Acta Technica *62* (2017), No. 6B, 759–778 © 2017 Institute of Thermomechanics CAS, v.v.i.

Target - aware event - driven remote sensing technology for unmanned aerial vehicle patrol

Song Wang[1], Jieqing Zhao[2], Liangliang Bu[2], Xin Zhang[3]

Abstract. Video multi-target tracking is a critical problem in the field of computer vision. Although many existing algorithms have the ability to track targets in certain environments, it is often difficult to achieve automated and robust tracking in video sequences if there are occlusion, various changes in target shape, or complex backgrounds. To surmount these difficulties, a multi-target tracking algorithm based on ORB (Oriented Brief) feature point is proposed in this paper. First, the "Moving Edge Growth" algorithm is used to get target blocks. Next, targets in two sequential frames are associated by matching the ORB features of target blocks with feature templates. The matching process enables target templates to be updated in real time by eliminating the outdated feature points and adding new feature points. In this way, effectiveness of feature template is guaranteed in real time, resulting in greater reliability of matching during the tracking process. Experimental results demonstrate the ability of the proposed algorithm in robustly tracking even if the targets are deformed or partially occluded.

1. Introduction

Multi-target tracking is one of an important applications of the computer vision technology. It has great potential of being widely used for intelligent monitoring, mobile robotics, navigation and guidance as well as unmanned driving. In recent years, the researchers have devoted a lot of efforts into visual tracking and yielded a lot of theories and methods. These achievements can be classified into three categories, i.e. filtering-based methods [1-2], target pattern matching-based methods[3-4] and the classification-based methods[5-6]. Typical examples of the filtering-based methods are UKF (Unscented Kalman Filter) and PF (Particle filter). But some of these methods are unable to process non-linear non-Gaussian scenarios, while some are

[1]School of Aeronautic Science and Engineering, BeiHang University, Beijing, 100191, China
[2]Shenzhen Power Supply Co., Ltd., Shenzhen 518048, China

very computationally intensive and thus not suited for situations in which the targets are expected to be tracked in real-time. Typical example of the target pattern matching-based methods is MS (Mean Shift) which entails slight calculation burden and is thus able to track targets in real time. However, MS needs to model the target using a color feature alone. As a result, when the background is complicated, target is occluded or moves fast, the performance of the algorithm is poor. Due to these reasons, it is usually combined with other target features or tracking methods. Yang et al. [9] proposes a method of using background subtraction to get the target, and then uses template matching to track the target. Yang's method can track multiple targets in indoor and outdoor scenes, but it can not segment the target effectively in the case of occlusion. The classification-based target tracking method must learn and train samples using some machine learning algorithm, leading to a high computational complexity, and its performance is largely dependent on the design of classifier and the number of samples. In order to solve the problems caused by the rotation, scaling, illumination change and so on, SIFT feature[10] has been introduced to yield performance gain in target tracking stability[11-12]. But the SIFT algorithm is very computationally intensive. Performing SIFT transform on each image or several image blocks at the same time is very time-consuming and makes it hard to track target in real time. Erublee et al.[13]proposed a new target tracking algorithm ORB. In addition to being rotation- and scale-invariant, it is able to produce high-quality feature points and operates faster than SIFT by two orders of magnitude, making it suited for the real-time situations.

By the using of the visual attention mechanism, people is able to quickly locate several highly salient targets or regions in a scene. In this way, the chosen targets or regions would be processed first to improve information processing efficiency. Furthermore, the visual system is very sensitive to a moving object. Hence, introducing the motion saliency to target tracking and detection will provides a very promising means of producing performance gains. Zhang et al.[14] combines motion saliency map and particle filter to realize the stable tracking of multi targets, but the extracted target zones are not accurate enough.

In the above context, this paper propose an ORB feature-based target tracking algorithm. Saliency of moving edges of targets are used to segment targets from background, ORB features are then extracted from the partitioned targets and used for tracking and motion analysis. Experimental results demonstrate the ability of the proposed algorithm to stably track multiple targets, reduce tracking error and address partial occlusion of targets effectively.

2. Moving target detection

Target detection is the premise of the target tracking. Human visual system is highly sensitive to the edge of a target, and tends to get the specific shape of the target and analyze it via edge information [15-17]. In this section, we propose a novel detection algorithm which referred as Moving Edge Growth to get the foreground targets.

2.1. Saliency of moving edge

Edge is an essential factor deciding the shape and contour of a moving target, and also tends to be in the most prominent position in movement. In this sense, an algorithm called Three-Frame Edge Difference is proposed to detect moving edge. The algorithm has an advantage that it is simple to calculate, and the most important moving edge can be extracted, which greatly reduces the follow-up optical flow computation. Since it is realizable to get relatively continuous edge via edge detection with Canny operator [18], we figure out the binary edge images E_{t-1}, E_t and E_{t+1} of the three adjacent frame images I_{t-1}, I_t and I_{t+1} via Canny operator first. Then, the binary moving edge map DE_t can be calculated as:

$$DE_t = |E_t - E_{t-1}| \cap |E_{t+1} - E_t|, \tag{1}$$

where "$\cap$" denotes intersection. Then the DE_t is to be treated by morphological close operation with 3×3 square structure element, which contribute to both the fusion of edges and the elimination of small noise.

The motion saliency of the moving edge is got by LK optical flow calculation method [19].LK is a kind of sparse optical flow estimation algorithm based on two-frame difference, in which only a small number of discrete points are handled. Thus, the computational cost of LK is small, so it can satisfies the real - time requirement.

It is assumed that all pixels in the $w_1 \times w_1$ neighborhood of an edge pixel p take the consistent motion, i.e., velocity vector $\mathbf{v}^T = (v_x, v_y)$ is a constant. Then the optical flow constraint equations can be obtained from all pixels in the neighborhood:

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{n^2}) & I_y(p_{n^2}) \end{bmatrix} \begin{bmatrix} v_x \\ v_y \end{bmatrix} = \begin{bmatrix} -I_t(p_1) \\ -I_t(p_2) \\ \vdots \\ -I_t(p_{n^2}) \end{bmatrix}, \tag{2}$$

where $I_x(.)$, $I_y(.)$ and $I_t(.)$ are the differential values in x, y and t directions respectively. The equation (2) can be reformulated as $\mathbf{Av} = \mathbf{b}$, then the optical flow vector $\mathbf{v}$ is got by the method of least squares:

$$\mathbf{v} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{b} \tag{3}$$

Optical flow vector has two properties of intensity and direction. In general, the bigger intensity the movement takes, the more eye attention it can attracts. We use the magnitude $|\mathbf{v}|$ of the optical flow vector to represent the motion saliency $M_{i,j}$ of the moving edge point at position (i, j). After the saliency values of all edge points are got, the motion saliency map with the same size as the source image is got, where the saliency values of all the non- edge points have been set to 0. At the same time, the orientation characteristic of each edge point is represented by a 3-dimensional vector $\Theta_{i,j}$: $\Theta_{i,j} = (i, j, \theta)$, where θ represents the radian value of the orientation.

The brief process of extracting edge saliency is shown in Fig. 1, where three adjacent frames ($t-1$th, tth and $t+1$th)are extracted from a video clip of a basketball bounce. In the three frames, the basketball is falling from a high point (with a certain rotation). First, the edge frame difference method mentioned above is used to get a binary moving edge map, and then the LK algorithm is used to extract the optical flow vector, so the motion saliency map is got according to the magnitudes of optical flow vectors at every points of the moving edge map.

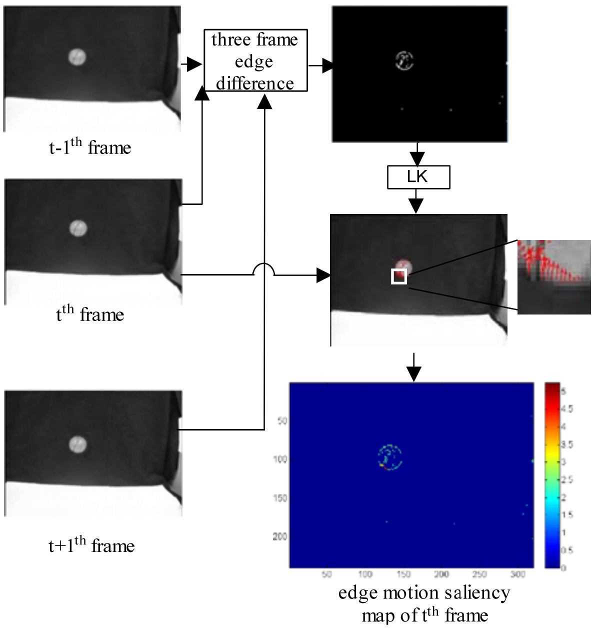
three frame edge difference
t-1th frame
LK
tth frame
t+1th frame
edge motion saliency map of tth frame

Fig. 1. Acquisition process of edge motion saliency

2.2. *Moving targets segmentation*

There are obvious edge discontinuities in the binary moving edge map DE_t of the tth frame I_t by the method of the Three-Frame-Edge-Difference, which make it difficult to segment targets according to these moving edges. To solve this problem, this paper proposes an effective strategy of "Moving Edge Growth" to process the discontinuous edges. The following is an outline of the procedure of our strategy:

Step 1: Denote the edge points coordinate set of E_t and the moving edge points coordinate set of DE_t as Z_E^t and Z_{DE}^t respectively. It is worth noting that values corresponding to the coordinates in the two sets all are equal to 1. In the set Z_{DE}^t, initialize the element serial number k to 1. The binary edge image E_t covered with the map DE_t.

Step 2: Select the kth moving edge point DE_{t,i_k,j_k} from the set Z_{DE}^t, and establish the initial queue $\Psi_{t,i_k,j_k} = \{\varphi_1\}$. Wherein, φ_1 is the coordinates of the first element of the queue, and $\varphi_1 = (i_k, j_k)$. Set the initial value of the queue length variable $Total$ to be 1. The initial values of the mean saliency $M_{t,i_k,jk}^{SM}$ and the mean orientation M^{Θ}_{t,i_k,j_k} of total points in Ψ_{t,i_k,j_k} are set to be the saliency and orientation of the point DE_{t,i_k,j_k} respectively.

Step 3: A 3×3 neighborhood centered at φ_1 is established in DE_t. If there is a point $p : p \notin Z^t_{DE} \&\& p \in Z^t_E \&\& DE_{t,p} = 0$ belongs to 8 adjacent connected points of φ_1, and p satisfy (4):

$$|SM_{t,p} - M^{SM}_{t,i_k,j_k}| < T_{sm} \&\& |\Theta_{t,p} - M^{\Theta}_{t,i_k,j_k}| < T_\theta \, , \tag{4}$$

where T_{sm} and T_θ are thresholds, then p is considered similar to DE_{t,i_k,j_k} in motion, and put it in the end of the queue Ψ_{t,i_k,j_k}. Update DE_t: $DE_{t,p} = 1$. Update $Total : Total = Total + 1$.

Step 4: Calculate the mean saliency $M^{SM}_{t,i_k,jk}$ and mean orientation M^{Θ}_{t,i_k,j_k} of points corresponding to the coordinates in set Ψ_{t,i_k,j_k}. Delete the first element of Ψ_{t,i_k,j_k}, and prepose the rest elements in sequence. Update $Total : Total = Total - 1$. If $Total \neq 0$, take out the first element φ_1 of Ψ_{t,i_k,j_k}, return to Step 3. Otherwise, implement $k = k + 1$, return to Step 2.

Step 5: Repeat step 2∼4 until the end of the growth.

After edge growth, the final moving edge is to be got via getting rid of noisy points and morphological dilation. The minimum bounding rectangle is used upon target edge to get the target block , and thus identify the target region.

3. Moving targets tracking

The overall framework of target tracking is shown in Fig. 2. The local invariant feature descriptors are extracted from the moving target blocks got by the detection process, and the real time association are realized by the matching of the ORB features between blocks and templates. Meanwhile, the template matching process update the target templates, the coordinates of the feature points and so on.

3.1. ORB features

ORB is based on FAST feature point detection[20] and BRIEF feature descriptor[21].

(1) Feature point detection

Consider the intensity values of pixels on a circle centered at p with a radius of r(usually taken as 9 pixels), if there are at least N(usually taken as 12)) consecutive points whose intensity values satisfies $I(p) - I(q_i) > \varepsilon_d$ or $I(q_i) - I(p) > \varepsilon_d$, then p is taken as a keypoint. $I(p)$ and $I(q_i)$ are intensity values of pixels p and q_i respectively, i is the serial number of the first pixel satisfies the condition, and $M \geq N$.

For all the keypoints, we use Harris algorithm to sort and select the desired number of points as feature points. In order to add scale characteristics to feature points, ORB set up an image pyramid, detecting feature points on each layer. Moreover, ORB calculates the angle between a feature point and the centroid of the neighborhood, then append the angle to each feature point as its orientation information. For the center point p, the $(a + b)$ order geometric moment of its neighborhood are

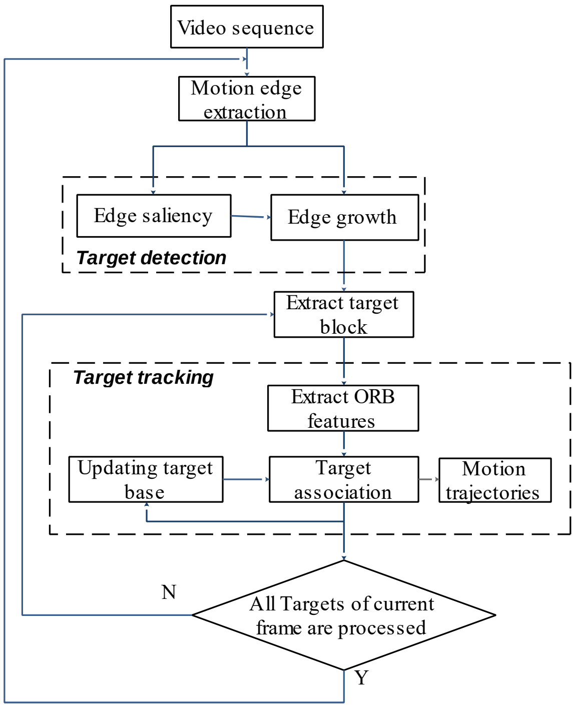

Fig. 2. Tracking system diagram

defined as:

$$m_{a,b} = \sum_{i,j} i^a j^b I_{i,j} \ , \tag{5}$$

where $I_{i,j}$ represents intensity value at (i, j). The centroid coordinates are denoted as:

$$C = (C_x, C_y) = (m_{10}/m_{00}, m_{01}/m_{00}) \ . \tag{6}$$

Thus, the orientation information can be calculated as:

$$\theta = arc\tan\left(m_{01}/m_{10}\right) \ . \tag{7}$$

(2) Feature descriptor

The ORB constructs feature point descriptor based on BRIEF algorithm[13]: n point pairs are selected around each feature point, two points in each point pair are compared to produce a binary bit, and the binary string of length n is used as the descriptor of the feature point.

After processing the image with Gaussian filter, a 31×31 neighborhood window P centered at a feature point is set, and a pair of 5×5 sub-windows are randomly chosen from P as test points $P(\mathbf{x})$ and $P(\mathbf{y})$. The intensity values of the two testing points are compared to get a binary bit:

$$\tau(P;\mathbf{x},\mathbf{y}) = \begin{cases} 1 & P(\mathbf{x}) < P(\mathbf{y}) \\ 0 & otherwise \end{cases} \tag{8}$$

where $\mathbf{x}$ and $\mathbf{y}$ are the coordinates of the two test points respectively. The steps above will be repeated n times to yield an n-*bit* binary string as the initial descriptor $f_n(P)$. Since the BRIEF descriptor is not rotation-invariant, ORB endows it with rotation invariance. The above mentioned n pairs of testing points constitute a matrix M:

$$M = \begin{bmatrix} \mathbf{x}_1, ..., \mathbf{x}_n \\ \mathbf{y}_1, ..., \mathbf{y}_n \end{bmatrix}. \tag{9}$$

The neighborhood direction θ and the corresponding rotation matrix R_θ are used to get a corrected version of M, i.e. $M_\theta = R_\theta \times M$. Afterwards, the final descriptor is given by:

$$g_n(P,\theta) = f_n(P)|(\mathbf{x}_i, \mathbf{y}_i) \in M_\theta\,. \tag{10}$$

3.2. Adaptive template update based tracking algorithm

In the present paper, our tracking algorithm is divided into two processes: target matching and template updating. Target matching is designed to find identical targets and establish target associations by examining the matching status of present targets' ORB features and each target template in TIL. Template updating is the real-time update of each target template during tracking, aiming at ensuring the sustainability of target recognition.

(1) Target matching

The algorithm presented in the paper uses the feature points of a target to describe the target. Therefore, the matching of a target and a template equivalent to the matching of their feature points.

1) Target information library:

The information of each target block includes the serial number of the target, the feature template, the position matrix of all feature points, and the block's top-left coordinates and center coordinates in history frames. In this paper, the feature template of each target is expressed as a $m \times n$ dimension matrix, where m is the count of feature points and n is the dimension of each feature point descriptor. We use a structure array to store all the target information to form a TIL, or TIL for short in later sections.

2) Matching of feature points:

In this process, the matching degree is measured by Euclidean distance between descriptors. In traditional feature points based matching algorithms, a certain feature point of a target in present frame need to find its matching point in everywhere of the last frame. In other words, this feature point should try to match with each feature point of each target template in the TIL, which would result in abundant computation expense and a low efficiency. To solve this problem, we propose a "near neighbor bilateral matching" strategy to improve the performance efficiency of the algorithm based on the following reason: the matching point in the last frame have no large space displacement. Let's assume that p_i is the ith feature point of the present target O, then the detailed process is:

The matching threshold of a descriptor is written as T_r; set a $w_2 \times w_2$ search window W_{cur} in the present frame with p_i as the center; set a $w_2 \times w_2$ search window W_{pre} in the previous frame at the same position as p_i; the matching point of p_i is denoted as p_{match}.

a. Backward searching:

The Euclidean distances between p_i and the feature points which not only lie in W_{pre} but also belongs to the TIL will be measured. Then, the two feature points with the shortest distance (indicating high similarity) should be selected as candidate feature points which denoted as p_1 and p_2 with distances are ξ_1 and ξ_2, respectively(Fig.3(a)). If only ξ_1 is lower than the threshold value T_r, then p_1 is immediately taken as the matched feature point, $p_{match} = p_1$; if both ξ_1 and ξ_2 are lower than the threshold value T_r and the ratio of ξ_1 to ξ_2 is less than threshold value T_d, then $p_{match} = p_1$; if both ξ_1 and ξ_2 are lower than the threshold value T_r but the ratio of ξ_1 to ξ_2 is not less than T_d, then step$b$ should be executed.

b. Forward searching:

The points p_1 and p_2 search their matched feature points in the window W_{cur} of the present frame, and respectively take the first two minimum distance points as the candidate points, which are denoted by p_{11}, p_{12} and p_{21}, p_{22}, their corresponding matching distances are denoted by ξ_{11}, ξ_{12} and ξ_{21}, ξ_{22}. If p_{11} or p_{12} is p_i (Figure 3(b)), then $p_{match} = p_1$; similarly, if p_{21} or p_{22} is p_i, then $p_{match} = p_2$; if p_{11} or p_{12} and p_{21} or p_{22} are p_i (Fig. 3(c)), then p_{match} should be selected according to the principle as follows:

$$p_{match} = \begin{cases} p_2 & p_{12} = p_i \&\& p_{21} = p_i \&\& d_2 < d_1 \\ p_1 & otherwise \end{cases} \tag{11}$$

where d_1 and d_2, respectively stand for the spatial distances from p_i to p_1 , p_i to p_2; if none of p_{11}, p_{12}, p_{21} and p_{22} is p_i, then $p_{match} = (p_m | m = \arg\min_{k \in \{1,2\}} (\xi_k + \xi_{k1} + \xi_{k2}))$.

3) Target matching:

All matched points of the present target O are extracted, each count of matched points corresponding to a different template forms a set Z: $Z = \{N_{\varphi_1}, ..., N_{\varphi_i}, ..., N_{\varphi_I}\}$, $\varphi_i \in \{1, ..., L\}$, where N_{φ_i} stands for the count of feature points belonging to the φ_ith target template among all the matched points, I for the total number of elements in set Z, and L for the total number of templates in the TIL. Find out the maximum of set Z, which is denoted by N_{φ_j}, and suppose N to be the count of feature points in the present target block, then the maximum matching ratio is denoted by $P_o = N_{\varphi_j}/N$. Set a threshold T_p of matching ratio, if $P_o > T_p$, it can be considered that the present target matches the φ_jth target of the TIL, and the association is established accordingly; otherwise, rematch the present target with the target templates numbered $\varphi_1, ..., \varphi_i, ..., \varphi_I$ in the TIL respectively, and make judgment again. Fact shows that since the relative displacement for a same target in 2 adjacent frames are small, there's usually no need of rematching.

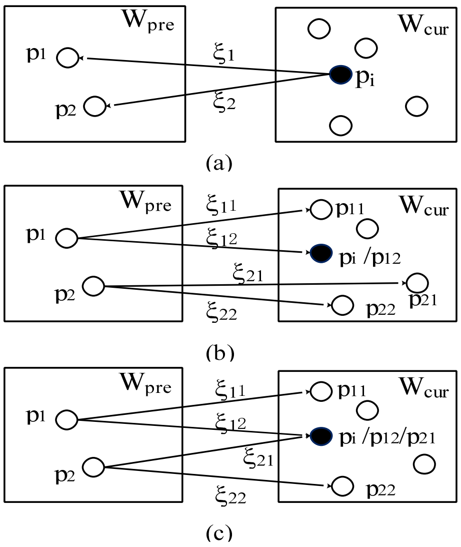

Fig. 3. Backward searching and forward searching. (a) Backward searching.(b)One of p_1's candidate points is p_i in forward searching.(c) One of p_1's candidate points and one of p_2's candidate points are p_i in forward searching.

(2) Template updating

To maintain the effectiveness of the TIL during the targets tracking process, it must be updated in real time, including the update of the existing templates as well as the preservation of the new target templates. In this paper, we refer to [22] to set a preservation level for every feature point in the TIL during template updating. For example, the preservation level of *jth* feature point of target O can be denoted by $R_{O,j}$ whose maximum value is denoted by $R_{\max}$, indicating the confidence of this feature point belongs to the target.

1) Add new target templates:

If a target in the present frame doesn't match any target template in the TIL, it is considered that the target is a new target whose serial number, target block coordinates and target template should be added into the TIL. For new targets, the preservation levels of all feature points are all set to $R_{\max}$.

2) Update the matched target templates:

If the present target O matches the *ith* target template of the TIL, every matched feature point in the *ith* target template should be replaced with that of the present target, the position of the feature point in the position matrix should be updated, and the preservation level of the point is set to $R_{\max}$; as to every unmatched feature point in the target template, the value of its preservation level minus 1. The unmatched feature points in the present target O should be added to the target template whose corresponding preservation levels all should be set to $R_{\max}$, and the feature point position matrix also need to be updated. Considering illumination change, target deformation, noise interference, and other external factors, that the preservation level of a feature point reduces to 0 means that this feature is not matched for a long time. Thus, it is deemed that this feature point cannot represent the target any longer, and all the information of this feature point should be deleted from TIL.

3) Update of unmatched target templates:

If there still exist a target template that do not matches any target even after the ending of matching process of the present frame, it is deemed that that target stops, is completely covered, or has left the scene. Then, it is supposed to inspect the center coordinates of that target in the last matching frame and previous $l-1$ frames.

For each one of the l frames, the minimum distance from the center of the target block to the four boundaries of the scene is calculated respectively, which are denoted by $dist_1, ..., dist_t$ in chronological order. If the l distances decrease progressively and $dist_t \leq T_e$, with T_e as the presupposed threshold value, then it is considered that the target has left the scene, the preservation level of all feature points of the template remains unchanged, and the central coordinates are set to be $(-1,-1)$ which is simply a sign of "leave a scene" ; if $dist_t > T_e$, the image patch of the present frame at the position of the last matched target with the equal size of the last matched target block , is still able to match the target template, then it is considered that the target stops temporarily, and the preservation level of feature points remains unchanged. At this moment, the center of target bounding box in tracking is the center of the target block preserved in the TIL; otherwise, it is judged that the target is occluded completely, the central coordinates of the target block is set to be $(-2,-2)$ which is simply a sign of "being occluded completely".

This algorithm does not remove any template, which is due to taking into account the actual situation: the target leaving the scene is likely to return. So, the template at the time before leaving the scene is need to be retained.

3.3. Partial occlusion between targets

During multi-target tracking, because of the change of the positions of the targets, there will be occlusions between targetsIn this paper, a method is proposed to judge if there is occlusion according to the proportion of matched feature points among which targets are separated according to the relative positions.

(1) Occlusion occurrence

During tracking, when different moving targets are partially occluded, a single rectangle bounding box will be formed. As shown in Fig. 4, target 1 and target 2 are blocked at a certain moment.

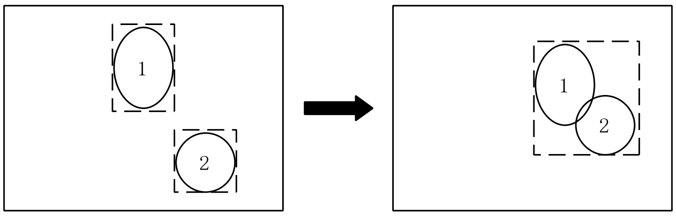

Fig. 4. Partial occlusion diagram between two targets.

The multiple target blocks that are mutually occluded will form a "cluster target block". Occlusion judgment is based on the following reason: when it just happens, most of feature points of the corresponding target templates can be found in the "cluster target block". Take 2 targets for example(Fig.4), hypothetically, the set that consist of target serial numbers and corresponding count of matching points that the cluster target block gets from backward matching is denoted as Z: $Z = \{N_1, N_2, N_{other\_1}, ..., N_{other\_k}\}$, where N_1 and N_2 are the count of points belongs to target No. 1 and target No.2 respectively among all matched points, $N_{other\_i}, i = 1, ..., k$ is the count of the matched points that belongs to each of the other k targets. For each $N_j, j = 1, 2, other\_1, ..., other\_k$ in Z, let N_j^T be the count of features points in the j^{th} template. If there are at least two different j values that meet $T_m \leq N_j/N_j^T \leq 1$, where T_m is the ratio threshold, then it is thought that there exists partial occlusion in the cluster target block of the present frame. Naturally the above method can be applied to the case of multiple targets.

(2) Occlusion handling

When occlusion occurs, we do not update the target templates that are occluded from each other because of their instable states until the occlusion is over.

In the present frame, let $x_{i,j}^{cur}$ and $y_{i,j}^{cur}$ be x and y coordinates of the j^{th} feature point of the cluster target block, which matches the i^{th} target template. Then, $x_{i,j}^T$ and $y_{i,j}^T$ represent the corresponding x and y coordinates preserved in the TIL. The center coordinates $(x_{i,c}^{cur}, y_{i,c}^{cur})$ of the target i in the present cluster target blocks can be computed as:

$$\begin{cases} x_{i,c}^{cur} = x_{i,c}^T + \dfrac{1}{N_i} \displaystyle\sum_{j=1}^{N_i} (x_{i,j}^{cur} - x_{i,j}^T) \,, \\ y_{i,c}^{cur} = y_{i,c}^T + \dfrac{1}{N_i} \displaystyle\sum_{j=1}^{N_i} (x_{i,j}^{cur} - x_{i,j}^T) \,, \end{cases} \tag{12}$$

where $x_{i,c}^T$ and $y_{i,c}^T$ are the last center coordinates of thei^{th} target in TIL, N_i is the count of feature points belongs to the cluster target block, which match the i^{th} target template.

4. Results and discussions

In order to verify the effectiveness of the proposed algorithm in targets tracking, we test and compare it with the particle filter algorithm based on visual attention [14] and the color feature template matching algorithm [9] on different video sequences. The experimental environment is: PC with Core^{TM}-i7 CPU, 8GB of memory and Windows7 operating system; using Matlab programming. A total of 10 video sequences are used for testing, whose names and their parameters are shown in Tab. 1. They include indoor and outdoor scenes, as well as rigid and nonrigid targets, and come from:ATON project of UCSD(sequence 1, http://cvrr. ucsd. edu/aton/), KIT (sequence 7 and sequence 9, http://i21www.ira.uka.de/image_sequences/), QMUL (sequence 3, sequence 8 and sequence 10, http://www.eecs.qmul.ac.uk/~andrea/spevi.html), Hanyang University (sequence 4~ sequence 6, http://cvlab.hanyang. ac.kr/ index.html), and MATLAB computer vision toolbox(sequence 2).In these sequences, all the ground truth of targets in sequence 3, sequence 8 and sequence 10 and the ground truth of one of targets in sequence 4~ sequence 6 have been given, the ground truth of other targets are manually marked by ourselves.

Table 1. Video sequences in experiments

No.	name	frame size	total frames
1	highway2	320×240	499
2	traffic	160×120	120
3	motinas_room160	360×288	1073
4	walking2	384×288	500
5	subway	352×288	175
6	walking	768×576	412
7	bad	512×512	50
8	motinas_room105	191×256	1077
9	dt	512×512	50
10	motinas_Chamber	360×288	1121

4.1. Detection results and analysis

To validate the proposed detection algorithm in section 2, parameters in the algorithm is set as: the length of the neighborhood $w_1 = 25$ pixels, threshold values T_{sm} and T_θ are 0.2 and $\pi/36$ respectively.

Fig. 5 shows a sampling result of moving multi-targets detection on sequence 1(221th frame) for the purpose of comparison between frame difference method, background subtraction method and ours.

In the figure, row 1 demonstrates the original image and ground truth target block, from left to right;row 2 and row 3 are detection results of frame difference method and background subtraction method respectively, where the detected target blocks are got after filtering noise. And the background updating model used in background subtraction method is

$$B_{i,j,t} = \alpha \times I_{i,j,t} + (1-\alpha)B_{i,j,t-1} \tag{13}$$

where $\alpha = 0.15$. As we can see that the frame difference method is prone to bring "holes" into the interior of the targets while the background subtraction method leading to a "tail" effect. That will cause ambiguities in the process of identifying moving targets. The last row of the Fig. 5 shows the detection result derived from our method, better performance is achieved.

We don't compare with optical flow method and Gaussian mixture background modeling method and so on, because of their high computational complexity. In this paper, moving target detection is just the predecessor of the overall algorithm of the target tracking, so from the point of view of real-time, the target detection method should be done simply and rapidly.

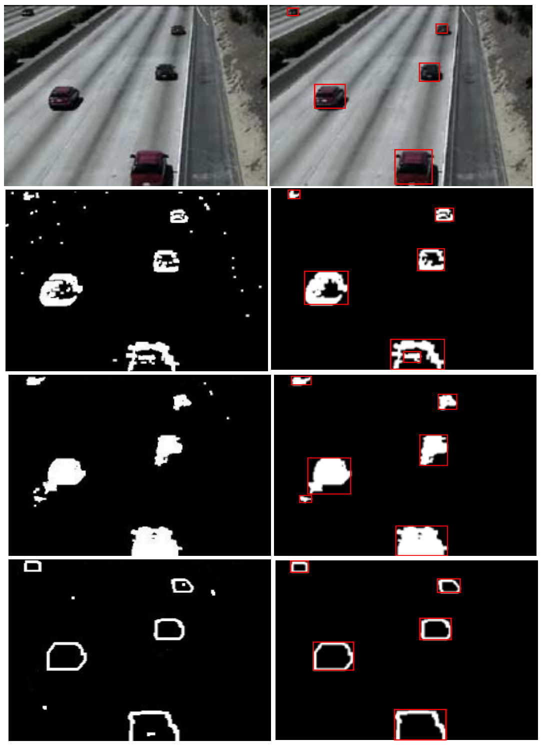
Fig. 5. Sampling results of different algorithms for target detection. The 1st row is the original frame image and its segmentation, the 2nd , 3rd and 4th rows are the detection and segmentation results(from left to right) of frame difference, background subtraction, and our algorithms respectively.

4.2. Tracking results and analysis

The related parameters are set as:

The side length value w_2 of the search window W_{cur} in the process of feature points matching is 10 pixels, the dimension n of ORB descriptor is 256,

the matching threshold T_r of feature point is 2.5, the distance ratio threshold T_d is 0.4, the matching rate threshold T_p equals 0.6, the preservation rank $R_{\max}$ is 5, and the threshold T_e of leaving scene is 15 piexels, $l = 7$; the threshold T_m of partial occlusion is 0.8.

(1) Qualitative analysis

We compare our result with two previous methods: Yang et.al[9] andZhang et al.[14] , which are referred to as YG and ZG respectively later in this paper.

Fig. 6 shows the tracking sampling result of automobile(s) in sequence 2 highway scene, Fig. 7 shows the tracking sampling result of person(s) in sequence 3 indoor scene, and Fig. 8 shows the tracking sampling result of walker(s) in sequence 4 corridor scene. The bounding boxes in different colors in the figures correspond to different targets, and the rectangular bounding boxes of dash-dotted line, solid line, dashed line, and dotted line denote the ground truth of targets, the tracking results achieved with the our algorithm, ZG algorithm and YG algorithm respectively; in the three figures, the first line refers to the ground truth of each target, the second line shows the comparison of tracking results achieved with our algorithm and the ZG algorithm, and the third line shows the comparison of tracking results achieved with our algorithm with the YG algorithm. It is clear that our algorithm performs better in tracking, which is not only reflected by the accuracy of target centers, but also by the accuracy of detection of target regions(target blocks). Figs. 9 shows the target tracking trajectories of sequence 2(left, from1st frame to the 55th frame), sequence 3(middle, from1st frame to the 450th frame) and sequence 4 (from the 1st frame to the 311th frame) respectively. The colors of the trajectories correspond with the targets in Fig. 6 to 8. The trajectories in other colors, such as the blue line in the upper left of Fig.9 (right), refer to a tracking trajectory of another target occurring in this period.

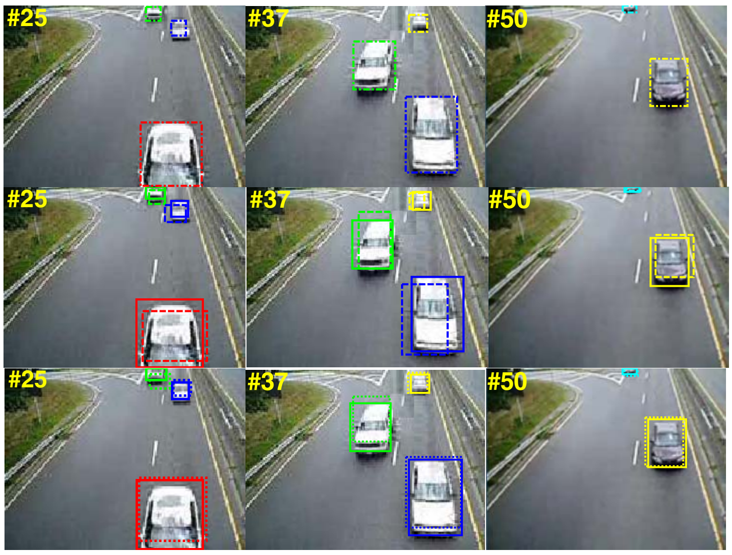

Fig. 6. Examples of tracking results for rigid targets in sequence 2 of outdoor scene. The 1st row shows the ground truth segmentations, the 2nd and 3rd rows are the results comparing our algorithm with ZG and YG algorithms respectively.

Figure 10 shows the tracking samples of 3 algorithms when moving targets are partially occluded. The 1st row in the figure consists of the 39th and 55th frame of original image. A pair of targets in the central-left part of the 39th frame are occluded with one another; two pairs of targets in the middle of the 55th frame are occluded. The second row in the figure shows the comparison of the proposed algorithm with ZG. It can be observed that the two methods are able to track a single target continuously. But the tracking accuracy of the proposed method is higher than ZG. The third row in the figure shows the comparison of the proposed

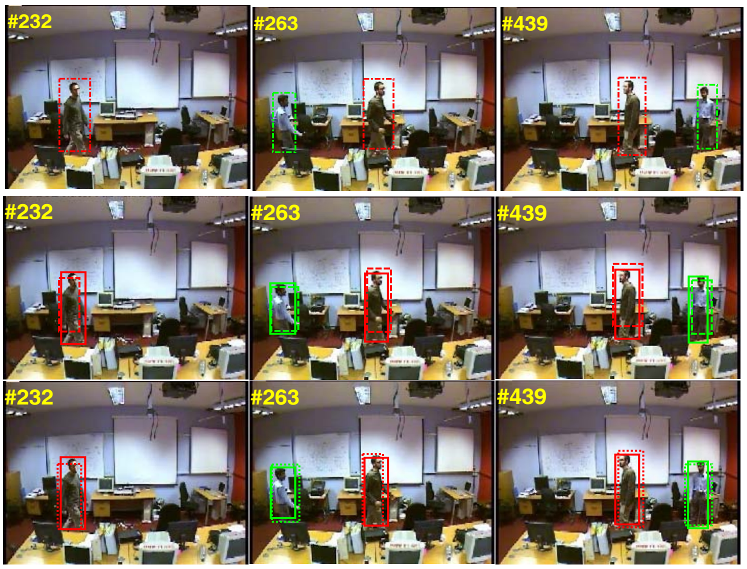

Fig. 7. Examples of tracking results for non-rigid targets in sequence 3 of an indoor scene. The 1st row shows the ground truth segmentations, the 2nd and 3rd rows are the results comparing our algorithm with ZG and YG algorithms respectively.

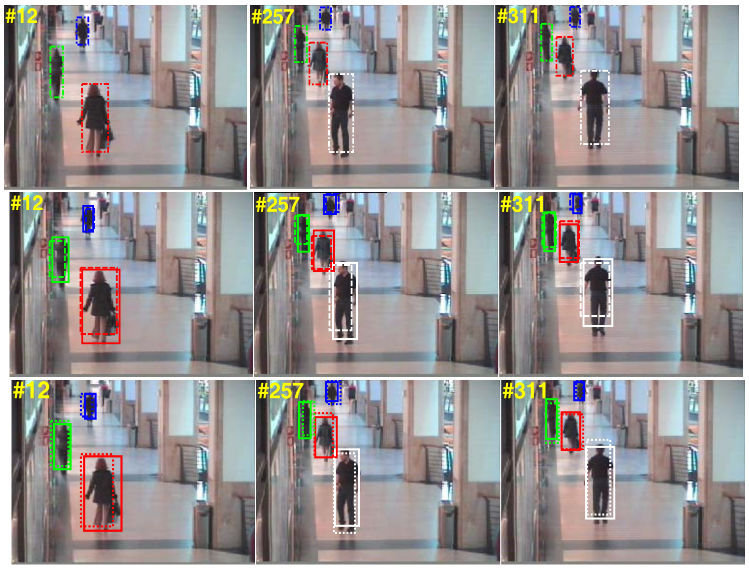

Fig. 8. Examples of tracking results for non-rigid targets in sequence 4 of an outdoor scene. The 1st row shows the ground truth segmentations, the 2nd and 3rd rows are the results comparing our algorithm with ZG and YG algorithms respectively.

algorithm with YG. In the case of target occlusion, YG regards the occluded targets

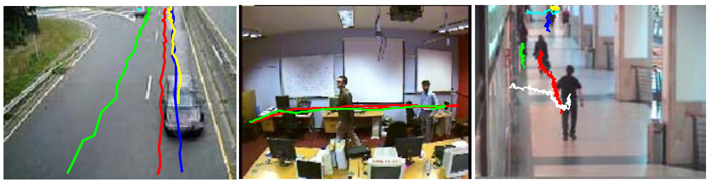

Fig. 9. Target tracking trajectories of our algorithm in sequence 2, sequence 3 and sequence 4(from left to right).

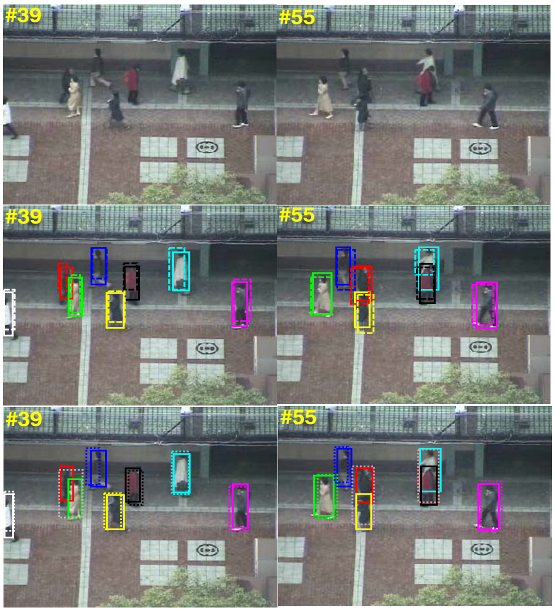

Fig. 10. Examples of tracking results under partial occlusion. The 1st row is the original frame images, the 2nd and 3rd rows are the results comparing our algorithm with ZG and YG algorithms respectively.

as a whole and then track, as shown in the large grey box in the 39th frame and the large gray and purple boxes in the 55th frame. As a result, YG is unable to distinguish between the targets until occlusion ends. But the proposed algorithm is able to track the occluded targets separately.

(2) Quantitative analysis

For the purpose of analyzing the tracking effect quantitatively, we investigated two indicators: tracking accuracy and success rate. Taking the overlap rate of target blocks as the criteria for tracking accuracy [23], which is defined as follows:

$$R_t = \frac{|r_t \cap r_t^{GT}|}{|r_t \cup r_t^{GT}|}, \tag{14}$$

where r_t and r_t^{GT} are tracked bounding box and Ground truth bounding box, $\cap$ and $\cup$ represent the intersection and union of two regions respectively, and $|.|$

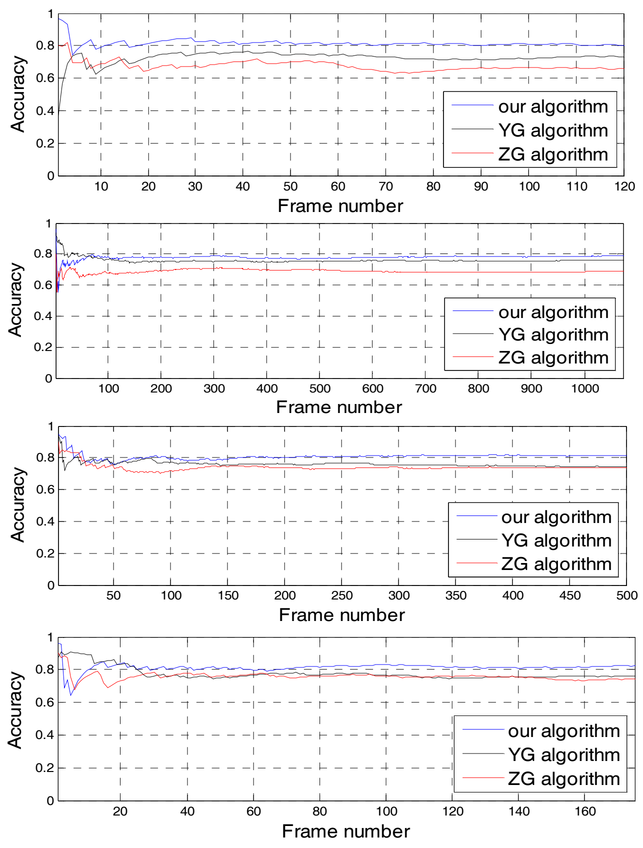

Fig. 11. Tracking accuracy. From top to bottom: sequence 2, sequence 3, sequence 4, and sequence 5.

denotes the number of pixels in the region.

In the experiments, we calculate the overlap rate of each target of a frame and then take the mean as the overlap rate of the frame. In addition, the cumulative overlap rate of present frame is taken as its accuracy measure:

$$\overline{R}_t = \frac{1}{t}\sum_{k=1}^{t} R_t \,. \tag{15}$$

Fig. 10 shows the tracking accuracies of sequence 2∼ sequence 5. It can be seen that the accuracy of each algorithm tends to be stable with the development of tracking, and our algorithm take the highest tracking accuracy. The exact values of the tracking accuracy for all 10 sequences are given in the third column of Tab. 2.

Tab. 2 denotes the comparison of tracking successful rate on 10 sequences for 3 algorithms. Assuming that it will fail when the accuracy of tracking is less than 50% for a frame. Our algorithm works the best.

Table 2. Comparison of tracking successful rate for 3 algorithms

No.	Algorithm	Accuracy	Successful tracked frames	Successful rate
1	Z's	69.30%	372	74.60%
	Y's	74.80%	350	70.10%
	Our	85.50%	405	81.20%
2	Z's	65.70%	81	67.50%
	Y's	73.40%	89	74.20%
	Our	80.40%	94	78.30%
3	Z's	68.20%	851	79.30%
	Y's	76.10%	834	77.70%
	Our	78.60%	872	81.30%
4	Z's	74.20%	324	64.80%
	Y's	74.80%	349	69.80%
	Our	78.60%	378	75.60%
5	Z's	74.10%	133	76.30%
	Y's	76.00%	137	78.30%
	Our	82.70%	151	86.30%
6	Z's	77.30%	243	60.00%
	Y's	73.60%	288	69.90%
	Our	79.80%	304	73.80%
7	Z's	70.40%	27	54.00%
	Y's	68.20%	30	60.00%
	Our	75.00%	34	68.00%
8	Z's	65.50%	815	75.70%
	Y's	72.10%	797	74.40%
	Our	78.50%	881	81.80%
9	Z's	73.00%	25	50.30%
	Y's	80.10%	38	76.00%
	Our	85.60%	39	78.00%
10	Z's	74.20%	914	81.50%
	Y's	76.70%	869	77.50%
	Our	79.90%	932	83.10%

5. Conclusion and future work

Focusing on some difficulties in target tracking, a target tracking algorithm based on ORB feature points is put forward in this paper. The algorithm is to extract edge motion saliency with Three-Frame Edge Difference algorithm and LK optical flow first, then perform Moving Edge Growth algorithm to get the edge of each motion target so as to segment target blocks. The ORB features of target blocks are extracted, which is use to realize tracking of each target by matching it with target templates. During tracking, TIL is updated in real time, where the updating of target template is to preserve the correct feature points in the template and effectively remove the outdated feature points, so as to ensure the persistent effectiveness of template. Experimental results show that our algorithm can achieves good tracking robustness in real time in the case of deformation and rotation of motion targets or partial inter-target occlusion.

However, we should also notice that our algorithm could not cope with the case in which the target is being "split" by small background obstacles, such as when one part of a target is on one side of obstacle, and the other part is on the other side. In that case, the target will be determined to be two different motion targets in the target detection phase as mentioned above. In future work we plan to solve this problem by taking other target features into consideration, thus further improving the robustness of target detection. In addition, how to apply the algorithm to the target tracking in the case of active backgrounds is one of the focuses.

Acknowledgement

The paper is funded by the National Natural Science Foundation of China (No.612-71407, No. 61301242).

References

[1] G. M Rao: (2013) *Visual Target Target Tracking Using Particle Filter: A Survey*, International Journal of Image, Graphics and Signal Processing, vol. 5, no. 6, pp. 57-71.

[2] A. Toloei and S Niazi: (2014) *State Estimation for Target Tracking Problems with Nonlinear Kalman Filter Algorithms*, International Journal of Computer Applications, vol. 98, no. 17, pp. 30-36.

[3] Y. Choi and Y Kim: (2014) *A target model construction algorithm for robust real-time mean-shift tracking*, Sensors, vol. 14, no. 11, pp. 20736-20752.

[4] S. Korman, D. Reichman, G. Tsur, and S: (2013) *Avidan, FasT-Match: Fast Affine Template Matching*, in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2331-2338.

[5] L. Li, Z. Han, Q. Ye, and J. Jiao: (2010) *Visual target tracking via one-class SVM, in Asian Conference on Computer Vision*, pp. 216-225.

[6] Z. Kalal Z, Mikolajczyk, and J. Matas: (2010) *Face-TLD: Tracking-Learning-Detection applied to faces*, in 17th IEEE International Conference on Image Processing, pp. 3789-3792.

[7] J. S. Liu and R. Chen: (2012) *Sequential Monte Carlo Methods for Dynamic Systems*, Journal of the American Statistical Association, vol. 93, no. 443, pp. 1032-1044.

[8] D. Comaniciu and P. Meer: (2002) *Mean shift: a robust approach toward feature space analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 5, pp. 603-619.

[9] T. Yang, Q. Pan, J. Li , and S. Z. Li: (2005) *Real-time multiple targets tracking with occlusion handling in dynamic scenes*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 970-975.

[10] D. G. Lowe: (2004) *Distinctive Image Features from Scale-Invariant Keypoints*, International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110.

[11] H. Y. Zhou, Y. Yuan, and C. M. Shi: (2009) *Target tracking using SIFT features and mean shift*, Computer Vision and Image Under-standing, vol. 113, no. 3, pp. 345-352.

[12] T. Tuytelaars and K. Mikolajczyk: (2008) *Local Invariant Feature Detectors: A Survey*, Foundations and Trends in Computer Graphics and Vision, vol. 3, no. 3, pp. 177-280.

[13] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski: (2011) *ORB: An efficient alternative to SIFT or SURF.* in *IEEE International Conference on* on Computer Vision, pp. 2564-2571.

[14] Y. Zhang , Z. L. Zhang, Z. K. Shen , and X. Y. Lu: *The images tracking algorithm using particle filter based on dynamic salient features of targets*, Acta Electronica Sinica, vol. 36, no. 12, pp. 2306-2311, 2008.

[15] [K. Joulan, N. Hautiere, and R. Bremond: (2011) *A unified CSF-based framework for edge detection and edge visibility*, in IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2011: 21-26.

[16] J. Q. Chen, T. N. Pappas, A. Mojsilovic, and B. E. Rogowitz: (2005) *Adaptive Perceptual Color- texture Image Segmentation*, IEEE Trans. on Image Processing, vol. 14, no. 10, pp. 1524-1536.

[17] K. B. Wang, B. Z. Yu, Q. Wang, and W. Xi: (2008) *Texture image segmentation using the without re-initialization geodesic active contour model*, Journal of Xidian University, vol. 35, no. 3, pp. 542-545.

[18] E. S. Li, S. L. Zhu, B. S. Zhu, Y. Zhao, C. G. Xia, and H. S. Li: (2009) *An adaptive edge-detection method based on the canny operator.* in International Conference on Environmental Science and Information Application Technology, pp. 465-469.

[19] S. Baker and I. Matthews: (2004) *Lucas-Kanade 20 Years On: A Unifying Framework*, International Journal of Computer Vision, vol. 56, no. 3, pp. 221-255.

[20] E. Rosten and T. Drummond: (2006) *Machine learning for high-speed corner detection, in European Conference on Computer Vision*, pp. 430-443.

[21] M. Calonder, V. Lepetit, C. Strecha C, and P. Fua: *BRIEF: binary robust independent elementary features*, in European Conference on Computer Vision, pp. 778-792, 2010.

[22] H. F. Lin, Y. F. Ma, and T. Song: (2010) *Research on Target Tracking Algorithm Based on SIFT, Acta Automatica Sinica*, vol. 36, no. 8, pp. 1204-1208.

[23] W. Y. Li, P. Wang, and H. Qiao: (2014) *A survey of visual attention based methods for target tracking*, Acta Automatica Sinica, vol. 40, no. 4, pp. 561-576.

Received May 7, 2017