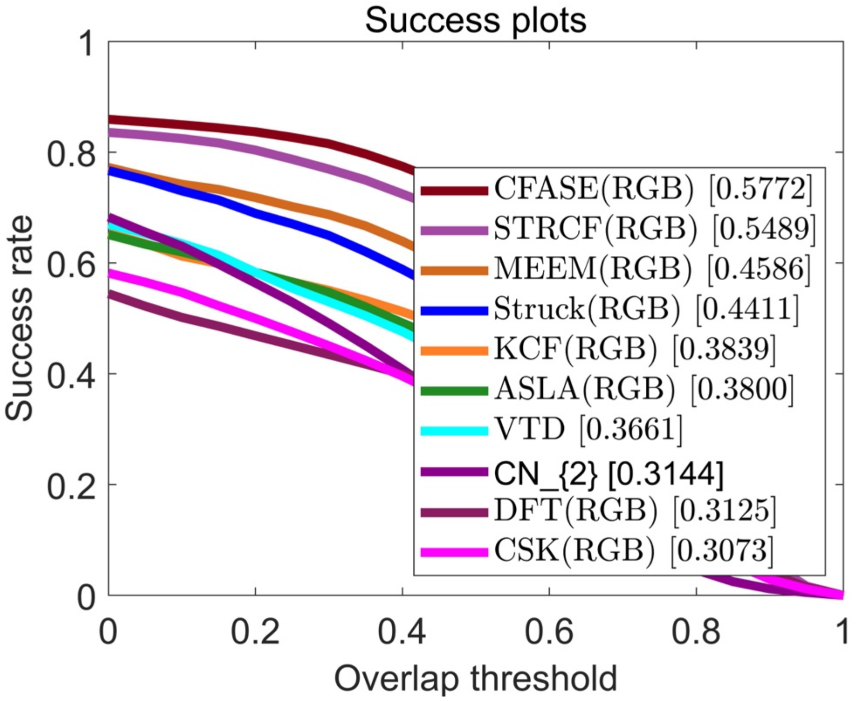


Figure 4.9 Comparisons with state-of-the-art trackers on TC-128 dataset in terms of success plot.

Our method obtains the outperformance both in precision plot and success plot on TC128 benchmark database. As shown in Fig 4.13 and Fig 4.14, our method achieves the best performance on the TC-128 benchmark. Compared to STRCF, CFASE obtains a gain of 2.1% and 2.9% in precision scores and success scores, respectively.

# Chapter 5

# Visual Tracking via Robust and Efficient Temporal Regularized Correlation Filters

In this section, background-aware correlation filter tracker with spatial-temporal regularization (BASTR) is proposed to achieve robustness and efficient performance in visual tracking. The background-Aware Correlation Filter (BACF) model [38] takes full advantage of the background information to enhance the robustness of the DCF model. However, BACF adopts the fixed learning rate to update the feature and DCF models. This method is not suitable for the state of the object changes. We introduce temporal regularization to mitigate the adverse effects of online updates. Besides, the DCF model coefficients are adjusted with adaptive spatial regularization. Thus, a discriminative correlation filters model becomes more robust. In terms of that, two different scale estimation methods are selected to obtain the object's scale adaptively. The proposed tracker can get the outperformance on different tracking databases.

## 5.1 Robust and Efficient Temporal Regularized Correlation Filters model

The efficiency of the standard DCF trackers benefits from the periodic assumption at both training and detection. However, the periodic assumption generates needless boundary effects. Boundary effects lead to the inaccurate representation of the image patch content since the training patches contain periodic repetitions. These inaccurate negative training patches reduce the discriminative power of the DCF model. Moreover, the response value near the center of the response map is reliable, and the boundary effect heavily influences the response values of other parts. Traditional CF trackers mitigate the negative influence of the boundary effect by restricting the search region size (be restricted to a 2.5 times target size). Although this method has some effect, it also limits the performance of the DCF trackers.

BACF introduces a binary matrix to crop all training samples to reduce the influence from the inaccurate negative training patches. Therefore, the center of each training sample is retained, the border is discarded. Since this method mitigates the boundary effects' influence, the search region of BACF is set to 5 times the object size. The crop operation enhances the quality of each training sample, and the larger research region leads to increase training samples. Therefore, the discriminative power of the DCF model is enhanced.

Although the obtained DCF model of BACF has robustness for the boundary effects, BACF adopts the online method to update the appearance model and the detection model. The online update method can meet up the variation of the environment and the change of the object to a certain extent. However, that does not mean it is the best one. Concerning deformation and full occlusion, the online update method cannot obtain a robust appearance model. We introduce the temporal regularization into BACF, the performance of the obtained model will be better. Besides, most object trackers do not consider the variation of the correlation filter center. When deformation or rotation occurs, it is easy for trackers to produce drift because the detection model does not keep up with the change of the object. We introduce an adaptive weight into regularization. The spatial regularization is updated as the same as the temporal regularization. The weight of the current frame keeps similar to the weight of the previous frame. A spatial-temporal regularized correlation filters (BASTR) model can be expressed as follows.

$$E(\boldsymbol{f}) = \tfrac{1}{2}\|y - \textstyle\sum_{c=1}^{C}(P^T\boldsymbol{f}_c) * x_c\|^2 + \tfrac{\gamma}{2}\textstyle\sum_{c=1}^{C}\|w(P^T\boldsymbol{f}_c)\|^2 + \tfrac{\mu}{2}\textstyle\sum_{c=1}^{C}\|P^T\boldsymbol{f}_c^t - P^T\boldsymbol{f}_c^{t-1}\|^2 + \tfrac{u}{2}\|w - w_{t-1}\|^2 \quad (5.1)$$

Here, $y$ is the desired response which is expressed as Gaussian distribution. $P$ is a binary matrix which crops the center of the classifier. $c$ is the channel number and $*$ denotes circular convolution. $T$ denotes matrix transpose. $\boldsymbol{f}_c^t$ denotes the classifier in the *t*-th frame, and $\boldsymbol{f}_c^{t-1}$ is the classifier in the (*t* - *1*)-th frame. $w$ is the weight of the spatial regularization. $w_{t-1}$ is the spatial weight in the *t*-th frame. $\mu$ and $u$ denotes a regularization parameter.

Eq. (5.1) can be efficiently solved via Alternating Direction Method of Multipliers (ADMM). Firstly, the correlation filter model of Eq. (5.1) is transformed into the frequency domain, to efficiently obtain the local optimal solution. Thus, the Augmented Lagrangian form of Eq. (5.1) can be expressed as

$$\begin{aligned} L(\hat{g}, \boldsymbol{f}, \hat{\zeta}) = \frac{1}{2}\left\|\hat{y} - \sum_{k=1}^{K}\hat{g}\widehat{x_k}\right\|^2 + \frac{\gamma}{2}\sum_{k=1}^{K}\left\|w\sqrt{L}FP^T\boldsymbol{f}_k\right\|^2 + \frac{\mu}{2}\sum_{k=1}^{K}\left\|\widehat{g_k^t} - \widehat{g_k^{t-1}}\right\|^2 + \sum_{k=1}^{K}\widehat{\zeta_k^T}\left(\widehat{g_k} - \sqrt{L}FP^T\boldsymbol{f}_k\right) \\ + \tfrac{\lambda}{2}\textstyle\sum_{k=1}^{K}\left\|\widehat{g_k} - \sqrt{L}FP^T\boldsymbol{f}_k\right\|^2 + \tfrac{u}{2}\|w - w_{t-1}\|^2 \qquad (5.2) \end{aligned}$$

Where $\widehat{g_k} = \sqrt{L}FP^T\boldsymbol{f}_k$ is an auxiliary variable matrix, $\wedge$ denotes the discrete Fourier transform, $\zeta$ is the Lagrange Multiplier, $\hat{\zeta}$ is the corresponding Fourier transform, and F is the orthonormal $L \times L$

matrix of complex basis vectors, any $L$ dimensional vectorized signal is transformed into the Fourier domain (such as $\hat{Q} = \sqrt{L}FQ$). $\lambda$ is the step size parameter.

Then, the following subproblems are alternatingly solved:

**Subproblem** $\hat{g}$: The optimal $\hat{g}$ can be formulated as

$$\hat{g} = \underset{g_k}{argmin}\frac{1}{2}\|\hat{y} - \sum_{k=1}^{K}\widehat{g_k}\widehat{x_k}\|^2 + \frac{\mu}{2}\sum_{k=1}^{K}\left\|\widehat{g_k}^{t} - \widehat{g_k}^{t-1}\right\|^2 + \sum_{k=1}^{K}\widehat{\zeta_k^T}\left(\widehat{g_k} - \sqrt{L}FP^T\boldsymbol{f}_k\right)$$

$$+\frac{\lambda}{2}\sum_{k=1}^{K}\left\|\widehat{g_k} - \sqrt{L}FP^T\boldsymbol{f}_k\right\|^2 \tag{5.3}$$

$$= \frac{\hat{x}\hat{y} + L\mu\sum_{k=1}^{K}\widehat{g_k}^{t-1} + L\lambda\sum_{k=1}^{K}\widehat{\boldsymbol{f}}_k - L\sum_{k=1}^{K}\widehat{\zeta_k^T}}{\hat{x}^T\hat{x} + L(\mu + \lambda)}$$

Where $\hat{x} = \{\widehat{x_1}, \widehat{x_2}, \dots, \widehat{x_k}\}$. $P$ is a binary matrix (For $P$, $P^TP = P$), and $f_k = \frac{1}{\sqrt{L}}PF^T\hat{\boldsymbol{f}}_k$.

**Subproblem** $\hat{\boldsymbol{f}}$: If the other variable is given, the optimal $\hat{\boldsymbol{f}}$ can be express as

$$\hat{\boldsymbol{f}} = \underset{f_k}{argmin}\frac{\gamma}{2}\sum_{k=1}^{K}\left\|w\sqrt{L}FP^T\boldsymbol{f}_k\right\|^2 + \sum_{k=1}^{K}\widehat{\zeta_k^T}\left(\widehat{g_k} - \sqrt{L}FP^T\boldsymbol{f}_k\right) + \frac{\lambda}{2}\sum_{k=1}^{K}\left\|\widehat{g_k} - \sqrt{L}FP^T\boldsymbol{f}_k\right\|^2 \tag{5.4}$$

$$= \frac{\sum_{k=1}^{K}(\zeta_k^T + \lambda g_k)}{\lambda + \gamma w^T w}$$

Where $g_k = \frac{1}{\sqrt{L}}PF^T\hat{g}_k, \zeta_k = \frac{1}{\sqrt{L}}PF^T\hat{\zeta}_k$.

**Subproblem** $w$: If $\boldsymbol{f}$ is given, the closed-form solution about $w$ can be express as:

$$w = \underset{w}{argmin}\frac{\gamma}{2}\sum_{k=1}^{K}\left\|w\sqrt{L}FP^T\boldsymbol{f}_k\right\|^2 + \frac{u}{2}\|w - w_{t-1}\|^2 \tag{5.5}$$

$$= \frac{uw_{t-1}}{u + \gamma P\boldsymbol{f}^T\boldsymbol{f}}$$

**Update the Lagrange Multiplier** $\hat{\zeta}$: The Lagrange Multiplier is update as

$$\hat{\zeta}^{i+1} = \hat{\zeta}^{i} + \lambda\left(\hat{g}^{i+1} - \hat{\boldsymbol{f}}^{i+1}\right) \tag{5.6}$$

$$\lambda^{i+1} = \min(\lambda_{max}, \beta\lambda^{i}) \tag{5.7}$$

Where $\hat{\zeta}^i$ is the Fourier transform of the Lagrange Multiplier in the previous state, $\lambda_{max}$ denotes the maximum of $\lambda$ and $\beta$ is the parameter.

## 5.2 Adaptive Scale Estimation

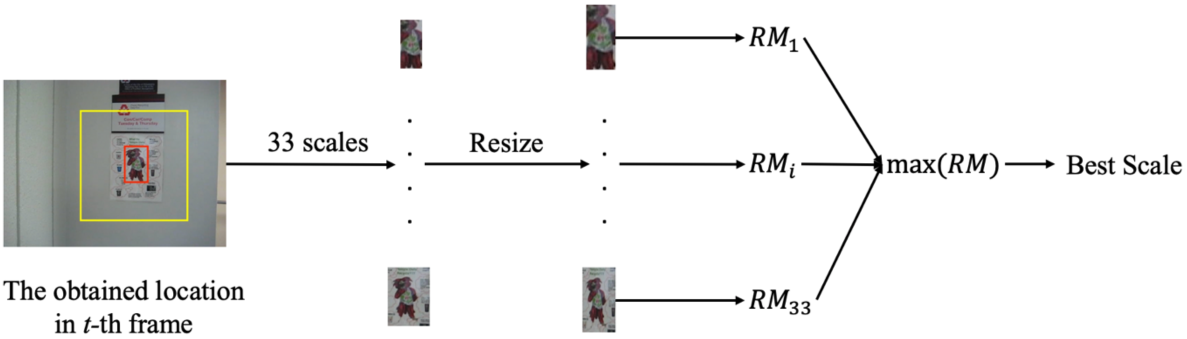


Figure 5.1 The process of the DSST.

There are two widely used scale estimations in the correlation filter tracking field: Scale Pool and DSST [63, 65]. Scale Pool has been discussed in section 3.1.2. DSST is an efficient method for estimating the target scale by training a scale estimation classifier on a scale pyramid. Scale Pool calculates the location of the object while estimating the object scale. However, DSST separates scale estimation and detection objects into two independent parts. The detection classifier and scale estimation classifier are trained, respectively. As shown in Fig 5.1, DSST extracts image patches of varying sizes (general 33 scales) based on the obtained object location from the detection process and produces the appearance model. The multiple feature models will be resized to the same size and transformed into multiple channel vectors. The obtained scale estimation classifier is combined to generate a confidence map; the maximum value of the confidence map corresponds to the optimal scale in the current frame. The advantages of DSST are that it can be introduced into any tracking framework, and the computation burden is small. Moreover, DSST meets the scale variation requirement of a large object. The disadvantage of DSST is that the precision of scale estimation is lower than Scale Pool. From the result of the experiment, scale pool method is suitable for image sequences where the size of the target is small. DSST is better suited to deal with a large object.

In recent years, different tracking evaluation benchmark databases have been proposed. The scale variation does significant works in the tracking. Most trackers obtain the outperformance on one tracking database and poorly on another. In this work, we take full advantage of Scale Pool and DSST to estimate scale. The proposed tracker is better applied to different databases. OTB and UAV benchmarks are different tracking databases. The size and aspect ratio of the target object in an image sequence of the OTB benchmark does not change much. In contrast, the target object's size and aspect ratio in an image sequence of the UAV benchmark changes much. Since DSST performs scale estimation based on the obtained location, the precision of scale estimation relies on the accuracy of the location. Therefore, the big scale gap of DSST is beneficial to enhance the accuracy of scale estimation.

However, since scale pool is used in the process of locating, the bigger scale gap will affect the accuracy of the location. So, the scale gap of scale pool is set to small. The smaller scale gap meets the requirement of the small object's scale variation. Accurate scale estimation is more suitable for small target objects. Moreover, the body of humans (In general, the aspect ratio of humans is bigger than 2) is more complex in the tracking process. When DSST performs scale estimation for humans, it is more inclined to the larger direction, leading the model to contain more background information. In this case, scale pool is selected to estimate scale. Selecting the accurate scale estimation method according to the target size and aspect ratio in an image sequence will improve the algorithm's performance. As shown in Eq.5.8, the scale pool method is selected when the small object or the object with a large aspect ratio is tracked. Otherwise, select DSST to estimate scale.

$$Scale = \begin{cases} Scale\ Pool, & Width * Height \leq thr1, or \frac{Width}{Height} > thr2 \\ DSST, & else \end{cases} \tag{5.8}$$

Here, $Width$ and $Height$ are the width and the height of the target. $thr1$ and $thr2$ are the scale estimation selecting thresholds.

# 5.3 Experiments

The evaluation experiments of the proposed tracker are introduced in this section. Firstly, the evaluation benchmark database and analysis metric are introduced. Next, the comparison of different trackers to analyze the strength and weaknesses of the tracker. In addition to OTB and TC128 benchmarks, UAV123 [71] is introduced to evaluate the trackers.

## 5.3.1 Parameter Setting

All the analysis experiments are conducted on the Matlab2019b platform and a PC machine with an Intel (R) Core (TM) i7-9700F CPU (3.00GHZ), 16GB memory. The shape of the search region is defined as square, the size = $\sqrt{5WH}$, ($W$ is the object's width, $H$ is the height of the object). The cell size of the HOG features is set to 4. The regularization parameters μ, $u$ and γ are set to 14, 0.1 and 0.01. The scaling step of Scale Pool and DSST are set to 1.01, 1.03, respectively.

Table 5.1 Success score of BASTR on OTB-2013 with the scale method selecting thresholds $thr1$ and $thr2$. The best result is shown in red font.

| | $thr1 = 400$ | $thr1 = 500$ | $thr1 = 600$ | $thr1 = 700$ |
|---|---|---|---|---|
| $thr2 = 2$ | 0.690 | 0.692 | 0.693 | 0.694 |
| $thr2 = 2.5$ | 0.696 | 0.698 | 0.701 | 0.700 |
| $thr2 = 3$ | 0.696 | 0.698 | 0.700 | 0.700 |
| $thr2 = 3.5$ | 0.695 | 0.697 | 0.699 | 0.699 |

Table 5.1 illustrates the impact of the scale method selecting thresholds $thr1$ and $thr2$. Overall, the value of $thr1$ and $thr2$ do not affect the performance of the tracker significantly. When $thr1$=600, $thr2$=2.5, BASTR achieves outstanding performance.

### 5.3.2 UAV123 Benchmark Databases

Figure 5.2 The image sequence of UAV123 dataset.

UAV benchmark database is an aerial video dataset and benchmark for low altitude UAV target tracking. UAV123 dataset is the second-largest object tracking that contains 123 video sequences and more than 110K frames. The total number of frames of OTB and TC is only around 90K. UAV123_10fps (All image sequences are recorded at frame rate 10 FPS) datasets is the subset of the UAV123. All image sequences are recorded at the frame rate 10FPS. UAV123_10fps is more challenging because the amplitude of the target object motion in two adjacent frames is large, closer to the real scenario. In this thesis, we select UAV123_10fps to evaluate our tracker. Twelve attributes are used to annotate each sequence, such as ARC (Aspect Ratio Change), BC (Background Clutter), CM (Camera Motion), FM (Fast Motion), FOC (Full Occlusion), IV (Illumination Variation), LR (Low Resolution), OV (Out-of-View), POC (Partial Occlusion), SOB (Similar Object), SV (Scale Variation), VC (Viewpoint Change).

The evaluation metrics still adopt precision scores, success scores, and FPS. The detailed introduction about these evaluation metrics is shown in section 3.4.1.

### 5.3.3 Analysis of OTB Benchmark Database

In this section, we offer comprehensive assessments to evaluate the performance of the proposed BASTR on the OTB-2013, OTB-2015 benchmark databases. The compared tracking methods include STRCF [31], SRDCF [66], BACF [38], CFHA [98], AutoTrack [95], KCF [47], SAMF [63], DSST [65], Staple [57], CSK [46], ASRCF [27] (with Hand-crafted feature).

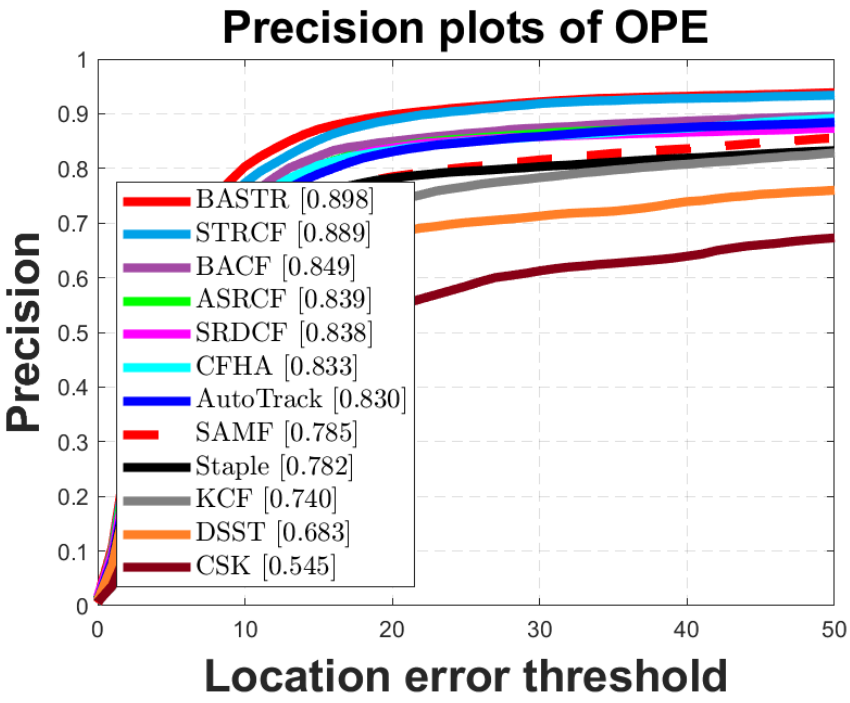


Figure 5.3 Comparisons with state-of-the-art DCF trackers on OTB-2013 benchmark in terms of precision plot.

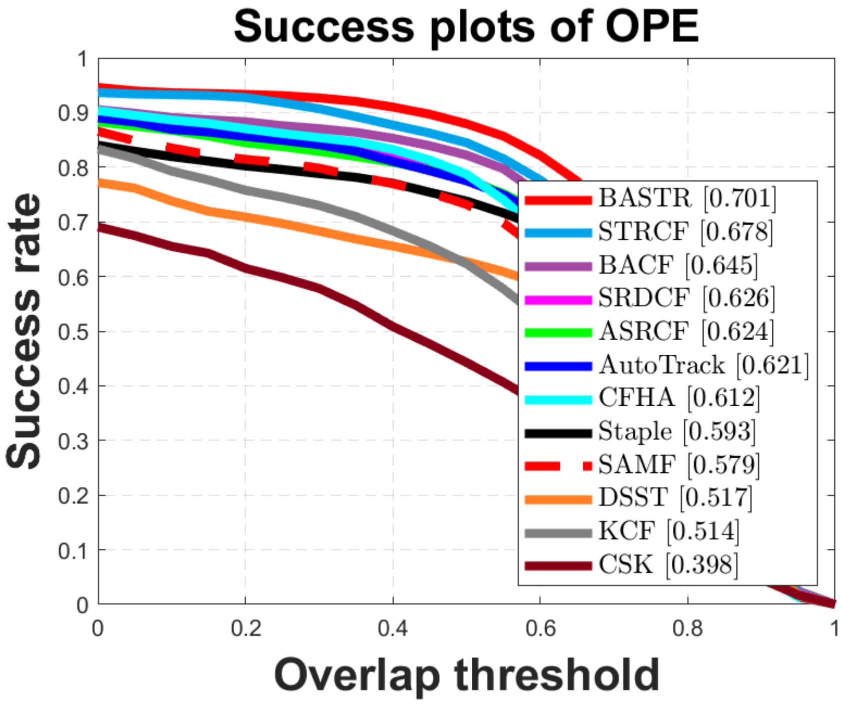


Figure 5.4 Comparisons with state-of-the-art DCF trackers on OTB-2013 benchmark in terms of success plot.

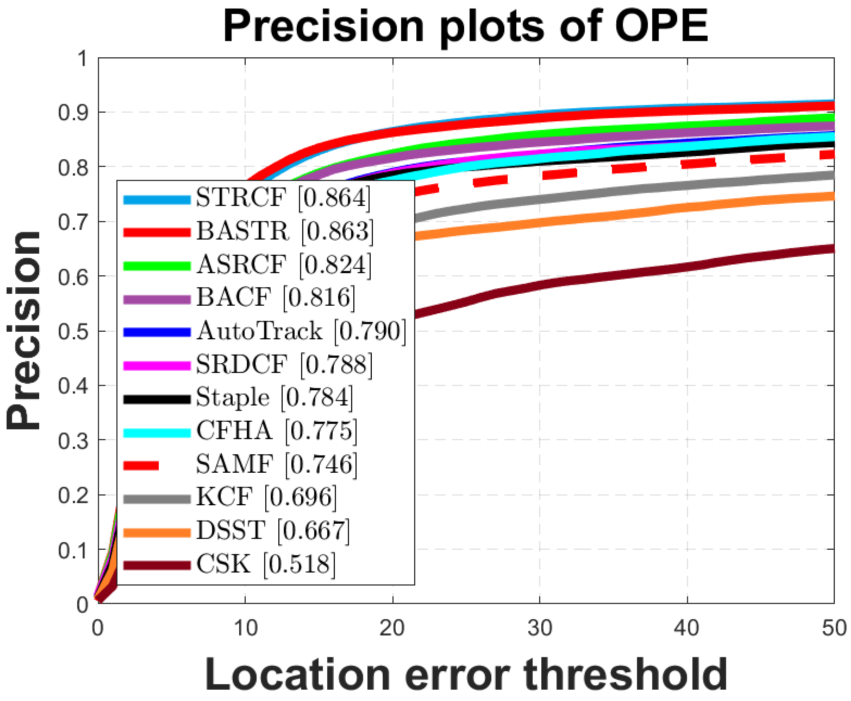


Figure 5.5 Comparisons with state-of-the-art DCF trackers on OTB-2015 benchmark in terms of precision plot.

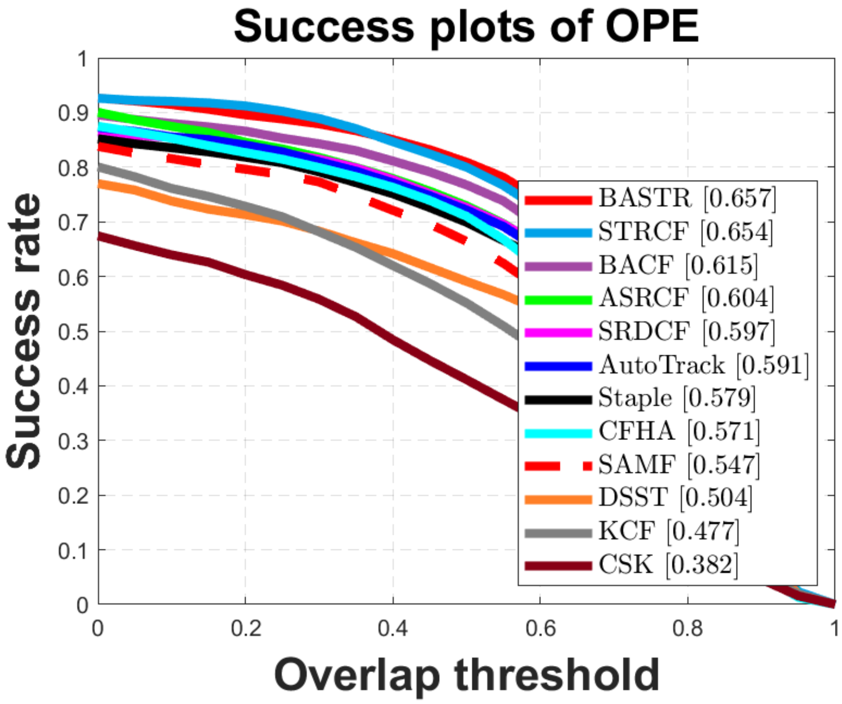
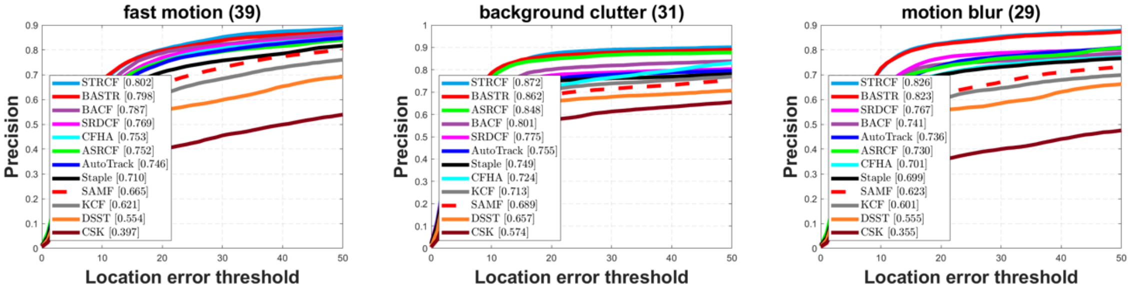


Figure 5.6 Comparisons with state-of-the-art DCF trackers on OTB-2015 benchmark in terms of success plot.

As shown in Fig 5.3 and Fig 5.4, BASTR achieves the excellent performance on OTB-2013, obtains a precision score of 89.8% and a success score of 70.1%, respectively. Compared to the baseline BACF, BASTR gets a gain of 4.9% and 5.6% on the precision and success scores, respectively. On OTB-2015 (Fig 5.5 and Fig 5.6), BASTR achieves a precision score of 86.3% and a success score of 65.7%, respectively. With the increase of the image sequences, the experimental data decreased. However, BASTR improvement to BACF is unchanged, and it is 4.8% and 4.2%.



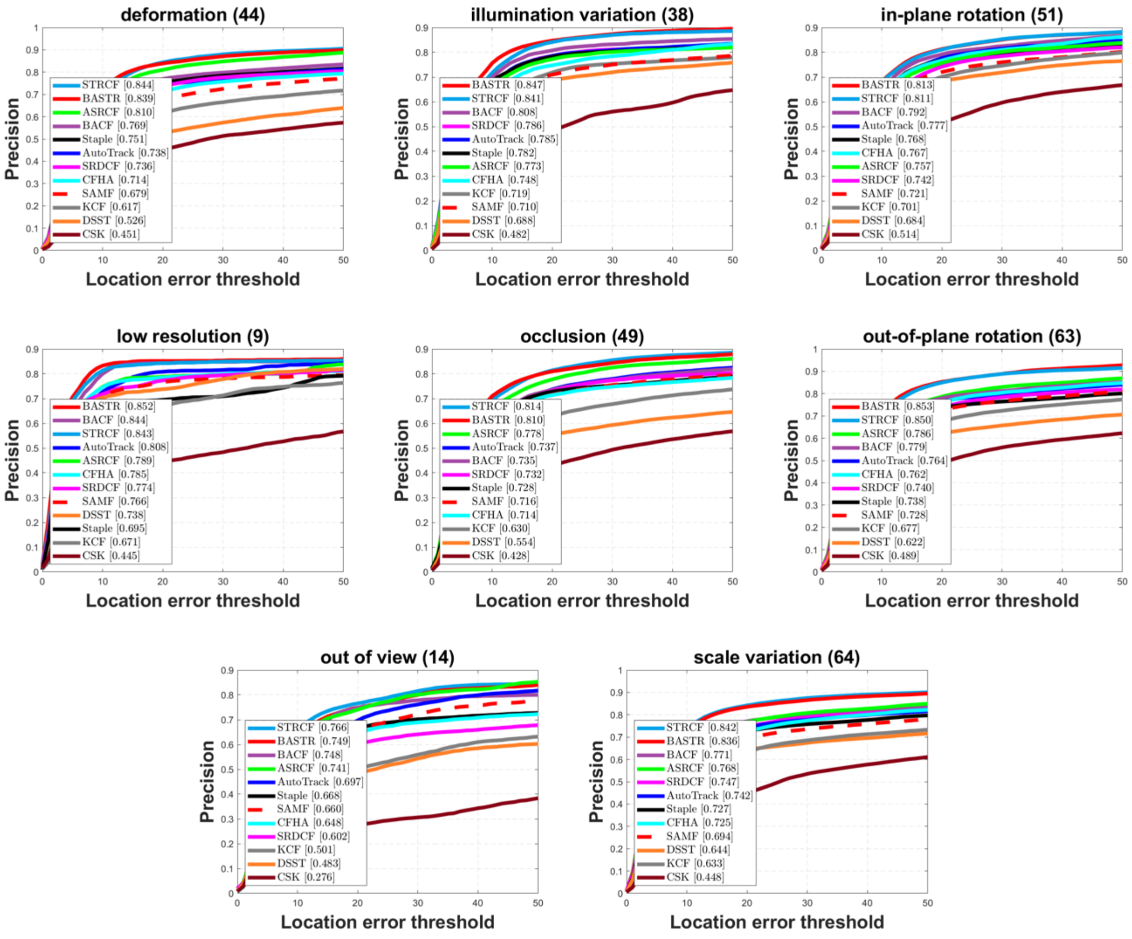
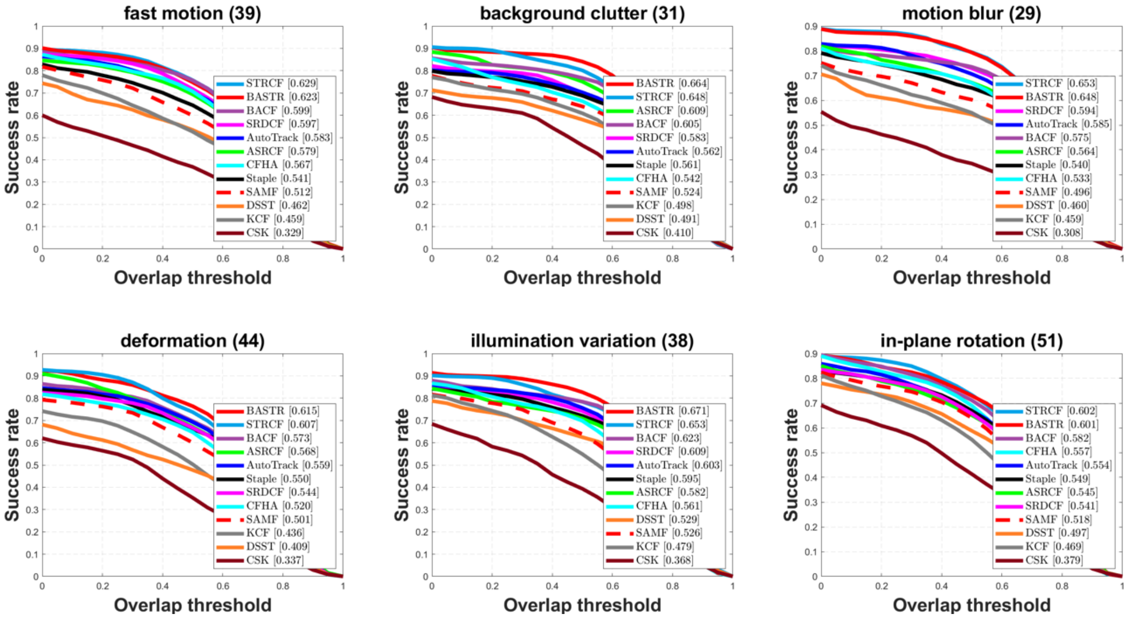


Figure 5.7 Attribute-based analysis of different trackers on the OTB-2015 dataset with 100 videos. The precision plots are shown for eleven attributes.



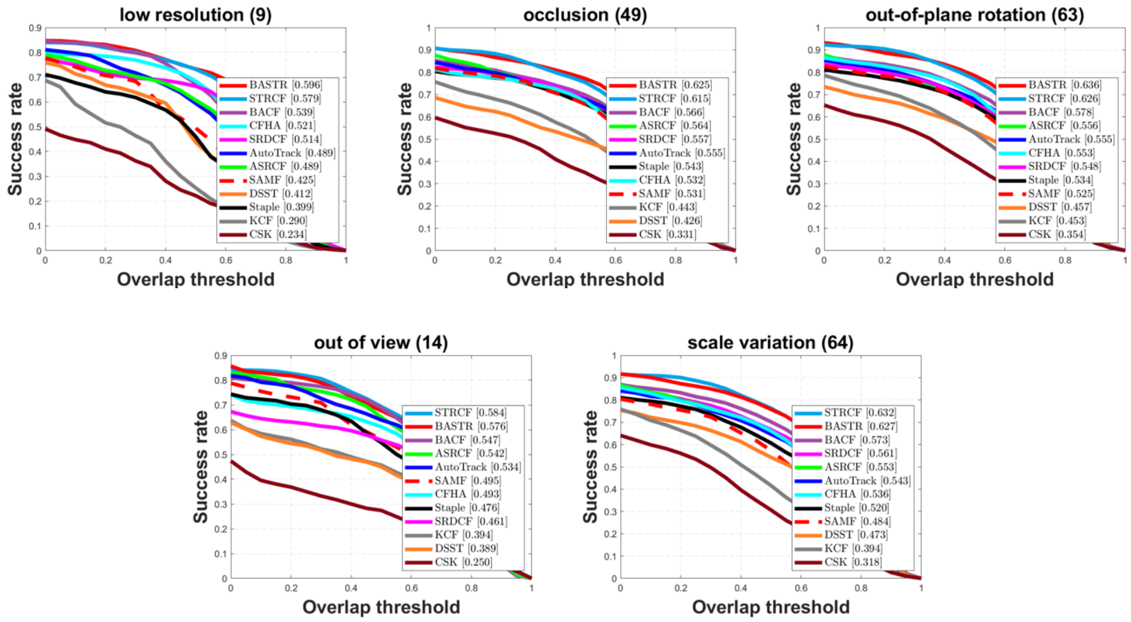


Figure 5.8 Attribute-based analysis of different trackers on the OTB-2015 dataset with 100 videos. The success plots are shown for eleven attributes.

The evaluation of the different DCF trackers by all attributes of OTB-2015. As shown in Fig 5.7, compared to BACF, BASTR obtains the greater improvement of 1.1%, 6.1%, 8.2%, 7.0%, 3.9%, 2.1%, 0.8%, 7.5%, 7.4%, 0.5%, 6.5% on the 11 attributes (FM, BC, MB, DEF, ILL, IPR, LR, OCC, OPR, OV, and SV), respectively. On FM, LR, and OV, the obtained improvement of BASTR is poor, however, it gets the significant improvement on the other attributes, especially BASTR obtains bigger than 6.0% on BC, MB, DEF, OCC, OPR, and SV. This experiment results illustrate the validity of the accuracy estimation about the object position.

Fig 5.8 shows the success scores of the evaluated trackers on the 11 attributes. Compared to BACF, BASTR obtains a gain of 2.4%, 5.9%, 7.3%, 4.2%, 4.8%, 1.9%, 5.7%, 5.9%, 5.8%, 2.9%, 5.4% on the 11 attributes (FM, BC, MB, DEF, ILL, IPR, LR, OCC, OPR, OV, and SV), respectively. On the success scores, BASTR gets the improvement is not more pronounced than the precision rates. The improvement of BASTR gets a 2.0% or more increase on the most attributes. The scale variation of the OTB benchmark is slow, the precision of the proposed scale estimation is not very high. However, BASTR still achieves the best performance than other trackers.

Table 5.2 Success and speed of top-5 trackers on the OTB-2015. The best two results are shown in red and blue fonts, respectively.

| | SRDCF | BACF | ASRCF | STRCF | BASTR |
|---|---|---|---|---|---|
| Success | 0.597 | 0.615 | 0.604 | **0.654** | **0.657** |
| FPS | 10.4 | **45.0** | **40.6** | 33.3 | 38.1 |

The DCF trackers with top-5 success scores on the OTB-2015 are selected to analyze the performance of the trackers. As shown in Table 5.2, BASTR obtains the best performance, and achieves the real-time.

### 5.3.4 Analysis of TC128 Benchmark Database

Most modern trackers employ color information to train the location model. OTB benchmarks contain some of the grayscale image sequences. It is not enough for some state-of-the-art trackers with color features to get the best evaluation, so we conduct our method on Temple color 128 benchmarks with STRCF [31], MEEM [45], Struck [83], BACF [38], ASLA [91], KCF [47].

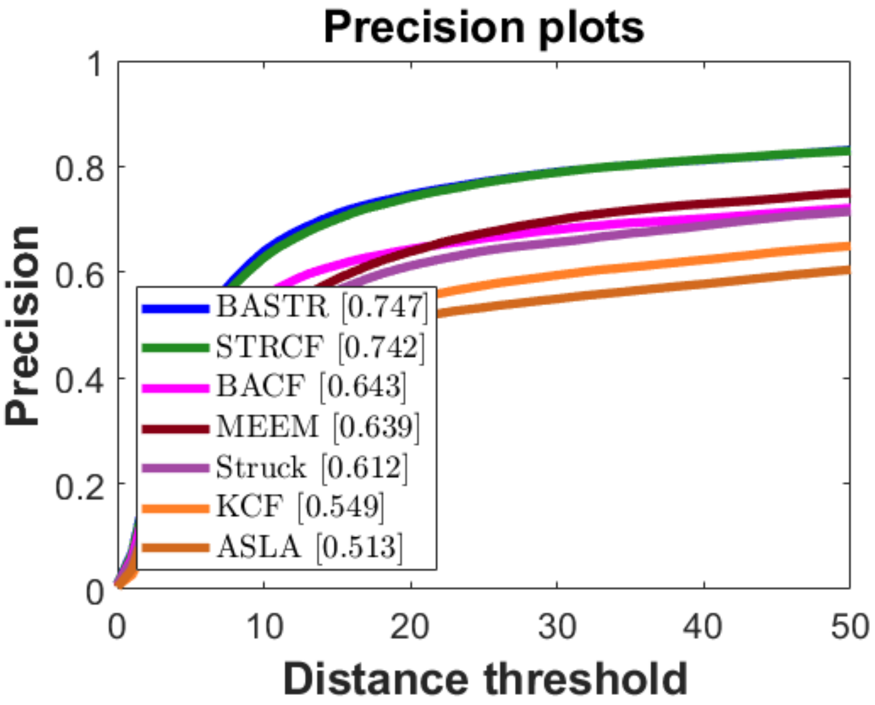


Figure 5.9 Comparisons with state-of-the-art DCF trackers on TC-128 dataset in terms of precision plot.

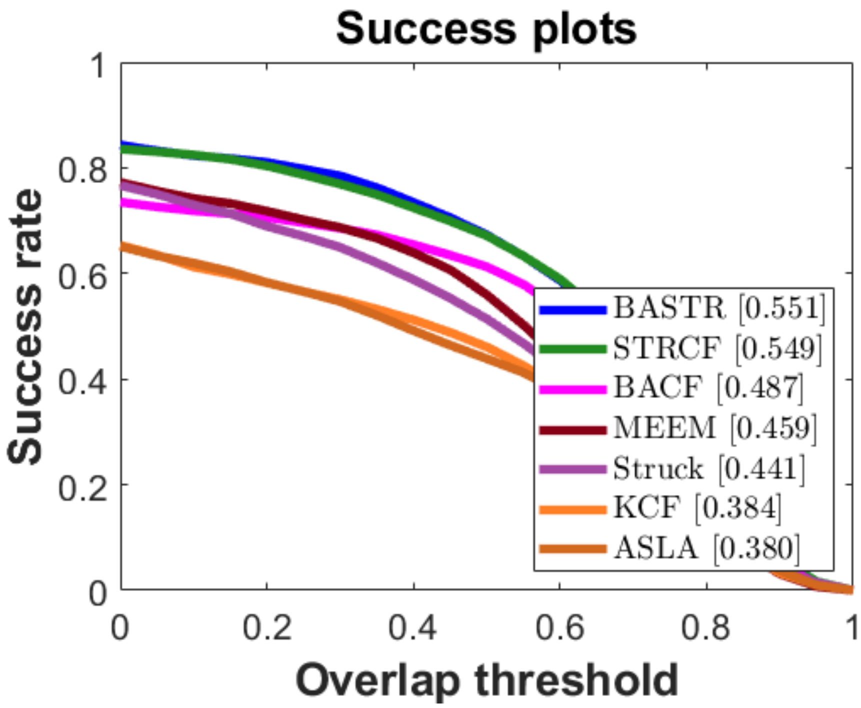


Figure 5.10 Comparisons with state-of-the-art DCF trackers on TC-128 benchmark in terms of success plot.

Our method obtains the outperformance both in precision plot and success plot on TC128. As shown in Fig 5.9 and Fig 5.10, our method achieves the best performance on the TC-128 benchmark. Compared to BACF, BASTR obtains a gain of 10.4% and 6.4% in precision scores and success scores, respectively.

### 5.3.5 Analysis of UAV123 Benchmark Database

In this section, we offer comprehensive assessments to evaluate the performance of the proposed BASTR on UAV123_10fps benchmark database. The compared tracking methods include STRCF [31], BACF [38], AutoTrack [95], KCF [47], SAMF [63], DSST [65].

As shown in Fig 5.2, some target objects are small in the initial frame. However, this does not mean the target remains the same scale in the rest of the image sequence. The objects of the UAV123 benchmark have scale variation significantly. Selecting an appropriate scale estimation method can improve the accuracy of the algorithm.

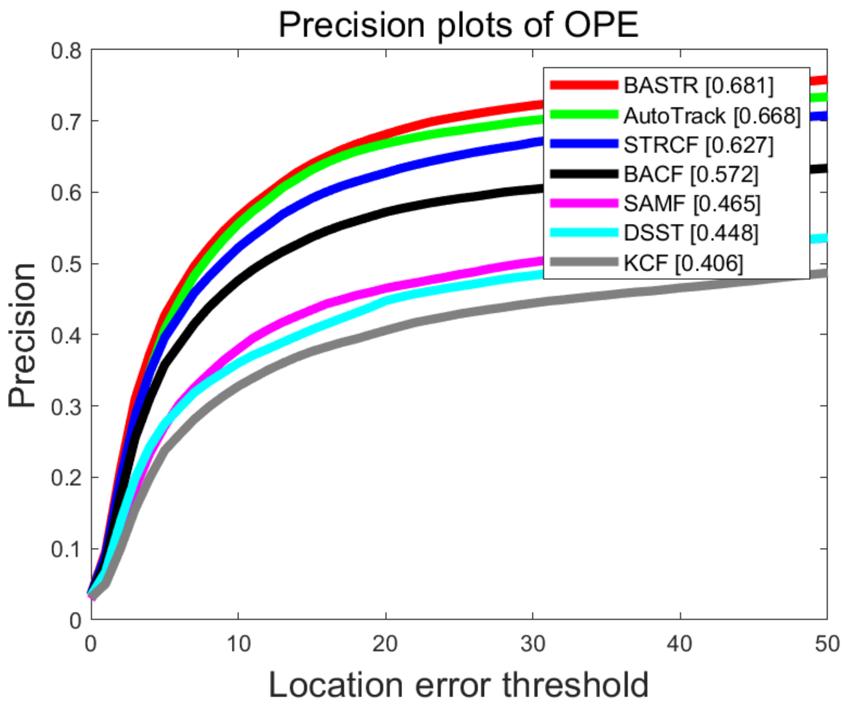


Figure 5.11 Comparisons with state-of-the-art DCF trackers on UAV123_10fps benchmark in terms of precision plot.

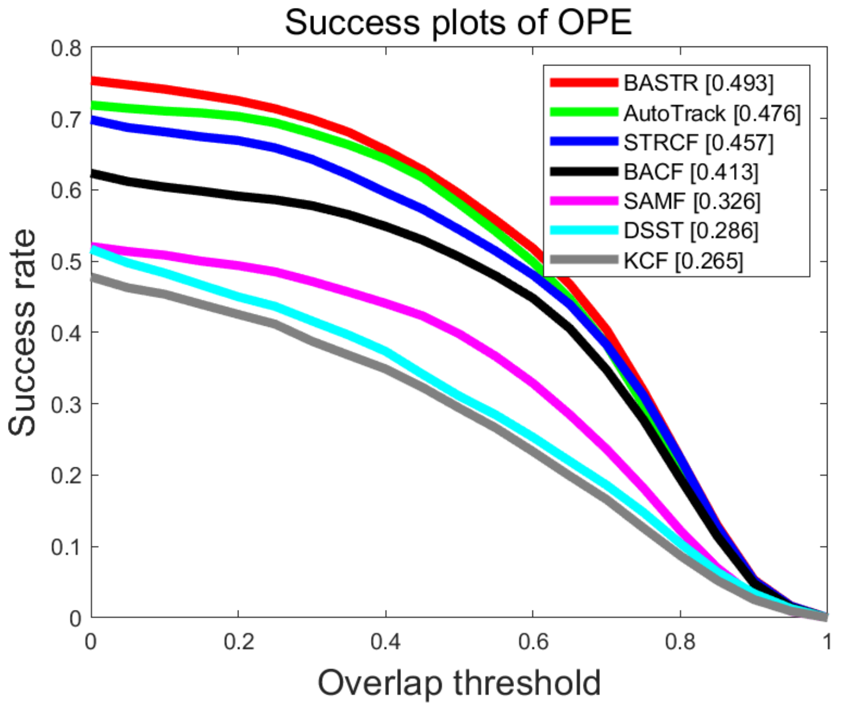


Figure 5.12 Comparisons with state-of-the-art DCF trackers on UAV123_10fps benchmark in terms of success plot.

As shown in Fig 5.11 and Fig 5.12, BASTR achieves the best performance on UAV123_10fps, obtains a precision score of 68.1% and a success score of 49.3%, respectively. Compared to the result of OTB, the precision scores and the success scores of all trackers are decreased. This also illustrates the difficulty of the UAV123 dataset. Since AutoTrack [95] is proposed to be applied in UAV localization, the performance is worse on the OTB benchmark. However, BASTR obtains better performance than AutoTrack of UVA123 datasets. STRCF and BASTR achieve nearly performance on the OTB benchmark, BASTR gets an improvement of 5.4% and 3.6% better than STRCF in precision score and success score on UAV123, respectively. The result demonstrates that the adaptive scale estimation makes the tracker suitable for different databases.

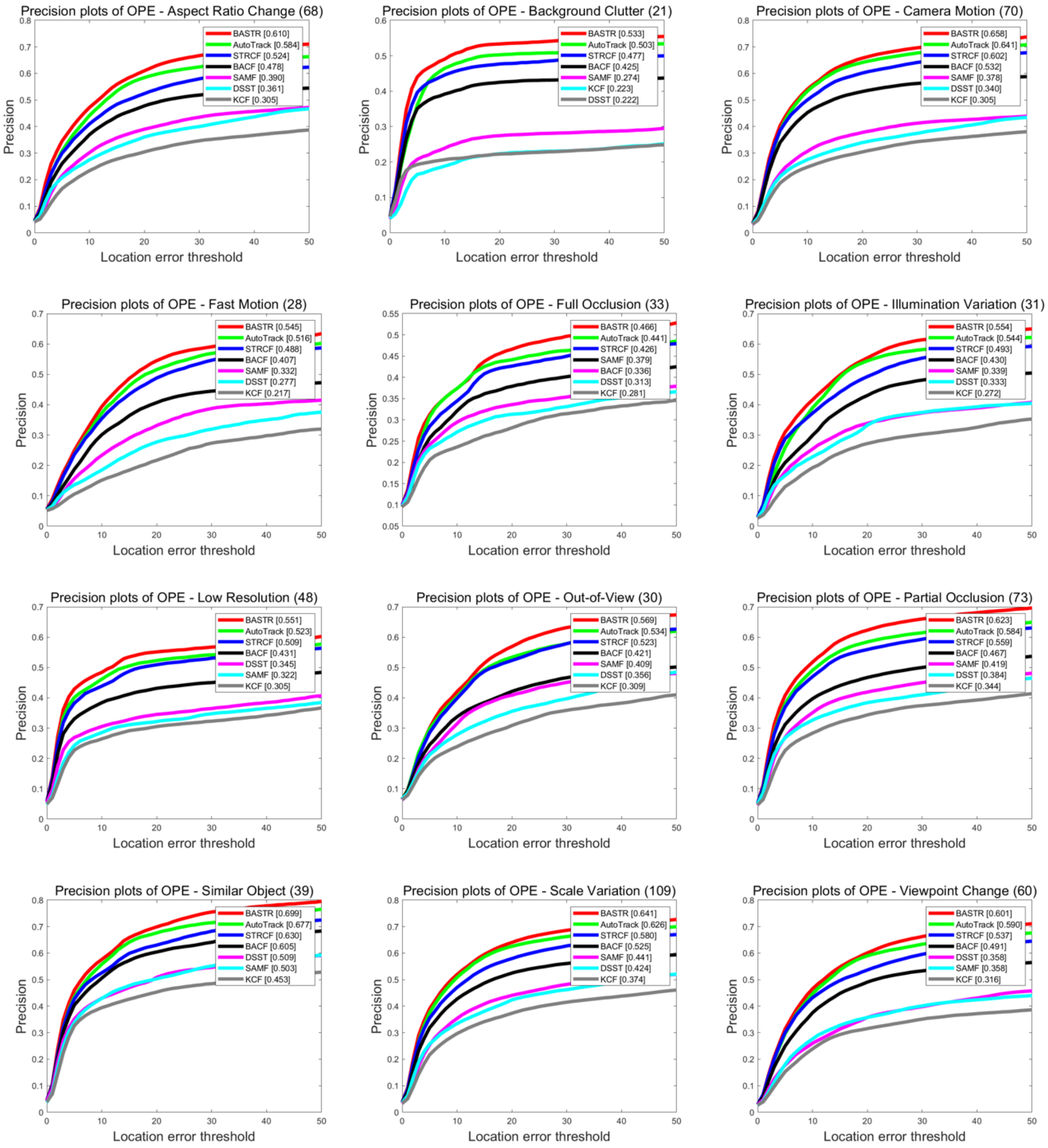


Figure 5.13 Attribute-based analysis of different trackers on the UAV123 dataset. The precision plots are shown for twelve attributes.

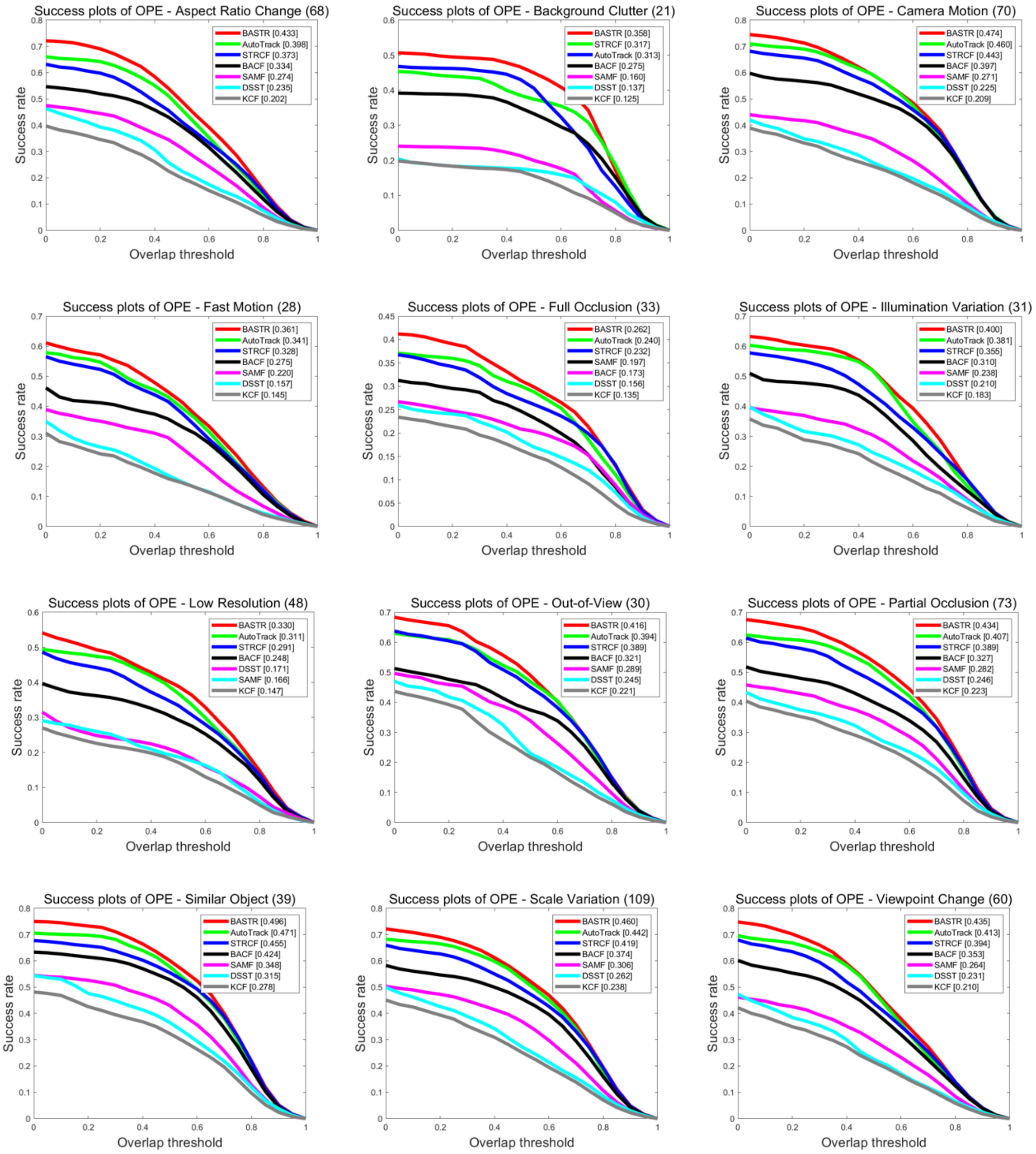


Figure 5.14 Attribute-based analysis of different trackers on the UAV123 dataset. The success plots are shown for twelve attributes.

Fig 5.13 and Fig 5.14 illustrate the evaluation of the different DCF trackers by all attributes of UAV123_10fps. STRCF and BASTR get almost the same tracking performance on the OTB benchmark and TC-128 benchmark. However, as shown in Fig 5.13, BASTR obtains a greater improvement on STRCF and gets bigger than 4.0% improvement on all attributes of UAV123 benchmark, especially BASTR obtains bigger than 6.0% on ARC, IV, POC, SOB, SV, and VC. Compared to baseline BACF, BASTR obtains a gain of 13.2%, 10.8%, 12.6%, 13.8%, 13.0%, 12.4%, 12.0%, 14.8%, 15.6%, 9.4%, 11.6%, 11.0% on the 12 attributes (ARC, BC, CM, FM, FOC, IV, LR, OV, POC, SOB, SV, and VC), respectively. This experiment results illustrate the validity of the accuracy estimation about the object position.

Fig 5.14 shows the success scores of the evaluated trackers on the 12 attributes. BASTR also gets better performance than STRCF. On the success scores, BASTR gets the improvement is not more pronounced than the precision scores. The improvement of BASTR gets a 4.0% or more increase on the most attributes. Compared to BACF, BASTR gets a great improvement of 9.9%, 8.3%, 7.7%, 8.6%, 8.9%, 9.0%, 8.2%, 9.5%, 10.7%, 7.2%, 8.6%, 8.2% on the 12 attributes (ARC, BC, CM, FM, FOC, IV, LR, OV, POC, SOB, SV, and VC), respectively.

# Chapter 6

# Experiments

This section combines the core of the three algorithms mentioned above and analyzes and discusses the algorithms in depth. Based on the results of the experiment, we elaborate on the connections and differences between the three algorithms.

In section 3.2, CFCA proposes a robust method to improve detection accuracy, combining the four maximum scores of the confidence map with the luminance histogram similarity. CFASE and BASTR adopt the same method as SRDCF to increase the location estimation accuracy. In section 6.1, I demonstrate the influence of location estimation by analyzing these two location estimation methods.

Scale is an essential factor for visual tracking algorithms. It is impossible for tracking objects to maintain the same scale in many scenarios. If trackers ignore scale variation in tracking, the performance of trackers cannot get robust performance. We attach great importance to scale estimation in the proposed three trackers. In section 6.2, we analyze the influence of scale estimation for visual tracking. The qualitative analysis experiments are conducted on different benchmark databases to evaluate the reliability of the proposed scale estimation methods.

In section 6.3, I demonstrate the connection among the mathematical model of the three proposed trackers. The three trackers are proposed based on discriminative correlation filters. With the development of research, the mathematical models of CFCA, CFASE, and BASTR become more and more complex, and the calculation burden becomes heavy. To comprehensively assess three trackers, we evaluate our trackers against the state-of-the-art trackers on different benchmark databases.

## 6.1 The influence of location

From chapters 3 to 5, we illustrate the details of CFCA, CFASE, and BASTR. CFCA is proposed based on the traditional CF tracker KCF. Like KCF, most CF trackers detect the target object's location by the maximum value of the confidence map, and this method produces a certain degree of errors. For example, when the target's appearance changes drastically or similar objects in the background, the confidence map fluctuates more, with multiple peaks. Therefore, it is not accurate to use only the maximum value position as the target's position. In addition to this, most CF trackers carry out the HOG feature to train samples and detect the target object. The search region will be narrowed down because the cell size of the HOG feature is set to 4. In other words, the training and detection samples are constructed using a grid strategy with a stride greater than one pixel. The accuracy of the location estimation is decreased because of these two reasons.

CFCA improves the detection accuracy by combining the four highest response scores of the confidence map and luminance histogram similarity. From the result of the experiments in chapter 3, the validity of this method is demonstrated. To address the negative influence of the HOG feature in detection, some DCF trackers [31, 66] obtain the optimization location using Newton’s method. This method is first proposed in SRDCF. Since the research region is narrowed down, the obtained confidence map also is narrowed down. SRDCF re-construct a pixel-dense confidence map by employing an interpolation approach and locating the best optimization location using Newton’s method. The computation burden of this process is low because only a few iterations are sufficient for convergence. CFASE and BASTR also adopts Newton’s method to detect the object's location.

This section mainly analysis the difference between the two methods. Firstly, we replace Newton’s method of CFASE with the proposed method of CFCA (called ASECA). CFASE adopts the maximum score to detect the object location instead of Newton’s method (called ASEN). Based on the OTB benchmark database, the evaluation experiments are conducted between CFCA, ASECA, CFASE, and ASEN. The remaining parts of the tracker do not change, including parameters.

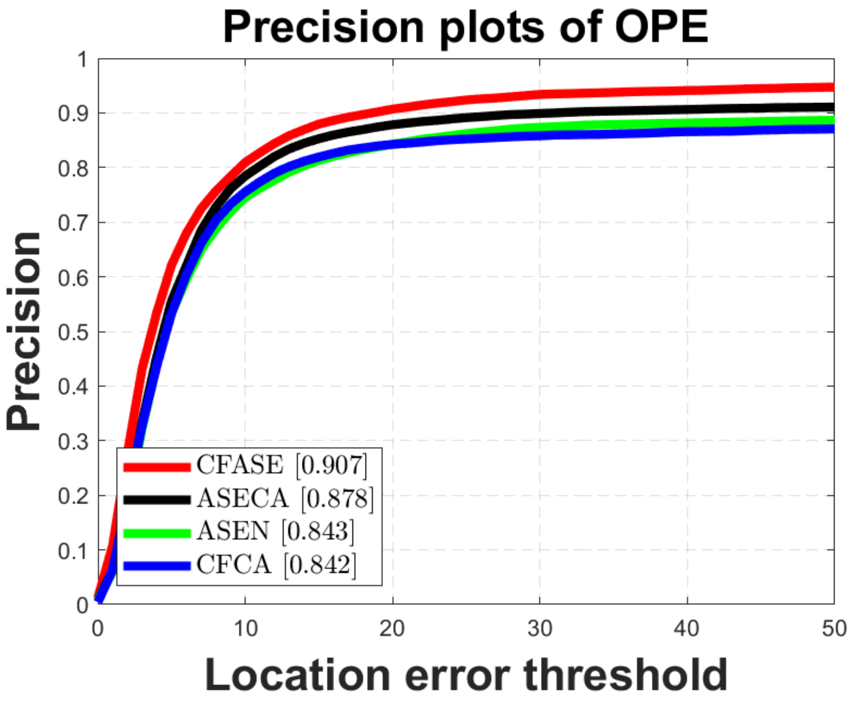


Figure 6.1 Comparisons with state-of-the-art DCF trackers on OTB-2013 benchmark in terms of precision plot.

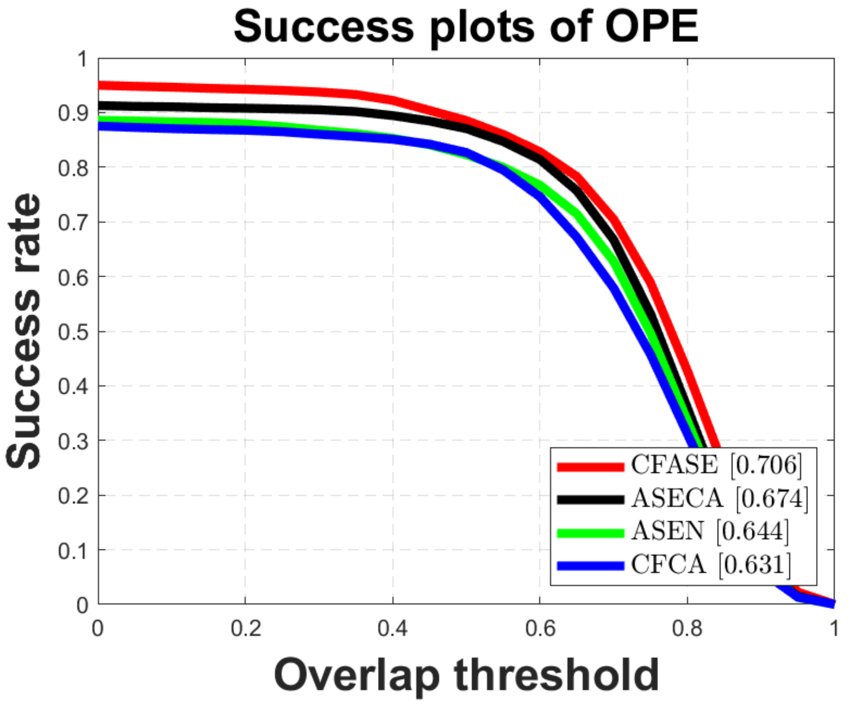


Figure 6.2 Comparisons with state-of-the-art DCF trackers on OTB-2013 benchmark in terms of success plot.

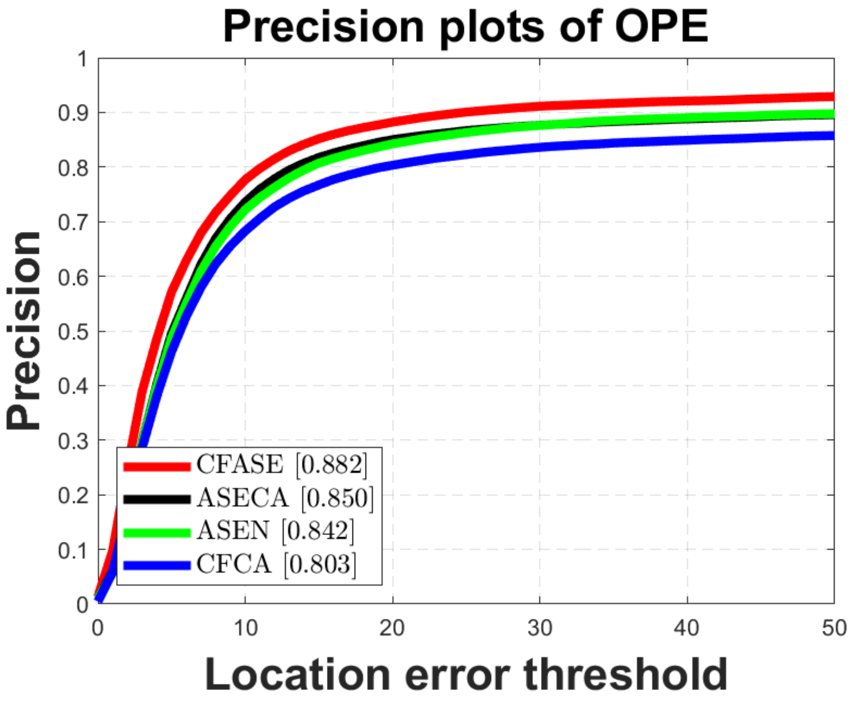


Figure 6.3 Comparisons with state-of-the-art DCF trackers on OTB-2015 benchmark in terms of precision plot.

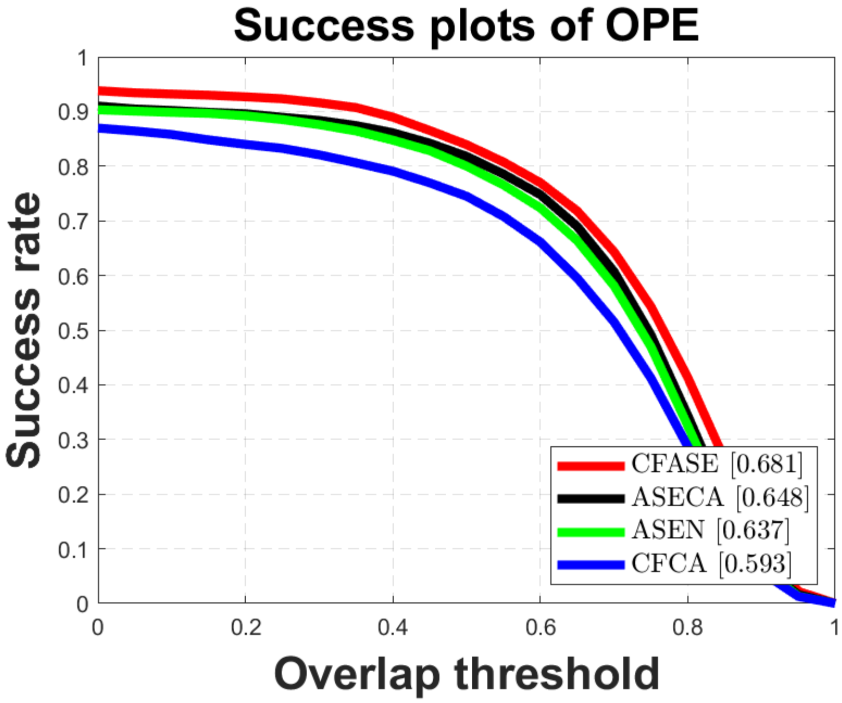


Figure 6.4 Comparisons with state-of-the-art DCF trackers on OTB-2015 benchmark in terms of success plot.

The precision and success plots of the four trackers (CFCA, ASECA, CFASE, and ASEN) are shown in Fig 6.1 and Fig 6.2. CFASE obtains the highest scores on both the precision plot and the success plot. If CFASE does not adopt Newton's method to improve the accuracy of the location estimation, instead, directly use the maximum score of the confidence map to detect the location (ASEN). The performance of CFASE has a significant loss, falling 6.4% in precision scores and 6.2% in success rate on OTB-2013 benchmark database. This result illustrates the importance of location estimation accuracy. The location estimation of CFCA is introduced into ASEN (ASECA), the performance of the tracker is improved, obtains the improvement of 3.5% and 3.0% in the precision score and the success scores, respectively. The location estimation of CFCA is further verified.

From the result of OTB-2015 (see Fig 6.3 and Fig 6.4), compared to ASEN, CFASE achieves a gain of 4.0% and 4.4% on precision scores and success scores, respectively. ASECA only gets an improvement of 0.8% and 1.1% on precision scores and success scores, respectively. The location estimation of CFASE is more robust than CFCA. The main reason is that CFCA combines the luminance histogram similarity and the confidence map to improve the performance of the tracker. When background clutter or illumination variation occurs, the negative influence from the luminance histogram becomes more.

CFCA is proposed based on the traditional CF tracker KCF, and CFASE is proposed based on

STRCF. Although both KCF and STRCF belong to the correlation filter tracker, some differences exist between the two frameworks. For feature extraction, CFCA only uses the HOG feature to train the CF model. CFASE trains the DCF model with hand-crafted features (HOG feature + CN feature + Gray feature). To better evaluate our tracker, we extract all gray image sequences (26 image sequences) from the OTB-2015 benchmark database.

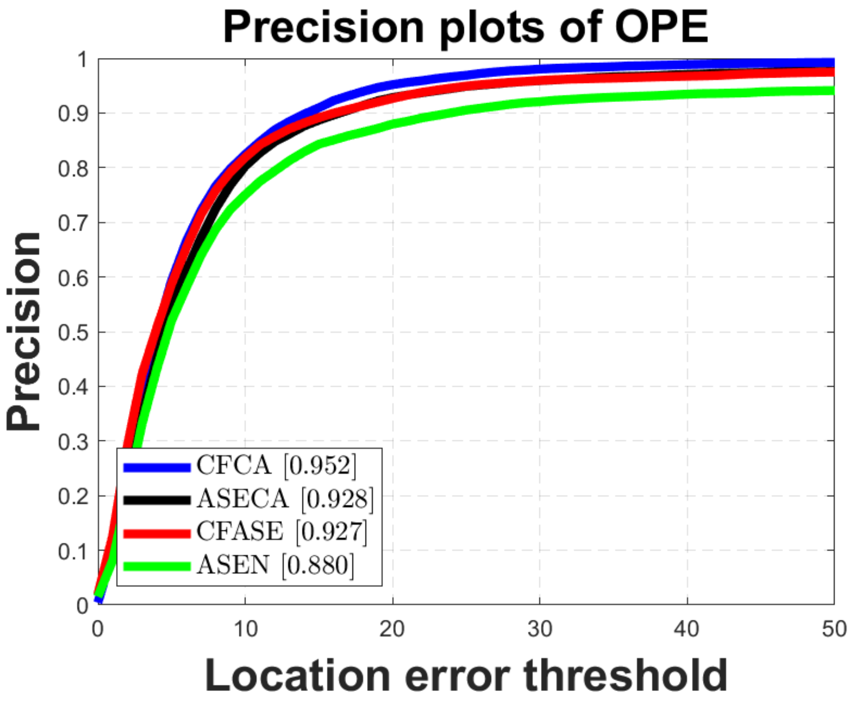


Figure 6.5 Precision plot of different DCF trackers on Gray image sequences.

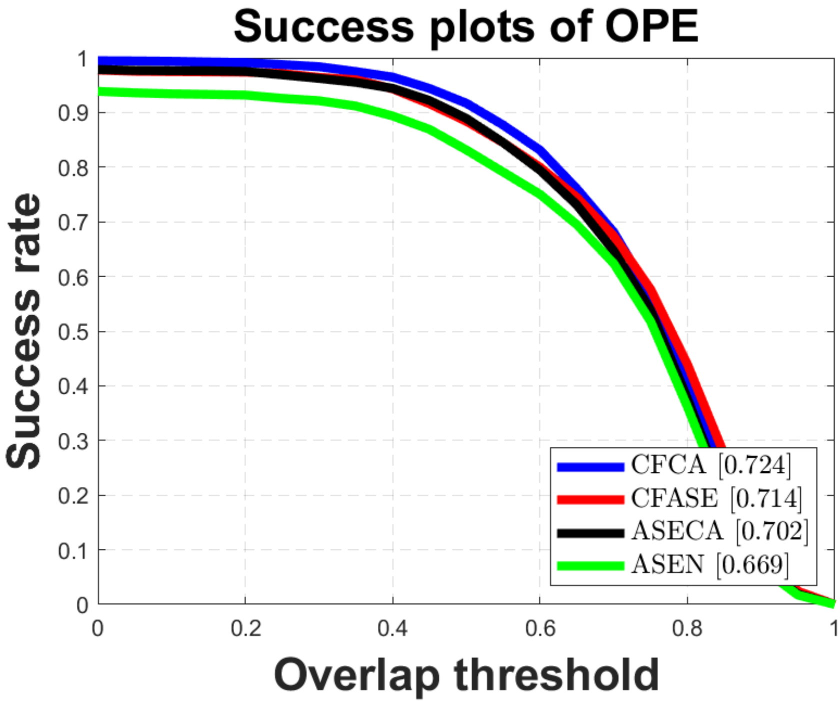


Figure 6.6 Precision plot of different DCF trackers on Gray image sequences.

Fig 6.5 and Fig 6.6 show the precision scores and the success scores of the gray image sequences. In contrast to the OTB-2013 database and the OTB-2015 database, CFCA unexpectedly achieves the best performance. The precision score of CFCA is as high as 95.2%, and the success rate is 72.4%. Compared to CFASE, CFCA obtains a gain of 2.5% and 1.0% on the precision score and the success score, respectively. This result shows the robustness of CFCA in tracking on the low-dimensional feature. ASECA obtains a precision score similar to CFASE and achieves a gain of 4.8% compared to ASEN. From the success score point of view, CFASE still gets better performance than ASECA. However, ASECA obtains an improvement of 3.3% on ASEN. Based on the same DCF framework, our proposed location estimation method can get a robust performance in low-dimensional features.

## 6.2 The importance of scale estimation

In this section, I demonstrate the influence of scale estimation by conducting evaluation experiments on databases. So far, many scale estimation methods have been proposed in visual tracking. In prior traditional trackers, scale-invariant key points [6, 13, 39] are used to estimate the scale variation, such as the Harris Corner Detector, scale-invariant feature transform (SIFT) [64], and speeded up robust features (SURF) [13]. However, the computation burden of key points-based scale estimation is severe. Scale pool uses the scale pyramid principle [29] to solve the scale variation of correlation filtering algorithms. Although the inaccurate scale estimation will affect the performance of the tracking detector, the multi-scale search can naturally promote each other with the detector positioning. So the mutual combination is relatively stable, and it is not easy to cause the scale estimation offset to be too large unless the detector completely takes the target. The computation burden of the scale pool depends on the positioning cost of the detector, N-1 times the positioning cost of the detector (N is the number of pyramid levels). Scale pool is a widely adopted scale estimation method in correlation filter tracking. DSST is a novel and innovative scale estimation method. Since it estimates the object's scale based on the obtained location, DSST requires high location accuracy of the detector. DSST is the best choice for low frame-rate tracking. Besides, there is a scale estimation method (bboxRR) [34] that be used in deep learning tracking. As the same as DSST, bboxRR estimates the scale based on the obtained location. However, it is much more robust to target changes than DSST, and the accuracy requirements of the detector are not as high as DSST. Since there is no good update strategy, few people study bboxRR in target tracking for the time being. We adopt scale pool and DSST to adaptive estimate scale in tracking. Since the three proposed trackers adopt different frameworks to train the CF models, the scale estimation methods cannot be introduced into each other. However, the scale selection of BASTR can be introduced into CFASE. To evaluate the influence of the scale estimation method, we conduct the analysis experiments with eight trackers, such as CFCA, CFCANS (CFCA without scale estimation), CFASE, CFASENS (CFASE without scale estimation), CFASESE (CFASE with scale election), BASTR, BASTRNS (BASTR without scale estimation), BASTRS (BASTR with scale pool).

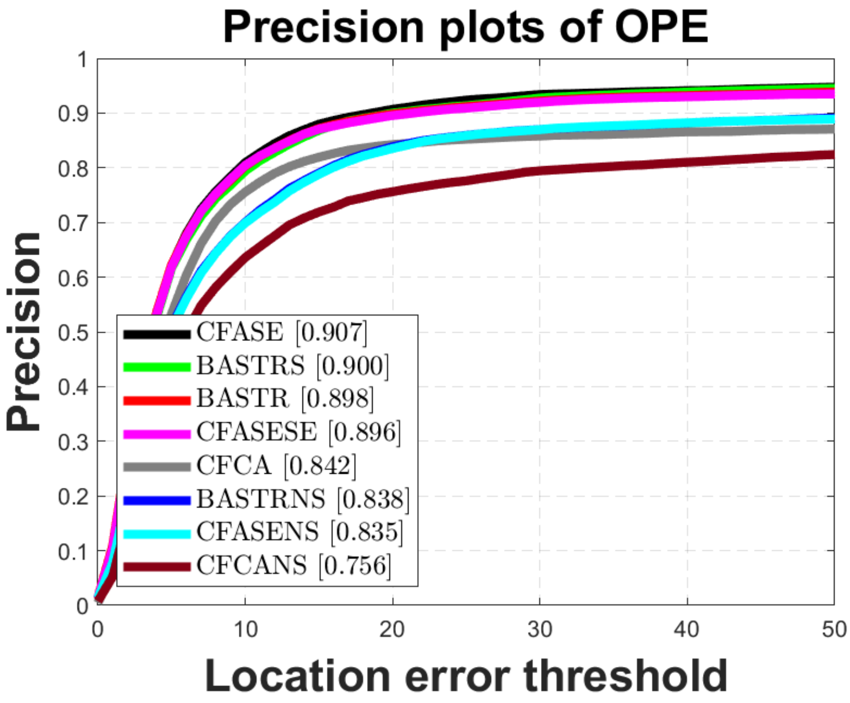


Figure 6.7 Precision plot of different DCF trackers on OTB-2013.

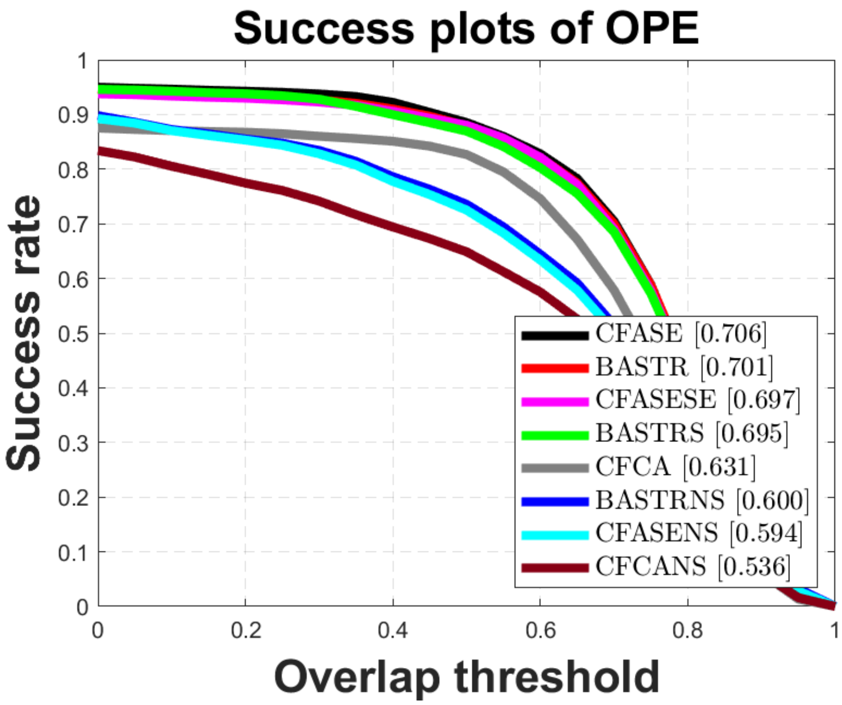


Figure 6.8 Success plot of different DCF trackers on OTB-2013.

Fig 6.7 and Fig 6.8 illustrate the precision and success scores of the proposed trackers on the OTB-2013 benchmark. CFASE gets the best performance on the OTB-2013 benchmark. However, if CFASE does not adopt the scale estimation to solve the scale variation, the performance dramatically decreases. Compared to CFASE, CFASENS achieves a decrease of 7.2% and 11.2% on precision score and success score, respectively. When CFASE adopts the same scale selection method as BASTR to estimate scale, the performance of CFASESE has decreased. However, BASTR adopts the scale pool to estimate scale, BASTRS keep the same performance as BASTR on the precision score. BASTR gets better performance than BASTRS on the success score. The performance of BASTRNS also obtains a worse performance than BASTR, decreasing 6.0% and 10.1% on precision score and success score, respectively. In six trackers, CFCANS achieves the worst result. Compared to CFCANS, CFCA gets a gain of 8.6% and 9.5% on precision and success score, respectively. From the results of experiments, scale estimation is a significant factor for trackers. Even if CFASE and BASTR adopt robust frameworks, the performance of CFASE and BASTR without scale estimation is worse than CFCA.

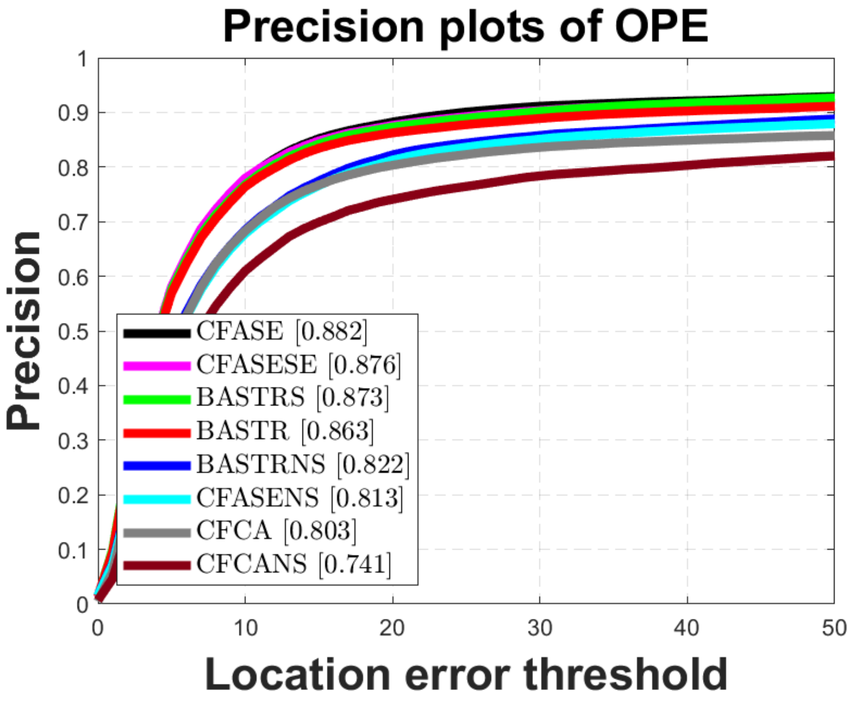


Figure 6.9 Precision plot of different DCF trackers on OTB-2015.

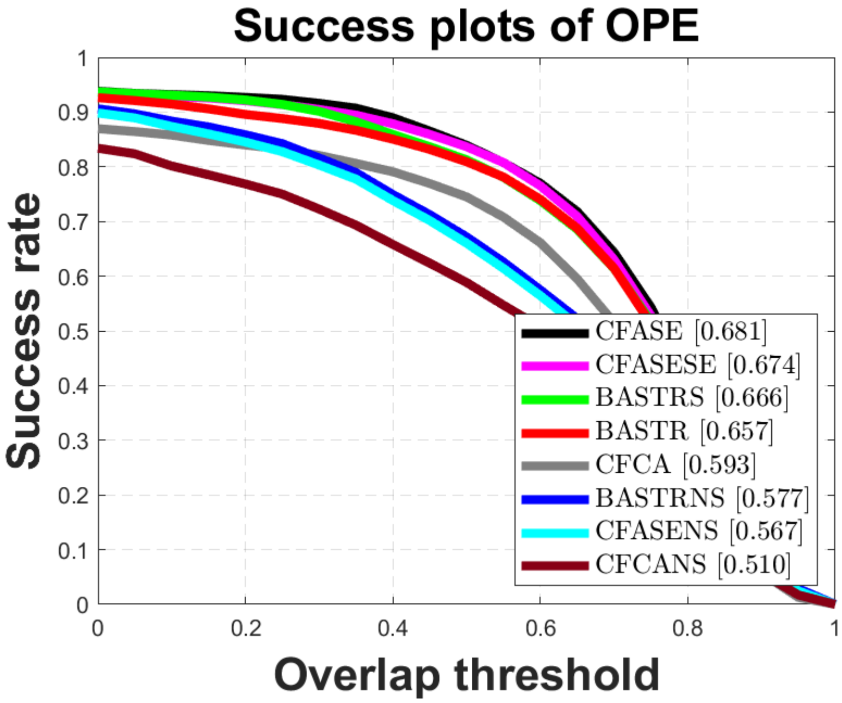


Figure 6.10 Success plot of different DCF trackers on OTB-2015.

Fig 6.9 and Fig 6.10 illustrate the proposed trackers' precision and success scores on the OTB-2015 benchmark. CFASE still gets the best performance on the OTB-2015 benchmark. The gap between CFASESE and CFASE is closing. This result denotes the scale selection of BASTR can also achieve great performance in the high FPS video. However, since CFASENS does not adopt the scale estimation to solve the scale variation, compared to CFASE, CFASENS achieves a decrease of 6.9% and 11.4% on precision and success scores, respectively. The negative influence of no scale estimation is not improved with the database increasing. The performance of BASTRNS still obtains a worse performance than BASTR, decreasing 4.1% and 8.0% on precision score and success score, respectively. However, BASTRS obtains better performance than BASTR, achieving a 1.0% increase on precision score and success score, respectively. Finally, CFCANS achieves the worst result. Compared to CFCANS, CFCA gets a gain of 6.2% and 8.3% on precision and success score, respectively. Although CFASENS and BASTRNS perform better than CFCA on the precision score, the success score is worse than CFCA. Scale estimation is more critical for success score.

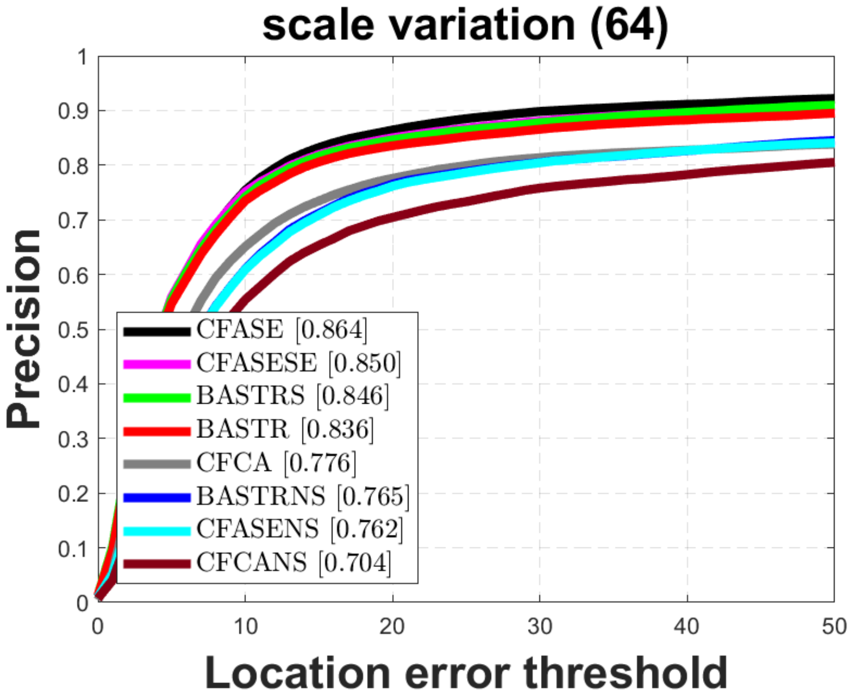


Figure 6.11 Precision plot of different DCF trackers on scale variation attributes on OTB-2015.

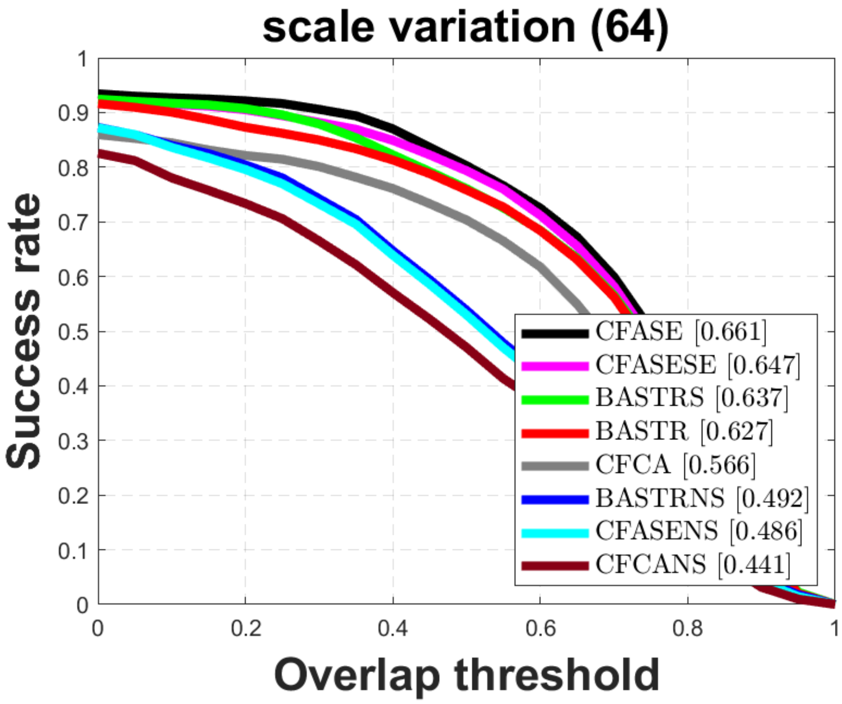


Figure 6.12 Success plot of different DCF trackers on scale variation attributes on OTB-2015.

We demonstrate the experiment results of the scale variation attributes on the OTB-2015 benchmark database (As Fig 6.11, Fig 6.12). For 64 image sequences of the OTB-2015 benchmark with scale variation, trackers which use the scale estimation method obtain better performance than trackers without scale estimation. Compared to CFASENS, BASTRNS, and CFCANS, CFASE, BASTR, and CFCA get an improvement of 10.2%, 7.1%, and 7.2% on the precision score, respectively, 17.5%, 13.5%, and 12.5% on the success score, respectively. For different trackers, scale estimation can significantly improve performance.

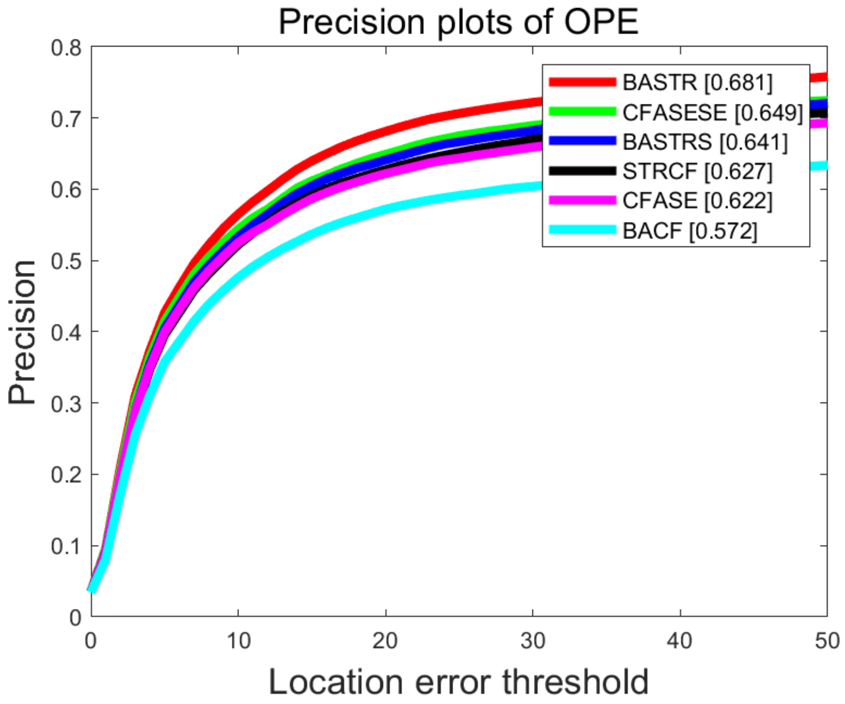


Figure 6.13 Precision plot of different DCF trackers on UAV123_10fps.

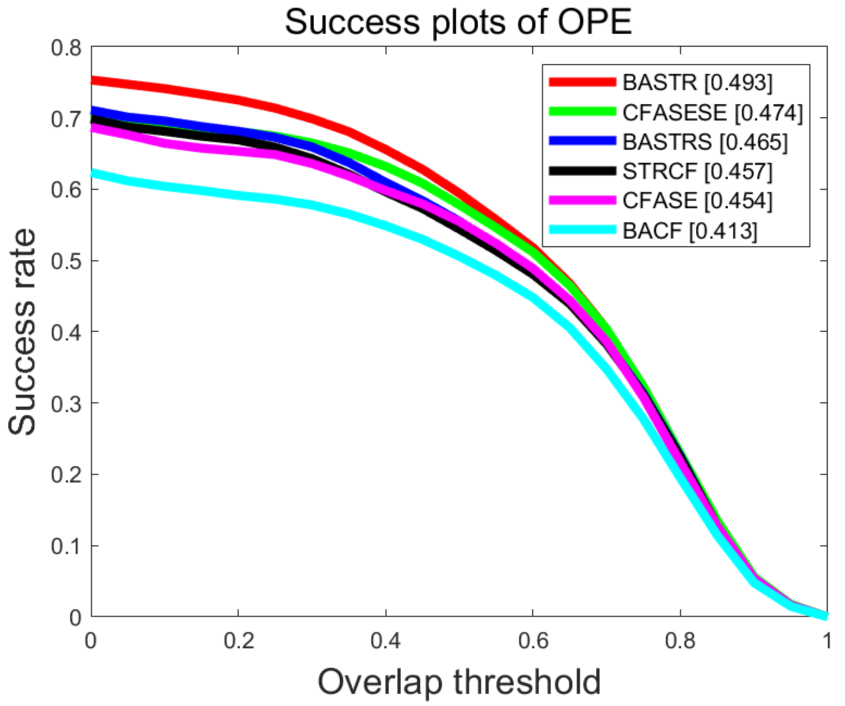


Figure 6.14 Success plot of different DCF trackers on UAV123_10fps.

Fig 6.13 and Fig 6.14 illustrate the proposed trackers' precision and success scores on the UAV-123 benchmark. In contrast to the OTB benchmark, BASTR gets the best performance on the UAV-123 benchmark. As shown in the OTB benchmark database, BASTRS achieves better than BASTR. On the UAV benchmark database, BASTR obtains better performance. Compared to BASTRS, BASTR gains 4.0% and 2.8% on the precision and success scores, respectively. Although the performance of

CFASESE is imperfect CFASE on the OTB benchmark, CFASESE achieves better performance than CFASE on the UAV benchmark. This result also illustrates the validity of BASTR's scale estimation method.

Moreover, CFASESE achieves better performance than BASTR on the OTB benchmark. However, if the temporal regularization parameters and the learning rate of CFASESE are adjusted on the UAV benchmark, CFASESE also obtains better performance. When $\mu1 = 13$, $\alpha1 = 0.3$; $\mu2 = 9$, $\alpha2 = 0.5$, CFASESE can achieve 67.8% and 49.9% on precision score and success score, respectively. In this case, CFASESE gets better performance than BASTR. BASTR obtains significant performance on the OTB and UAV benchmarks without parameters adjustment. Overall, BASTR's performance will be more consistent. UAV benchmark contains more complex scenarios and rigid targets than the OTB benchmark. The adaptive spatial regularization is suitable to improve the location precision. Therefore, the performance of BASTR achieves the best precision score on the UAV benchmark. Fig 6.13 and Fig 6.14 illustrate the proposed trackers' precision and success scores on the UAV-123 benchmark. In contrast to the OTB benchmark, BASTR gets the best performance on the UAV-123 benchmark. And, CFASESE achieves better performance than CFASE. This result denotes the scale selection of BASTR is suitable for long-term low frame-rate tracking.

Table 6.1 Running speed of six trackers on OTB-2015.

| Trackers | CFASE | BASTR | CFCA | CFASENS | BASTRNS | CFCANS | CFASESE | BASTRS |
|---|---|---|---|---|---|---|---|---|
| FPS | 31.1 | 38.1 | 114.1 | 43.6 | 54.9 | **166.7** | 27.5 | 34.1 |

Table 6.1 shows the running speed of six proposed trackers on the OTB-2015 benchmark database. The best result is shown in red font. Although trackers without scale estimation perform worse results than trackers using scale estimation, the running speed of trackers without scale estimation is faster than trackers using scale estimation. For example, CFCANS gets 166.7 FPS, 52 FPS faster than CFCA. If the target object’s scale has not significantly changed in some applications, the scale estimation can be considered discarded.

## 6.3 The importance of framework

Discriminative correlation filters trackers regard object tracking to a regression task. Based-DCF trackers aim to construct a robust correlation filter model based on the ridge regression model. As shown in Equation 2.1, CFCA adopts the simplest ridge regression model to improve the tracking performance. The excellent performance and real-time illustrate the effectiveness of CFCA. CFASE is proposed on STRCF. Compared to the L2 regularization of CFCA, STRCF adds spatial weight into the second term of regularization, and introduces a temporal regularization to maintain the consistency of the DCF model. CFASE proposes an improved method of temporal regularization so that the trained DCF model can maintain more useful appearance information. Since the complications of the framework, the calculations burden of CFASE is heavy than CFCA. Compared to CFASE, the mathematical model of BASTR becomes more complex. To improve the robustness of BACF's framework, BASTR introduces temporal regularization and adaptive spatial regularization. This leads to the framework of BASTR can adapt to more scenarios.

To analyze the effectiveness of the proposed trackers, we conduct evaluation experiments between the proposed trackers (CFCA, CFASE, and BASTR) and the baseline trackers (KCF, STRCF, and BACF) on different benchmark databases (OTB benchmark [93, 94], Temple Color benchmark [75], UAV benchmark [71]).

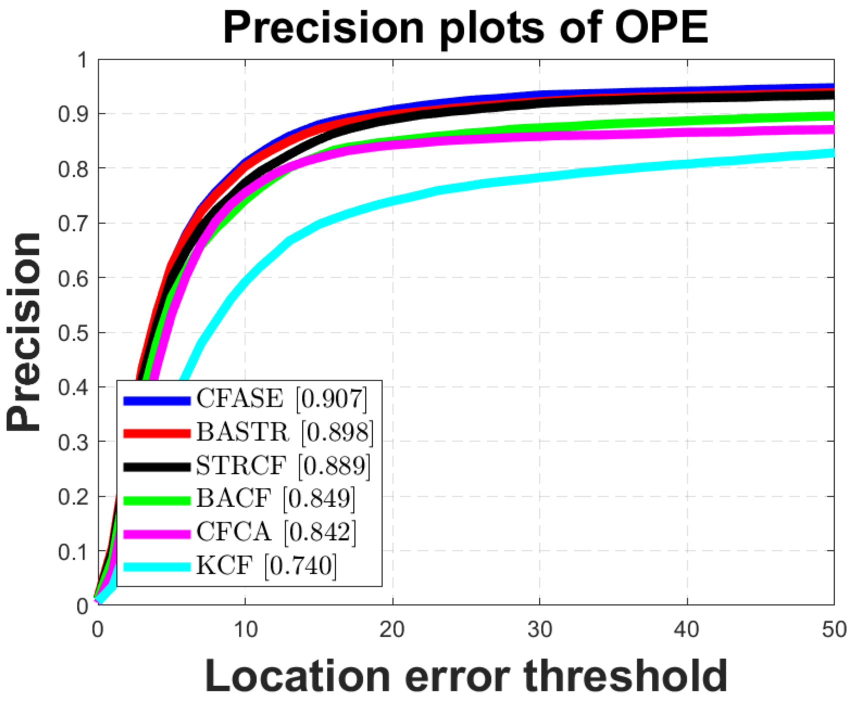


Figure 6.15 Precision plot of the proposed trackers and baseline on OTB-2013.

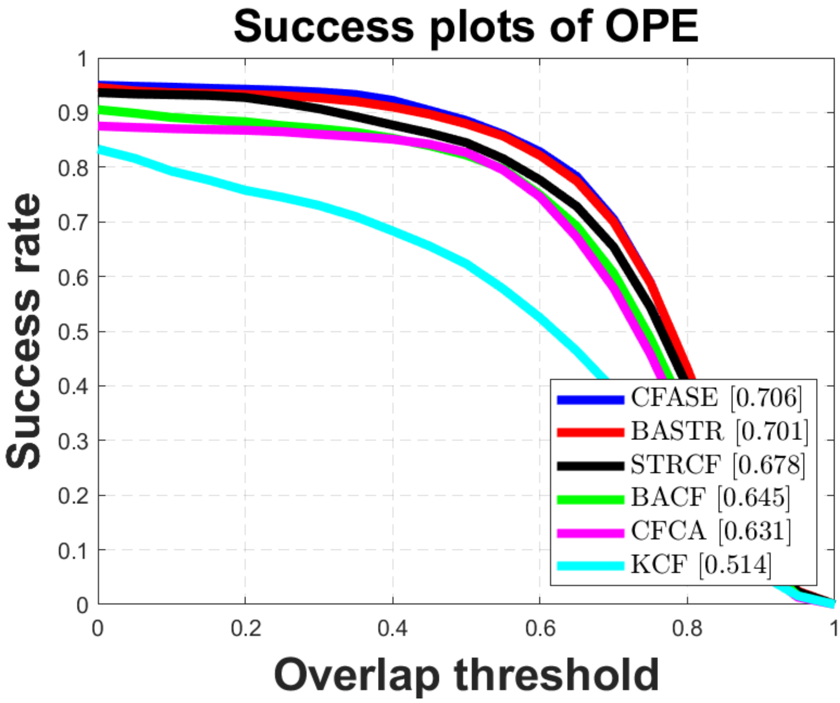


Figure 6.16 Success plot of the proposed trackers and baseline on OTB-2013.

Firstly, we analyze the performance of proposed trackers on the OTB-2013 benchmark database (As Fig 6.15 and Fig 6.16). Compared to baseline KCF, CFCA obtains a gain of 10.2% on the precision score and 11.7% on the success score, respectively. Since BACF enlarges the research region by solving the boundary effect, this leads to the performance of BACF being better than CFCA. BASTR gets better performance than BACF, increasing 4.9% and 6.1% on precision and success score, respectively. CFASE achieves the best performance on the OTB-2013. The experiment results of the OTB-2013 illustrate the importance of the mathematical model in visual tracking.

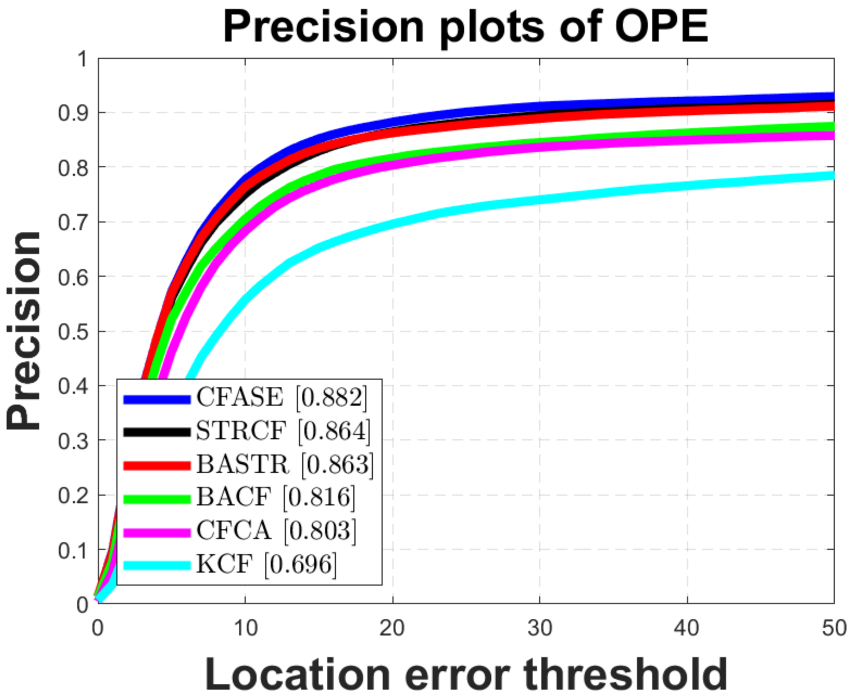


Figure 6.17 Precision plot of the proposed trackers and baseline on OTB-2015.

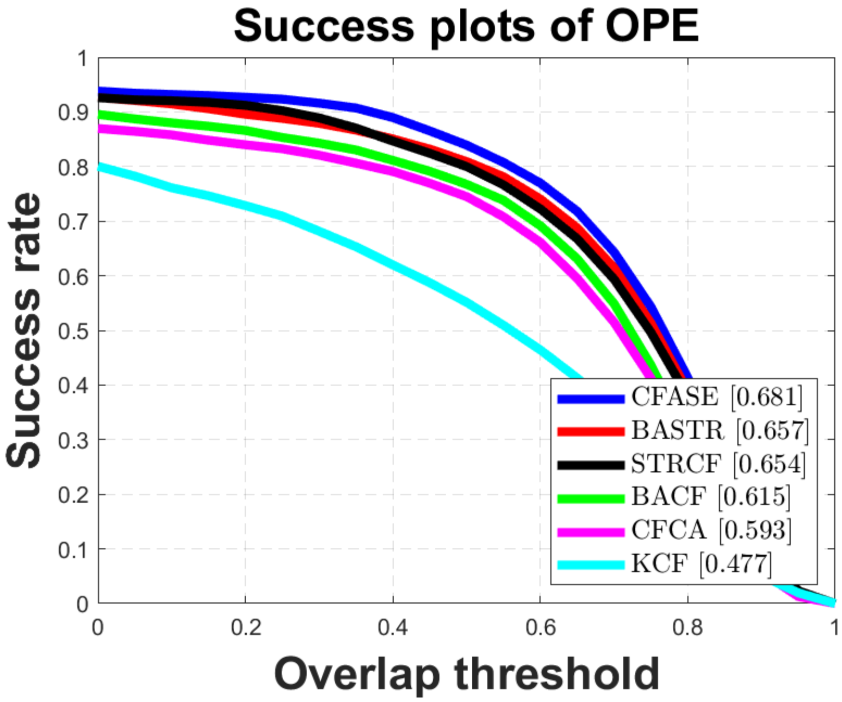


Figure 6.18 Success plot of the proposed trackers and baseline on OTB-2015.

Fig 6.17 and Fig 6.18 show the precision score and the success score on the OTB-2015 benchmark database. Compared to baselines, the proposed trackers obtain a certain degree of improvement. CFCA obtains a gain of 10.7% on the precision score and 11.6% on the success score, respectively. Compared to STRCF, CFASE obtains a gain of 1.8% on the precision score and 2.7% on the success score, respectively. As discussed in chapter 4, CFASE achieves remarkable improvement in some scenarios. BASTR obtains a gain of 4.7% on the precision score and 4.2% on the success score, respectively.

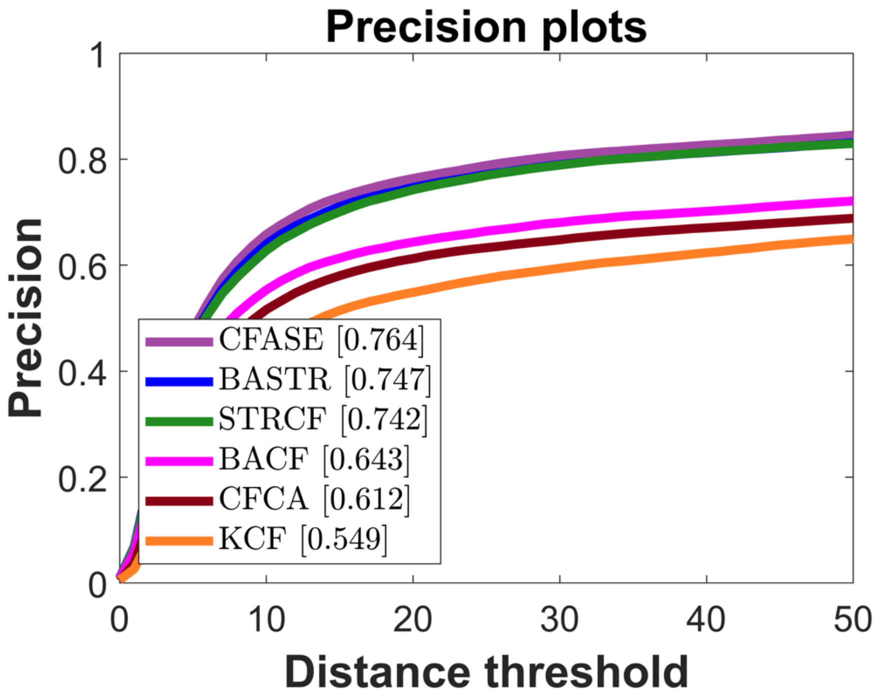


Figure 6.19 Precision plot of the proposed trackers and baseline on TC128.

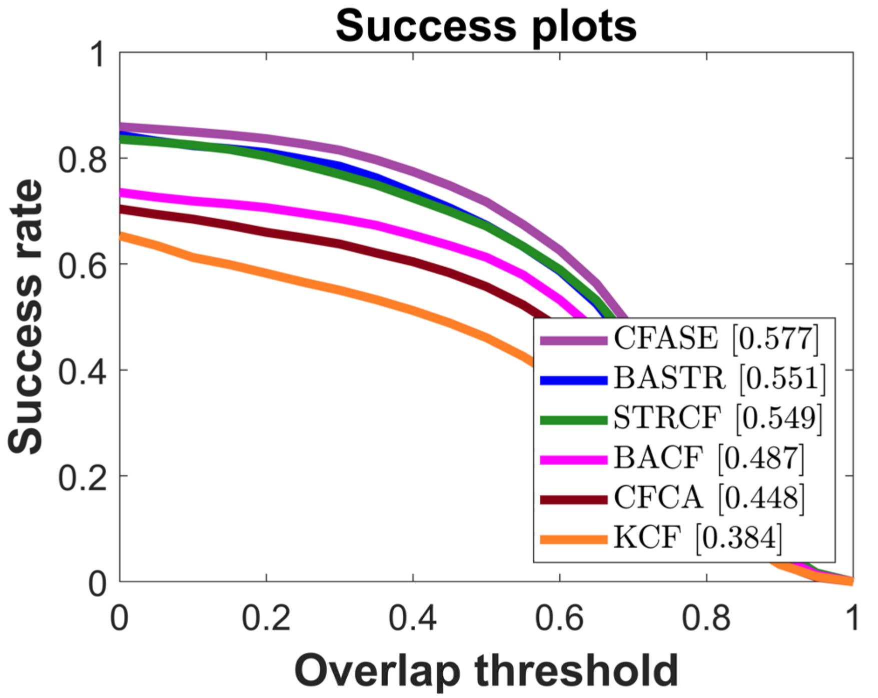


Figure 6.20 Success plot of the proposed trackers and baseline on TC128.

Since the TC128 benchmark only contains color image sequences, CFCA and KCF do not achieve significant performance. However, CFCA improves 6.3% and 6.4% on the precision score and the success score, respectively. CFASE still maintains the best performance in six trackers and obtains a gain of 2.2% and 2.8% on the precision score and the success score, respectively. In contrast to the OTB benchmark, BASTR obtains more greatly improvement than BACF on the TC128 benchmark and gains 10.4% and 6.4% on the precision score and the success score, respectively.

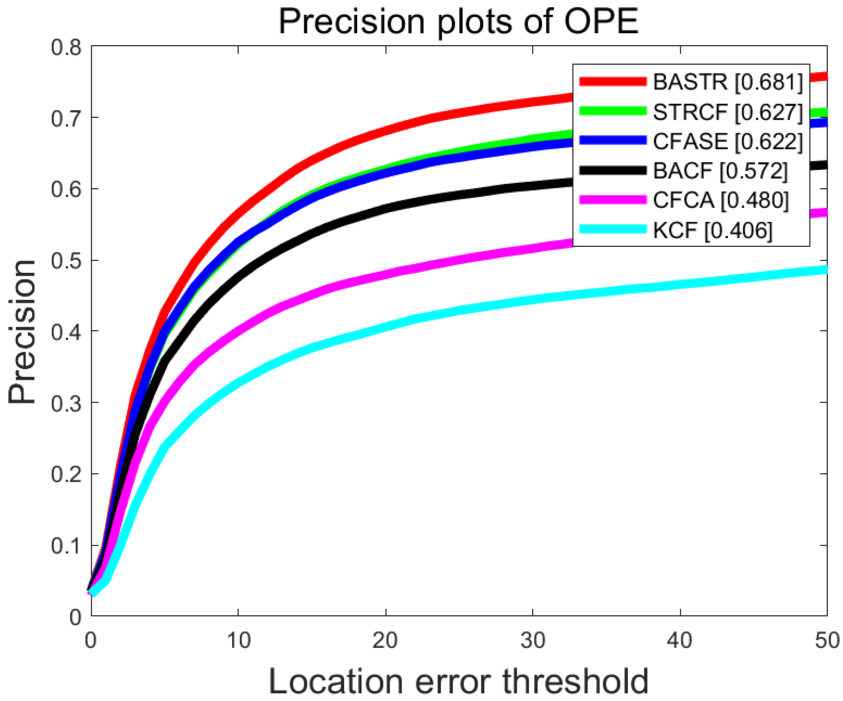


Figure 6.21 Precision plot of the proposed trackers and baseline on UAV123-10fps.

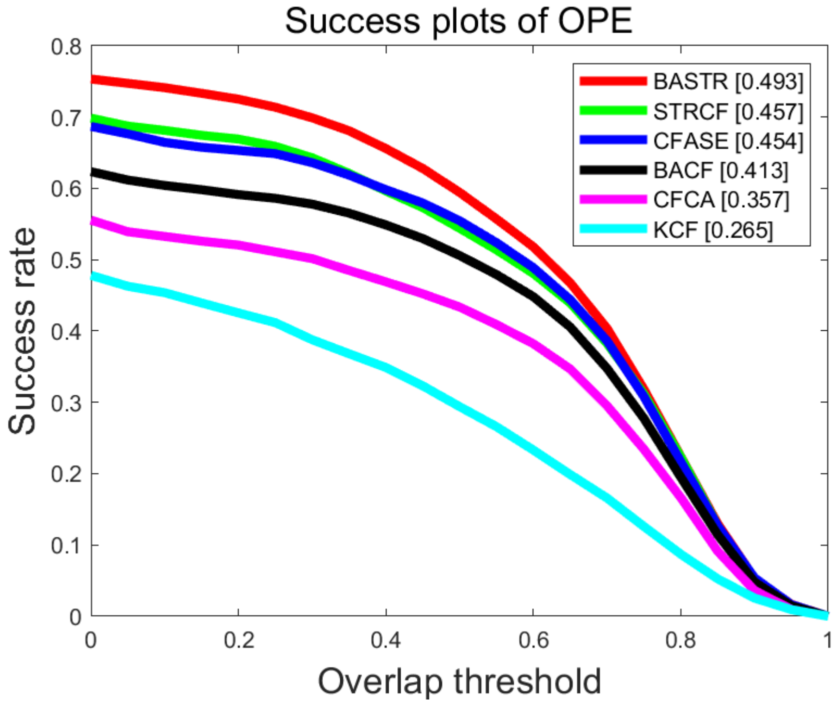


Figure 6.22 Success plot of the proposed trackers and baseline on UAV123-10fps.

In six trackers, BASTR is proposed with consideration of UAV. As shown in Fig 6.21 and Fig 6.22, BASTR achieves great success on the UAV database. Compared to BACF, BASTR obtains a gain of 10.9% on the precision score and 8.0% on the success score, respectively. The performance of CFASE which gets the best performance on the OTB benchmark and TC128 benchmark worse than STRCF on the UAV123 benchmark. Since a high precision scale pool cannot meet the scale requirement of the UAV database, the performance of the tracker will be impaired. The framework of CFCA and KCF is the simplest, and the performance is the worst on the UAV123 benchmark. The results of the UAV benchmark demonstrate that it is vital for trackers to construct a robust mathematical model.

Table 6.2 Running speed of DCF trackers on OTB-2015.

| Trackers | CFASE | BASTR | CFCA | STRCF | BACF | KCF |
|---|---|---|---|---|---|---|
| FPS | 31.1 | 38.1 | 114.1 | 33.4 | 45.0 | **168.7** |

The experiment results of different benchmarks show the effectiveness of the proposed trackers. Table 6.2 shows the running speed of the six trackers on the OTB-2015 benchmark database. The best result is shown in red font. Although the performance of CFCA and KCF is worse than other DCF trackers, the running speed is faster than others. The slowest trackers also meet the requirement of real-time.

# Chapter 7

# Conclusion

In this chapter, we summarize the most significant results and analysis discussed in this thesis. The present study demonstrates the importance of the high precision location, scale estimation, and the mathematical model by analyzing the proposed object tracking algorithms based on a discriminative correlation filter. As an integral part of computer vision, object tracking has been a hot research topic. However, object tracking still has some unsolved problems. Blindly pursuing significant performance and ignoring real-time algorithms cannot be considered an excellent algorithm. The proposed three trackers maintain the balance between outperformance and real-time.

In chapter 3, I proposed an improved strategy CFCA based on the most straightforward correlation filter framework. CFCA adopts the novel scale pool to solve scale variation and increases scale estimation precision by adaptively adjusting the research region's size. At the same time, I combine the four highest scores and luminance histogram similarity to improve the detection accuracy. In the model update, I analyze the target object's state to determine whether update the appearance model and the correlation filter model. Certainly, when an object has drifted, the re-detection occurs by the peak location of the confidence map. The evaluation experiments demonstrate the effectiveness of the proposed tracker, and CFCA gets a high running speed (114.1FPS) with a single CPU.

I proposed a robust tracker CFASE based on the complex DCF model in chapter 4. Although the temporal regularization improves the performance of the tracker, the trained DCF model only tends to the obtained DCF model in the previous frame. I improve the temporal regularization to learn more information from the previous few frames. For scale variation, I train the detection model and scale estimation model, respectively. I adopt a more effective method to determine the target object's state in CFASE. The judgment of the object's state is not used to update the model instead of determining the accuracy of scale estimation and location. The experiment results show the effectiveness of CFASE.

In chapter 5, a robust DCF model BASTR is proposed. BASTR adaptive selects the scale estimation method to meet the requirement of more scenarios. Based on background awareness, I introduce adaptive spatial regularization and temporal regularization to increase the robustness of the DCF model.

I demonstrate the influence of detection location, the importance of scale estimation, and the importance of the mathematical model. By the experiment results of different benchmarks, I illustrate the effectiveness of the proposed trackers.

In the future, there will be several works for improvement in the proposed trackers. Firstly, the proposed trackers perform many parameters to obtain outstanding performance. Although these parameters enhance the robustness of the model, they also reduce the stability of the trackers. The tracker will get a significant improvement with adaptive parameters. In addition, the complexity of the model improves the performance of the trackers. It also dramatically reduces the speed of the trackers. Simplification of the model is also essential to work in the future.

# Reference

[1] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun. 3d traffic scene understanding from movable platforms. PAMI, 2014.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In NIPS, pp. 1097–1105, 2012.

[3] A. Mahalanobis, B. V. K. V. Kumar, and D. Casasent. Minimum average correlation energy filters. In Appl. Opt., Vol. 26, No. 17, pp. 3633, 1987.

[4] A. Nilski. An evaluation metric for multiple camera tracking systems: The i-LIDS 5th scenario. In Proc. SPIE, 2008.

[5] A. T. Nghiem, F. Bremond, M. Thonnat and V. Valentin. Etiseo performance evaluation for video surveillance systems. In Proc. AVSS London U.K., pp. 476-481, 2007.

[6] Alvarez, L. and Morales, F. Affine morphological multiscale analysis of corners and multiple junctions. In International Journal of Computer Vision, Vol. 2, No. 25, pp. 95–107, 1997.

[7] Arnold W. M. Smeulders,Rita Cucchiara,Afshin Dehghan.Visual Tracking: An Experimental Survey. In TPAMI, Vol. 36, No. 7, pp. 1442 - 1468, 2014.

[8] B. Babenko, M. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. In TPAMI, 2011.

[9] B. Kumar, A. Mahalanobis, S. Song, S. Sims, and J. Epperson. Minimum squared error synthetic discriminant functions. In Optical Engineering, 1992.

[10] B. Kumar. Minimum-variance synthetic discriminant functions. In J. Opt. Soc. of America., Vol. 3, No. 10, pp. 1579–1584, 1986.

[11] B. Ristic and M. L. Hernandez. Tracking systems. In IEEE RADAR, pp. 1-2, 2008.

[12] B. Schölkopf and A. Smola. Learning with kernels: Support vector machines, regularization, optimization, and beyond. 2002.

[13] Bay, Herbert; Tuytelaars, Tinne & Gool, Luc Van. SURF: speeded up robust features. In ECCV, Vol. 110, No. 3, pp. 346-359, 2007.

[14] Bergen J.R., Anandan P., Hanna K.J., Hingorani R. (1992) Hierarchical model-based motion estimation. In ECCV, 1992.

[15] Besag, J., York, J. & Mollié, A. Bayesian image restoration with two applications in spatial statistics. In Ann Inst Stat Math 43, pp. 1–20,1991.

[16] Bo Li, Wei Wu, and Qiang Wang. SiamRPN++: Evolution of Siamese Visual Tracking with Very Deep Networks. In CVPR, 2019.

[17] Bretzner, L. and Lindeberg, T. Feature tracking with automatic selection of spatial scales. In Computer Vision and Image Understanding, Vol. 71, No. 3, pp. 385–392, 1998.

[18] C. Dicle, M. Sznaier, and O. Camps. The way they move: Tracking multiple targets with similar appearance. In ICCV, 2013.

[19] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In CVPR, pp. 5388–5396, 2015.

[20] C. Xie, M. Savvides, and B. Vijaya-Kumar. Kernel correlation filter based redundant class-dependence feature analysis (KCFA) on FRGC2.0 data. In Analysis and Modelling of Faces and Gestures, 2005.

[21] C.R. Wren, A. Azarbayejani, T. Darrell, and A.P. Pentland. "Pfinder: Real-time Tracking of the Human Body". IEEE Trans. on Pattern Recognition and Machine Intelligence, Vol. 19, No. 7, pp. 780-785, 1997.

[22] Claudine Badue, Ranik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius B. Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago M. Paixao, Filipe Mutz, Lucas de Paula Veronese, Thiago Oliveira-Santos, Alberto F. De Souza.Self-driving cars: A survey. Expert Systems with Applications, Vol. 165, 2021.

[23] Cornelis, N., Leibe, B., Cornelis, K. et al. 3D Urban Scene Modeling Integrating Recognition and Reconstruction. Int J Comput Vis 78, pp.121–141, 2008.

[24] Cyriel Diels, Jelte E. Bos, Self-driving carsickness. Applied Ergonomics, Vol. 53, Part B, pp.374-382, 2016.

[25] D. A. Klein, D. Schulz, S. Frintrop and A. B. Cremers, "Adaptive real-time video-tracking for arbitrary objects", Proc. IEEE IROS, pp. 772-777, 2010.

[26] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Yui M. Lui. Visual object tracking using adaptive correlation filters. In CVPR, 2010.

[27] Dai, Kenan and Wang, Dong and Lu, Huchuan and Sun, Chong and Li, Jianhua. Visual Tracking via Adaptive Spatially-Regularized Correlation Filters. In CVPR 2019.

[28] David M. J. Tax; Robert Duin; Dick De Ridder. Classification, Parameter Estimation and State Estimation: An Engineering Approach Using MATLAB. John Wiley and Sons. 2004.

[29] E. H. Adelson, C. H. Anderson, J. R. Bergen, P. J. Burt, and J. M. Ogden. Pyramid methods in image processing. RCA engineer, 1984.

[30] F. Bashir and F. Porikli. Performance evaluation of object detection and tracking systems. In IEEE Int. Workshop PETS, 2006.

[31] F. Li, C. Tian, W. Zuo, L. Zhang, and M.-H. Yang. Learning spatial-temporal regularized correlation filters for visual tracking. In IEEE Conference on Computer Vision and Pattern Recognition, 2018.

[32] Felzenszwalb, P., Girshick, R., McAllester, D., Ramanan. Object detection with discriminatively trained part-based models. In TPAMI, 2010.

[33] G. Rigoll, S. Eickeler and S. Muller. Person tracking in real-world scenarios using statistical methods. In Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition,pp. 342-347, 2000.

[34] Girshick, Ross B., Jeff Donahue, Trevor Darrell and Jitendra Malik. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In IEEE Conference on Computer Vision and Pattern Recognition, pp. 580-587, 2014.

[35] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In ECCV, 2008.

[36] H. K. Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. In ICCV, 2013.

[37] H. T. Nguyen and A. W. M. Smeulders. Fast occluded object tracking by a robust appearance filter. IEEE Trans. Pattern Anal. Mach. Intell, Vol. 26, No. 8, pp. 1099-1104, 2004.

[38] Hamed Kiani Galoogahi, Ashton Fagg, and Simon Lucey. Learning background-aware correlation filters for visual tracking. In ICCV, 2017.

[39] Harris, C. & Stephens, M. A combined edge and corner detector. In Alvey Vision Conference, pp. 147-151, 1988.

[40] J. F. Henriques, J. Carreira, R. Caseiro, and J. Batista. Beyond hard negative mining: Efficient detector learning via block-circulant decomposition. In ICCV, 2013.

[41] J. Kwon and K. M. Lee. Visual tracking decomposition. In CVPR, pp. 1269–1276, 2010.

[42] J. Mairal, M. Elad and G. Sapiro. Sparse Representation for Color Image Restoration. In IEEE Transactions on Image Processing, Vol. 17, No. 1, pp. 53-69, 2008.

[43] J. Popoola and A. Amer. Performance evaluation for tracking algorithms using object labels. In ICASSP, 2008.

[44] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In CVPR, 2016.

[45] J. Zhang, S. Ma, and S. Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In ECCV, 2014.

[46] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. Exploiting the Circulant Structure of Tracking-by-detection with Kernels. In ECCV LNCS, Vol.7575, pp.702-715, 2012.

[47] João F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. In TPAMI, Vol. 37, No. 20, pp. 583-596, 2014.

[48] Jungong Han; Ling Shao; Dong Xu; Jamie Shotton. Enhanced Computer Vision With Microsoft Kinect Sensor: A Review. In IEEE Transactions on Cybernetics, Vol: 43, Issue: 5, pp.1318 - 1334, 2013.

[49] K. Briechle and U. D. Hanebeck. Template matching using fast normalized cross correlation. In SPIE, Vol. 4387, pp. 95-102, 2001.

[50] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In arXiv, pp. 1409-1556, 2014.

[51] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In ECCV, 2012.

[52] K.-K. Sung and T. Poggio. Example-Based Learning for ViewBased Human Face Detection. In Image Understanding Workshop, 1994.

[53] Khan, F.S., Anwer, R.M., van de Weijer, J., Bagdanov, A., Lopez, A., Felsberg, M.: Coloring action recognition in still images. In IJCV, Vol. 105, No.3, pp. 205-221, 2013.

[54] Khan, F.S., Anwer, R.M., van de Weijer, J., Bagdanov, A., Vanrell, M., Lopez, A. Color attributes for object detection. In CVPR, 2012.

[55] Khan, F.S., van de Weijer, J., Vanrell, M.: Modulating shape features by color attention for object recognition. In IJCV, Vol. 98, No. 1, pp. 49-64, 2011.

[56] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In European Conference on Computer Vision workshops, 2016.

[57] L. Bertinetto, J. Valmadre, S. Golodetz, O. Miksik, and P. Torr. Staple: Complementary learners for real-time tracking. In CVPR, 2016.

[58] L. Figueiredo, I. Jesus, J. A. T. Machado, J. R. Ferreira and J. L. Martins de Carvalho. Towards the development of intelligent transportation systems. In ITSC, pp. 1206-1211, 2001.

[59] L. Sevilla-Lara and E. Learned-Miller. Distribution Fields for Tracking. In CVPR, 2012.

[60] L. Wang, W. Ouyang, X. Wang, and H. Lu. Stct: Sequentially training convolutional networks for visual tracking. In CVPR, 2016.

[61] L. Wen, D. Du, Z. Lei, S. Z. Li, and M.-H. Yang. Jots: Joint online tracking and segmentation. In IEEE Conference on Computer Vision and Pattern Recognition, 2015.

[62] L.M. Brown, A.W. Senior, Y.-L. Tian, J. Connell, A. Hampapur, C.-F. Shu, H. Merkl, and M. Lu. Performance Evaluation of Surveillance Systems Under Varying Conditions. In Proc. IEEE Workshop on Performance Evaluation of Tracking and Surveillance, 2005.

[63] Li Y, Zhu J. A scale adaptive kernel correlation filter tracker with feature integration. In ECCV LNCS, Vol. 8926, pp. 254-265, 2014.

[64] Lowe, D.G. Distinctive image features from scale invariant key points. In t. J. Computer Vision, Vol. 60, No. 2, pp. 91–110, 2004.

[65] M. Danelljan, G. Häger, F. Khan, M. Felsberg. Accurate Scale Estimation for Robust Visual Tracking. In BMVC, 2014.

[66] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In ICCV, 2015.

[67] M. Everingham, L. J. V. Gool, C. K. I. Williams, J. M. Winn and A. Zisserman. The Pascal visual object classes VOC challenge. In IJCV, Vol. 88, No. 2, pp. 303-338, 2010.

[68] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin, T. Voj ˇ ´ır, G. Hager, A. Luke ¨ ziˇ c,ˇ G. Fernandez, et al. The visual object tracking vot2016 challenge results. In European Conference on Computer Vision, 2016.

[69] M. Meyer, M. Hötter, and T. Ohmacht. A New System for Video-B ased Detection of Moving Objects and its Integration into Digital Networks. In Proc. 30th Intern. Carnahan Conf on Security Technology, pp. 105-110, 1996.

[70] Martin Danelljan, Fahad Shahbaz Khan, Michael Felsberg and Joost van de Weijer. Adaptive Color Attributes for Real-Time Visual Tracking. In CVPR, 2014.

[71] Matthias Mueller, Neil Smith, and Bernard Ghanem. A Benchmark and Simulator for UAV Tracking. In ECCV, 2016.

[72] Mengmeng Wang, Yong Liu, and Zeyi Huang. Large Margin Object Tracking with Circulant Feature Maps. In CVPR, pp. 4800-4808, 2017.

[73] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In IEEE Conference on Computer Vision and Pattern Recognition, 2005.

[74] P. Chockalingam, S. N. Pradeep, and S. Birchfield. Adaptive fragments-based tracking of non-rigid objects using level sets. In ICCV, pp. 1530–1537, 2009.

[75] P. Liang, E. Blasch, and H. Ling. Encoding Color Information for Visual Tracking: Algorithms and Benchmark. In IEEE Trans. on Image Processing (T-IP), Vol. 24, No. 12, pp. 5630-5644, 2015.

[76] P. Viola and M. J. Jones. Robust real-time face detection. In IJCV, Vol.57, pp. 137–154, 2004.

[77] P.J. Philips, H. Moon, S.A. Rizvi, and P.J. Rauss. The FERET Evaluation Methodology for Face-Recognition Algorithms. In IEEE Trans. on Pattern Recognition and Machine Intelligence, Vol. 22, No. 10, pp. 1090-1104, 2000.

[78] Qiang Wang, Li Zhang, Luca Bertinetto. Fast Online Object Tracking and Segmentation: A Unifying Approach. In CVPR, 2019.

[79] R. Patnaik and D. Casasent. Fast FFT-based distortion-invariant kernel filters for general object recognition. In Proceedings of SPIE, Vol. 7252, 2009.

[80] R. Unnikrishnan, C. Pantofaru and M. Hebert. Toward Objective Evaluation of Image Segmentation Algorithms. In IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 29, No. 6, pp. 929-944, 2007.

[81] Rostyslav Demush, Hacker Noon. A Brief History of Computer Vision (and Convolutional Neural Networks), 2019.

[82] S. Avidan. Support Vector Tracking. In IEEE Trans. on PAMI, Vol. 26, pp. 1064–1072, 2004.

[83] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured Output Tracking with Kernels. In ICCV, 2011.

[84] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards realtime object detection with region proposal networks. In NIPS, pp. 91–99, 2015.

[85] Stephen Boyd, Neal Parikh, Eric Chu Borja Peleato and Jonathan Eckstein. Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. Foundations and Trends in Machine Learning, Vol.3, No.1, 2010.

[86] T. Makovski, G. A. Vazquez, and Y. V. Jiang. Visual learning in multiple-object tracking. In PLoS One, 2008.

[87] Than, Ker. How the Human Eye Works. In LiveScience. TechMedia Network, 2010.

[88] U. Prabhu, K. Seshadri and M. Savvides. Automatic facial landmark tracking in video sequences using kalman filter assisted active shape models. In ECCV, 2010.

[89] V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation, and signal processing. In Advances in Neural Information Processing Systems 9, pp. 281– 287, 1997.

[90] V. Vapnik. The Nature of Statistical Learning Theory. In Springer, 1995.

[91] X. Jia, H. Lu, and M.-H. Yang. Visual Tracking via Adaptive Structural Local Sparse Appearance Model. In CVPR, pp. 1822-1829, 2012.

[92] Xu, L., Kim, P., Wang, M. et al. Spatio-temporal joint aberrance suppressed correlation filter for visual tracking. In Complex Intell. Syst, 2021.

[93] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. In TPAMI, Vol. 37, No. 9, pp. 1834– 1848, 2015.

[94] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In CVPR, pp. 2411-2418, 2013.

[95] Yiming Li, Changhong Fu, Fangqiang Ding, Ziyuan Huang, Geng Lu. AutoTrack: Towards High-Performance Visual Tracking for UAV with Automatic Spatio-Temporal Regularization. In CVPR, 2020.

[96] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learningdetection. In TPAMI, 2012.

[97] Zhaoqian Tang, Kaoru Arakawa. Kernel Correlation Filter via Adaptive Model. In ISPACS, 2018.

[98] Zhaoqian Tang, Kaoru Arakawa. Visual Tracking via Correlation Filter using Luminance Histogram and Adaptive Model. In SISA, 2019.