An Edge-Based Approach to Motion Detection*

Angel D. Sappa and Fadi Dornaika

Computer Vison Center
Edifici O Campus UAB
08193 Barcelona, Spain
{sappa, dornaika}@cvc.uab.es

Abstract. This paper presents a simple technique for motion detection in steady-camera video sequences. It consists of three stages. Firstly, a coarse moving edge representation is computed by a set of arithmetic operations between a given frame and two equidistant ones (initially the nearest ones). Secondly, non-desired edges are removed by means of a filtering technique. The previous two stages are enough for detecting edges corresponding to objects moving in the image plane with a dynamics higher than the camera's capture rate. However, in order to extract moving edges with a lower dynamics, a scheme that repeats the previous two stages at different time scales is performed. This temporal scheme is applied over couples of equidistant frames and stops when no new information about moving edges is obtained or a maximum number of iterations is reached. Although the proposed approach has been tested on human body motion detection it can be used for detecting moving objects in general. Experimental results with scenes containing movements at different speeds are presented.

1 Introduction

A number of techniques for motion detection have been proposed during last years (e.g., [1], [2], [3]). An extensive survey of the current state of the art in image change detection is given in [4]. The most common approaches compute a background image and then threshold the difference between each frame and this estimated background. This difference will automatically unveil moving objects (foreground) present in the scene. Background modeling and subtraction approaches have been extensively used and mainly rely on the use of color or luminance information (e.g. [5], [6]). [7] utilizes color and edge information in order to improve the quality and reliability of the results. It requires several frames to compute an initial estimation of the background image. Since the background is exposed to permanent changes, it has to be updated periodically. Typical approaches update background model by means of Gaussian mixtures [8].

In contrast to iterative updating algorithms, [9] proposes a background estimation algorithm that utilizes a global optimization to identify the periods of time in which

* This work was supported by the Government of Spain under the CICYT project TIN2005-09026 and The Ramón y Cajal Program.

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part I, LNCS 3991, pp. 563 – 570, 2006.
© Springer-Verlag Berlin Heidelberg 2006

background content is visible in a small block of the image. Since foreground regions are excluded, no bias towards the foreground color will occur in the reconstructed background. The main drawback of background-modeling techniques appears when moving objects always overlap the same area.

On the contrary to previous approaches, the difference between consecutive images was also used to detect motion. For instance, [10] and [11] propose techniques based on the difference between consecutive frames. In [10], moving objects are detected by a combination of three edge maps: *a)* a background edge map, *b)* an edge map computed from the difference of two consecutive frames, and *c)* an edge map from the current frame. In both approaches an interframe scheme that only considers two consecutive frames is proposed; therefore, objects moving with a low dynamics are only detected by both an edge labeling process and a parameters' tuning process.

Our work is closely related to the work presented in [10]. However, it is more advantageous than [10] since: *(1)* efficiency is higher due to the fact that there is no need to compute a background model, *(2)* moving objects are directly extracted by means of their moving edges without tuning any user defined parameter and *(3)* complex scenes, containing objects with different dynamics, can be processed. The proposed technique is based on the use of arithmetic operations between the current frame and other two equidistant ones (backward and forward along the video sequence). It allows handling scenes containing bodies moving at different speeds.

The proposed technique consists of three stages. Firstly, a coarse representation of moving edges is computed. Secondly, that representation is filtered given rise to an image only containing those objects moving with a speed higher than the camera's capture rate. Finally, these two stages are applied iteratively in order to extract all the moving objects present in the current frame. The paper is organized as follows. Section 2 introduces the moving edge detection stage. Section 3 presents the filtering stage, proposed to remove non-moving edges generated in noisy regions. The iterative process is presented in section 4. Experimental results are presented in section 5 and conclusions are finally given in section 6.

2 Moving Edge Detection

Given a video sequence defined by f frames, the algorithm starts by computing their corresponding edges by means of the Canny edge detector [12]. These segmented frames, E_i, contain all the edges of the input frames, F_i. At this first stage the objective is to extract a coarse description of those edges defining moving objects.

In order to detect those edges, a set of arithmetic operations is applied over three consecutive frames $\{n\text{-}m, n, n+m\}$. The philosophy of this first stage is to detect moving edges based on the fact that they will be placed at different positions when consecutive frames are considered. Firstly, the signed differences between edges extracted from a central frame and edges corresponding to two nearest neighbors are computed ($DE_l = \lfloor E_n - E_{n-1} \rfloor$ and $DE_r = \lfloor E_n - E_{n+1} \rfloor$). From these differences, only positive pixels are considered; pixels with a negative value are set to zero. Each one of these new images (DE_l , DE_r) essentially contains moving edges together with some background edges occluded by the non-overlapped difference (DE_l , DE_r). The latter will be called δ edges, see Fig. 3(*bottom*). The amount of δ edges depends on the speed of the moving objects in the image plane. In addition to the previous edges, the new images also contain edges generated by noisy data or by small differences in the edge representation computed by the Canny edge detector (edges are quite sensitive to light variations). All these non-moving edges will be removed during the next stage by a filtering algorithm, while δ edges are easily removed by merging the computed images (DE_l , DE_r) through an AND logical operation:

$$ME = DE_l \cap DE_r \tag{1}$$

δ edge removal stage is one of the differences with respect to [10], where occluded edges are removed by using an edge map generated by combining background edges and the edges of the current frame.

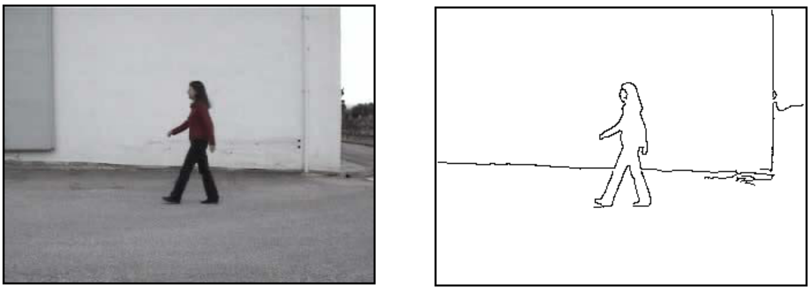


Fig. 1. (*left*) Original frame. (*right*) Edge representation computed by the Canny edge detector.

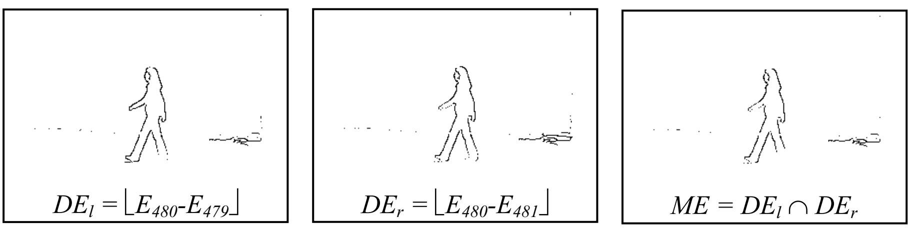


Fig. 2. (*left*) Edges computed by subtracting to the central frame the previous one. (*center*) Result after subtracting the next one. (*right*) Final edge representation *ME*, computed from DE_l and DE_r.

As mentioned above, an image *ME* still contains edges belonging to non-moving objects generated by noisy data. They are removed next by a filtering stage. Fig. 2(*right*) shows an illustrations of the resulting *ME* image, corresponding to Fig. 1, computed after merging Fig. 2(*left*) with Fig. 2(*center*). Notice that at this particular sequence, motion is performed with a low dynamics—a walking displacement; hence, there are not unveiled δ edges in Fig. 2(*left*) neither in Fig. 2(*center*). Notice that DE_l, DE_r and therefore *ME,* contain some edges corresponding to noisy data from Fig. 1(*right*), which will be removed next.

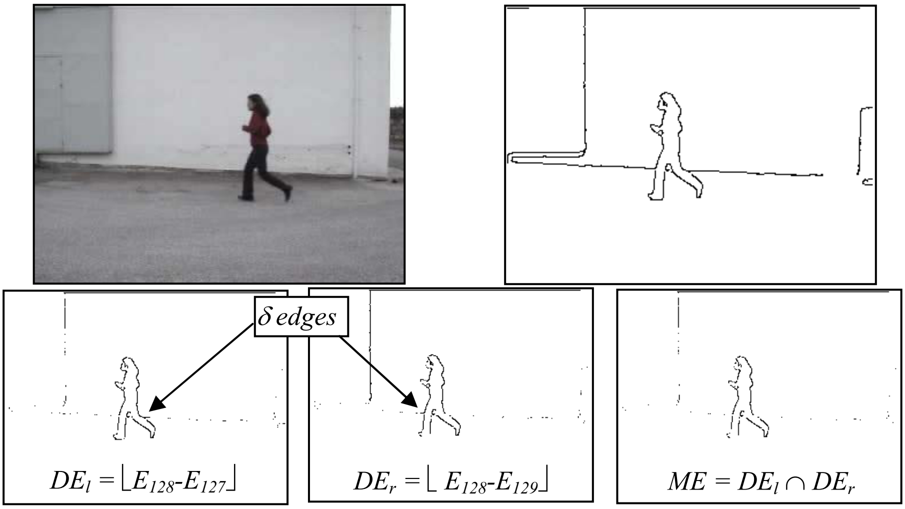


Fig. 3. (*top-left*) Original frame. (*top-right*) Edge representation computed by the Canny edge detector. (*bottom-left*) Edges computed by subtracting to the central frame the previous one. (*bottom-center*) Result after subtracting the next one. (*bottom-right*) Final edge representation ME, computed from DE_l and DE_r.

Fig. 3 shows the result obtained with a scene containing a movement having higher dynamics. Differently to the previous case, Fig. 3(*bottom-left*) and Fig. 3(*bottom-center*) show some δ edges. The final edge representation is shown in Fig. 3(*bottom-right*), again there are some edges corresponding to noisy data.

3 Non-moving Edge Removal

The outcome of the previous stage is an image containing edges belonging to objects moving with a speed, in the image plane, higher than the camera capture rate. In addition, that image contains edges belonging to non-moving objects, which are originated due to the fact that the random noise created in one frame is different from the one created in other frames. These differences generate slight changes in the edge position (or new edges), which make that even stationary background edges are not removed when the differences between the current frame and its neighbors is computed (DE_l , DE_r) (see Fig. 2(*right*) and Fig. 3(*bottom-right*)). The objective at this stage is to remove all these non-moving edges.

As shown in [10] and [11], an easy and robust way to extract a noiseless edge representation of moving edges is to apply the Canny operator over the difference of two original frames $\zeta(| F_n - F_{n-1} |)$, instead of performing the difference of the computed edges. It is because Gaussian convolution, included in the Canny operator, suppresses the noise in the luminance difference by smoothing it:

$$\zeta(| F_n - F_{n-1} |) = \theta (\nabla G * | F_n - F_{n-1} |) \tag{2}$$

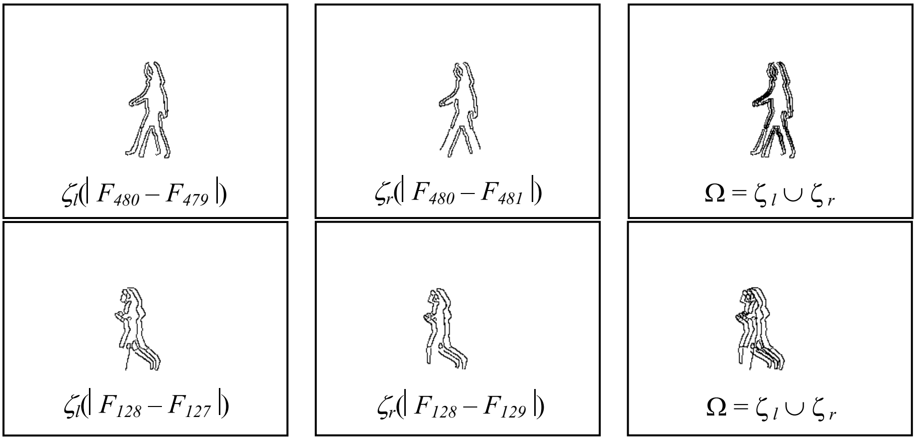


Fig. 4. (*top*) Filter mask for merged edges (*ME*) shown in Fig. 2(*right*). (*bottom*) Filter mask for merged edges (*ME*) shown in Fig. 3(*bottom-right*).

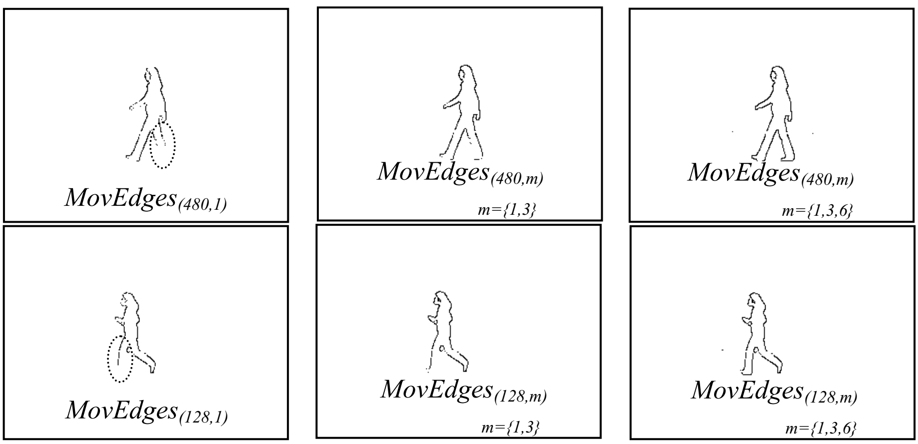


Fig. 5. (*top*) Moving edges extracted from frame 480, Fig. 1(*top-right)*, after two iterations. (*bottom*) Moving edges extracted from frame 128, Fig. 3(*top-right*), after two iterations.

where the edges of the original input frames difference, $\zeta(|F_n - F_{n-1}|)$, are computed by the Canny edge detector by performing a gradient operation ∇ on the Gaussian convoluted image $G*F$, followed by applying the nonmaximum suppression to the gradient magnitude to thin the edges and the thresholding operation with hysteresis to detect and link them (θ). This strategy has already been used in [10] to extract the edges of moving objects by merging this representation with other two edge map representations (edges from the background, automatically or manually computed, and edges from the current frame). In the current implementation we propose to take advantage of this noiseless edge representation and to use it as a filtering mask.

Similarly, two representations are computed: $\zeta_l(|F_n - F_{n-1}|)$ and $\zeta_r(|F_n - F_{n+1}|)$. These representations are merged together, by means of an OR logical operation, giving rise to a single image that is the sought filtering mask:

$$\Omega = \zeta_l \cup \zeta_r \tag{3}$$

Fig. 4 shows filter masks for the two examples previously presented. Finally, this mask is applied over the edge representation computed in (*1*), through an AND logical operation. The resulting representation only contains those moving edges present at the frame n, when its two nearest frames ($n\pm 1$) are considered:

$$MovEdges_{(n,1)} = \Omega \cap ME \tag{4}$$

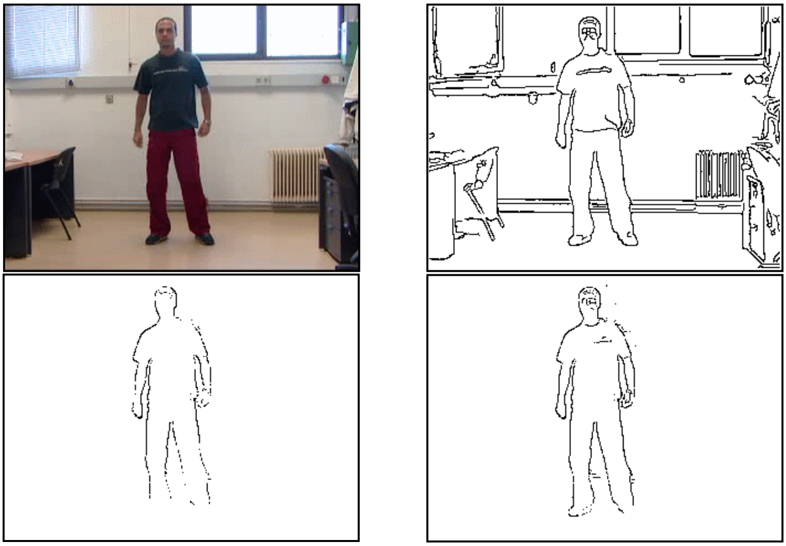
The previous scheme only detects objects moving with a speed higher than the camera's capture rates (only two nearest frames were used). It cannot work properly with all the possible situations—low dynamics or temporarily still moving objects. In order to handle these situations the following scheme is proposed.

4 Detecting Moving Objects

The previous stages can easily be extended by considering not only the two nearest frames but a combination of two frames equidistant to the one under study. In this way, an iterative process has been proposed to detect all the spectra of moving edges present in the scene.

Let E_n be the edges extracted from frame n by using the Canny operator. The technique presented in previous sections, is now used by taking into account a couple of frames placed at m backward and forward positions from n (m>1). Again, moving edges computed by (*1*) are filtered by means of (*3*), also computed from the frame n together with both $n\pm m$ frames. The variable m is incremented after every iteration and the computed moving edges, $MovEdges_{(n,m)}$, are merged with previous results—OR operation. This iterative process is applied until no new information about moving edges is extracted or a maximum number of iterations is reached. In this case the algorithm stops and moving objects are defined by the extracted moving edges.

In order to speed up the process, in the current implementation the variable m has been increased by a step of three frames after each iteration (m += *3*). An attractive point of the proposed scheme, when human motion is considered, is that this iterative approach allows detecting all body parts independently of their particular dynamics. Human body displacement (e.g. walking, running) is a good example of a movement involving different dynamics. Its particularity, over other rigid moving objects, is that in spite of the fact that the center of gravity could have associated a constant velocity, each body part has a different non-constant velocity; this velocity, for example during a walking period, is temporarily null for the foot that is in contact with the floor. Hence, detection of human body displacement is an attractive topic, where, up to our knowledge none of those algorithms based on the use of only two consecutive frame differences is able to efficiently detect without further considerations.



$MovEdges_{(128,1)}$ $MovEdges_{(128,m)}$ $m=\{1,3\}$

Fig. 6. (*top-left*) Original frame. (*top-right)* Edge representation computed by the Canny edge detector. (*bottom-left*) Moving edges extracted after one iteration. (*bottom-right*) Moving edges extracted after two iterations.

5 Experimental Results

The proposed technique has been tested with several video sequences depicting body motion having different dynamics. In the paper two different illustrations have been used (one with low dynamics and the other with high dynamics). Fig. 5 shows final results of both illustrations. Fig. 5(*top-left*) has been obtained after filtering Fig. 2(*right*) with the mask presented in Fig. 4(*top-right*). While Fig. 5(*top-center*) and Fig. 5(*top-right*) present moving edges obtained after two and three iterations respectively—edges corresponding to the highlighted region in Fig. 5(*top-left*) have been recovered when frames further than one position were considered. Fig. 5(*bottom-left*) has been obtained after filtering Fig. 3(*bottom-right*) with the mask presented in Fig. 4(*bottom-right*). Similarly, the highlighted region corresponds to the body part with lowest dynamics. Fig. 5(*bottom-center*) and Fig. 5(*bottom-right*) present moving edges obtained after two and three iterations respectively.

Fig. 6 presents results obtained with an indoor video sequence. Fig. 6(*top-left*) shows an original frame, while its corresponding edges, computed by Canny edge detector, are presented in Fig. 6(*top-right*). Moving edges obtained after one and two iterations are presented in Fig. 6(*bottom*).

Finally, although at the current implementation segmenting the bodies' region is not addressed, they can be easily handled by detecting regions bounded by the first and last edge points along rows and columns [10]. Extracted points will define the moving body regions.

6 Conclusions

This paper described a simple technique for recovering moving objects by extracting their defining edges—moving edges. Further works will consider labeling those non-moving edges as background edges. In this way, if it necessary, a background representation could be incrementally generated; moreover after computing a full background image, where some measure of confidence is reached, the algorithm could switch from moving edge detection to a background subtraction approach, probably reducing CPU time.

Improvements of the proposed technique with respect to [10] are mainly in two aspects. First, all the spectra of moving objects is recovered; and second there is no need to generate a background edge map neither to tune particular parameters. The main advantage over those background modeling techniques is that the proposed approach can be applied whenever it is required, without having to process a large part of the video.

References

1. Kim, J., Kim, H.: Efficient Region-Based Motion Segmentation for a Video Monitoring System. Pattern Recognition Letters **24** (2003) 113-128
2. Sheikh, Y., Shah, M.: Bayesian Object Detection in Dynamic Scenes. IEEE Int. Conf. on Computer Vision and Pattern Recognition, San Diego, CA, USA, June 2005
3. Kellner, M., Hanning, T.: Motion Detection Based on Contour Strings. IEEE Int. Conf. on Image Processing, Singapore, October 2004
4. Radke, R., Andra, S., Al-Kofahi, O., Roysam, B.: Image Change Detection Algorithms: A Systematic Survey. IEEE Trans. On Image Processing, **14** (2005) 294-307
5. Bichsel, M.: Segmenting Simply Connected Moving Objects in a Static Scene. IEEE Trans. on Pattern Analysis and Machine Intelligence **16** (1994) 1138-1142
6. Francois, A., Medioni, G.: Adaptive Color Background Modeling for Real-Time Segmentation of Video Streams. Int. Conf. on Imaging Science, Systems and Technology, Las Vegas, NA, June 1999
7. Jabri, S. , Duric, Z., Wechsler, H., Rosenfeld, A.: Detection and Location of People in Video Images Using Adaptive Fusion of Color and Edge Information. 15th. Int. Conf. on Pattern Recognition, Barcelona, Spain, Sep. 2000
8. Lee, D.: Effective Gaussian mixture learning for video background subtraction. IEEE Trans. on Pattern Analysis and Machine Intelligence **27** (2005) 827-832
9. Farin, D., de With, P., Effelsberg, W.: Robust Background Estimation for Complex Video Sequences. IEEE Int. Conf. on Image Processing, Barcelona, Spain, Sep. 2003.
10. Kim, C., Hwang, J.: Fast and Automatic Video Object Segmentation and Tracking for Content-Based Applications. IEEE Trans. on Circuits and Systems for Video Technology **12** (2002) 122-129
11. Marqués, F., Molina, C.: Object Tracking for Content-Based Functionalities. SPIE Vis. Commun. Image Processing, vol. 3024, San Jose, CA, Feb. 1997
12. Canny, J.F.: A Computational Approach to Edge Detection. IEEE Trans. on Pattern Analysis and Machine Intelligence **6** (1986) 679-698