Tracking a Human Fast and Reliably Against Occlusion and Human-Crossing

Xuan-Phung Huynh, In-Ho Choi, and Yong-Guk Kim(✉)

Department of Computer Engineering, Sejong University, Seoul, Korea
ykim@sejong.ac.kr

Abstract. Tracking a human using the computer vision techniques is essential in the automatic surveillance task. Not only its accuracy and speed but also how it deals with occlusion and human-crossing are the challenges for a reliable tracking framework. Among many, Kernelized Correlation Filter (KCF) has become a state-of-the-art tracker partly because of its high speed, although its performance in dealing diverse situations requires some improvement. We present a new tracking method whereby the reliability is greatly enhanced while maintaining its speed by integrating a Kalman filter with the KCF. The tracker works as follow. After the KCF estimates target's position based on the prediction by the Kalman filter, then the estimated value is given to the updating step of the Kalman filter. During the KCF learning phase, the kernel model is updated using the correct state. Evaluation result using the standard tracking databases suggests that the present tracker outperforms the standard KCF, MOSSE and MIL trackers, respectively. In particular, it is the only tracker that can deal very well with the occlusion and human-crossing tasks, which are the crucial requirements for the high-end surveillance.

Keywords: Surveillance · Human tracking · Occlusion · Human-crossing · Kernelized correlation filters · Kalman filter

1 Introduction

Human tracking has played an important role in the automatic surveillance system [15]. Currently, surveillance [7,13] is contributing vital roles not only in the research but also in the market. Visual tracking is the process of estimating the trajectory of an object in the image plane as it moves around in the scene. According to [15], tracking objects can be complex due to its accuracy and speed as well as how it deals with occlusion and human-crossing cases. Numerous approaches for object tracking have been proposed. Among them, Kernelized Correlation Filter [5,6] has been a state-of-the-art tracker partly because of its high speed and simple implement with only a few lines of code. It is the third performing tracker among many in term of accuracy [10]. And yet, its performance in dealing such occlusion requires some improvement for robust tracking.

T. Bräunl et al. (Eds.): PSIVT 2015, LNCS 9431, pp. 461–472, 2016.
DOI: 10.1007/978-3-319-29451-3_37

In general, a Kalman filter estimates the state of a linear system where state is supposed to be distributed by a Gaussian [2,8,12,15]. It is known that the Kalman filter successfully tracks objects even in the case of occlusion if the assumed type of motion is correctly modeled [2]. This assumption is very strict and it is only suitable for tracking very small objects [15].

Combination of Kalman filter with the kernel method has also been proposed based upon the mean-shift tracking [9,11,17,18]. The mean-shift tracker eliminates the brute force search in the standard template matching and shortens the computation time although it requires that a portion of the objects should be inside a circular region during the initialization stage [15].

In this paper we present a new tracking method whereby the reliability is greatly enhanced while the speed is also maintained by combining a Kalman filter with the KCF. Once KCF estimates a target position based on the prediction by the Kalman filter, the estimated value becomes the observation in updating the object's state. During the KCF learning phase, the correct state of the Kalman filter is utilized to update the kernel model. However, when the tracker meets an occlusion, the Kalman filter omits observation values from KCF and adjusts the state based on the previous state. Experimental results show that the present tracker outperforms the standard KCF, MOSSE (Minimum Output Sum of Squared Error) [1] and MIL (Multiple Instance Learning) trackers [16], respectively. In particular, it is the only tracker that can deal very well with occlusion and human-crossing task, which are critical requirements for the high-end surveillance task.

The rest of the paper is organized as follows. In Sects. 2 and 3, we review the Kernelized Correlation Filter and Kalman Filter, respectively. In Sect. 4, we describe our proposed tracker in detail. In Sect. 5, we present experimental results. Finally, Sect. 6 summarizes this paper.

2 Kernelized Correlation Filters

2.1 Circulant Matrices

KCF trains a linear classifier using both a base sample, i.e. a positive example, and several virtual samples, which serve as the negative examples, obtained by translating it. Here, a cyclic shift operator is utilized in modeling this translation. Because of the cyclic property [4], the signal x becomes identical after n^{th} shifts, and the full set of shifted signals is

$$\{P^u x | u = 0, ..., n-1\} \tag{1}$$

with P is the permutation matrix [5] and n is the number of translation step.

Element of Eq. (1) is a row of a circulant matrix X

$$X = C(x) = \begin{bmatrix} x_1 & x_2 & x_3 & \dots & x_n \\ x_n & x_1 & x_2 & \dots & x_{n-1} \\ x_{n-1} & x_n & x_1 & \dots & x_{n-2} \\ . & . & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ x_2 & x_3 & x_4 & \dots & x_1 \end{bmatrix} \tag{2}$$

Since all circulant matrices are made diagonal by the discrete Fourier transform (DFT) [4], X can be expressed as

$$X = Fdiag(\hat{x})F^H \tag{3}$$

where F is Discrete Fourier transform (DFT) matrix, and $\hat{x}$ is the DFT of the generating vector.

Equation 3 expresses the eigendecomposition of a general circulant matrix and why KCF is fast when it is implemented.

2.2 Fast Kernel Regression

The kernel matrix $K(n \text{ x } n)$ stores the dot-products between all pairs of samples

$$K_{ij} = \kappa(x_i, x_j) = \varphi^T(x_i)\varphi(x_i) \tag{4}$$

with high-dimensional space $\varphi(.)$.

The following kernels satisfy the condition to claim that K is circulant [6]:

- *Radial Basic Function kernels –e.g., Gaussian.*
- *Dot-product kernels –e.g., linear, polynomial.*
- *Additive kernels –e.g., intersection, χ^2 and Hellinger kernels.*
- *Exponentiated additive kernels.*

Therefore, the kernelized version of Ridge Regression is possible to diagonalize

$$\hat{\alpha}^* = \frac{\hat{y}^*}{\hat{k}^{xx} + \lambda} \tag{5}$$

where $K = C(k^{xx})$ is kernel matrix.

Learning phase utilizes Eq. (5) to update the model for next frame.

2.3 Fast Detection

Several candidate patches, $\mathbf{z}$, that can be modeled by cyclic shifts are evaluated by the regression function $f(z)$. To compute efficiently, detection phase diagonalizes regression function

$$\hat{f}(z) = \hat{k}^{xz} \odot \hat{\alpha} \tag{6}$$

where k^{xz} is a kernel correlation of $\mathbf{x}$ and $\mathbf{z}$.

During detecting phase, Eq. (6) predicts where is the center of target within the given frame with a learned coefficients α.

2.4 Fast Kernel Correlation

The kernel correlation of two arbitrary vectors, **x** and **x'**, is the vector $k^{xx'}$ with elements

$$k_i^{xx'} = \kappa(x', P^{i-1}x) = \varphi^T(x')\varphi(P^{i-1}x) \tag{7}$$

Kernel correlation consists of computing the kernel for all relative shifts of two input vectors. This quadratic complexity can be resolved efficiently by diagonalization with DFT.

Depending on the kernel value being unchanged by unitary transformations, such as the DFT, we can use fast kernel correlation for such kernels: dot-product and radial basic function.

Firstly, with dot-product, we have

$$k_i^{xx'} = g(F^{-1}(\hat{x}^* \odot \hat{x'})) \tag{8}$$

where F^{-1} is inverse of DFT. In particular, for a polynomial kernel

$$k_i^{xx'} = (F^{-1}(\hat{x}^* \odot \hat{x'}) + a)^b \tag{9}$$

Secondly, for radial basic function

$$k_i^{xx'} = h(\|x\|^2 + \|x'\|^2 - 2F^{-1}(\hat{x}^* \odot \hat{x'})) \tag{10}$$

As a particularly, useful special case, a Gaussian kernel

$$k_i^{xx'} = exp(-\frac{1}{\sigma^2}(\|x\|^2 + \|x'\|^2 - 2F^{-1}(\hat{x}^* \odot \hat{x'}))) \tag{11}$$

[6] proposed simple Matlab code, with a Gaussian kernel, that can run very fast.

3 Kalman Filter

By assuming the system noise has Gaussian distribution, Kalman filter utilizes the linear dynamical systems to resolve the linear optimal filtering problem. More specifically, x_n is a discrete time system with state at time n. In the next time step $n+1$, the state is

$$x_{n+1} = F_{n+1,n}x_n + w_{n+1} \tag{12}$$

where $F_{n+1,n}$ is the transition matrix from state x_n to x_{n+1}, and w_{n+1} is white Gaussian noise with zero mean and covariance matrix Q_{n+1}.

While the measurement vector z_{n+1} is given by

$$z_{n+1} = H_{n+1}x_{n+1} + v_{n+1} \tag{13}$$

where H_{n+1} is the measurement matrix and v_{n+1} is white Gaussian noise with zero mean and covariance matrix R_{n+1}, that is independent of noise w_{n+1}. The system gets measurement value for each step then estimates the correct state based on the minimum mean-square error of Eq. (13). The solution is a recursive procedure [2,8] as illustrated in Algorithm 1.

Algorithm 1. Kalman filter

1. Initialization:

$$\hat{x}_0 = E[x_0]$$
$$P_0 = E[(x_0 - E[x_0])(x_0 - E[x_0])^T]$$

2. Prediction:

$$\hat{x}_n^- = F_{n,n-1}\hat{x}_{n-1}$$
$$P_n^- = F_{n,n-1}P_{n-1}F_{n,n-1}^T + Q_n$$
$$G_n = P_n^- H_n^T [H_n P_n^- H_n^T + R_n]^{-1}$$

3. Estimation:

$$\hat{x}_n = \hat{x}_n^- + G_n(z_n - H_n\hat{x}_n^-)$$
$$P_n = (I - G_n H_n)P_n^-$$

Goto the Prediction step for the next prediction.

4 The Proposed Method

We propose a new tracker that improves the KCF tracker by correcting target's position with Kalman filter. It is known that KCF often makes failed prediction for the case of occlusion and human crossing simply because the object is disappear. Moreover, performance of the KCF tracker is often deteriorated for rotation, illumination variation, motion blur and etc. We thought that Kalman filter can improve these limitations. In fact, after the KCF estimates target's position based on the prediction by the Kalman filter, and the estimated value is given to the updating step of the Kalman filter. During the KCF learning phase, our tracker uses the correct state to update the kernel model.

First, the fast detection would acquire the peak value $f_{max}(k)$ at $(x(k), y(k))$ according to formula [6] in the current frame. Then, Kalman filter uses this position to adjust the system state to correct the position by using Algorithm 1. After several frames of progress, when Kalman filter is stable, it can correct the position of the target in the current frame. Then, during the KCF's learning phase, it is able to extract the target's feature successfully for the forthcoming frames.

Secondly, in order to resolve the occlusion as well as human crossing cases, we propose a novel approach. When the peak value is less than threshold T, it is assumed that this is the occlusion case. In other words, KCF would predict the incorrect position in that frame, and we could not use it as a measurement value of Kalman filter. During such case, our tracker will use the target's position based on the prediction step by Algorithm 1, and only this step will run. Figure 1 and Algorithm 2 describe it in detail.

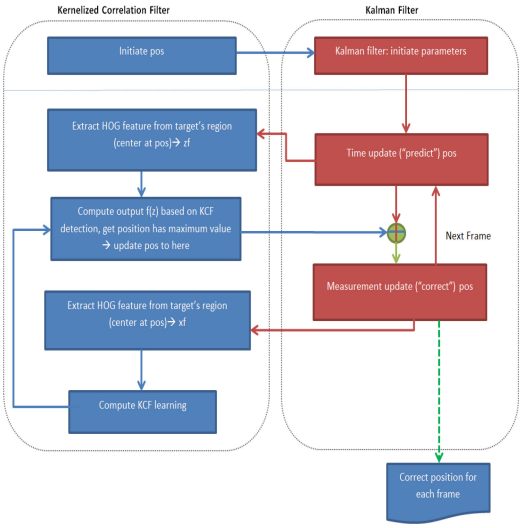


Fig. 1. Flowchart of our tracker, which basically combines KCF and Kalman filter.

Algorithm 2. Kernelized Correlation Filter with Kalman Filter

1. Initialization
   $pos \leftarrow target's\ center\ (first\ frame)$
   $\hat{x}_0 = pos$
   $P_0 = E[(x_0 - E[x_0])(x_0 - E[x_0])^T]$
2. Kalman prediction
   $pos = \hat{x}_n^-$
3. KCF detection
   $zf \leftarrow HOG\ target's\ feature\ (center\ at\ pos)$
   $f_{peak}, pos' \leftarrow max(\hat{k}^{xz} \odot \hat{\alpha})$
   if $f_{peak} < T$ **then**
      Goto step 5
   else
      $z_n = pos'$
      Goto step 4
   end if
4. Kalman estimation
   $pos = \hat{x}_n$
5. KCF learning
   $xf \leftarrow HOG\ target's\ feature\ (cente\ at\ pos)$
   $\hat{\alpha}^* = \frac{\hat{y}^*}{\hat{k}^{xx}+\lambda}$

Goto step 2 for new frame.

Table 1. Average precision and fps of 5 trackers

	Mean precision (%)	Standard deviation precision (%)	Mean fps	Standard deviation fps
Our tracker	**96.7**	5.7	138	125.3
KCF on HOG	90.7	18.3	148	121.6
KCF on raw pixels	67.9	33.6	152	135.1
MOSSE	55.3	33.4	**254**	207.4
MIL	41.3	33.2	145	41.45

5 Experiments

5.1 Implementation Details

The present tracker is implemented using Matlab library. The pipeline for the tracker is illustrated in Algorithm 2. Some heuristics are used for threshold value T and Kalman filter. For instance, the sampling rate, acceleration magnitude, process noise, and measurement noise are given as 1 frame/s, 0.2, 0.5, and 1.0, respectively. Performance of our tracker is compared against KCF, MIL and MOSSE trackers. All trackers are implemented using Matlab on Windows 7 running on a computer having an i7 CPU with 16 GB RAM.

5.2 Evaluation

Tracking Dataset. To evaluate the performance, we use 38 video sequences, which are taken from the standard tracking benchmark dataset [3,14][1]. These videos consist of different challenges for tracking such as illumination variation, rotation, scale, motion blur, occlusion, and human crossing. Performance of the present tracker is compared with those of KCF, MOSSE, and MIL tracker, respectively, by implementing all trackers within the same computer.

Results. The precision curve and the frame rate (fps) are evaluated for each tracker [6,14]. In this metrics, the ratio of successfully tracked frame is assessed by a set of thresholds within the precision plot. By setting the threshold at 20, we compute the average precision and fps as well as the standard deviation as shown in Table 1. The mean precisions of MIL, MOSSE, KCF, KCF on HOG and our tracker are 41.3, 55.3, 67.9, 90.7 and 96.7 %, respectively. Figure 2 shows the performance curves of five trackers by varying the threshold, suggesting that the present tracker is the best among them. Regarding to the speed, the mean fps of the KCF on HOG case is 148 and the present one runs at 138 fps. Both can run in the real-time applications.

[1] Dataset is available at https://goo.gl/2bPKi7.

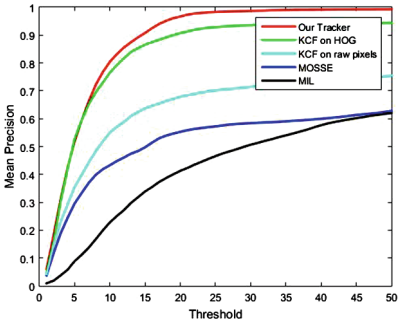
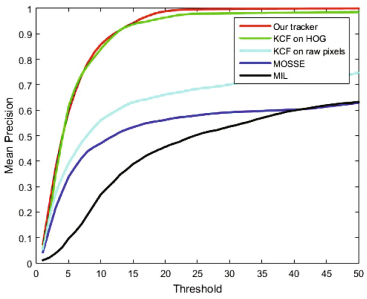


Fig. 2. Comparing the trackers' performance for full datasets.

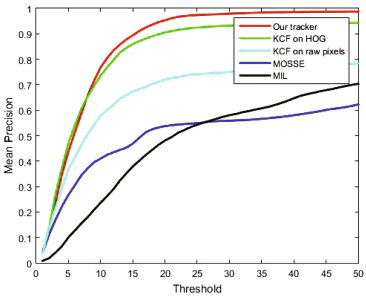
Table 2. Average precision of 5 trackers with different categories and datasets (%)

Category	Datasets	Our tracker	KCF on HOG	KCF on raw pixels	MOSSE	MIL
Illumination variation	Boy, Car1,Fish, Car2, Crossing, Car24, RedTeam, Car4, CarDark, Crowds, Human8, Man, Mhyang, Singer1, Singer2, Skater, Subway, Sylvester, Trellis, Walking, Suv, Woman	**98.8**	96.4	66.2	56.2	45.6
Rotation	Board, Boy, CarScale, Dancer, Dancer2, David2, David3, Doll, Dudek, Gym, Liquor, Mhyang, MountainBike, RedTeam, Singer2, Skater, Sylvester, Tracjectory10, Tracjectory16, FaceOcc2, Trellis	**95.3**	90.6	72.1	53.7	48.1
Scale variation	Board, Car4, CarDark, CarScale, Dancer, Doll, Dudek, Fish, Jogging, Liquor, Mhyang, RedTeam, Singer1, Singer2, Skater, Sylvester, Tracjectory10, Tracjectory16,Walking, Woman	**94.8**	86.3	65.6	52.1	42.8
Motion blur	Car1, Car24, Woman, CarDark, Dancer, Dancer2, Suv, Doll, Dudek, Fish, Gym, Mhyang, RedTeam, Singer2, Subway, Boy	**98.2**	95.6	67.7	44.5	51.5
Occlusion	Car1, Car24, CarScale, David3, Doll, FaceOcc1, FaceOcc2, Jogging, Liquor, Suv,Tracjectory16, Walking, Woman	**95.1**	81.9	57.8	47.7	36.4
Human-crossing	Crowds, Jogging, Subway, Tracjectory10, Tracjectory16	**97.3**	69.9	60.8	60.2	15.2

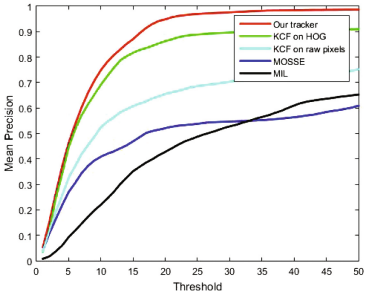
To analyze the detail characteristics, the dataset is divided into six categories: illumination variation, rotation, scale, motion blur, occlusion, and human-crossing. The mean precision for each tracker is obtained for each category as shown in Table 2. Figure 3 shows six graphs. As you can see, the present tracker shows great improvement compared with the KCF on HOG tracker, particularly in the categories of the occlusion and human-crossing cases. This result suggests



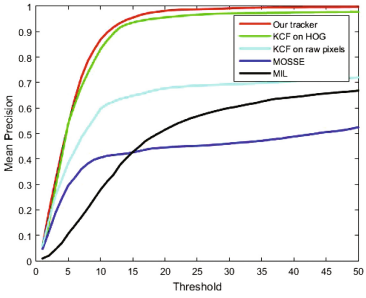
(a) Illuminated datasets.



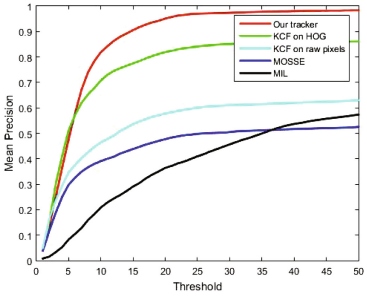
(b) Rotated datasets.



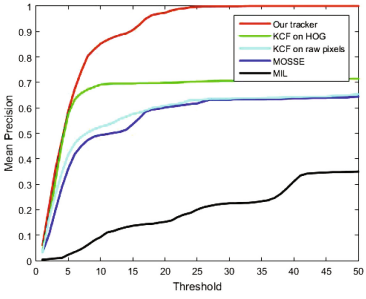
(c) Scaled datasets.



(d) Motion datasets.



(e) Occlusion datasets.



(f) Human-crossing datasets.

Fig. 3. Comparing the trackers' performance for sequence attribute.

that Kalman filter plays an important role in making a directional decision when the object drifts into occlusion or other object such as human.

Further Analysis for the Human Crossing Case. Since the present tracker shows the best performance for the human-crossing case, it would be necessary to analyze a bit further. Figure 4 demonstrates how 3 trackers, i.e. MIL, MOSSE, and KCF, could not track the walking man correctly except the present one.

(a) MIL tracker starts to fail at these frames.

(b) MOSSE tracker starts to fail at these frames.

(c) KCF tracker starts to fail at these frames.

(d) Our tracker is successful.

Fig. 4. Comparing the trackers' performance for the human-crossing case.

It seems obvious that those trackers could not deal well with occlusion and human-crossing.

6 Conclusions

In this paper, we present a new tracking framework that combines the best features of the Kernelized Correlation filter and Kalman filter. Using the KCF, we acquire an estimation of the target's location, which is corrected by Kalman filter for adapting object's position. It is found that our tracker outperforms the state-of-art trackers such as KCF, MOSSE and MIL. It is particularly excellent in the occlusion and human-crossing cases. Given that KCF is one of the fastest trackers so far, our tracker can be used for the real-time applications such as the high-end surveillance.

Acknowledgments. This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education, Science and Technology (NRF-2013R1A1A2006969) and by the MSIP(Ministry of Science, ICT and Future Planning), Korea, under the Global IT Talent support program (IITP-2015-R0134-15-1032) supervised by the IITP(Institute for Information and Communication Technology Promotion).

References

1. Bolme, D.S., Beveridge, J.R., Draper, B., Lui, Y.M., et al.: Visual object tracking using adaptive correlation filters. In: 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2544–2550. IEEE (2010)
2. Cuevas, E.V., Zaldivar, D., Rojas, R., et al.: Kalman filter for vision tracking (2005)
3. Ferryman, J., Ellis, A.: Pets 2010: dataset and challenge. In: 2010 Seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), pp. 143–150. IEEE (2010)
4. Gray, R.M.: Toeplitz and Circulant Matrices: A Review. Now Publishers Inc, Hanover (2006)
5. Caseiro, R., Martins, P., Batista, J., Henriques, J.F.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012)
6. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. IEEE Trans. Pattern Anal. Mach. Intell. **37**(3), 583–596 (2015)
7. Huang, T.: Surveillance video: the biggest big data. Computing Now **7**(2) (2014). http://www.computer.org/web/computingnow/archive/february2014. IEEE Computer Society
8. Kalman, R.E.: A new approach to linear filtering and prediction problems. J. Fluids Eng. **82**(1), 35–45 (1960)
9. Karavasilis, V., Nikou, C., Likas, A.: Visual tracking by adaptive kalman filtering and mean shift. In: Konstantopoulos, S., Perantonis, S., Karkaletsis, V., Spyropoulos, C.D., Vouros, G. (eds.) SETN 2010. LNCS, vol. 6040, pp. 153–162. Springer, Heidelberg (2010)
10. Kristan, M., et al.: The visual object tracking VOT2014 challenge results. In: Agapito, Lourdes, Bronstein, Michael M., Rother, Carsten (eds.) ECCV 2014 Workshops. LNCS, vol. 8926, pp. 191–217. Springer, Heidelberg (2015)
11. Lu, H., Zhang, R., Chen, Y.W.: Head detection and tracking by mean-shift and kalman filter. In: 3rd International Conference on Innovative Computing Information and Control, ICICIC 2008, pp. 357–357. IEEE (2008)
12. Mohinder, S.G., Angus, P.A.: Kalman filtering: theory and practice using matlab. John Wileys and Sons, Hoboken (2001)
13. Ojha, S., Sakhare, S.: Image processing techniques for object tracking in video surveillance-a survey. In: 2015 International Conference on Pervasive Computing (ICPC), pp. 1–6. IEEE (2015)
14. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: A benchmark. In: 2013 IEEE Conference on Computer vision and pattern recognition (CVPR), pp. 2411–2418. IEEE (2013)
15. Yilmaz, A., Javed, O., Shah, M.: Object tracking: a survey. ACM Computing Surveys (CSUR) **38**(4), 13 (2006)

16. Zhang, K., Song, H.: Real-time visual tracking via online weighted multiple instance learning. Pattern Recogn. **46**(1), 397–411 (2013)
17. Zhao, J., Qiao, W., Men, G.Z.: An approach based on mean shift and kalman filter for target tracking under occlusion. In: 2009 International Conference on Machine Learning and Cybernetics, vol. 4, pp. 2058–2062. IEEE (2009)
18. Zhu, Z., Ji, Q., Fujimura, K., Lee, K.: Combining kalman filtering and mean shift for real time eye tracking under active IR illumination. In: Proceedings 16th International Conference on Pattern Recognition, vol. 4, pp. 318–321. IEEE (2002)