Journal of
Electronic Imaging

JElectronicImaging.org

Real-time tracking of deformable objects based on combined matching-and-tracking

Junhua Yan
Zhigang Wang
Shunfei Wang

SPIE. IS&T imaging.org

Junhua Yan, Zhigang Wang, Shunfei Wang, "Real-time tracking of deformable objects based on combined matching-and-tracking," *J. Electron. Imaging* **25**(2), 023011 (2016), doi: 10.1117/1.JEI.25.2.023011.

Real-time tracking of deformable objects based on combined matching-and-tracking

Junhua Yan,[a,b,*] **Zhigang Wang,**[a] **and Shunfei Wang**[a]
[a]Nanjing University of Aeronautics and Astronautics, College of Astronautics, Nanjing 210016, China
[b]613th Research Institute of Aviation Industry Corporation of China, Science and Technology on Electro-optic Control Laboratory, Luoyang, Henan 471009, China

Abstract. Visual tracking is very challenging due to the existence of several sources of variations, such as partial occlusion, deformation, scale variation, rotation, and background clutter. A model-free tracking method based on fusing accelerated features using fast explicit diffusion in nonlinear scale spaces (AKAZE) and KLT features is presented. First, matching-keypoints are generated by finding corresponding keypoints from the consecutive frames and the object template, then tracking-keypoints are generated using the forward–backward flow tracking method, and at last, credible keypoints are obtained by AKAZE-KLT tracking (AKT) algorithm. To avoid the instability of a statistical method, the median method is adopted to compute the object's location, scale, and rotation in each frame. The experimental results show that the AKT algorithm has strong robustness and can achieve accurate tracking especially under conditions of partial occlusion, scale variation, rotation, and deformation. The tracking performance shows higher robustness and accuracy in a variety of datasets and the average frame rate reaches 78 fps, showing good performance in real time. [DOI: 10.1117/1.JEI.25.2.023011]

Keywords: AKAZE matching; KLT; the fusion feature; real-time tracking; median function.

Paper 15541 received Jul. 4, 2015; accepted for publication Mar. 2, 2016; published online Mar. 28, 2016.

1 Introduction

Visual object tracking, which is the process of estimating the motion parameters such as location, scale, and rotation of the object in an image sequence given the initial box in the first frame, is a popular problem in computer vision, with wide-ranging applications including visual navigation, military reconnaissance, and human–computer interaction.[1,2] Although significant progress has been made in recent years, the problem is still difficult due to factors such as partial occlusion, deformation, scale variation, rotation, and background clutter.[3] To solve these problems, numerous algorithms have been proposed.[4–6]

The online learning algorithm is one of the useful algorithms that has been widely used to solve the problem of objects' changing appearance. As some information of the objects to be tracked is known in advance in various scenarios, it is possible to employ prior knowledge to design the tracker. However, for other applications, as nothing about the objects of interest is known beforehand, no prior knowledge can be of use. Also, it is impossible to employ offline machine learning techniques to achieve efficient tracking because the appearance of an object is likely to vary due to its constant movements and also under different environmental conditions, such as varying level of brightness.[7,8] Instead, online learning algorithms have been employed to adapt the object model to the abovementioned uncertainties. In practice, however, updating a model often introduces errors as it is difficult to explicitly assign hard class labels.

To efficiently track the constantly changing object and avoid the errors caused by an online learning algorithm, a model that precisely represents the object is needed. Various forms of representation of the object are used in practice, for example: points,[9,10] contours,[11,12] optical flow,[13,14] or articulated models.[15,16] Models that decompose the object into parts are more robust,[17,18] as local changes only affect individual parts. Even when individual parts are lost or in an erroneous state, other object parts can still represent the object well. Keypoint, such as SIFT,[19] SURF,[20] ORB,[21] AKAZE,[22] and so on, is a representative kind of local feature that has been widely used in image fusion, object recognition, and other fields.

In this paper, a model-free tracking method based on fusing AKAZE and KLT features is proposed. The brief procedure is as follows: first, generate matching-keypoints by finding corresponding keypoints from the consecutive frames and the object template, then generate tracking-keypoints using the forward–backward flow tracking method, and at last, obtain credible keypoints by AKT fusion algorithm. To avoid the instability of a statistical method, the median method is adopted to compute object's location, scale, and rotation in each frame.

2 Background Work

AKAZE[22] is regarded as the improved version of SIFT features and SURF. It is a more stable feature detection algorithm. Traditional SIFT and SURF feature detection algorithms build scale space by the linear Gaussian pyramid. However, this kind of linear decomposition can cause loss of accuracy, object's edge blur, and loss of details. In order to solve these problems, AKAZE algorithm uses the method based on nonlinear scale space. The fast explicit diffusion (FED)[23] is used to construct scale space. By using this method, any step length can be applied. Compared to

*Address all correspondence to: Junhua Yan, E-mail: yjh9758@126.com.

SIFT and SURF, the computational complexity is greatly reduced and the robustness is improved. In the following subsections, the detailed procedures of constructing nonlinear scale space using FED scheme will be illustrated. The process of feature detection and the effects of feature description of AKAZE algorithm based on modified-local difference binary (M-LDB) will then be discussed.

2.1 *Building Nonlinear Scale Space*

Similar to SIFT in the construction of the nonlinear scale space, scale level increases logarithmically. The scale space constructed has $P^t_{ai}(x, y)$ octaves and each octave has V^a layers. Different octaves and layers are marked with serial numbers o and s, respectively. The relationship between them and the scale parameter σ is shown in the equation below:

$$\sigma_i(o, s) = 2^{o+s/S}, \tag{1}$$

where $o \in [0 \dots O-1]$, $s \in [0 \dots S-1]$, $i \in [0 \dots M-1]$. M is the total number of images that remain after filtration by the filter. Since the nonlinear diffusion filter is based on the scale of time, scale parameters σ_i with the unit of pixel is transformed to the unit of time, as shown below:

$$t_i = \frac{1}{2}\sigma_i^2, i \in [0 \dots M], \tag{2}$$

where t_i is $o \in [0 \dots O-1]$, $s \in [0 \dots S-1]$, $i \in [0 \dots M]$ called evolutionary time. For each input image, a Gaussian filter is first applied, then the gradient histogram of the image is calculated. The contrast factor $P^t_i(x, y)$ is set as 70% of the gradient histogram. In the case of two-dimensional (2-D) images, since the image derivative is one pixel grid size, the maximal step size $t_{\max}$ is 0.25 without violating stable conditions. Then by using a set of evolutionary time t_i, all the images of scale space can be obtained using FED scheme.

2.2 *Feature Detection*

Feature detection of AKAZE is achieved by computing the Hessian local maxima after normalization of various scales for the filtered images in the nonlinear scale space. Calculation of a Hessian matrix is as follows:

$$L^i_{\text{Hessian}} = \sigma^2_{i,\text{norm}}(L^i_{xx}L^i_{yy} - L^i_{xy}L^i_{xy}), \tag{3}$$

where $\sigma_{i,\text{norm}} = \sigma_i/2^{o^i}$. For computing the second order derivatives, the concatenated Scharr filters with step size $\sigma_{i,\text{norm}}$ are applied. First, search for maxima of the detector response in spatial location. Check that the detector response is higher than a predefined threshold and that it is a maxima in a window of 3×9 pixels of three adjacent sublevels. Finally, the 2-D position of the keypoint is estimated with subpixel accuracy by fitting a 2-D quadratic function to the determinant of the Hessian response in a 3×3 pixels neighborhood and finding its maximum.

2.3 *Feature Description*

The diagram in Fig. 1, as supplied by Ref. 22, demonstrates LDB[24] and M-LDB tests between grid divisions around a keypoint. The intensity is expressed by colorful grids and the gradients in x are expressed by the arrows. The feature description of AKAZE algorithm is based on M-LDB that exploits gradient and intensity information from the nonlinear scale space. And there are two main improvements of M-LDB compared with LDB: (1) rotation invariance is obtained by estimating the main orientation of the keypoint, as is done in KAZE,[25] and rotating the grid of LDB accordingly. (2) A function of the scale σ is used as the subsample grids in steps instead of using the average of all pixels inside each subdivision of the grid. The scale-dependent sampling in turn makes the descriptor robust to changes in scale.

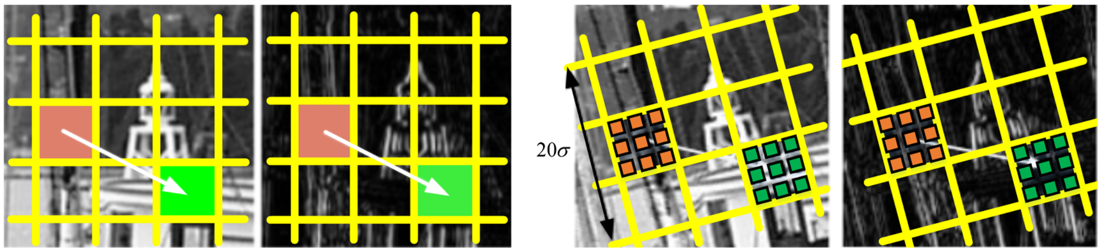
3 Fusing AKT Tracking

3.1 *Forward–Backward Flow Tracking*

Because of the environmental impact or object's appearance change, the results of KLT often produce deviation, an evaluation method needs to be established to judge the accuracy of tracking results. Forward–backward error,[26] which is based on the forward–backward continuity assumption, can effectively estimate the trajectory error of keypoints, i.e., if the object tracking is correct, then the tracking results are independent of time.

As shown in Fig. 2, for two adjacent frame I_{t-1} and I_t, x_{t-1} is a random keypoint from object template in the frame I_{t-1}, x_t is the corresponding keypoint of x_{t-1} in the frame I_t using forward tracking, and $\hat{x}_{t-1}$ is the corresponding keypoint of x_t in the frame I_{t-1} using backward tracking. Forward–backward error is defined as the Euclidean distance between two keypoints in frame I_{t-1}, i.e., $e^{FB}_{t-1} = \|x_{t-1} - \hat{x}_{t-1}\|$. If error e^{FB}_{t-1} is bigger than a threshold which we set, the keypoint will be tracked falsely.

We set the location of keypoint and status of forward–backward error as a pair pair (keypoint, status). If the status corresponding to keypoint is TRUE, which means the status



(a)

(b)

Fig. 1 Binary test: (a) LDB and (b) M-LDB.

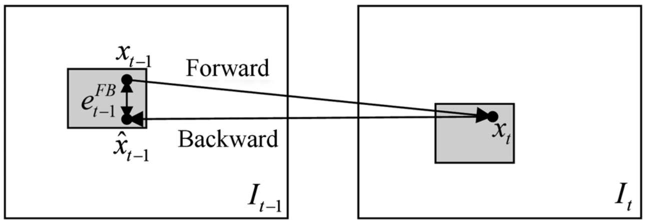


Fig. 2 Forward–backward error in two adjacent frames.

of forward KLT and backward KLT themselves both must be TRUE, and error e^{FB}_{t-1} is smaller than the Euclidean distance threshold, then we call the keypoint with TRUE status tracking-keypoint. The rest are called failing tracking-keypoint.

3.2 *Model of AKT*

When calculating the homographic matrix between the initial keypoints and the current keypoint based on the traditional AKAZE algorithm, robust statistical methods, such as RANSAC and LMEDS, are usually adopted. However, when the number of outliers is too much, homographic matrix estimation will get poor results. So, in this paper, we put forward a tracking model called AKT, which can fundamentally eliminate the false matching-keypoints and reduce the proportion of outliers to effectively solve the problem of inaccurate parameter estimation.

The diagram in Fig. 3 demonstrates how the AKT algorithm fuses the matching-keypoints and tracking-keypoints by AKT algorithm. The collection of V^a is composed of matching-keypoints $P^t_{ai}(x, y)$ in the t'th frame corresponding to the keypoints in object template obtained by AKAZE matching algorithm. And these matching-keypoints are represented by black circles in Fig. 3. The collection of V^k is composed of tracking-keypoints $P^t_{ki}(x, y)$ in the t'th frame corresponding to the keypoints in object template obtained by KLT algorithm. And these tracking-keypoints are represented by gray circles in Fig. 3. There is a one-to-one correspondence between matching-keypoints and tracking-keypoints. Keypoints surrounded by the curve are credible keypoints in the t'th frame, which will make contributions to calculating an object's location, scale, and rotation. The rest of the key points are outliers and thus, they are deleted.

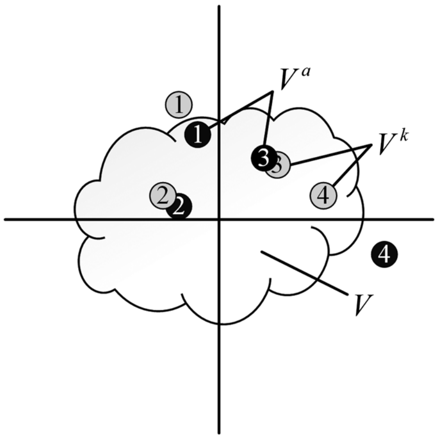


Fig. 3 The model of AKT.

The credible keypoints are obtained by fusing matching-keypoints and tracking-keypoints. Its collection is V.

Sort the Euclidean distance l^t_i between the i'th pair of matching-keypoints and tracking-keypoints in the t'th frame in descending order, then the experiments show that the optimal value l^t_{Th} to be set as maximum allowable deviation threshold is in 0.26th of the distance sequence because enough credible keypoints are ensured, and the obvious false matching-keypoints can be removed. This means that the all but the bottom 0.74th pairs of points are valid matches. Set keypoint $P^t_i(x, y)$ as the center, a as the width and the height of the patch as M^t_i. The degree of similarity between two patches is defined as

$$\alpha(M_i, M^t_i) = 0.5[\beta_{\text{NCC}}(M_i, M^t_i) + 1], \tag{4}$$

where β_{NCC} is the normalization correlation coefficient. Set minimum allowed similarity threshold to be α_{Th}, the set V of credible keypoints is composed of three parts: (1) when the Euclidean distance between the i'th pair of matching-keypoints and tracking-keypoints satisfies $l^t_i \le l^t_{Th}$, keypoints $P^t_{ai}(x, y) \in V$; (2) when $l^t_i > l^t_{Th}$, AKAZE match or KLT track may cause an error, lead to an excessively large deviation, so mistakenly deleted credible keypoints can be screened out by referring to similarity, namely if $\alpha(M_i, M^t_i) > \alpha_{Th}$, matching-keypoints $P^t_{ai}(x, y) \in V$; and (3) if $\alpha(M_i, M^t_i) > \alpha_{Th}$, tracking-keypoints $P^t_{ki}(x, y) \in V$.

3.3 *Bounding Box*

The traditional ways to calculate the homographic matrix are statistical methods, such as RANSAC and LMEDS. However, experiments show that the estimation of homography gives poor results for nonplanar objects, even though the keypoint association was performed correctly.[27] So, in this paper, the median method is put forward to compute object's location, scale, and rotation in each frame.

As shown in Fig. 4, $P_{\text{center}}(x, y)$ and $P^t_{\text{center}}(x, y)$ represent the center of the initial template and the object's bounding box in the t'th frame, respectively. $P_i(x, y)$ and $P^t_i(x, y)$ represent credible keypoints of the initial template and that in the t'th frame. θ_n and θ^t_n represent the angle between the i and $i + 1$ keypoints of the initial template and that in the t'th frame. d_n and d^t_n, respectively, represent the Euclidean distance between the keypoints in the initial template and that in the t'th frame. With the following equations, the relative changing rate of position, scale and rotation angle can be calculated:

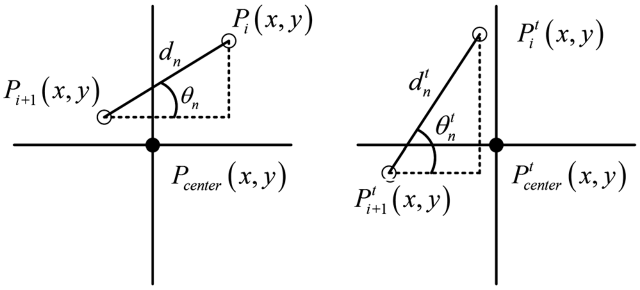


Fig. 4 The median method to get object's location, scale, and rotation.

$$d^t_{\text{center}}(x,y)=\text{median}(\|P^t_i(x,y)-P_i(x,y)\|),i\in[1,N], \tag{5}$$

$$s^t_{\text{center}}=\text{median}(d^t_n/d_n),n\in[1,(N-1)!], \tag{6}$$

$$\theta^t_{\text{center}}=\text{median}(\theta^t_n-\theta_n),n\in[1,(N-1)!], \tag{7}$$

where median represents the function of calculating median. Set the four vertices' coordinates of initial tracking box as $P_{ri}(x,y), i=[1,4]$, its relative offset to the center of initial tracking box is $P_{di}(x,y)$, $i=[1,4]$, in the t'th frame, the vertices' coordinates of tracking box can be obtained by the following equations:

$$P^t_{\text{center}}(x,y)=P_{\text{center}}(x,y)+d^t_{\text{center}}(x,y), \tag{8}$$

$$x^t_{\text{rotate}}=\cos\,\theta^t_{\text{center}}\cdot x_{P_{di}}-\sin\,\theta^t_{\text{center}}\cdot y_{P_{di}}, \tag{9}$$

$$y^t_{\text{rotate}}=\cos\,\theta^t_{\text{center}}\cdot y_{P_{di}}+\sin\,\theta^t_{\text{center}}\cdot x_{P_{di}}, \tag{10}$$

$$P^t_{ri}(x,y)=P^t_{\text{center}}(x,y)+s^t_{\text{center}}\cdot P^t_{\text{rotate}}(x^t_{\text{rotate}},y^t_{\text{rotate}}),i=[1,4], \tag{11}$$

where x^t_{rotate} and y^t_{rotate}, respectively, represent the x-coordinate and y-coordinate after rotation. $P^t_{ri}(x,y)$ are the four vertices' coordinates of tracking box in the t'th frame. The tracking box $B=(b_1,b_2,\dots b_n)$ of each frame can be obtained through the calculation above.

3.4 *Algorithm Procedure*

Given a sequence of images $I_1,\dots,I_n$ and an initializing region b_1 in I_1, our aim in each frame of the sequence is to recover the box of the object of interest. Steps of the AKT Algorithm 1 are as follows:

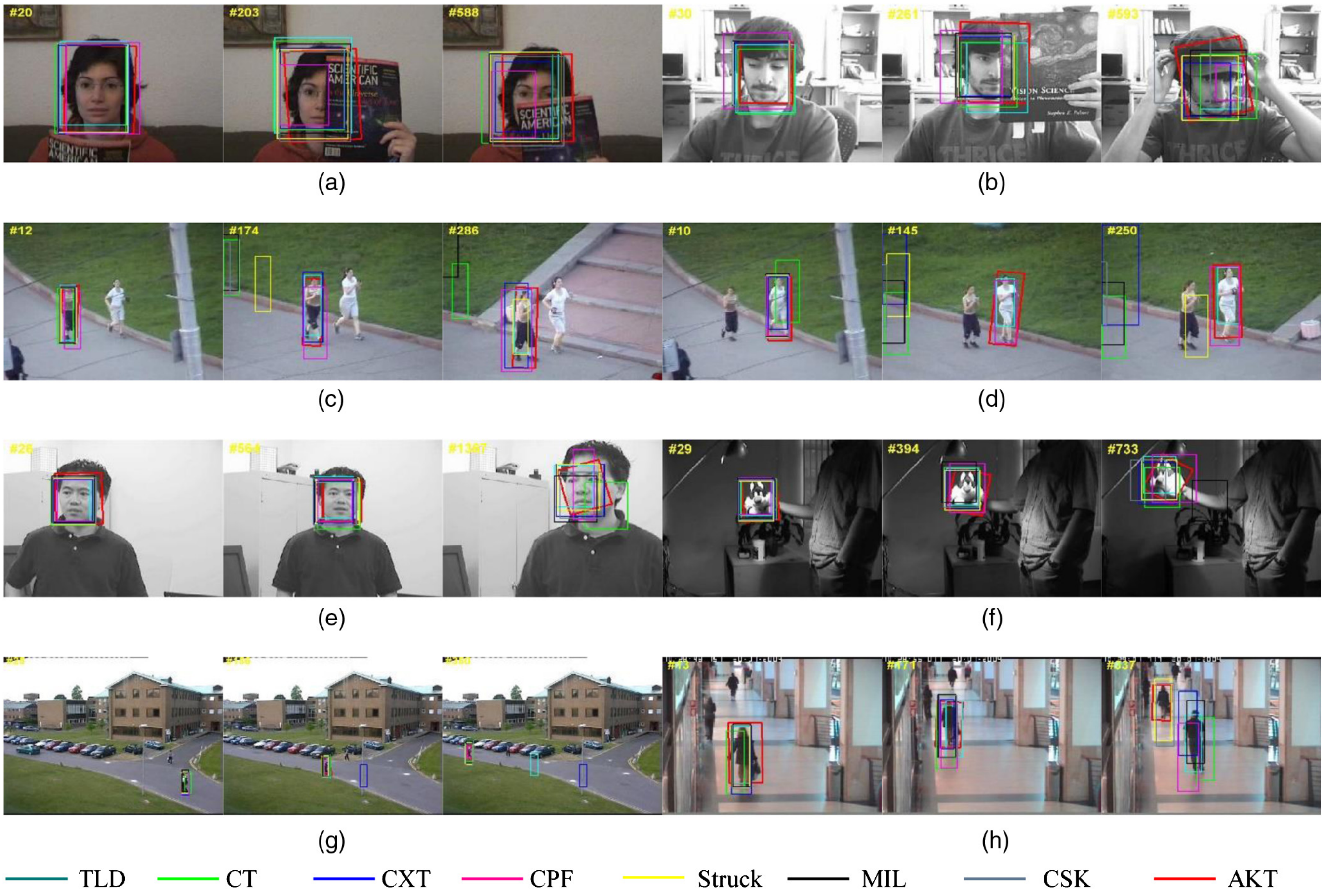
4 Experimental Results

We evaluated the proposed tracking algorithm based on fusing AKAZE and KLT (AKT) algorithm using sequences, as supplied by Ref. 28, with challenging factors including partial occlusion, drastic illumination changes, nonrigid deformation, background clutter, and motion blur. We compared the proposed AKT tracker with seven state-of-the-art methods: tracking-learning-detection (TLD),[14] compressive tracker (CT),[29] context tracker (CXT),[30] color-based probabilistic tracking (CPF),[31] structured output tracking with kernels (Struck),[32] multiple instance learning tracker (MIL)[33] and the circulant structure of tracking with kernels (CSK).[34] All data in the experimental results and the quantitative evaluation are based on the unified dataset and the same initial state conditions. Since our algorithm focuses primarily on the challenges of partial occlusion, deformation, rotation, and scale variation, we only include eight of the videos that mainly contain these challenges and neglect the others in the following discussions. Additionally, the results of precision and success rate are based on 22 videos, in which the good ones are as shown in Fig. 5 and Table 1. Experimental environment: Visual Studio 2013 + OpenCV3.1.0. Equipment is configured to: 2.00 GHz, dual processor, a 64-bit operating system, the 32-Gb installed memory.

There are a range of measures available in previous research for assessing the performance of tracking algorithms quantitatively. Many authors employ the center-error measure that expresses the distance between the centroid of the algorithmic output and the centroid of the ground truth. This measure is only a rough assessment of the localization. Since it is not bounded, the comparison of results obtained

Algorithm 1: Fusing AKAZE-KLT tracking.

```
Input: Sequences of images $S=(I_1,I_2,\dots,I_n)$ and initializing object template $b_1$.
$P_i(x,y)\leftarrow$AKAZE_detect($I_1$), detect and describe keypoints of object template in the first frame using AKAZE algorithm.
for $t=2\dots n$ do
    $P^t_{di}(x,y)\leftarrow$AKAZE_detect[$I_t$(ROI)], detect and describe keypoints of search window in the $t^{th}$ frame
    $P^t_{ai}(x,y)\leftarrow$AKAZE_match($P_i,P^t_{di}$), match keypoints of object template and search window using AKAZE algorithm.
    $P^t_{ki}(x,y)\leftarrow$KLTtrack[$P_i(x,y),I_1,I_t$], track keypoints of object template in search window in the $t^{th}$ frame using forward-backward KLT algorithm.
    $P^t_i(x,y)\leftarrow$fuse[$P^t_{ai}(x,y),P^t_{ki}(x,y)$], fuse the results of AKAZE matching and KLT tracking using AKT algorithm.
    $d^t_{\text{center}}(x,y)=\text{median}(\|P^t_i(x,y)-P_i(x,y)\|)$
    $s^t_{\text{center}}=\text{median}(d^t_n/d_n),n\in[1,(N-1)!]$
    $\theta^t_{\text{center}}=\text{median}(\theta^t_n-\theta_n),n\in[1,(N-1)!]$
    $b_t\leftarrow\{P^t_{r1}(x,y),\dots,P^t_{r4}(x,y)\}$, the tracking box is acquired by coordinates of four vertices.
end for
Output: Tracking box $B=(b_1,b_2,\dots b_n)$, tracking location $d^t_{\text{center}}(x,y)$, tracking scale $s^t_{\text{center}}$, tracking rotation $\theta^t_{\text{center}}$.
```



(a) (b) (c) (d) (e) (f) (g) (h)

Fig. 5 The tracking results of AKT algorithm on different sequences: (a) FaceOcc1, (b) FaceOcc2, (c) Jogging1, (d) Jogging2, (e) Mhyang, (f) Sylvester, (g) Walking, and (h) Walking2.

Table 1 The CLE and average frame per second (pixel/FPS).

Sequence[26]	TLD[14]	CT[27]	CXT[28]	CPF[29]	Struck[30]	MIL[31]	CSK[32]	AKT
FaceOcc1	32.9/12.3	32.0/42.3	22.6/10.1	31.7/25.2	**2.6**/9.8	31.0/24.0	16.9/**108.2**	**12.0/41.0**
Gym	15.7/19.1	26.5/49.7	8.7/6.5	21.8/50.5	9.3/7.2	16.8/23.8	11.0/**109.9**	23/**52.1**
Jogging1	**11.3**/20.0	92.7/58.5	49.5/23.1	21.9/51.6	49.0/10.2	94.4/23.8	236.0/**170.8**	**11.7/82.5**
Jogging2	**14.3**/16.1	138.6/59.4	125.4/25.5	20.8/45.9	89.0/10.0	136.8/26.4	98.6/**134.5**	**7.9/72.4**
Mhyang	8.9/15.1	25.8/46.3	5.5/11.0	15.5/**102.5**	**5.3**/9.5	15.2/27.5	**5.4/148.4**	8.2/67.7
Sylvester	12.5/16.3	13.5/45.2	20.5/4.5	16.2/57.2	**7.8**/7.0	14.3/25.9	**10.2/150.5**	13.7/**85.7**
Walking	64.5/18.8	78.6/32.0	168.8/9.8	**4.6**/53.1	6.4/10.5	5.6/25.0	7.7/**186.4**	**5.2/117.5**
Walking2	24.3/20.1	65.6/48.3	30.4/14.8	49.9/52.1	**13.9**/10.6	35.5/31.5	28.8/**150.9**	**13.1/104.1**
Average CLE	23.0	59.2	53.9	**22.8**	25.2	43.7	39.3	**11.9**
Average FPS	17.2	47.6	13.2	54.8	9.3	26.0	**144.9**	**77.9**

on different sequences is difficult. So, we also employed the widely used overlap measure

$$o(b_T, b_{GT}) = \frac{b_T \cap b_{GT}}{b_T \cup b_{GT}}, \tag{12}$$

where b_T is the tracker output and b_{GT} refers to the manually annotated bounding box, ∪ represents union, namely, the overlap of b_T, and b_{GT}, ∩ represents intersection of these boxes. The overlap rate is a better indicator for per-frame success when bounded between 0 and 1.[35]

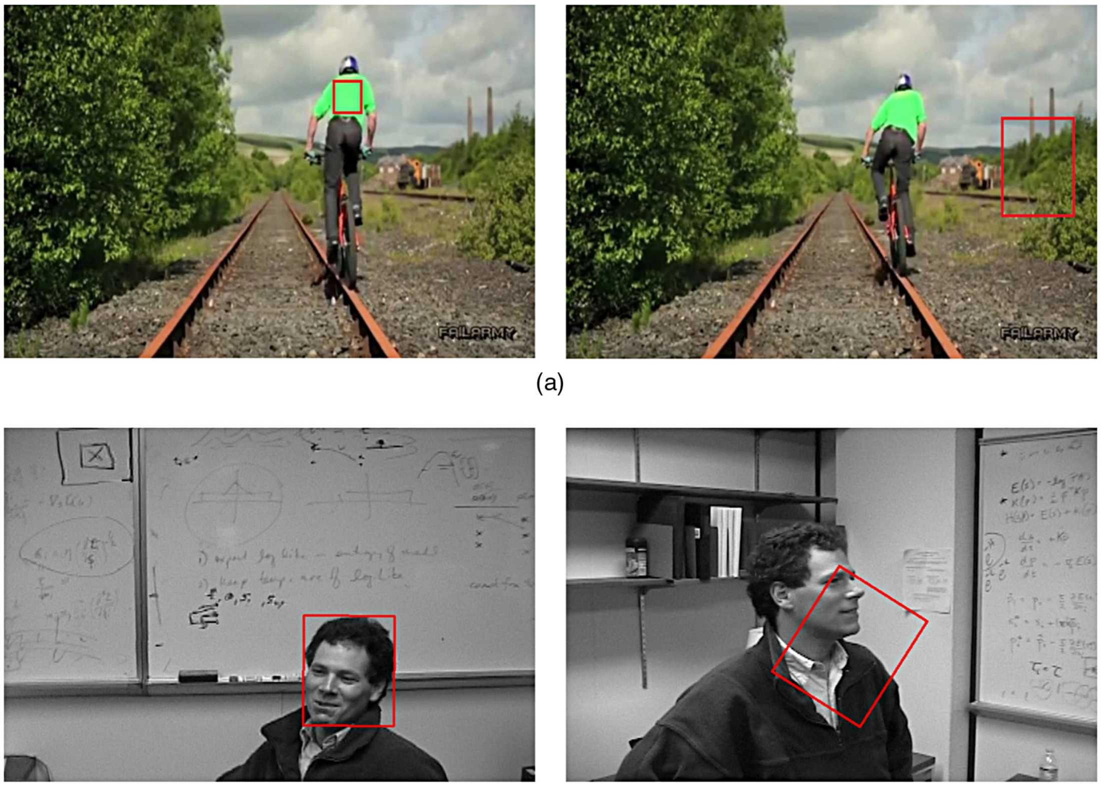


Fig. 6 The AKT algorithm suffers from texture less object and the changed appearance: (a) the tracking box is given falsely because of fewer keypoints and (b) the tracking box drifts because of the changed appearance.

Since the rotation is not considered in the ground truth of the benchmarks, it is excluded in the overlap comparisons between our results and the benchmarks.

4.1 *Accuracy Comparison of Methods for Tracking*

The tracking performance of the AKT algorithm on different datasets[28] is as shown in Fig. 5. Sequences (a) and (b) mainly contain the challenging aspect of partial occlusion. Sequences (c) and (d) mainly contain deformation. Sequences (e) and (f) mainly contain plane rotation and out-of-rotation. Sequences (g) and (h) mainly contain scale variation, and so on. The results show that facing different situations, the AKT algorithm can accurately track the object and has a very good robustness.

Although the AKT algorithm shows good tracking results in these videos, there are still some challenges that are hard to deal with. Since the AKT algorithm is based on keypoints, when the object's appearance is smooth or the texture is not rich, it may struggle, as shown in Fig. 6(a). Also, when the object's appearance is almost or totally changed, the tracking box may drift. For example, the initial object is the face, but

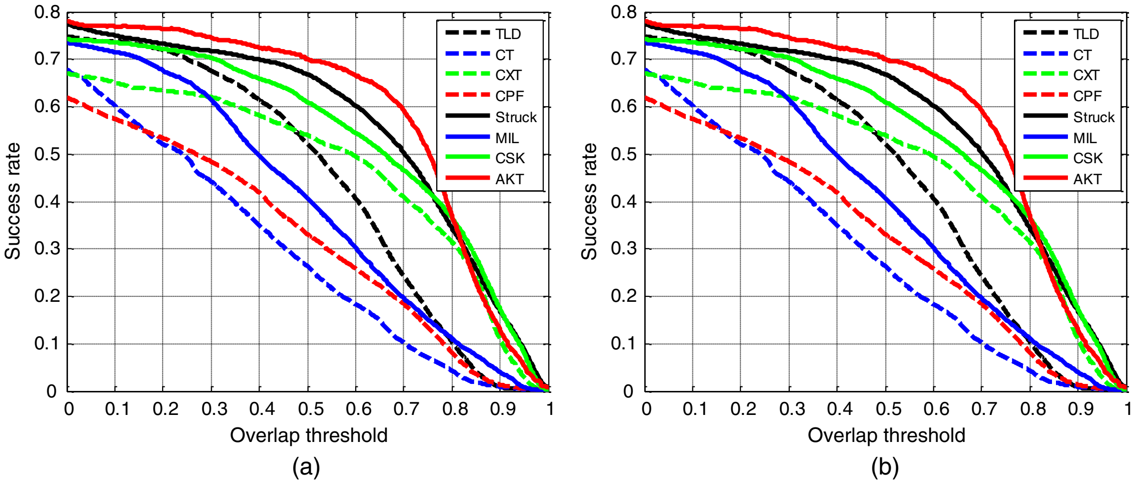


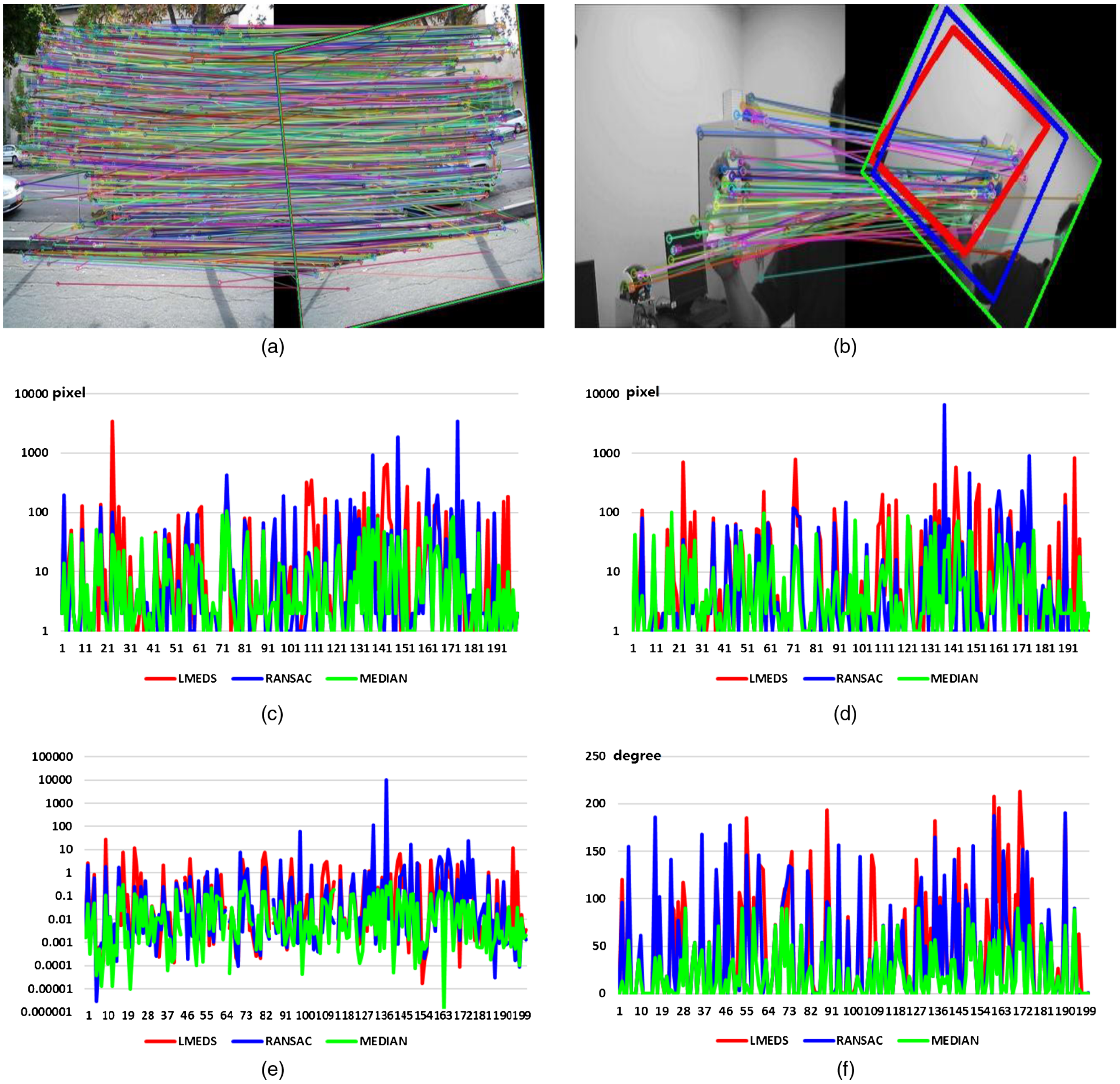
Fig. 7 (a) Precision and (b) success rate.

when the person turns around, it is hard to track because of the changed appearance, as shown in Fig. 6(b).

4.2 *Performance Comparison of Methods for Tracking*

The center location error (CLE) and average frame per second (fps) of AKT algorithm and other seven kinds of tracking algorithms are shown in Table 1 (bold fonts indicate the best or second best performance), the results of the other seven kinds of tracking methods on different sequences in the table comes from Ref. 26. In Table 1, the results show that among the tracking on the eight datasets, the frame rate of AKT algorithm is 77.9 fps, showing a high real-time performance (the average fps comes in the top two 7 times), and achieving a high tracking accuracy with the average CLE of 11.9 pixels (the average CLE comes in the top two 5 times), the tracking performance is better than the other seven methods.

The CLE is defined as the average Euclidean distance between the center locations of the tracking boxes using our method and the manually labeled ground truths. Then the average CLE over all the frames of one sequence is used to summarize the overall performance for that sequence. Precision plot shows the percentage of frames whose estimated location is within the given threshold distance T_{th} of the ground truth, as shown in Fig. 7(a). The results show that



(a) (b) (c) (d) (e) (f)

Fig. 8 Comparison results of methods for homography estimation: (a) similar accurate results for homography estimation, (b) LMEDS and RANSAC gives poor results while MEDIAN gives good result, (c) errors of *x*-coordinate displacement, (d) errors of *y*-coordinate displacement, (e) errors of scale, and (f) errors of rotation.

Table 2 The AE and AEN of center location, scale, and rotation (pixel/1/deg).

	x-coordinate (pixel)		*y*-coordinate (pixel)		Scale		Rotation (deg)	
Methods	AE	AEN (<100)	AE	AEN (<100)	AE	AEN (<10)	AE	AEN (<150)
LMEDS	59.495	18.216	41.185	14.620	0.848	0.538	47.211	38.858
RANSAC	55.330	11.055	55.725	13.384	52.999	0.504	41.792	33.185
MEDIAN	12.385	11.369	10.320	9.864	0.043	0.043	19.396	19.396

precision of AKT tracking is higher than the other algorithms and similar to Struck.

To measure the performance of success rate on a sequence of frames, we count the number of successful frames whose overlap o is larger than the given threshold T'_{th}. The success plot shows the ratios of successful frames at the thresholds varied from 0 to 1, as shown in Fig. 7(b). The results show that AKT algorithm is superior to other algorithms.

4.3 *Error Comparison of Methods for Homography Estimation*

In order to evaluate the different methods for homography estimation, we developed our own dataset because the data supplied by Ref. 26 did not include rotation data. We gained a total of 200 frames randomly as original frames. Then we transformed these frames using the affine model, as shown in Eq. (13).

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = s \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} dx \\ dy \end{bmatrix}, \tag{13}$$

where $[x'y']^T$ represents the coordinate of a point in the original frame. $[xy]^T$ represents the coordinate of a point in the transformed frame. s, α, and $[dxdy]^T$, respectively, represent scale, rotation, and displacement of the affine model. After transforming, we can get the dataset composed of original frames and transformed frames with known affine homography.

Then, under the condition that the keypoints of original frames and that of transformed frames are the same, we calculate the errors of displacement (pixel), scale (1) and rotation (deg) to get the error figures (method LMEDS in red, RANSAC in blue, MEDIAN in green), as shown in Fig. 8. The independent variable of error figures is the number of frames, whereas the dependent variable is the error.

The average error (AE) is used for comparison as the first evaluation criterion, as shown in Table 2. There will be noises causing by obvious variable estimation error, so to make better comparison of the methods for homography estimation, we set up average error without noise (AEN) as the second evaluation criterion. From the error figures, we set 100 pixels as location noise threshold, 10 as scale noise threshold, 150 deg as rotation noise threshold. The lower the AE and AEN, the better the performance of method for homography estimation. The smaller the difference between AE and AEN, the more stable the method for homography estimation. Therefore, the experimental results show not only that the median method is more stable, not having apparent noises, but also that its value of AE and AEN is less than that of the traditional statistical method.

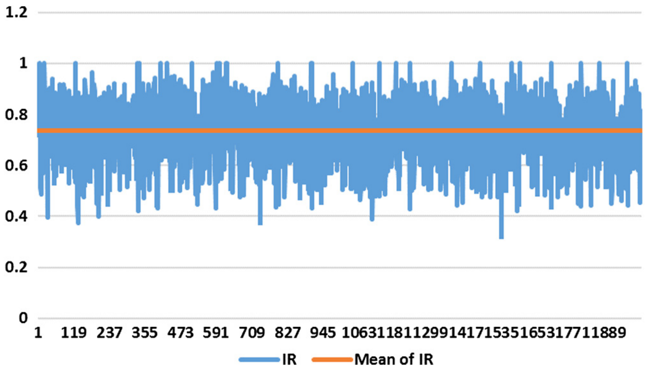


Fig. 9 IR and mean of IR.

4.4 *Selection of Threshold for Tracking Results*

The ratio of the number of inliers to the total number of matching-keypoints is called inlier ratio (IR). The larger the IR, the better the estimation of homographies. We impose that the error in location for two corresponding keypoints has to be less than 2.5 pixels, i.e., $\|F'_b - H(F_a)\| < 2.5$, where H is the true homography between the frames, F_a is the location of keypoint a in original frame F, and F'_b is the location of keypoint b in transformed frame F_b. The keypoint meeting above condition is called inlier. To find the threshold for better tracking, we still use the dataset put forward in Sec. 4.3 with the total number changed to 2000. We calculate the IR of these corresponding frames and the mean of IR is 0.74, as shown in Fig. 9. Therefore, we set optimal value l^t_{Th} for tracking as the mean of IR to avoid outliers.

5 Conclusion

In this paper, in an effort to reduce an excess of outliers when using traditional AKAZE match-tracking algorithm and solve the problems caused by poor homography estimates produced by statistical methods, AKT algorithm is put forward. The experimental results on different datasets show that the AKT algorithm can deal with challenges, such as partial occlusion, deformation, scale variation, rotation, and background clutter, showing high real-time performance and accuracy. However, since the tracking method used is based on keypoints, when the objects appearance is smooth, and texture is not rich, using the AKT algorithm may result in reduction of the effectiveness of tracking. Therefore, in future work, we will address the problems mentioned above.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (Grant No. 61471194), Science and

Technology on Electro-optic Control Laboratory and Aeronautical Science Foundation of China (Grant No. 20135152049), CASC (Grant No. China Aerospace Science and Technology Corporation) Aerospace Science and Technology Innovation Foundation Project; the Fundamental Research Funds for the Central Universities; Nanjing University of Aeronautics and Astronautics Graduate School Innovation Base (Laboratory) Open Foundation Program (Grant No. kfjj20151505).

References

1. A. Yilmaz, O. Javed, and M. Shah, “Object tracking: a survey,” *ACM Comput. Surv.* **38**(4), 13 (2006).
2. K. Cannons, “A review of visual tracking,” Technical Report CSE-2008-07, Department of Computer Science Engineering, York University, Toronto, Canada (2008).
3. E. Maggio and A. Cavallaro, *Video Tracking: Theory and Practice*, Wiley Online Library, Hoboken, New Jersey (2011).
4. T. K. Lee et al., “Reliable tracking algorithm for multiple reference frame motion estimation,” *J. Electron. Imaging* **20**(3), 033003 (2011).
5. A. W. Smeulders et al., “Visual tracking: an experimental survey,” *IEEE Trans. Pattern Anal. Mach. Intell.* **36**(7), 1442–1468 (2014).
6. Y. Junhua et al., “Real-time tracking of targets with complex state based on ICT algorithm,” *J. Huazhong Univ. Sci. Technol. (Natural Sci. Ed.)* **43**(3), 107–112 (2015).
7. A. Saffari et al., “On-line random forests,” in *IEEE 12th Int. Conf. on Computer Vision Workshops*, pp. 1393–1400 (2009).
8. B. Babenko, M. H. Yang, and S. Belongie, “Robust object tracking with online multiple instance learning,” *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(8), 1619–1632 (2011).
9. P. Sand and S. Teller, “Particle video: long-range motion estimation using point trajectories,” *Int. J. Comput. Vision* **80**(1), 72–91 (2008).
10. G. Nebehay and R. Pflugfelder, “Consensus-based matching and tracking of keypoints for object tracking,” in *IEEE Winter Conf. on Applications of Computer Vision*, pp. 862–869, IEEE (2014).
11. C. Bibby and I. Reid, “Robust real-time visual tracking using pixel-wise posteriors,” in *European Conf. on Computer Vision* (2008).
12. C. Bibby and I. Reid, “Real-time tracking of multiple occluding objects using level sets,” in *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1307–1314 (2010).
13. T. Brox et al., “High accuracy optical flow estimation based on a theory for warping,” in *European Conf. on Computer Vision*, pp. 25–36 (2004).
14. Z. Kalal, K. Mikolajczyk, and J. Matas, “Tracking-learning-detection,” *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(7), 1409–1422 (2012).
15. D. Ramanan, D. A. Forsyth, and A. Zisserman, “Tracking people by learning their appearance,” *IEEE Trans. Pattern Anal. Mach. Intell.* **29**(1), 65–81 (2007).
16. P. Buehler et al., “Long term arm and hand tracking for continuous sign language TV broadcasts,” in *British Machine Vision Conf.* (2008).
17. A. Adam, E. Rivlin, and I. Shimshoni, “Robust fragments-based tracking using the integral histogram,” in *IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, pp. 798–805, IEEE (2006).
18. S. M. S. Nejhum, J. Ho, and M. H. Yang, “Online visual tracking with histograms and articulating blocks,” *Comput. Vision Image Understanding* **114**(8), 901–914 (2010).
19. D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. Comput. Vision* **60**(2), 91–110 (2004).
20. H. Bay, T. Tuytelaars, and L. Van Gool, “SURF: speeded up robust features,” in *European Conf. on Computer Vision*, pp. 404–417, Springer, Berlin Heidelberg (2006).
21. E. Rublee et al., “ORB: an efficient alternative to SIFT or SURF,” in *Int. Conf. on Computer Vision*, pp. 2564–2571 (2011).
22. P. Alcantarilla, J. Nuevo, and A. Bartoli, “Fast explicit diffusion for accelerated features in nonlinear scale spaces,” In *Proc. British Machine Vision Conf.*, 1–11 (2013).
23. S. Grewenig, J. Weickert, and A. Bruhn, “From box filtering to fast explicit diffusion,” in *Pattern Recognition*, pp. 533–542, Springer, Berlin Heidelberg (2010).
24. X. Yang and K. T. Cheng, “LDB: an ultra-fast feature for scalable augmented reality on mobile devices,” in *IEEE Int. Symp. on Mixed and Augmented Reality*, pp. 49–57, IEEE (2012).
25. P. F. Alcantarilla, A. Bartoli, and A. J. Davison, “KAZE features,” in *European Conf. on Computer Vision,* pp. 214–227, Springer, Berlin Heidelberg (2012).
26. Z. Kalal, K. Mikolajczyk, and J. Matas, “Forward-backward error: automatic detection of tracking failures,” in *20th Int. Conf. on Pattern Recognition*, pp. 2756–2759, IEEE (2010).
27. G. Nebehay and R. Pflugfelder, “Consensus-based matching and tracking of keypoints for object tracking,” in *IEEE Winter Conf. on Applications of Computer Vision*, pp. 862–869 (2014).
28. Y. Wu, J. Lim, and M. H. Yang, “Online object tracking: a benchmark,” in *Computer Vision and Pattern Recognition*, pp. 2411–2418 (2013).
29. K. Zhang, L. Zhang, and M. H. Yang, “Real-time compressive tracking,” in *European Conf. on Computer Vision*, pp. 864–877, Springer, Firenze, Italy (2012).
30. T. B. Dinh, N. Vo, and G. Medioni, “Context tracker: exploring supporters and distracters in unconstrained environments,” in *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 1177–1184 (2011).
31. P. Pérez et al., “Color-based probabilistic tracking,” in *European Conf. on Computer Vision*, pp. 661–675, Springer, Berlin Heidelberg (2002).
32. S. Hare, S. Amir, and P. H. Torr, “Struck: Structured output tracking with kernels,” in *Int. Conf. on Computer Vision*, pp. 263–270 (2011).
33. B. Babenko, M. H. Yang, and S. Belongie, “Visual tracking with online multiple instance learning,” in *IEEE Conf. on Computer Vision and Pattern Recognition*, pp. 983–990 (2009).
34. J. F. Henriques et al., “Exploiting the circulant structure of tracking-by-detection with kernels,” in *European Conf. on Computer Vision*, pp. 702–715, Springer, Berlin Heidelberg (2012).
35. B. Hemery, H. Laurent, and C. Rosenberger, “Comparative study of metrics for evaluation of object localisation by bounding boxes,” in *Fourth Int. Conf. on Image and Graphics*, pp. 459–464, IEEE (2007).

Junhua Yan is an assistant professor at Nanjing University of Aeronautics and Astronautics, a visiting researcher in Science and Technology on Electro-Optic Control Laboratory. She received her BSc, MSc, and PhD degrees from Nanjing University of Aeronautics and Astronautics in 1993, 2001, and 2004, respectively. She is the author of more than 30 journal papers and has 5 patents. Her current research interests include multisource information fusion, and target detection, tracking, and recognition.

Zhigang Wang received his BSc degree from Nanjing University of Aeronautics and Astronautics in 2013. Now, he is a MSc degree candidate at Nanjing University of Aeronautics and Astronautics. His main research direction is object detection and tracking.

Shunfei Wang received his BSc degree from Nanjing University of Aeronautics and Astronautics in 2014. Now, he is a MSc degree candidate at Nanjing University of Aeronautics and Astronautics. His main research direction is object detection and tracking.