ROBUST BACKGROUND ESTIMATION FOR COMPLEX VIDEO SEQUENCES

Dirk Farin[1], *Peter H. N. de With*[2], *and Wolfgang Effelsberg*[1]

[1] Dept. of Computer Science IV
University of Mannheim
68131 Mannheim, Germany
farin@uni-mannheim.de

[2] LogicaCMG / Univ. of Technol. Eindhoven
5600 MB Eindhoven, Netherlands
P.H.N.de.With@tue.nl

ABSTRACT

Knowing the background image of a video scene simplifies the general video-object segmentation problem and therefore it is required by several automatic segmentation algorithms. This paper presents a new background estimation algorithm which is applicable to complex video sequences where many objects are simultaneously visible and the background is visible for a short time period only. The algorithm applies a rough segmentation of the input images into foreground and background regions to exclude the foreground objects from background synthesis. This prevents a bias of the synthesized background image towards the color of foreground objects. Experiments show that the obtained background images differ significantly less from the real background than those obtained with previous algorithms.

1. INTRODUCTION

Several automatic video-object segmentation algorithms (e.g., [1], [2]) assume that a background image of the scene is known. This assumption allows to use the difference between video input frame and background image as a first estimate for the foreground object masks. However, in many cases it is impractical to record a separate background image without objects, since one cannot control the scenario (e.g., in surveillance applications), or the background changes slowly over time and has to be adapted. Hence, an automatic background estimation algorithm is required which reconstructs the background image even though it has never been available as such.

Background estimation is subject to several problems for which a good survey can be found in [3]. These include sudden global illumination changes or a “background” that is changing its appearance (like a clock at the wall). In this paper, we will concentrate on the bootstrapping problem. This involves obtaining the background image of a scene with many, arbitrarily moving objects and where the background is visible for only short periods of time.

Especially if the reconstructed background should be used for automatic object segmentation, it is important that there is no luminance bias towards a temporary foreground object color. This phenomenon is a common problem with previous algorithms.

2. PREVIOUS WORK

Most of the existing algorithms for background estimation apply a frame-based update strategy. They maintain a current background estimate which is updated iteratively after each new input frame. The update factor can be a constant aging factor, or it can be regulated using a Kalman filter [4]. In [1] and [5], it is suggested to update adaptively, where the update factor is decreased if the frame-to-frame or frame-to-background difference is high. The intention is to slow down the update in areas where foreground is assumed. The disadvantage of this strategy is that errors in the background estimate are only removed slowly for exactly the same reason. Hence, this class of algorithms only works reliably when foreground objects do not move too slowly and background is visible during most of the time. Otherwise, the reconstruction remains unstable or converges to an average of foreground and background color.

A different approach is to use a pixel-based temporal median-filter over a large number of frames [6]. However, in the worst case, each pixel has to contain background content for at least half of the input frames. An advantage of the median-filter algorithm is that the effect of blurring is not as severe as with the weighted update algorithms. However, even with the median-filter algorithm, the background image content can deviate from the correct background color. To understand this, assume that a bright background is occluded for some period by a dark object. Since the median is computed including the dark foreground object pixels as well as the bright background pixels, the median will not be at the mean background luminance, but it will be shifted to slightly darker values, caused by shadows or image noise. The effect can be observed in an example result shown in Fig. 6b. Even though all foreground objects are dark and the background is bright, objects appear half-transparent in the reconstruction. This bias from the correct background color is not always perceivable, but it can cause difficulties in automatic segmentation algorithms.

3. BACKGROUND RECONSTRUCTION ALGORITHM

The principal idea of our algorithm is to apply a rough segmentation of the input images into foreground and background regions. The background image is synthesized using the median algorithm [6], but we exclude foreground regions from the synthesis process. Since foreground regions are excluded, no bias towards the foreground color will occur in the reconstructed background. The classification is carried out on small blocks to make the background/foreground decision more robust. Periods of background content are identified by searching for the subset of frames that show stable content in the block. The similarity of block contents over time is collected into a matrix, which contains the difference between the image content at the block position for each pair of frames. Low values correspond to stationary background regions, whereas high values are present for each pair of frames that contains differences in this block. This matrix is then decomposed to obtain the foreground/background classification. Depending on the subset of frames that contain background content, the matrix elements can be classified into stationary and non-stationary elements. Background periods are obtained by searching for the subset of frames so that the sum of stationary matrix elements is minimized, while the sum of non-stationary matrix elements is maximized. This is described in greater detail in the next two sections.

3.1. Block Similarity Matrix

Let the block size be $N \times N$ pixels and let the sequence be of length L. Furthermore, let $f_i(x, y)$ be the luminance of pixel (x, y) in input frame i. We assume that $f_i \in [0; 1]$. For each block (u, v) with top left pixel at position (uN, vN), we calculate a symmetric similarity matrix $M^{(u,v)}$ of size $L \times L$ with

$$M_{a;b}^{(u,v)} = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} \left| f_a(uN+i, vN+j) - f_b(uN+i, vN+j) \right|.$$

This states that each matrix element $M_{a;b}$ is set to the *sum of absolute differences* (SAD) measure between the blocks in frame a and frame b at the same position.

For time periods in which the content in the block does not change, the corresponding square block centered at the matrix diagonal will contain low values (Fig. 1). If a specific block content disappears for some time and reemerges later, another rectangle off the matrix diagonal will show low values. Periods of time with moving content show as high-valued matrix elements. If the content is only visible for a short time, the corresponding matrix rows and columns will contain mostly high values.

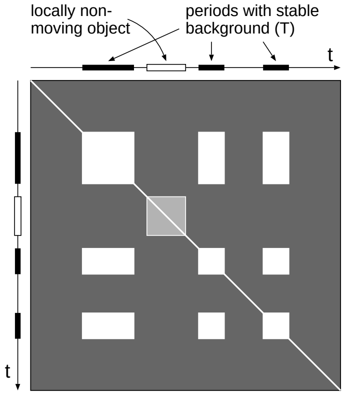


Fig. 1. Structure of a block similarity matrix. Low matrix elements are shown in white, high values as dark areas in the matrix.

3.2. Matrix Decomposition

To identify the periods in which only background is visible in a block, we separate the matrix into two parts: the stationary elements (small difference values), and the non-stationary elements (large difference values). Let $T^{(u,v)} \subseteq \{1, \dots, L\}$ be the set of frames in which block (u, v) only contains background content[1]. A matrix element $M_{a;b}$ is considered stationary iff $a, b \in T$.

Since stationary elements should be small values and non-stationary elements should mostly be large values, we can separate them by choosing T such that the stationary elements are as small as possible and the non-stationary elements are as large as possible. More specifically, we optimize the cost function

$$\min_T C = \min_T \sum_{a,b \in T} M_{a;b} + \sum_{a \notin T \vee b \notin T} (1 - M_{a;b}).$$

Optimization is carried out using an iterative process. Starting with a good estimate of T (see Section 3.4 for explaining how this is obtained), we calculate the cost difference that results from adding or removing each of the input frames to T. If adding or removing the frame decreases the cost, T is modified accordingly. Optimization is stopped when none of these changes can further decrease the cost. We have found that this process converges fast in only about two or three iterations over the input frames. Note that instead of the naive way of computing the cost by summing over the complete matrix, it is sufficient to compute the cost difference, which can be obtained by summing only over a single matrix row. In the case of adding a frame k to T, the cost difference is

$$\Delta C_{+k} = 2\Bigg(\underbrace{\sum_{a \in T} M_{a,k} + \sum_{a \notin T} (1 - M_{a,k})}_{\text{new costs}} - \underbrace{\sum_{a \in \{1,\dots,L\}} (1 - M_{a;k})}_{\text{old costs}} \Bigg)$$
$$= 2\Bigg(\sum_{a \in T} (2M_{a;k} - 1) \Bigg),$$

and hence for the case of removing a frame k from T, it is the negative value $\Delta C_{-k} = -\Delta C_{+k}$.

Since the above matrix decomposition process converges to a local minimum close to the initialization, an initialization near the correct minimum must be chosen. Note that the global minimum need not necessarily correspond to the correct background periods. If the sequence contains many foreground objects of the same color, and if the objects are visible during most of the time, the global optimum can correspond to those periods in which foreground objects are visible. To solve this problem, we apply two additional steps, preceding the optimization step. First, we exclude periods from T for which we observe motion in the block. Since we assume that camera motion has been compensated beforehand, moving content cannot belong to the background. Second, we exploit the correlation of background periods between neighboring blocks. Both steps are described in the next two sections.

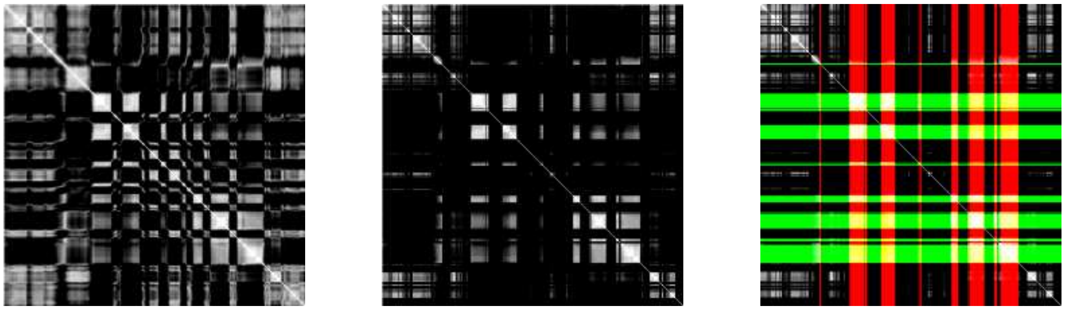
3.3. Integration of Motion Information

Motion estimation is carried out for each block using a block-matching algorithm. If the minimum SAD matching error is lower than 90% of the null-vector matching error, the block is considered as moving and the matrix row and column corresponding to the current input frame are artificially set to 1. This prevents the optimization algorithm from selecting the block in this frame as a background block. Figure 2 shows an example how this exclusion disambiguates an otherwise unclear situation.

3.4. Background Periods Prediction

If an object moves across a block, it will most probably also move across a neighboring block during a comparable time period. Hence, when calculating $T^{(u,v)}$, we use the previously calculated $T^{(u-1,v)}$ and $T^{(u,v-1)}$ to initialize the optimization process. If an input frame a is contained in $T^{(u-1,v)}$ and $T^{(u,v-1)}$, it is also included in $T^{(u,v)}$. If it is only contained in one of both, it is included randomly. At the left and top border, predictions are formed directly from the solution of the block above or to the left, respectively. The very first block (top-left) is initialized with all input frames active in $T^{(0,0)}$. This is a sensible assumption, since image activity is usually centered in the image such that the border contains mainly background content.

[1] We will omit the superscript (u, v) to simplify notation when the meaning is clear. Since we are only considering single blocks in this section, no ambiguities will occur.



(a) Sample similarity matrix.

(b) Same matrix after invalidating periods with motion.

(c) Decomposition result.

Fig. 2. A sample block similarity matrix taken from the sequence shown in Fig. 6.

The spatial prediction scheme has two advantageous properties. First, it provides an accurate initialization of the optimization, leading to fast convergence. Second, the prediction helps to select the correct local minimum, even when the object is visible for more time than the background. Since the prediction provides the initialization, even a strong minimum has not enough support in the beginning that the optimization could be attracted to it. This is illustrated in Figure 3.

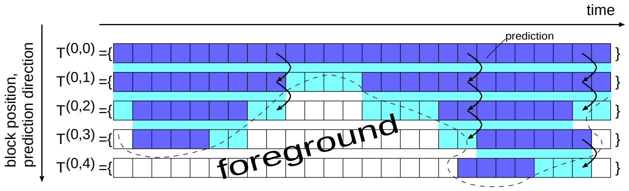


Fig. 3. Spatial background-period prediction (first column of blocks in an image). The block at the top left $T^{(0,0)}$ contains background content throughout the sequence (background marked in a dark shade). The background periods of $T^{(0,i)}$ form the initialization for background periods of $T^{(0,i+1)}$ (prediction is drawn in a light shade). The matrix decomposition step then refines this prediction to get the final result for this block. Since optimization is started with the last block's result, the optimization will converge to the correct minimum even for blocks that are clearly dominated by foreground objects (e.g., $T^{(0,4)}$).

4. RESULTS

We have applied our algorithm to a variety of popular test sequences like the *hall-and-monitor* (see Fig. 4), *road1*, *road2*, and *urbicande* sequences, for which the background could be reconstructed without any visible errors. Even the background from *seq_17* of the Video Quality Expert Group (VQEG) test set was recovered without error (see Fig. 5). Hence, we applied our algorithm to a special sequence containing very difficult background-object behaviour (see Fig. 6). This sequence contains many people where some persons are walking around and some are standing still for a long time. Part of the background is even impossible to reconstruct, because the background is never visible during the whole sequence. Apart from these impossible regions, our algorithm reconstructs the background without any blurring.

To evaluate the quality of the background image with respect to applicability for automatic segmentation algorithms, we measured the difference between the reconstructed background image and the ground-truth background image. Since the real background image is only available for the *hall-and-monitor* sequence (in the first frames of the sequence), we used this sequence to obtain the results. We measured the PSNR of several reconstruction algorithms and estimated the camera noise by calculating the PSNR between the first two frames of the sequence. Since the median algorithm cannot remove the foreground objects completely, we calculated the PSNR a second time, now with the erroneous regions excluded. Even with these regions excluded, our algorithm achieves considerably higher PSNR than the median algorithm, which makes it a better choice for segmentation applications.

	PSNR
average	29.86 dB
median	30.89 dB
median (cropped)	32.02 dB
our algorithm	35.15 dB
camera noise	38.74 dB

Table 1. PSNR between reconstructed background and real background (*hall-and-monitor* sequence)

5. CONCLUSIONS

We presented a new algorithm for background estimation of complex video sequences. In contrast to iterative updating algorithms, our approach utilizes a global optimization to identify the periods of time in which background content is visible in a small block of the image. The background is reconstructed by applying a temporal median-filter over the identified periods of background content.

Experimental results show no visible reconstruction error for common test sequences. In cases that are nearly impossible to reconstruct, it provides considerably better results than the popular median algorithm. Since our algorithm does not average between foreground and background content, there is no luminance bias, thereby improving the usability of the reconstructed backgrounds for automatic segmentation algorithms.

6. REFERENCES

[1] Andrea Cavallaro and Touradj Ebrahimi, “Video object extraction based on adaptive background and statistical change detection,” in *Proc. of SPIE VCIP*, Jan. 2000, pp. 465–475.

[2] Richard J. Qian and M. Ibrahim Sezan, “Video background replacement without a blue screen,” in *Proc. of ICIP*, 1999, pp. 143–146.

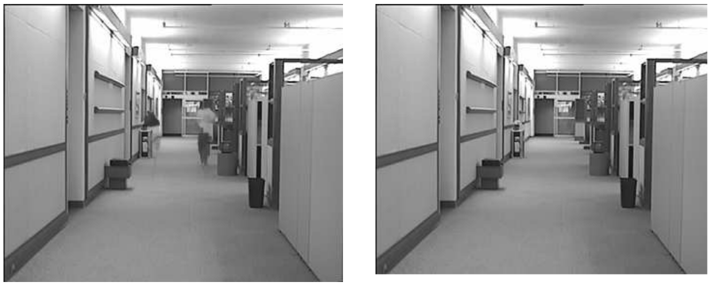
[3] Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers, “Wallflower: Principles and practice of background maintenance,” in *International Conference on Computer Vision*, 1999, p. 255.

[4] Christof Ridder, Olaf Munkelt, and Harald Kirchner, “Adaptive background estimation and foreground detection using Kalman-filtering,” in *Proc. of ICRAM*, 1995, pp. 193–199.

[5] Patrick Piscaglia, Andrea Cavallaro, Michel Bonnet, and Damien Douxchamps, “High level description of video surveillance sequences,” in *Proc. of ECMAST*, 1999, pp. 316–331.

[6] M. Massey and W. Bender, “Salient stills: Process and practice,” *IBM Systems Journal*, vol. 35, no. 3&4, pp. 557–573, 1996.

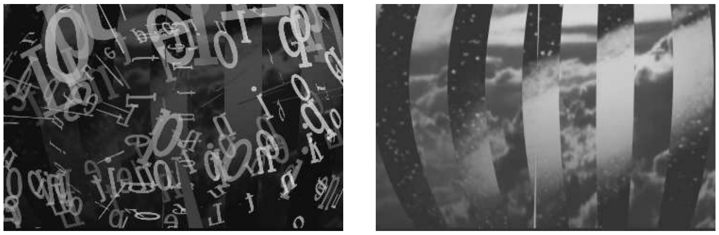
[7] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, “Background estimation and removal based on range and color,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 1999.



(a) Median algorithm.

(b) Our algorithm.

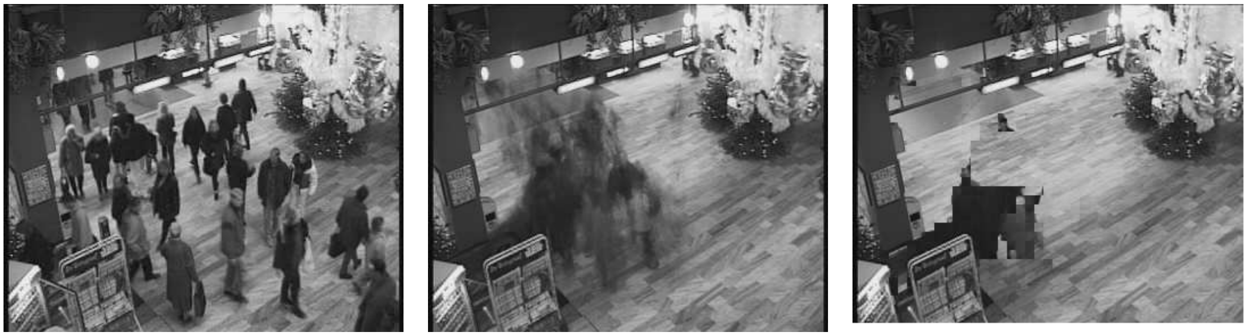
Fig. 4. Results for *hall-and-monitor* sequence. (a) The median operator cannot remove the two persons completely. (b) Our algorithm reconstructs the background without any visible errors.



(a) Typical input frame.

(b) Our algorithm.

Fig. 5. Results for VQEG test sequence 17: artificially created sequence where many letters are whirling around. No reconstruction errors are visible. Note that we have increased the background image brightness for clarity.



(a) Typical input frame.

(b) Median algorithm.

(c) Our algorithm.

Fig. 6. Results for a very complex scene with many walking and standing people. Note that the background cannot be reconstructed completely, since some background regions are never visible in this sequence.