

PAPER • OPEN ACCESS

A Camera Tracking System Based on Closed-loop Kernelized Correlation Filters

To cite this article: Muzi Li *et al* 2019 *IOP Conf. Ser.: Mater. Sci. Eng.* **533** 012037

View the [article online](#) for updates and enhancements.

A Camera Tracking System Based on Closed-loop Kernelized Correlation Filters

Muzi Li, Bo Cheng, Shuai Zhao, and Junliang Chen

State Key laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing, China

Corresponding author: cedricLi@bupt.edu.cn

Abstract. Camera tracking is an important application in the computer vision. With the progress of the tracking-by-detection algorithm, industrial cameras can track targets autonomously. However, during the long-time tracking, it's prone to miss target owing to heavy object occlusion, environment changing and objects appearance variations. Our camera tracking system is based on the kernelized correlation filter to track object, and we add a self-verification module to judge if the current tracking results are reliable. This can be useful in solving target missing when object meets occlusion or variations, and avoid model drift during long-time detection. Last but not the least, we keep the targets in the corner of the image by controlling our camera, avoiding object's moving out of view. The kernel correlation filter bounded with self-verification and re-detection modules boost the performance. Extensive experiments on the OTB-2013 benchmark show that it performs better than state-of-the-art methods.

1. Introduction

Visual object tracking is a very popular topic in computer vision for its numerous applications, such as intelligent vehicles, human-machine interaction, and surveillance[1]. Even though the significant progress has been made in object tracking, there still remains a lot of challenge due to object occlusion and significant appearance changes.

In recent years, there are many object trackers, The tracking mechanism may be similar, a tracker identify targets after a detector inspect it, and then update the tracker according to object moving and its variations[2]. Bolme et al. [3] introduced correlation filters to visual tracking for this first time by minimizing the output sum of squared error. And then Correlation Filter based trackers have been very popular for its high efficiency and robustness among those object trackers. There are two reasons for Correlation Filter based trackers' success. One thing is that this kind of trackers can produce thousands of training samples through image patch shifting. The lack of training data avoids other trackers' progressing. And the other thing is its fast training and detecting via circular correlation via Fast Fourier Transform(FFT)[4].

Even though Correlation Filter based trackers are so popular, there still remains a lot to be improved. In a long- time detection, the model update is easy to drift and leads to a wrong result under the situation of object occlusion and appearance variations. Most Correlation Filter trackers consider tracking process as an open-loop process. They neither check the reliability of their results, nor generate any feedback to reshape their filter. So it's not hard to know that the ignored abnormality and



localization error in a few frames will cause the bad results. There are two import things to fix the problem. The first one is to identify whether the current tracking result is reliable. And the other thing is to add re-detection module to the filter.

In order to solve the existing problems in long-time tracking, we designed a camera tracking system based on closed-loop kernelized filter tracker. We add a self-verification module after the tracking output to judge whether we need to run our re-detection module. What's more, a re-detection module is also added to compensate the localization error and reduce model drifting. Through a feedback loop, the re-detection model can retain the model's discriminative ability in long time tracking. We send control commands to our camera to turn a certain angle when the target deviates from the center of the image beyond the pre-set threshold..

2. Architecture

Our camera tracking system contains two parts in the figure 1. The first part is about our camera which provides the function of real-time video streaming and camera control. And the second part is our object tracker consists of kernelized correlation filter, self-verification and re-detection. This part helps track the target in a long time.

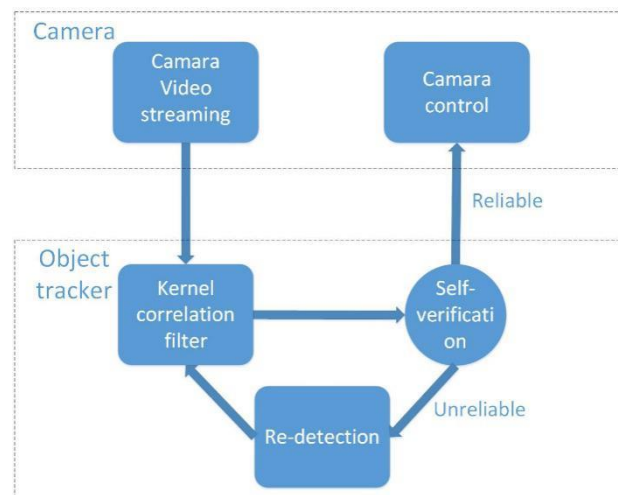


Figure 1. Architecture of the Camera Tracking System

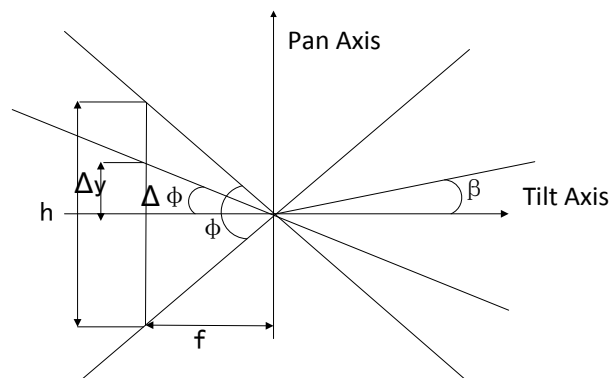


Figure 2. Relationship between Offset and Rotation Angle

2.1. Camera Function

The camera's real-time video stream is streamed in RTSP format. We use openCV tools to receive the video stream, and then extract the first frame as the input of our subsequent model initialization. The target position is also marked by the detection algorithm.

2.1.1. Real Time Video Streaming. The camera's real-time video stream is streamed in RTSP format. We use openCV tools to receive the video stream, and then extract the first frame as the input of our subsequent model initialization. The target position is also marked by the detection algorithm.

2.1.2. Camera Control. For a camera tracking system, target's being out of view should be avoided. It can be hard for a object tracker to achieve the goal. The target is always moving, we should control the moving of camera to keep the target in proper position. The way we do is to send control commands to our camera to turn a certain angle when the target deviates from the center of the image beyond the pre-set threshold.

Our experimental camera is a kind of PTZ camera. In the PTZ camera, there are three kinds of rotation axes including horizontal rotation axis of the turntable (Pan axis), the vertical axis of rotation (Tilt axis) and the optical axis of the camera. Every two of them are perpendicular to each other and intersect at point O, and the three meet the Cartesian coordinate system [5]. The X and Y axes of the camera coordinate system coincide with the Tilt and Pan axes, respectively, the Z axis is the optical axis of the camera.

When the target is at the edge of the image, it is necessary to control the motion of the camera to rotate around the Pan axis and the Tile axis, so that the coordinate position of the target is at a reasonable position of the image. During the control process, we should consider the coordinates' (Δx , Δy) changes and the angle of rotation($\Delta\theta$, $\Delta\phi$) as the most important parameters. And then we will discussed about the relationship between $\Delta\theta$, $\Delta\phi$ and Δx , Δy .

We suppose that the horizontal and vertical angels of the field captured by the camera are φ and ψ . The camera focus length is f when imaging. The horizontal length of the image is L pixels and the vertical direction is h pixels. Among these parameters, only f can't be measured. So we take the coordinate(x , y) as an example.

As the figure 2 shows, the Pan axis intersects the Tilt axis and they are perpendicular to each other. and then we get the $\Delta\phi$,

$$f \times \tan(\varphi/2) = h/2 \quad (1)$$

$$f \times \tan \Delta\phi = \Delta y \quad (2)$$

$$\Delta\phi = \arctan \frac{2\Delta y \tan(\varphi/2)}{h} \quad (3)$$

In the precise camera model, the Tilt axis is project to the plane of x and y plane and there exists a error angle is β .

$$\Delta\phi = \frac{\Delta y \tan(\varphi/2)}{h \tan(\varphi/2 + \beta)} \varphi \quad (4)$$

We can also get the relationship between $\Delta\theta$ and Δx . In this equation, Δy equals to the amount of the target's change in the y axis.

2.2. Kernelized Correlation Filter

Kernelized Correlation Filter tracker is a kind of tracking-by-detection algorithm. The core function is to cyclically shift the tracking target area to mark the target object as a positive sample and the rest of the surrounding environment is a negative sample[6]. It constructs a training discriminant classifier on the basis that the positive sample is 1, and the negative sample is 0. The sample labelling method is well reflected by the weight of each negative sample, and is assigned [0,1] according to the distance between the target and the center of the sample. The closer the distance is to the target, the closer the value is to 1, and it is closer to 0 in the contrast. The experiment can prove that the sample labelling works well in the discriminant classifier training. The Kernelized Correlation Filter tracker constructs the training samples of the classifier via the matrix cyclic offset, and it selects the candidate region with the largest similarity as the new tracking target, so that the data matrix becomes a circular matrix. Then, based on the characteristics of the cyclic matrix, the solution of the un-linear kernel problem is

transformed into the discrete Fourier transform domain in order to avoid the process of inverting the matrix. This way can reduce both storage and computation of the algorithm by several orders of magnitude and greatly improve the processing speed.

2.2.1. Ridge Regression. Ridge regression can perform as well as those sophisticated methods. The goal is to minimize the squared error over samples x_i and regression targets.

$$\min_w \sum_i (f(x_i) - y_i)^2 + \lambda \|w\|^2 \quad (5)$$

Since ridge regression admits a simpler closed-form solution, w is given by:

$$w = (X^T X + \lambda I)^{-1} X^T y \quad (6)$$

In this eq, per row x_i and each regression target element of y are in the data matrix. To minimize the squared error w , there exists a large number of linear equations. And in the next section, the computational complexity will be greatly reduced with the introduction of the circulant matrix.

2.2.2. Circulant Matrices. Focusing on only single-channel signals can help reduce our computation complexity. We refer to an $n \times 1$ vector representing a sample patch as the base sample which is denoted as x . And then we can model the permutation matrix by a cyclic shift operator. Shifting x by one element, we model a small translation. A larger translation can be made by the permutation matrix. By this way we can both get half of the shifts set as positive samples, and the other half as negative samples. And the translation leads us to the circulant matrix.

The fact that all circulant matrices are made diagonal by the Discrete Fourier Transform (DFT), regardless of the generating vector x [7]. And the expression is:

$$X = F \text{diag}(X^\wedge) F^H \quad (7)$$

F is a DFT matrix and $X^\wedge$ is the DFT of the generating vector. $x^\wedge = F(x)$. And DFT is a kind of linear operation.

2.2.3. Kernel Trick. This section will map the inputs of a linear problem to a complicated non-linear spatial feature space with the kernel trick. The map function is $\phi(x)$, the solution w can be expressed as:

$$w = \sum_i \alpha_i \phi(x_i) \quad (8)$$

In this eq, $\alpha = \{\alpha_1, \alpha_2, \dots, \alpha_i, \dots\}$, α_i is the coefficient of the responding training sample x_i . Using the kernel function κ , we can get the expression

$$\phi^T(x) \phi(x') = \kappa(x, x') \quad (9)$$

$$K_{ij} = \kappa(x_i, x_j) \quad (10)$$

And the final regression function is eq11, the kernel trick brings the model to a high-dimensional feature space, however the complexity will grow larger as the number of samples grows.

$$f(z) = \omega^T z = \sum_{i=1}^n \alpha_i \kappa(z, x_i) \quad (11)$$

2.2.4. Fast Training and Detection. With the addition of kernel trick, the solution of the ridge regression is:

$$\alpha = (K + \lambda I)^{-1} y \quad (12)$$

Since K is a circulant matrix formed by cyclic shifting of the data sets, we can diagonalize Eq 12 as fast as the linear case. There are several kinds of kernel that meet the demand of circulant matrix, such as Radial Basic Function Kernels, Dot-product kernels, Additive kernels and Exponentiated additive kernels.

$$\alpha^{\wedge} = \frac{y^{\wedge}}{k^{\wedge}xx + \lambda} \quad (13)$$

Hat $\wedge$ is the DFT of a vector and the $k^{\wedge}(xx)$ is the first row of the kernel matrix.

When KCF calculates the regression function for a newly input image z , the calculation is of great computation complexity which will cause much time consuming. We can reconstruct the image to get the test samples and training samples of the kernel function, and the established core function matrix is

$$K^z = C(k^{xz}) \quad (14)$$

In the eq, k^{xz} is the vector of the first row. Then we can calculate the full detection response of the cyclic shift structure based on the testing sample z

$$f^z(z) = k^{xz} \odot \alpha^{\wedge} \quad (15)$$

2.3. Self-verification

location errors in a few frames can lead to the model's drifting and tracking failure. The self-verification module is to check the reliability of the detection results carefully. And in this section, we put up an easy way to tell a better result from a bad result. We utilize the responses map of HOG feature as the criterion for the reliability of the tracking results. The peak to sidelobe ratio (PSR) is regarded as the most import parameter. It can help quantify the sharpness of the correlation peak. When the correlation between current frame and previous frames is high, the PSR value is also higher. And in the contrast, it will be lower, if the correlation is down. So we propose the tracking results' score as:

$$S_h^{(i)} = \frac{\max(f_h^{(i)}) - \mu_i}{\sigma_i} \quad (16)$$

In the eq, $f_h^{(i)}$ is the i -th response map of HOG-based correlation filter, and μ_i and σ_i are the mean and standard deviation of the response. Minimizing the discrepancy between the tracking results and expected response can help us find better threshold. Under the assumption that the sub-image catered at the ground-truth position has the minimum discrepancy to the expected response. The discrepancy function can be written as:

$$\arg \min_{X_p} D(X_p, h_t, N) \quad (17)$$

the sub-image X_p is the filter response and the h_t is the filter trained from previous frames. And D is a function to calculate the discrepancy between the actual response map and the expected response map N .

$$D((X_p, h_t), N) = W_p (P_{expected} - P(X_p, h_t)) + W_s (S_{expected} - S(X_p, h_t)) \quad (18)$$

In the eq, $P_{expected}$ is the peak value of the expected fileter response and $P(X_p, h_t)$ is the real peak value of the tracking filter response. $S(X_p, h_t)$ is the PSR response map and $S_{expected}$ is the expected one. W_p and W_s are different weights.

2.4. Self-detection

Under the assumption that the object target is prone to appear in the area near the last reliable frame for a little interval, we only search from our predefined sub-image patches according to our $G(X_p, h_t)$ score.

$$\arg \max_{X_p} G(X_p, h_t) = w_p P(X_p, h_t) + w_s S(X_p, h_t) \quad (19)$$

The problem comes to select the best one which can maximum the G score. A step-wise method can be used in this proble:

$$x_{i,j} = x_0 + i \times \Delta T \cos(j \times \Delta \theta + \frac{(-1)^{i+1}}{4} \times \Delta \theta) \quad (20)$$

$$y_{i,j} = y_0 + i \times \Delta T \sin(j \times \Delta \theta + \frac{(-1)^{i+1}}{4} \times \Delta \theta) \quad (21)$$

In the eq, ΔT and $\Delta \theta$ stands for every distance step and angle step. R_d is the radius of the re-detection area circle, determined by the target size and the PSR value. And $m = R_d / \Delta T$, $n = 2\pi / \Delta \theta$, $m \times n$ candidate sub-images are extracted.

Our Kernelized Filter tracker can work well in most scenarios. In order to keep the computation efficiency, the re-detection module only works under the situation that the self-verification consider the tracking results as unreliable results.

3. Experiment Analysis

We implemented the proposed tracking algorithm in MATLAB and conducted all experiments on a computer with an Intel I7-4700HQ, 2.40GHZ CPU, 8GB RAM and conducted extensive experiments on OTB-2013 datasets. Evaluation Metrics: we use the success rate as our main evaluation metrics.

3.1. Evaluation of Our Proposed Filter Tracker

We evaluate the proposed algorithm on OTB-2013. It can be seen in figure 3 that, the precision rate of our kernelized filter tracker has been increased from 0.514 to 0.677. Among the compared tracking methods, our tracker achieves the second result with a success rate better than the threshold of 0.600. And in the figure 4, our tracker gets a better score than the ECO algorithm when target gets out of view.

3.2. Actual Scenario

In the actual scenario of our camera tracking system, the red frame area is the area where the target object is located, and the yellow one contains more area used by the re-detection module. We can see from the figure 4, the tracking results are considered as unreliable results in our self-verification module when the white woman passes the pole. And then, larger detection area is added in the re-detection part. The result can be seen that the woman can be detected right and our tracking system continues following the woman, which proves that our tracking system works well under the situation of object occlusion.

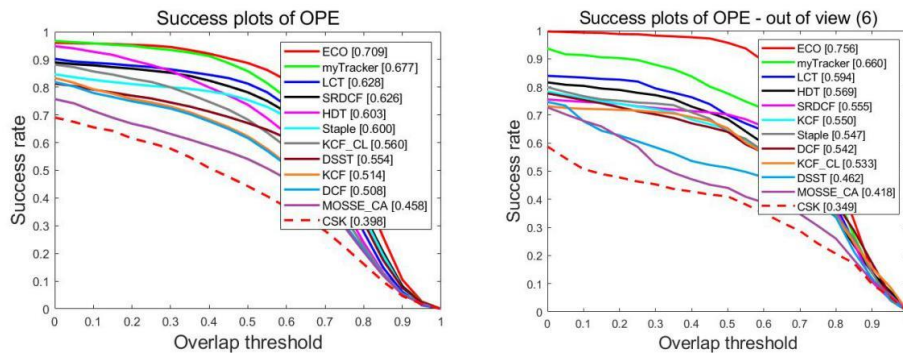


Figure 3. Experiment evaluation on OTB-2013 benchmark.

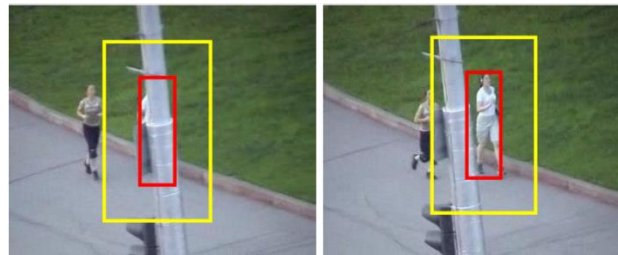


Figure 4. Actual senario of object occlusion.

4. Conclusion

The camera tracking system is of great significance in the era of rapid development of computer vision applications. The target tracking algorithm is applied to the camera. Based on the algorithm of the kernelized filter tracker, the system can keep a real-time object tracking and keep the model work well in a long term. The self-verification module help the system to judge the reliability of the tracking results. The self-detection module run under the unreliable situation, and help search more participants patches to localize target. What's more, we also keep the object target in the center of the screen via camera control. In all, Our camera tracking system effectively provides long-term target tracking and the correctness of the occlusion object to continue tracking.

In our future work, we hope to add color features and deep learning features to the tracker to improve the precision, and add scale feature to deal with the size variation of the tracking object due to distance changes.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Grant No. 61501048), Beijing Natural Science Foundation (Grant No. 4182042), Fundamental Research Funds for the Central Universities (No. 2017RC12), and National Natural Science Foundation of China (U1536111, U1536112).

References

- [1] Arnold WM Smeulders, Dung M Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.
- [2] Sayanan Sivaraman and Mohan Manubhai Trivedi. Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis. *IEEE Transactions on Intelligent Transportation Systems*, 14(4):1773–1795, 2013.
- [3] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2544–2550, 2010.
- [4] R. Patnaik and D. Casasent, “Fast FFT-based distortion-invariant kernel filters for general object recognition,” in *Proceedings of SPIE*, vol. 7252, 2009, p. 1.
- [5] A. Maksai, X. Wang, and P. Fua, “What players do with the ball: A physically constrained interaction modeling,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [6] João F Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.
- [7] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao, “Multistore tracker (muster): A cognitive psychology inspired approach to object tracking,” in *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 749–758.