Front. Electr. Electron. Eng. China 2007, 2(1): 1–7
DOI 10.1007/s11460-007-0001-3

RESEARCH ARTICLE

JIANG Shuhong, WANG Qin, ZHANG Jianqiu, HU Bo

An image-tracking algorithm based on object center distance-weighting and image feature recognition

Abstract A real-time image-tracking algorithm is proposed, which gives small weights to pixels farther from the object center and uses the quantized image gray scales as a template. It identifies the target's location by the mean-shift iteration method and arrives at the target's scale by using image feature recognition. It improves the kernel-based algorithm in tracking scale-changing targets. A decimation method is proposed to track large-sized targets and real-time experimental results verify the effectiveness of the proposed algorithm.

Keywords object tracking, video processing, mean-shift algorithm, feature recognition

1 Introduction

Real-time object tracking is a typical task in many computer vision applications, such as video surveillance, human-machine interaction systems, and object-based video compression, and so on. At present some object tracking and recognition methods always extract the motive target using successive frame difference detection method [1].The method regards the pixels as the motive target, which are larger than the threshold value after difference between the successive frames. However, when a target moves at a high speed, the displacement between the successive frames is large and this is likely to lead to the overlapping of target regions. The algorithm proposed by Isaac et al. is also a traditional tracking method [2]; it gives pyramid decomposition to the successive frames, extracts angular points, processes local correlation match and divides threshold to optical flow field so as to obtain the motive regions in the images. This algorithm does not give the accurate description of the target models and its computation is also very complex.

Translated from *Acta Electronica Sinica*, 2006, 34(7): 1175–1180 [译自: 电子学报]

JIANG Shuhong, WANG Qin, ZHANG Jianqiu (✉), HU Bo
Department of Electronic Engineering, Fudan University, Shanghai 200433, China
E-mail: jqzhang01@fudan.edu.cn

The kernel-based method proposed by Comaniciu and Meer [3] uses an isotropic kernel function to assign different weights to the color probability density function (PDF) of the target and achieves object tracking by measuring the similarity based on the Bhattacharyya (BH) coefficient between the successive frames, and finds the maximum value in the neighborhood by the mean-shift method [4] to get the motive target region. The kernel-based method is very simple and satisfies the real-time requirement; however, it is not quite accurate for using only $\pm 10\%$ bandwidth increment to estimate the current frame scale localization. Based on this algorithm, Peng Ningsong et al. proposed a kernel histogram filtering by Kalman filter [5, 6], in which $\pm \Delta h$, $h = 2$ was added to the kernel bandwidth to track target changing in size, but the scale estimate of the target was also relatively rough. The computational costs of the tracking algorithms using the mean-shift iterative method are directly proportional with the amount of the pixels in the target regions. So Comaniciu and Meer [3] and Peng Ningsong et al. [5] point out that they can allow tracking an object with the size 50×50 pixels at most in real time but not get a real-time track to a larger-sized target.

The algorithm proposed in this paper gives small weights to pixels farther from the object center and uses the quantized image gray scales as a template. It derives the target's location by the mean-shift method. At the same time, the algorithm sets up the binary image model, and arrives at the target's scale using image feature recognition. A decimation method is proposed to track large targets in real time whose size is in excess of some defined size (e.g., 31×31 pixel).

The remainder of this paper is organized as follows. Section 2 introduces the spatial localization procedures including the establishment of the histogram-based target model, the mean-shift iterative process and the decimation method. Section 3 describes the scale adaptation procedures including normalized moment of inertia (NMI) feature, shape feature introduction and image feature recognitions. The complete algorithm description and some experiments and comparisons are given in Section 4. Finally, Section 5 concludes the findings.

2 Spatial localization

2.1 Histogram-based target model

A target is represented by an ellipsoidal region [3], whose center is the central position of the target (x_c, y_c). All the pixels in the ellipsoidal region χ_i ($i=1, 2,\ldots,n$) are normalized to a unit circle

$$\chi_i^* = \left(\frac{x_i}{L_X}, \frac{y_i}{L_Y} \right)^{\mathrm{T}} \tag{1}$$

where L_X and L_Y are the horizontal axis and the vertical axis of the ellipse, respectively.

When tracking targets, the peripheral pixels are the least reliable, being often affected by occlusions (clutter) or interference from the background. In order to increase the robustness, we assigned different weights to all the pixels χ_i according to the distance between the pixels and the object center

$$w_d(\chi_i) = 1 - d_i^2 \tag{2}$$

where d_i is the distance of χ_i^* to the object center, and $d_i = \sqrt{\left(\frac{x_i - x_c}{L_X} \right)^2 + \left(\frac{y_i - y_c}{L_Y} \right)^2}$

A color histogram denotes the joint probabilities of the intensities of the three color channels [7]. We quantize the RGB color space into $m_0 \times m_0 \times m_0$ bins, and then its color histogram is defined as

$$u_i = R_i + m_0 G_i + m_0^2 B_i \tag{3}$$

On observing numerous tracking examples, we found that by using the grayscale histograms $m = m_0$, one can obtain the similar tracking effect as when using color histograms $m = m_0^3$; however, the computation cost to calculate the histograms is $1/m_0^2$. So our algorithm uses grayscale histograms: $b(\chi_i) \rightarrow \{1, 2,\ldots, m_0\}$. The weighted grayscale histograms of the target are then computed as

$$q_u = \frac{1}{C_q} \sum_{i=1}^{n} w_d(\chi_i)\, \delta\, [b(\chi_i) - u] \tag{4}$$

where $C_q = \sum_{i=1}^{n} w_d(\chi_i)$, by satisfying the condition $\sum_{u=1}^{m} q_u = 1$; n is the number of all the pixels located in the ellipsoidal region; δ is the Kronecker delta function.

2.2 The mean-shift iteration method

The weighted grayscale histograms of the kth frame target candidate are given by

$$p_{u,k}(\chi_{c,k}) = \frac{1}{C_{p,k}} \sum_{i=1}^{n_k} w_d(\chi_{i,k})\, \delta\, \left[b(\chi_{i,k}) - u \right] \tag{5}$$

The algorithm uses the BH coefficient between p and q as the similarity function among target model and candidates [3], and the bigger the BH coefficient is, the more similar the two images are.

The weighted grayscale histograms of the $(k-1)$th frame target are called $p_{u,k}(\chi_{c,k-1})$, shortening them as p_{u_0}. The search for the new target location in the kth frame starts at the location $\chi_{c,\,k-1}$ of the target in the previous frame using the mean-shift algorithm [4]. It is as follows

$$\chi_{c,k} = \frac{\sum_{i=1}^{n_k} \chi_{i,k} W_i}{\sum_{i=1}^{n_k} W_i} \tag{6}$$

where $W_i = \sum_{u=1}^{m} \left(\frac{q_u}{p_{u_0}} \right)^4 \delta\, [b(\chi_{i,k}) - u]$, which is the weight function.

If $\left\| \chi_{c,k} - \chi_{c,k-1} \right\| > \varepsilon$ (ε is less than one pixel distance), let $\chi_{c,k-1} = \chi_{c,k}$, and continue to compute p_{u_0} and W_i to get the new location $\chi_{c,k}$ until $\left\| \chi_{c,k} - \chi_{c,k-1} \right\| < \varepsilon$. The final $\chi_{c,k}$ is the new matched center location of the kth frame target.

2.3 Decimation

In order to track large-sized targets in real time we will employ a decimation method to the pixels in the ellipsoidal region when the major axis of the target ellipse L_{ma} is longer than L_T (suppose $L_T = 15$). Thus the biggest size of target is 31×31 pixel. The decimation proportion is

$$N_k = \left\lfloor \frac{L_{ma} - 1}{L_T} \right\rfloor + 1 \tag{7}$$

The decimation method is not employed to the kth frame image I_k but to the target ellipse. It starts at the central location $\chi_{c,k-1}$, preserves line $x_{c,k}$ and column $y_{c,k}$, and then samples one pixel every N_k pixels. The decimation method must be used around the new $\chi_{c,k-1}$ every time during the mean-shift iterative process.

Note: It cannot simply employ the decimation method to the kth frame image I_k and get the spatial localization result from the decimated I_k. If so, the smallest displacement during the mean-shift iterative process is N_k pixels, which largely increase the error. And line $x_{c,k}$ and column $y_{c,k}$ may not be sampled, which must have been given the biggest weight. All this most likely leads to the defeat of spatial localization.

3 Scale adaptation

In the kernel-based object tracking method [3], Comaniciu et al. add $\pm 10\%$ bandwidth increment to estimate the current frame scale localization. To avoid oversensitive scale adaptation, this method compels the scale change between

the successive frames to be $\pm 1\%$. It is a rough estimation and the inaccuracy will be accumulated by the increase in tracking frames. Moreover, the aspect ratio of the target is fixed. Yang Jie and Peng Ningsong [6] point out that the kernel-based algorithm can get good tracking result when the target size becomes smaller but when its size increases more and more, the kernel bandwidth will always become smaller instead of becoming larger.

We get the target's scale using image feature recognition, which improves the kernel-based algorithm in tracking scale-changing targets.

3.1 NMI feature and shape feature

Experimental results verify the scaling invariability, rotation invariability and translation invariability of NMI [8]. So NMI can be the object recognition feature of the tracking targets.

NMI of the grayscale image is defined as follows

$$\mathrm{NMI} = \frac{\sqrt{J_{(c_x, c_y)}}}{m} = \frac{\sqrt{\sum_{x=1}^{M}\sum_{y=1}^{N}((x-c_x)^2+(y-c_y)^2)f(x,y)}}{\sum_{x=1}^{M}\sum_{y=1}^{N} f(x,y)} \quad (8)$$

where $m = \sum_{x=1}^{M}\sum_{y=1}^{N} f(x,y)$ is the sum of all the gray scales of the image and (c_x,c_y) is the image centroid.

The shape feature of a target is its complexity, which is expressed by the ratio of its acreage A to its perimeter P (AVP). When two targets have the same acreage, the one that has the longer perimeter is the more complex one. The AVP feature only relates to shapes, and it has rotation invariability and translation invariability. And AVP is directly proportional to the target scaling. The shape factor of image [9] ($C = P^2/(4\pi A)$) has scaling, rotation and translation invariability. However, we found that AVP is more effective in the actual track examples obtained by using our algorithm.

3.2 Binary image model

In order to get the image feature for scale adaptation we must extract the edge of the target, that is to say, to set up the binary model according to some threshold value. Our algorithm defines the threshold value of binary image by the average grayscale value of the initial target region G_{tar} and the average grayscale value of the background region G_{b}

$$G_{\mathrm{thre}} = \frac{G_{\mathrm{tar}} - G_{\mathrm{b}}}{K} + G_{\mathrm{tar}} \ (K = 2) \quad (9)$$

Transfer $(2L_{X,0}+8)\times(2L_{Y,0}+8)$ pixels around target center $(x_{\mathrm{c},0}, y_{\mathrm{c},0})$ into binary image and divide the binary image into pieces of connective regions, then choose the biggest connective region as binary model. (Note: The target's gray scale is defined as 1 and the background's gray scale is 0 for the convenience of the algorithm program implementing. When $G_{\mathrm{tar}} < G_{\mathrm{b}}$, the binary image must be reversed and the reverse flag $f = 1$ is recorded.)

Compute NMI_0 and AVP_0 of the binary target model, record the size proportion of the binary model and the grayscale model

$$X_{\mathrm{scale}} = \frac{L_{X,0}}{L_{\mathrm{bw}X,0}},\ Y_{\mathrm{scale}} = \frac{L_{Y,0}}{L_{\mathrm{bw}Y,0}} \quad (10)$$

where, $L_{\mathrm{bw}X}$ is the farthest distance of the x axis in the binary model and $L_{\mathrm{bw}Y}$ is the farthest distance of the y axis in the binary model.

3.3 Image features recognition

The center of search scope for the new target location in the kth frame starts at $\chi_{\mathrm{c},k-1}$. We define the search region as S_{l} rows and S_{c} columns

$$\begin{aligned} S_{\mathrm{l}} &= \max\ (S_{\mathrm{l}0}+1,\ 4L_{X,k-1}+1) \\ S_{\mathrm{c}} &= \max\ (S_{\mathrm{c}0}+1,\ 4L_{Y,k-1}+1) \end{aligned} \quad (11)$$

In normal cases, the search scope is four times of the target size. However, when its size is too small, we limit the smallest search scope size as 65×65 pixel, that is $S_{\mathrm{l}0} = S_{\mathrm{c}0} = 64$.

The computation cost of the binary image search is small, so we use the decimation method only when the size of the kth frame ellipsoidal target is more than two times of the initial target. The decimation proportion is

$$M_k = \left\lfloor \frac{N_k - 1}{N_0} \right\rfloor + 1 \quad (12)$$

where N_0 and N_k are the decimation proportions of the initial grayscale model and the kth frame grayscale image.

We can divide the binary image of the decimated search region into pieces of connective regions, then find some candidate connective regions whose acreages are similar to the target model.

$$\alpha_1 < \frac{(M_k)^2 A_i}{A_0} < \alpha_2 \quad (13)$$

where $\alpha_1 = 0.5$, $\alpha_2 = 2$, and A_0 is the actually acreage of the binary model.

Next, calculate NMI_i and AVP_i of each candidate connective regions, then find the candidate targets whose image feature are similar to the binary model, that is, they should satisfy the following two conditions

$$\eta_{\mathrm{l}} < \frac{\mathrm{NMI}_j}{\mathrm{NMI}_0} < \eta_2,\ \eta_{\mathrm{l}} < \frac{M_k \mathrm{AVP}_j}{\mathrm{AVP}_0} < \eta_2 \quad (14)$$

where $\eta_1 = 0.8$, $\eta_2 = 1.2$.

To calculate

$$\begin{aligned} L_{X,(j)} &= M_k L_{\mathrm{bw}X,(j)} X_{\mathrm{scale}} \\ L_{Y,(j)} &= M_k L_{\mathrm{bw}Y,(j)} Y_{\mathrm{scale}} \end{aligned} \tag{15}$$

If $\eta_1 < \dfrac{L_{X,(j)}}{L_{X,k-1}} < \eta_2$ and $\eta_1 < \dfrac{L_{Y,(j)}}{L_{Y,k-1}} < \eta_2$, and there is only one candidate target, then the scale adaptation of the target is

$$L_{X,k} = L_{X,(j)}, L_{Y,k} = L_{Y,(j)} \tag{16}$$

If there are more than one candidate targets that satisfy these conditions, the scale adaptation has failed. Because there are some disturbing images in the background whose acreage, NMI and image feature are all similar to the tracking target, we cannot get the target's scale in such conditions and we conclude that the scale location has failed.

3.4 Updating binary image model

If scale adaptation is successful, update the binary image model as follows

$$\begin{aligned} A_0 &= M_k^2 A_j \\ \mathrm{NMI}_0 &= \mathrm{NMI}_j \\ \mathrm{AVP}_0 &= M_k \mathrm{AVP}_j \end{aligned} \tag{17}$$

4 Algorithm description and experiment results

4.1 Complete algorithm description

1) Read the current frame image.

2) Scale adaptation: get $L_{X,k}$ and $L_{Y,k}$ of the target ellipse according to the acreage, NMI and AVP in the search region of S_1 rows and S_c columns around $\chi_{c,k-1}$ of the target in the previous frame, and then update the binary image model. Hold $L_{X,k} = L_{X,k-1}$ and $L_{Y,k} = L_{Y,k-1}$ if we cannot find a binary target that satisfies all the image feature conditions by reason that the gray scale of background is too similar to the target, and so on.

3) Carry out the mean-shift process by $L_{X,k}$ as the horizontal axis and $L_{Y,k}$ as the vertical axis of the target ellipse starting at the location $\chi_{c,k-1}$ until $\left\|\chi_{c,k} - \chi_{c,k-1}\right\| < \varepsilon$, where the final $\chi_{c,k}$ is the

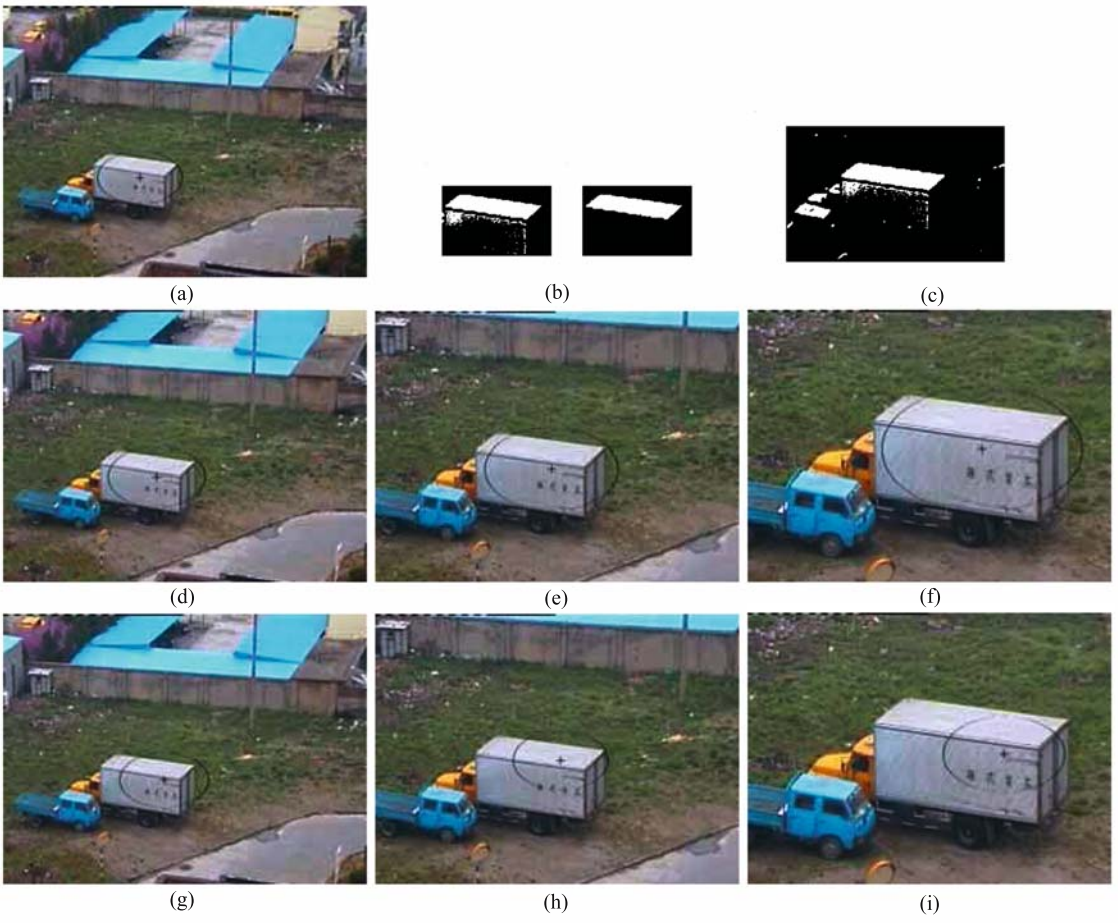

Fig. 1 Tracking a large-sized target
(a) Initial model; (b) Binary image and binary model; (c) Binary image of search regions in the second frame; (d)–(f) Frame 16, 54 and 96 of our algorithm; (g)–(i) Frame 16, 54 and 96 of the kernel-based algorithm

new matched center location of the kth frame target. (The decimation method must be used around the new $\chi_{c,k-1}$ every time during the mean-shift iterative process if $L_{ma} > L_T$).

4) Add $\pm 10\%$ bandwidth increment to estimate the current frame scale adaptation [3] as the supplement if the scale localization according to the image feature failed in the second step.

Note: If we find that the average gray scale of the background is very close to the target when setting up the binary model, namely the difference of their gray scale is less than the preestablished value in the algorithm program, we will cancel the second step and only perform the first, third, fourth steps.

4.2 Experiment results

The implementation of our tracking system has been tested on a P4 2.4 GHz PC where the tracker runs at a rate of 30 fps on Visual C^{++} implementation. The target histograms have been derived from the grayscale space with $m = 32$ bins.

The first example presented the tracking of a large-sized target. The image size was 360×480 pixels. The initial image is shown in Fig. 1(a), the target size was 71×115, which was in excess of the biggest size (50×50) that Comaniciu and Meer [3] and Peng Ningsong et al. [5] can track in real time. The connective regions in the $(2L_{X,0}+8) \times (2L_{Y,0}+8)$ pixels binary image were divided in Fig. 1(b) and the biggest one was chosen as the binary model. Fig. 1(c) shows the S_l rows and S_c columns search regions in the second frame. The tracking target became bigger in the tracking process and its size reached up to 181×281 in the 96th frame (Fig. 1(f) and Fig. 1(i)). It can be demonstrated from Fig. 1(d)–(f) that both spatial localization and scale adaptation were correct by our algorithm; however, Fig. 1(g)–(i) shows that scale adaptation using $\pm 10\%$ bandwidth increment in kernel-based algorithm was not accurate and that the error was accumulated through the tracking sequence, which resulted in spatial localization failure.

In the second example we carried out a tracking test to a motive target whose size became bigger through the entire tracking sequence. The average grayscale value of the initial target region was less than the one of the background ($G_{tar} < G_b$), so the binary image must be reversed, as shown in Fig. 2(b). It is clear from Fig. 2(e)–(h) that both spatial localization and scale adaptation were correct by our algorithm; however, Fig. 2(i)–(l) shows that the tracking ellipse became smaller instead of become larger using the kernel-based algorithm when the actual target became bigger. And scale adaptation failure led to spatial localization failure; Fig. 2(l) demonstrates that the tracking ellipse is located on the neck of the man instead of the man's face (the target).

The third example is shown in Fig. 3. Figure 3(f) is the binary target model; Fig. 3(g) shows the search regions and the divided binary target of the 66th frame; the proportion of the horizontal axis to the vertical axis changed according to the girl's gesture (however, the aspect ratio of the target is fixed in the kernel-based algorithm). When the girl walked close to the border there was some disturbance in the tracking because the gray scale of the background was very close to the target, which is demonstrated in Fig. 3(h). So we could not find a binary target that satisfied all the image feature

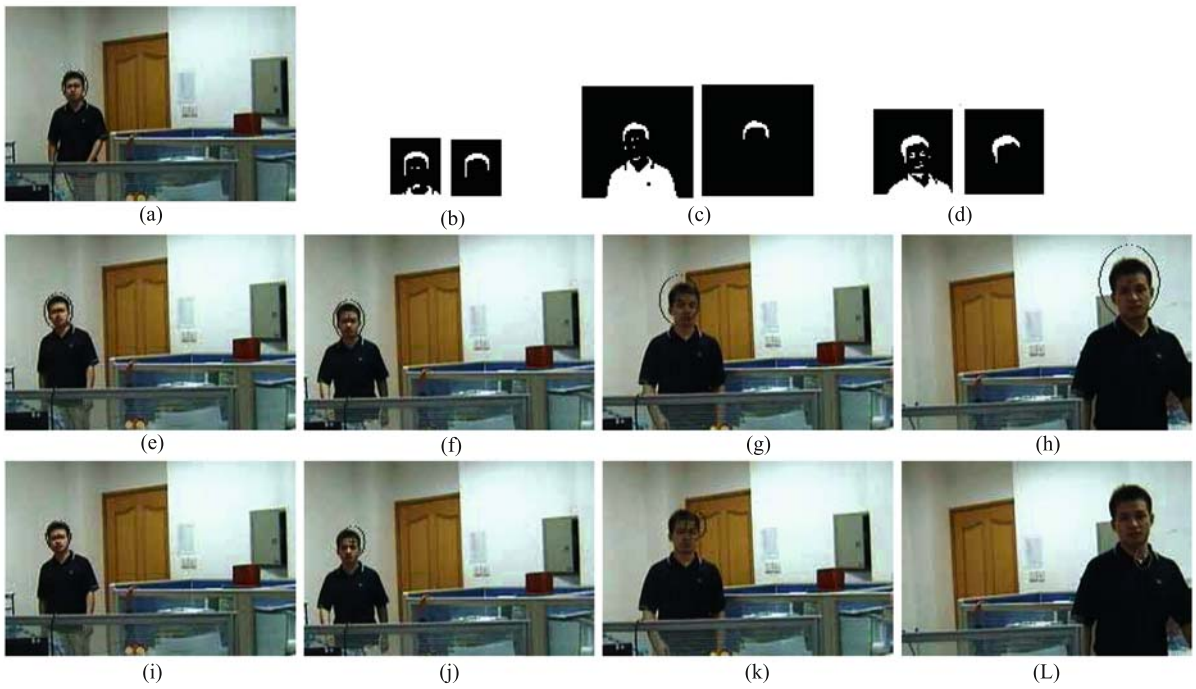

Fig. 2 Tracking of a motive target whose size becomes bigger
(a) Initial model; (b) Binary model; (c) Search regions and divided binary target of 2nd; (d) The 141st frame; (e)–(h) Frame 16, 34, 85 and 141 of our algorithm; (i)–(l) Frame 16, 34, 85 and 141 of the kernel-based algorithm

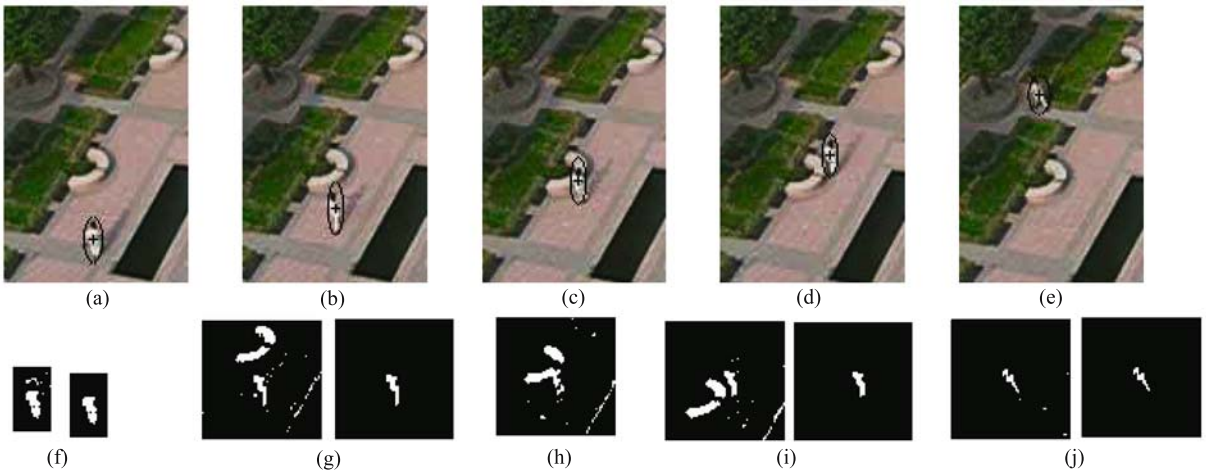

Fig. 3 Tracking background disturbing target
(a) Initial model; (b)–(e) Frame 66, 107, 150 and 262 of our algorithm; (g)–(j) The search regions and divided binary target of frame 66, 107, 150 and 262

conditions. Then we held the previous frame target scale and added $\pm 10\%$ bandwidth increment to estimate the current frame scale adaptation. When the girl walked away from the border (Fig. 3(d) and (i)) we could divide the binary target correctly to get accurate scale adaptation again. Finally Fig. 3(d) and Fig. 3(e) demonstrate that the aspect ratio of the 262nd frame was quite different from the 150th frame.

The fourth example is shown in Fig. 4. The target was a white dog, whose gray scale was very similar to the background. When setting up the threshold value of the binary model we could find the average gray scale of target ($G_{\text{tar}} = 135$) was too close to the background ($G_{\text{b}} = 144$), so this example would not employ scale adaptation by binary model. The white dog target was so similar to the background

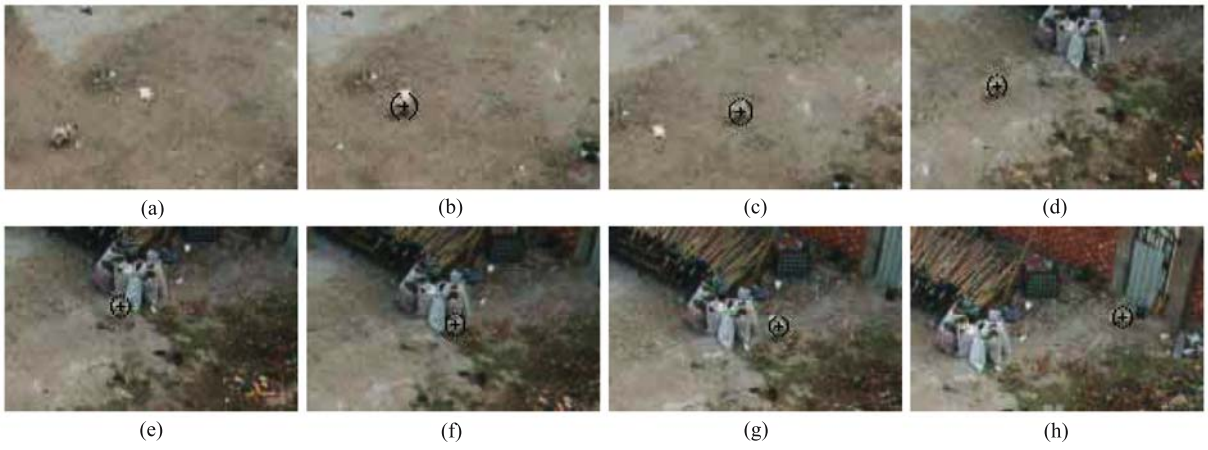

Fig. 4 Tracking a target similar to background
(a) Tracking target; (b)–(h) Frame 26, 54, 118, 149, 160, 176 and 234 images

Fig. 5 Tracking under low SNR circumstance
(a) Tracking target; (b)–(d) Frame 9, 56 and 121

dump in Fig. 4(e) and (f) that we even could not distinguish them by eye; however, our algorithm tracked the dog quite exactly.

The tracking target in the fifth example was a black car running at night. Similarly, this example would not employ scale adaptation by binary model. The night background was dark and the glistening of the road surface influenced the image grayscale histograms; so the task of tracking was quite difficult. However, our algorithm was still able to track the car through the entire sequence.

5 Conclusions

The main contribution of this paper is to introduce a real-time image-tracking algorithm, which separates scale adaptation from spatial localization. This algorithm gives small weights to pixels farther from the object center, uses the quantized image gray scales as a template and identifies the target's location by the mean-shift method. At the same time, the algorithm sets up the binary image model, and extracts the target image feature and gets the target's scale adaptation according to the binary target's acreage, NMI and AVP. It improves the kernel-based algorithm in tracking scale-changing targets. A decimation method is proposed to track large-sized targets. The experimental results indicate that the algorithm can track 181×281 pixel large-sized targets in real time. The proposed algorithm has already been applied to real-time tracking systems by data acquisition using PC camera.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant No. 60572023).

References

1. Thomas M. Automatic segmentation of moving objects for video object plane generation. IEEE Trans. on Circuits and Systems for Video Technology, 1998, 8(5): 525–528
2. Isaac C, Gerard M. Detection and tracking of objects in airborne video imagery. Workshop on Interpretation of Visual Motion CVPR, 1998
3. Comaniciu D, Meer P. Kernel-based object tracking. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2003, 25(5): 564–577
4. Comaniciu D, Meer P. Mean shift: A robust approach toward feature space analysis. IEEE Trans. on Pattern Analysis and Machine Intelligence, 2002, 24(5): 603–619
5. Peng Ningsong, Yang Jie, Liu Zhi. Mean shift blob tracking with kernel histogram filtering and hypothesis testing. Pattern Recognition letters, 2005, 26: 605–614
6. Yang Jie, Peng Ningsong. Visual tracking research based on robustness of mean-shift method. http://www.paper.edu.cn, April 2005 (in Chinese)
7. John R., Shih-Fu Chang. Tools and techniques for color image retrieval. Columbia University Department of Electrical Engineering and Center for Telecommunications Research, March 6, 1996
8. Yang Xiaogang, Fu Guangyuan, et al. An new approach to target recognition based on image NMI feature. Computer Engineering, 2002, 28(6): 149–152 (in Chinese)
9. Jan F. On the inverse problem of rotation moment invariants. Pattern Recognition, 2002, 35(12): 3015–3017