A Foreground Extraction Method by Using Multi-Resolution Edge Aggregation Algorithm

Wenqian Zhu, Bingshu Wang, Xuefeng Hu, and Yong Zhao(✉)

School of Electronic and Computer Engineering,
Shenzhen Graduate School of Peking University, Shenzhen 518000, China
{zhuwenqian,wangbingshu}@sz.pku.edu.cn
huxuefeng@pku.edu.cn, yongzhao@pkusz.edu.cn

Abstract. Foreground extraction is a fundamental step of video analysis. The common solution for foreground extraction is background subtraction which is based on color information. However, color is sensitive to intensity changes and may lose efficacy in complex scenes such as scene with low contrast or strong illumination. To overcome the disadvantages of color-based methods, we propose a new approach based on edge information. We get the edge of foreground from color-based background instead of edge-based background to reduce calculation amount. And a novel multi-resolution edge aggregation algorithm is used to solve the edge-filling problem, especially in the case when edge is not continuous. This algorithm obtains foreground region through expands the influence of the edges and reduces the gaps between the edges. Both visual and quantitative comparisons on various image sequences validate the efficacy of our method.

Keywords: Foreground extraction · Multi-resolution · Edge aggregation

1 Introduction

Recent years, more and more surveillance cameras have been used in schools, shopping malls, airports and governments to guarantee the safety and property of the people. The data volume of these surveillance cameras is too large to analyse all through manual work. So, extracting information from videos automatically is of great importance. Foreground extraction is the first step of intelligent video analysis and is the foundation of further processing such as object tracking, activity understanding and safety forecast. A good foreground extraction method can greatly reduce the follow-up workload.

Foreground extraction is usually based on background subtraction. Obtain a statistical model at first to represent the approximate ideal background and then detect the moving objects through background subtraction. Most background subtraction methods adopt color information to obtain foreground region. However, color is sensitive to intensity changes and may lose efficacy in complex scenes. Such as:

- **Low Contrast:** When the color difference between object and background is very small, we call it low contrast condition. Such as videos captured at night or captured by far-infrared cameras. The color-based foreground extraction usually has a predefined

Y.-J. Zhang (Ed.): ICIG 2015, Part I, LNCS 9217, pp. 75–85, 2015.
DOI: 10.1007/978-3-319-21978-3_8

threshold to distinguish foreground from background. If the color similarity between object and background is lower than the threshold, the object will be marked as background falsely.

- **Strong Illumination:** Sharp, strong or fast-changing illumination, such as vehicle headlight at night, usually changes the color characteristics of background completely and invalidates the color-based foreground extraction methods. A large part of the test frame will be detected as foreground due to the influence of changing light.
- **Slow Motion:** If an object moves very slowly, a large part of the object will be learned as background by the background update mechanism. When we use background subtraction, the slow moving objects will be totally or partly lost as it has been already learned in background model.

Different from color, edge is a good feature for foreground extraction due to its immutability to light variation. In this paper, we propose an edge-based method to solve the above-mentioned issues. First, background modeling is used to get the background of the video. Second, both edge information of test frame and corresponding background is calculated to get the edge of foreground. Then, a novel multi-resolution edge aggregation method is used to solve the problem of filling discontinuous edges. In edge aggregation, we expand the influence of foreground edge and reduce the gap between the edges to aggregate them into regions. Finally, we adjust the size of foreground region to get a more precise result.

The rest of our paper is organized as follows. In Sect. 2, we briefly review the related work of foreground extraction. The details of the proposed method are introduced in Sect. 3. In Sect. 4, experiments comparing our method with color-based method are presented. Section 5 concludes the paper.

2 Related Work

Up to now, numerous methods have been proposed for foreground extraction. These methods have been already summarized in some surveys [1–4]. Here, we classify these methods into three categories according to the feature used in the detection: color-based, edge-based and color-and-edge-based methods.

Color-based Methods: Color information is used in lots of foreground detection methods. MOG (Mixture of Gaussian) is one of the most popular schemes [5–7]. Several numbers of gaussian functions are used to learn the dynamic background and the foreground is detected by the intensity difference of coming frame and the background. Codebook algorithm [8, 9] uses codebook to represent each pixel. Each codebook models a long image sequence in a compressed form with a set of codewords. SOBS (Self-Organizing Background Subtraction) [10, 11] constructs background based on self organization through artificial neural networks and foreground is got if the euclidean distance of a coming pixel and the model in the HSV color hexcone is larger than a predefined threshold. However, color-based methods are sensitive to intensity changes which lead to a large mount of false detections.

Edge-based Methods: This kind of method has been already studied by some researchers. Jain et al. [12] detected foreground by modeling background through subpixel edges. Edges deviated from the model are masked as foreground. Adin et al. [13, 14] adopted kernel-density distributions to represent edges' behaviors and used segment features to overcome variations of edges. Jabid et al. [15] detected the edges of moving objects by using three consecutive frames and maintained the texture information with local directional pattern descriptor. The defect of these methods is that the edge of foreground is usually discontinuous and hard to be filled into regions.

Color-and-Edge-based Methods: Some methods used both color and edge information to obtain better solutions. These methods usually build color and edge model separately and then combine the results in a certain mechanism. Jabri et al. [16] introduced color and edge confidence map and combined them by taking their maximum. Li and Leung [17] used a mathematical method, edge difference is measured by correlation and then edge and color differences are combined by energy function minimization. Javed et al. [18] fused the two features in a straight forward way, foreground region detected by color model will be removed if the region is short of edges. The shortcoming of these methods is that they should model for both edge and color information which increases the complexity of computation.

In this paper, we propose an edge-based method. Most of the existing edge-based methods get foreground edges by using edge magnitude and direction to build a background and then subtract it from the edge of test frame. Different from them, we just use color-based background and detect the edges of both background and test frame to obtain the foreground edge. We believe that the foreground edges obtained from color-based background won't lose much accuracy than that obtained from edge-based background. But it is much easier to realize as images are stored in form of color. Besides, we solve the edge filling problem which faced by all the edge-based methods by a novel edge aggregation algorithm which is easy to implement.

3 Proposed Method

We take our inspiration from the observation that edges can get correct object profile when foreground derived by color difference is terrible, such as in low contrast, strong illumination and slow motion conditions as we discussed before. But the obtained edges of objects are usually incomplete and have much noise. In this situation, how to denoise and how to transform these incomplete edges into foreground regions become a big challenge. To solve these problems, a novel multi-resolution edge aggregation method is used here. The core idea of this algorithm is to enhance the influence of strong edges and suppress the weak edges, which may be the remnant of background edges in subtraction, at the same time. Edge pictures of foreground are present in the form of grayscale images. Strong edges have high pixel values, while weak edges have low pixel values. Pixels inside moving objects will have some strong edges nearby. The expansion of its surrounding edges will make it has a high value. Otherwise, if a pixel belongs to background without strong edges around, its intensity will keep in low level. The expansion of the strong edges will

finally aggregate the foreground edges into high-intensity regions. The whole procedure can be sketched in Fig. 1 and will be discussed in details below.

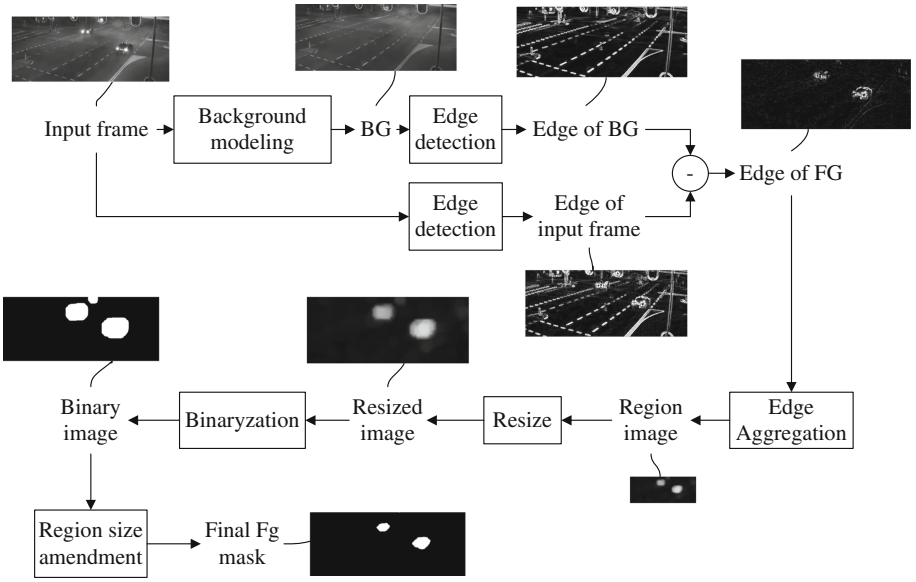

Fig. 1. Procedure of foreground extraction by multi-resolution edge aggregation

3.1 Background Modeling and Edge Detection

This step is used to obtain the edge of foreground, which is the foundation of our method. We get foreground edge by using color-based background rather than edge-based background, because the technique of color-based background modeling is much more mature and easier to realize. Though foreground edge obtained in this way may lose some precision, it won't much influence our final results. Because we designed an effective edge filling algorithm which works well even under the condition that the edge is not complete.

We don't focus on background modeling as it is only an auxiliary step of our method and has already been widely studied. The existing color-based background modeling algorithm is enough for us, we can choose any color-based background modeling method, such as MOG, Codebook, SOBS or any else. Here, we use MOG (Zivkovic) [2] as it is a mature method and can be easily obtained in OpenCV.

In edge detection part, a 3×3 Scharr filter is used to obtain the horizontal and vertical edge images G_x and G_y of input frames. And the pixel value at location (i, j) of the edge picture E is calculated by:

$$E(i,j) = \frac{1}{2}\left|G_x(i,j)\right| + \frac{1}{2}\left|G_y(i,j)\right| \tag{1}$$

Then the pixel value at location (i,j) in the edge pictures of foreground E_{fg} can be got from the edge picture of current frame E_{cur} and the edge picture of background E_{bg}:

$$E_{fg}(i,j) = \left| E_{cur}(i,j) - E_{bg}(i,j) \right| \tag{2}$$

3.2 Edge Aggregation

As for edge aggregation, we attempt to transform foreground edges into regions. The core idea of this algorithm is to expand the influence of edges. We elevate the values of pixels that near edges gradually to diminish the gaps between the edges. When the gaps disappear, the edges are aggregated into regions.

The algorithm is implemented iteratively. During each iteration, we zoom out the edge picture at first. In this way, the hole inner the edges will be diminished. Then, median filter is used to suppress noise and remove weak edges, which may be the remains of background edge. Later, dilation is adopted to strengthen edges and diminish the hole inner the edges again. After K times repeat of these operations, the picture will become very small and the edges will aggregated into regions. The flow chart of this step is shown in Fig. 2.

1. **Zoom Out Picture:** In this step, bilinear interpolation is used, which is one of the basic resampling techniques. For an input image with size $M \times N$, we shrink its size to $sM \times sN$. Here s is a factor smaller than 1. Smaller s helps to reduce calculation, while larger s will keep more details of the foreground. 0.5 is recommended here.
2. **Median Filtering:** This step is used to reduce noise and remove weak edges. We call edges with low intensity and narrow width as weak edges. The continuity of weak edges is broken during zooming out step, so they can be removed by median

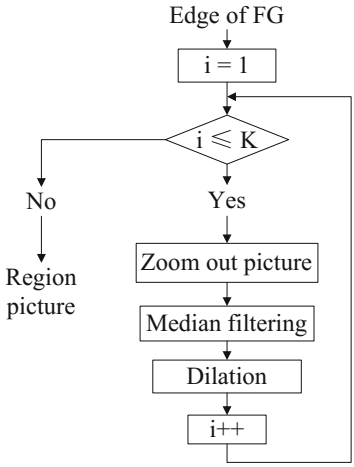

Fig. 2. Flow chart of edge aggregation

filter just like isolate noise. The window size of median filter we use is 3. Because after zooming out, the size of picture will become very small, large window size will make some foreground edges disappear.

3. **Dilation:** After zooming out and median filtering, the edges of object can be very thin. To enhance the edge and diminish the hole inner the edges again, a 3×3 dilation is used here.

Figure 3 illustrates the procedure with $K = 3$ and $s = 0.5$. It can be found that during each iteration, the image size reduced by half, the edges in the image are blurred and the holes inner the edges are diminished. Although the input edges are incomplete and have much noise, after K times processing, the foreground edges are aggregated into regions.

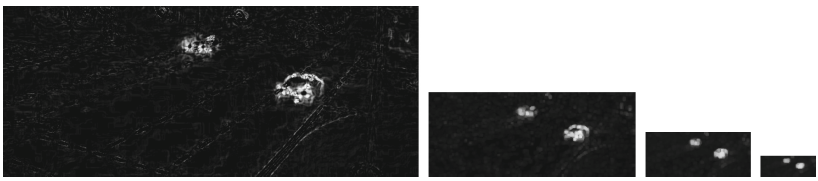

Fig. 3. Results of edge aggregation. (a) Edge of foreground. (b) The result of first iteration. (c) The result of second iteration. (d) The result of third iteration.

3.3 Resize

In step 3.2, we zoomed out the image K times, with deformation factor s. So we need to zoom in the image with factor $1/s^K$ to enlarge it to input frame size. Here we also use bilinear interpolation.

3.4 Binarization

The foreground picture we got from step 3.3 is grayscale. To obtain the binary foreground mask, binarization is employed here. A grayscale threshold T is used here.

$$T = \max\left(avg\left(E\right), 50\right) \tag{3}$$

Where $avg\left(E\right)$ here denotes the average pixel value of the image E. Pixels with value larger than T are marked as foreground and pixels with value smaller than T are marked as background. We require the regions of foreground objects have larger intensity than $avg\left(E\right)$. And to avoid false detection when no object exists in the frame, we require the pixel value of object region larger than 50, which is chosen through experiment.

3.5 Region Size Amendment

The foreground region after binary processing will be larger than the real region because dilation is used in edge aggregation. So we need to erode the regions to correct size. The question is how many pixels should we erode? Through analyzing our method, we can find that only zooming out, zooming in and dilation steps will affect the region size.

Consider horizontal direction only. When we zoom in or zoom out an image with factor s, the width of an edge w will change to sw. In dilation step, we use 3×3 structural element. It means the width of an edge will add 2 pixels, 1 pixel each side.

Let $w_i, \quad i = 1, 2, \ldots, K$ denotes the width of an edge after i times iteration in step 3.2. During each iteration, we zoom out and dilate the picture for one time. For the reasons discussed above, the width of edge after K times iteration will be:

$$w_K = sw_{K-1} + 2 = s(sw_{K-2} + 2) + 2 = \cdots = s^K w_0 + 2 \times \frac{1 - s^K}{1 - s} \tag{4}$$

After resizing picture to input size, the edge width becomes:

$$w_{re} = \frac{w_K}{s^K} = w_0 + 2 \times \frac{1 - s^K}{s^K (1 - s)} \tag{5}$$

The calculation shows that after these operations, the edge will have $2\left(1 - s^K\right) / s^K (1 - s)$ extra pixels than before, $\left(1 - s^K\right) / s^K (1 - s)$ pixels each side. The pixels inside the regions fill the edge but the pixels outside enlarge the region. To get correct region size, we need to erode the region with $\left(1 - s^K\right) / s^K (1 - s)$ pixels from the outside. Besides, it also means that the max gap size can be filled is $2\left(1 - s^K\right) / s^K (1 - s)$ pixels. It gives a numerical way to choose parameters. We can choose a smaller K if the gaps between edges are narrow and a larger K in reverse.

4 Experiment Results

Here we evaluate our method by comparing the performances with MOG (Zivkovic) and Codebook. The foreground gotten by our method is based on edge information, while foreground gotten by MOG and Codebook are based on color information.

Values of parameters in our method are chosen as $K = 3, s = 0.5$. For parameters in MOG, we choose model number as 3, learning rate as 0.005 and other parameters as default values in OpenCV. Codebook is a nonparametric method and we use the first 30 frames to learn background. Shadow detection is not considered in all the methods. The dataset we use is CDW-2014, which is available at www.changedetec-tion.net [19].

4.1 Visual Comparisons

Figure 4 shows the results of foreground extraction using our method, MOG and Codebook on test sequences. Both our method and MOG use the same background which gotten from Gaussian model while Codebook uses another background which represented by codebook.

It can be found that our method has many advantages through comparison. First, our method can conquer weak shadows as shown in (a) and (c), because weak shadow is lack of edges. Second, our method can obtain a relatively good result in low contrast conditions, as shown in (e), (f) and (g). Third, the side effect of light at night can be

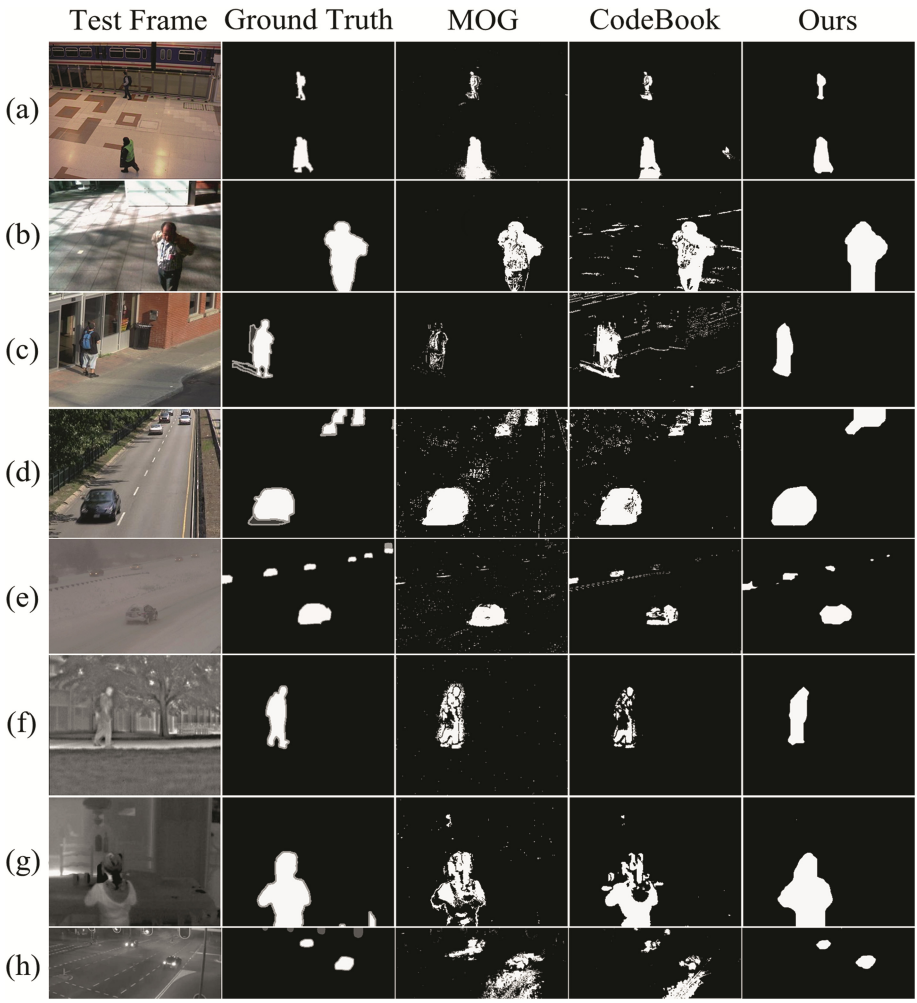

Fig. 4. Visual comparison results using MOG, Codebook and our method on test sequences. (a) PETS2006. (b) PeopleInShade. (c) BusStation. (d) Highway. (e) Blizzard. (f) Park. (g) DiningRoom. (h) StreetCornerAtNight.

overcome as edge is insensitive to light which is shown in (h). Fourth, our method can present a more complete result in slow motion condition as shown in (b) where the person in the video stayed for a while and has been partly learned as background.

But there are also some problems in our method. The details of objects are lost in our method, for example, the gaps between the legs in (a) and (f) are filled. Adjacent objects may stick together if the gaps between two objects are small as shown in (d). These are because every little gap between edges will be filled by the mechanism of edge aggregation. Further work is needed to deal with these problems.

4.2 Quantitative Comparisons

Recall, Precision and F-Measure are used for quantitative comparisons. Let TP represent for true positive, FP for false positive, FN for false negative and TN for true negative. Then

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{6}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7}$$

$$\text{F} - \text{Measure} = \frac{2 \times \text{Precison} \times \text{Recall}}{\text{Precison} + \text{Recall}} \tag{8}$$

Recall reflects how many percents of pixels in ground truth are detected. Precision reflects what percentage of pixels that are marked as foreground is correct. F-Measure is the weighted harmonic mean of precision and recall. High recall and high precision will lead to high F-Measure which means good result.

Table 1 presents the numerical comparison of our method, MOG and Codebook. Bigger values are shown in bold. From the values we can observe that proposed method presents a more superior result in most videos. Edge aggregation mechanism ensures the object integrity in our results, which leads to high recall. Noise and fake edges are suppressed, which guarantees high precision. The F-Measure of our method outperforms MOG's by an average of 11 % and outperforms Codebook's by an average of 8 %, which verifies the effectiveness of our method.

Table 1. Quantitative comparison among MOG, codebook and our method

Video names	Recall			Precision			F-Measure		
	MOG	CB	Ours	MOG	CB	Ours	MOG	CB	Ours
PETS2006	0.673	**0.835**	0.664	0.583	0.702	**0.838**	0.625	**0.763**	0.741
PeopleInShade	0.652	**0.966**	0.585	**0.772**	0.494	0.672	**0.707**	0.654	0.625
BusStation	0.486	**0.797**	0.739	**0.802**	0.386	0.750	0.605	0.520	**0.744**
Highway	0.878	0.827	**0.895**	**0.833**	0.749	0.749	**0.855**	0.786	0.816
Blizzard	**0.662**	0.650	0.589	0.739	0.926	**0.929**	0.698	**0.764**	0.721
Park	**0.731**	0.470	0.672	0.657	**0.870**	0.785	0.692	0.611	**0.724**
DiningRoom	0.371	0.820	**0.822**	0.902	0.795	**0.939**	0.526	0.807	**0.876**
StreetCornerAtNight	0.774	0.648	**0.893**	0.157	0.197	**0.491**	0.260	0.303	**0.634**
Average	0.653	**0.752**	0.732	0.681	0.640	**0.769**	0.621	0.651	**0.735**

5 Conclusions

We have presented a foreground extraction method based on edge information and solved the problem of edge filling by a novel multi-resolution edge aggregation algorithm. The method is easy to implement and performs well, especially in low contrast, strong illumination and slow motion scenes, where the color information doesn't work. Plenty of experiments are made to evaluate the effectiveness of our method. Compared with foreground extracted from color-based method, the result of our method has less noise and better integrity, which reflects on both visual perception and statistical data.

References

1. Brutzer, S., Höferlin, B., Heidemann, G.: Evaluation of background subtraction techniques for video surveillance. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 1937–1944 (2011)
2. Piccardi, M.: Background subtraction techniques: a review. In: IEEE International Conference on Systems, Man and Cybernetics, vol. 4, pp. 3099–3104 (2004)
3. Radke, R.J., Andra, S., Al-Kofahi, O., Roysam, B.: Image change detection algorithms: a systematic survey. IEEE Trans. Image Process. **14**, 294–307 (2005)
4. Benezeth, Y., Jodoin, P.M., Emile, B., Laurent, H., Rosenberger, C.: Review and evaluation of commonly-implemented background subtraction algorithms. In: 19th International Conference on Pattern Recognition, pp. 1–4 (2008)
5. Stauffer, C., Grimson, W.E.L.: Adaptive background mixture models for real-time tracking. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 246–252 (1999)
6. Zivkovic, Z.: Improved adaptive Gaussian mixture model for background subtraction. In: 17th International Conference on Pattern Recognition, vol. 2, pp. 28–31 (2004)
7. Lee, D.-S.: Effective Gaussian mixture learning for video background subtraction. IEEE Trans. Pattern Anal. Mach. Intell. **27**, 827–832 (2005)
8. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Background modeling and subtraction by codebook construction. In: International Conference on Image Processing, vol. 2, pp. 3061–3064 (2004)
9. Kim, K., Chalidabhongse, T.H., Harwood, D., Davis, L.: Real-time foreground-background segmentation using codebook model. Real-Time Imaging **11**, 172–185 (2005)
10. Maddalena, L., Petrosino, A.: A self-organizing approach to background subtraction for visual surveillance applications. IEEE Trans. Image Process. **17**, 1168–1177 (2008)
11. Maddalena, L., Petrosino, A.: The SOBS algorithm: what are the limits? In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 21–26 (2012)
12. Jain, V., Kimia, B.B., Mundy, J.L.: Background modeling based on subpixel edges. In: International Conference on Image Processing, vol. 6, pp. VI321–VI324 (2007)
13. Ramirez Rivera, A., Murshed, M., Chae, O.: Object detection through edge behavior modeling. In: 8th IEEE International Conference on Advanced Video and Signal Based Surveillance, pp. 273–278 (2011)
14. Ramirez Rivera, A., Murshed, M., Kim, J., Chae, O.: Background modeling through statistical edge-segment distributions. IEEE Trans. Circ. Syst. Video Technol. **23**, 1375–1387 (2013)

15. Jabid, T., Mohammad, T., Ahsan, T., Abdullah-Al-Wadud, M., Chae, O.: An edge-texture based moving object detection for video content based application. In: 14th International Conference on Computer and Information Technology, pp. 112–116 (2011)
16. Jabri, S., Duric, Z., Wechsler, H., Rosenfeld, A.: Detection and location of people in video images using adaptive fusion of color and edge information. In: 15th International Conference on Pattern Recognition, vol. 4, pp. 627–630 (2000)
17. Liyuan, L., Leung, M.K.H.: Integrating intensity and texture differences for robust change detection. IEEE Trans. Image Process. **11**, 105–112 (2002)
18. Javed, O., Shafique, K., Shah, M.: A hierarchical approach to robust background subtraction using color and gradient information. In: IEEE Workshop on Motion and Video Computing, pp. 22–27 (2002)
19. Yi, W., Jodoin, P.-M., Porikli, F., Konrad, J., Benezeth, Y., Ishwar, P.: CDnet 2014: an expanded change detection benchmark dataset. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp. 393–400 (2014)