# Reliable Background Suppression for Complex Scenes

Simone Calderara,
Rudy Melli
D.I.I. - Univ. of Modena and
Reggio Emilia
Via Vignolese, 905/b
Modena, Italy

Andrea Prati
D.I.S.M.I. - Univ. of Modena
and Reggio Emilia
Via Amendola, 2 - Pad.
Morselli
Reggio Emilia, Italy

Rita Cucchiara
D.I.I. - Univ. of Modena and
Reggio Emilia
Via Vignolese, 905/b
Modena, Italy

## ABSTRACT

This paper describes a system for motion detection based on background suppression, specifically conceived for working in complex scenes with vacillating background, camouflage, illumination changing, etc.. The system contains proper techniques for background bootstrapping, shadow removal, ghost suppression and selective updating of the background model. The results on the challenging videos provided in VSSN '06 Open Source Algorithm Competition dataset demonstrate that the proposed system outperforms the widely-used mixture-of-Gaussians approach.

## Categories and Subject Descriptors

I.4.8 [**Computing Methodologies**]: Image Processing and Computer Vision—*Scene Analysis*

## General Terms

Design,Algorithms,Security

## Keywords

background suppression, people detection and tracking, shadow detection

## 1. INTRODUCTION

In scene analysis from videos a key aspect to be evaluated is related to the moving parts of the scene. Motion analysis is, indeed, a key research topic for the computer vision and image processing communities. Reliable and effective motion analysis should pursue both high precision (with the two meanings of accuracy in shape detection and reactivity to changes in time) and flexibility in different scenarios (indoor, outdoor) or different light conditions.

In the absence of any a priori knowledge about target and environment, the most widely adopted approach for moving object detection is based on *background suppression* [3, 6, 8, 7, 4, 5, 9, 1]. An estimate of the background (often called a *background model*) is computed and evolved frame by frame: moving objects in the scene are detected by the difference between the current frame and the current background model. It is well known that background suppression carries two problems for the precision of moving object detection. The first problem is that the model should reflect the real background as accurately as possible. The second problem is that the background model should immediately reflect sudden scene changes such as the start or stop of objects, so as to allow detection of only the actual moving objects with high reactivity.

Among the many approaches proposed for background suppression, none of them can really work on a 24/7 basis, in almost every condition. One of the most cited and used work is the modeling of the background pdf by means of a mixture of Gaussians [2, 7]. However, as we will demonstrate in this paper, this approach is not enough reactive to slow illumination changes.

In this paper we present a robust system based on background suppression, that improves our previously-proposed system called SAKBOT (Statistical And Knowledge-Based Object Tracker) [1] by including new techniques for improving the reliability in complex scenes.

## 2. RELIABLE BACKGROUND SUPPRESSION

This section describes the main steps of the proposed algorithm for background suppression.

Let us denote with $BG_t$ the background model available at time $t$ and with $I_t$ the current input frame. We will also refer to $BG_t(i,j)$ to denote the vector $(R,G,B)$ of pixel $(i,j)$ in the background model. The same applies for $I_t(i,j)$.

### 2.1 Background bootstrapping

A crucial task in every background suppression approach is the background initialization or *bootstrapping*, that needs to be both fast and accurate. Unfortunately, it is frequently impossible to have a clear background for many frames in order to compute statistics for building the model. Therefore, it is important to implement a method that can initialize the background model as quickly as possible even starting from "dirty" frames.

Our approach basically partitions the image into blocks (of 16×16 pixels) and selectively updates the background model with a block whenever a sufficiently high number of pixels within the block are not in motion. Motion is evaluated with a thresholded single difference between two consecutive frames. If more than 95% of the pixels in the block

*VSSN'06,* October 27, 2006, Santa Barbara, California, USA.
Copyright 2006 ACM 1-59593-496-0/06/0010 ...$5.00.

are detected as not in motion, the model is updated in that area with the pixels not in motion. If this happens for more than 10 times (also non consecutive), the whole block is considered "stable" and no more evaluated. Once all the blocks have been set to "stable" the background model is ready. To avoid deadlocks and to speed up the bootstrapping, if no new blocks change state to "stable" for two consecutive frames, the threshold for the single difference is increased.

## 2.2 Background Updating and Foreground Extraction

After the bootstrapping stage, the background model is updated using a temporal median. A fixed $k$-sized circular buffer is used to collect values of each pixel over time. We sample, at a framerate lower or equal than the input framerate, the values of the pixel $I_t(i,j)$ and store the last $k$ values in the circular buffer. In addition to the $k$ values, the current background model $BG_t(i,j)$ is sampled and added to the buffer to account for the last reliable background information available. These $k+1$ values are then ordered according to their grey-level intensity, and the median value is used as an estimate for the current background model.

Once the background model has been created and updated, the foreground is extracted frame by frame using the background differencing technique. The difference between the current image $I_t$ and the background model $BG_t$ is computed:

$$M_t(i,j) = \frac{(I_t(i,j) - BG_t(i,j)) \cdot \mathbf{i^T}}{3} \tag{1}$$

where $\mathbf{i^T}$ is the 1×3 identity vector. $M_t(i,j)$ is the foreground mask containing the grey-level information of the difference. The mask is then binarized using two different thresholds: a low threshold $T_{low}$ to filter out the noisy pixel extracted due to small intensity variations; a high threshold $T_{high}$ to identify the pixels where a large intensity variation occurs. Both these thresholds are local, i.e. they have different values for each pixel of the image. Let $b_p(i,j)$ be the value at position $p$ inside the ordered circular buffer $b$ of pixel $(i,j)$ and, consequently, $b_{\frac{k+1}{2}+1}$ the median. The thresholds are computed as follows:

$$T_{low}(i,j) = \lambda \left( b_{\frac{k+1}{2}+l} - b_{\frac{k+1}{2}-l} \right) \tag{2}$$

$$T_{high}(i,j) = \lambda \left( b_{\frac{k+1}{2}+h} - b_{\frac{k+1}{2}-h} \right) \tag{3}$$

where $\lambda$ is a fixed multiplier, while $l$ and $h$ are fixed scalar values. We experimentally set $\lambda = 7$, $l = 2$ and $h = 4$, for a buffer of 9 values. It is straightforward to see that, being the vector $b$ ordered, $T_{high}$ is always higher or equal than $T_{low}$.

Our experiments demonstrated that these settings well perform in most of the common surveillance scenarios. The motivation for the adoption of the dynamic per-pixel thresholds is trivially explained by the fact that using fixed, per-frame thresholds makes the system less reactive to local illumination changes.

The final binarized motion mask $B_t$ is obtained as composition of the two binarized motion masks computed respectively using the low and the high thresholds: a pixel is marked as foreground in $B_t$ if it is presented in the low-thresholded binarized mask AND it is spatially connected to at least one pixel present in the high-thresholded binarized mask. Finally, the list $MVO_t$ of moving objects at time $t$ is extracted from $B_t$ using a two-pass labeling algorithm. Each $j^{th}$ element $MVO_t^j$ of the list is a candidate foreground object. A further refinement of the list is performed discarding all the small objects.

Foreground points resulting from the background subtraction could be used for the selective background update, i.e. avoiding to include them in the background updating process; nevertheless, in this case, all the errors made during background subtraction will consequently affect the selective background update. A particularly critical situation occurs whenever moving objects are stopped for a long time and become part of the background. When these objects start again, a so-called "ghost" [3, 1] is detected in the area where they were stopped. This will persist for all the following frames, preventing the area to be updated in the background image forever, causing *deadlock* [1]. Our approach substantially overcomes this problem since it performs selectivity not by reasoning on single moving points, but on detected and recognized moving objects. This object-level reasoning proved much more reliable and less sensitive to noise than point-based selectivity [1].

## 2.3 Shadow Removal

Moving and static shadows create big troubles to background suppression approach since they are likely to be confused for moving or static objects due to the difference in brightness with respect to the underlying background.

Without going too much into details, our system uses the shadow detection algorithm described in [1]. In practice, shadows are detected by assuming that they darken the underlying background, but do not significantly change its color. Thus, a model of shadow detection in the hue, saturation and value (HSV) color space is proposed:

$$SP^t(i,j) = \begin{cases} 1 & \text{if } \alpha \leq \frac{I_t(i,j).V}{BG_t(i,j).V} \leq \beta \\ & \wedge |I_t(i,j).S - BG_t(i,j).S| \leq \tau_S \wedge \\ & D(i,j) \leq \tau_H \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where the dotted notations with H, S, and V indicate the hue, saturation, and value components, respectively. The lower bound $\alpha$ is used to define a maximum value for the darkening effect of shadows on the background, and is approximately proportional to the light source intensity. Instead the upper bound $\beta$ prevents the system from identifying as shadows those points where the background was darkened too little with respect to the expected effect of shadows. $\tau_H$ and $\tau_S$, instead, account for the bounded change in color due to the shadow.

The distance $D(i,j)$ is computed as:

$$D(i,j) = \min\left(|I_t(i,j).H - BG_t(i,j).H|, \right. \\ \left. 360 - |I_t(i,j).H - BG_t(i,j).H|\right) \tag{5}$$

## 2.4 Object Validation

After the shadow removal, an object-level validation step is performed in order to remove all the moving objects generated by small motion of the background, for example by waving trees. This validation is performed accounting for joint contribution coming from color information and gradient of the objects.

The gradient is computed with respect to both spatial and temporal coordinates:

$$\frac{\partial I_t(i,j)}{\partial(x,t)} = I_{t-\Delta t}(i-1,j) - I_t(i+1,j)$$
$$\frac{\partial I_t(i,j)}{\partial(y,t)} = I_{t-\Delta t}(i,j-1) - I_t(i,j+1) \quad (6)$$

For stationary points, we can approximate the past sample $I_{t-\Delta t}$ with the background model $BG_t$:

$$\frac{\partial I_t(i,j)}{\partial(x,t)} = BG_t(i-1,j) - I_t(i+1,j)$$
$$\frac{\partial I_t(i,j)}{\partial(y,t)} = BG_t(i,j-1) - I_t(i,j+1) \quad (7)$$

The gradient module is obtained by using the following equation:

$$G_t = \left\{ g(i,j) \mid g(i,j) = \sqrt{\left\| \frac{\partial I_t(i,j)}{\partial(x,t)} \right\|^2 + \left\| \frac{\partial I_t(i,j)}{\partial(y,t)} \right\|^2} \right\} \quad (8)$$

From our tests, it is evident that this gradient module is quite robust to small motions in the background, mainly thanks to the use of temporal partial derivative. Moreover, the joint spatio-temporal derivative makes the object gradient computation more accurate, since it also detects gradient in the inner parts of the object.

Given the list of moving objects $MVO_t$, the gradient $G_t$ is compared, for each pixel $(i,j)$ of each moving object $MVO_t^h$, with the gradient (in the spatial domain) of the background $GBG_t$ in order to evaluate the coherence with it. This *gradient coherence* $GC_t$ is evaluated over a $k \times k$ neighborhood:

$$GC_t(i,j) = \min_{\substack{i-k \le x \le i+k \\ j-k \le y \le j+k}} |G_t(i,j) - GBG_t(x,y)| \quad (9)$$

where $|.|$ represents the absolute value of ., since $G_t$ and $GBG_t$ are grey-level images.

Unfortunately, when the gradient module (either $G_t$ or $GBG_t$) is close to zero data are not reliable. To overcome to this problem, we combine the gradient coherence with the *color coherence* $CC_t$:

$$CC_t(i,j) = \min_{\substack{i-k \le x \le i+k \\ j-k \le y \le j+k}} \|I_t(i,j) - BG_t(x,y)\| \quad (10)$$

where $\|.\|$ represents the norm in the RGB color space.

The overall validation score $VS_t^h$ for the considered $MVO_t^h$ is the normalized sum of the per-pixel validation score, obtained by multiplying the two coherence measures reported above:

$$VS_t^h = \frac{\sum\limits_{(i,j)\in MVO_t^h} GC_t(i,j) * CC_t(i,j)}{\sum\limits_{(i,j)\in MVO_t^h} 1} \quad (11)$$

where the denominator is the area (as number of pixels) of $MVO_t^h$. This value is then thresholded and, if below the threshold, $MVO_t^h$ is discarded and its pixels are marked as belonging to background.

### 2.5 Fast Ghost Suppression

As above mentioned, one of the problem of selective background updating is the possible creation of ghosts. Therefore, it is necessary to implement a method to detect ghosts and force them into the background model. The approach used is similar to that used for background bootstrapping (see section 2.1), but at region level instead of pixel level.

All the validated objects are used to build an image called $A_t$ that accounts for the number of times that a pixel is detected as stopped by the single difference:

$$A_t(i,j) = \begin{cases} A_{t-1}(i,j) + 1 & \text{if } SD(i,j) < T_{SD} \\ A_{t-1}(i,j)/2 & \text{otherwise} \end{cases} \quad (12)$$

where $SD(i,j)$ is the single difference between two consecutive frames and $T_{SD}$ a suitable threshold. A valid object $MVO_t^h$ is classified as ghost if:

$$\frac{\sum\limits_{(i,j)\in MVO_t^h} A_t(i,j)}{N_t^h} > T_{ghost} \quad (13)$$

where $N_t^h$ is the area of $MVO_t^h$ and $T_{ghost}$ is the threshold on the percentage of points of the $MVO_t^h$ stopped for sufficient time.

Practically speaking, in the case of pixels belonging to a ghost, the single difference will be lower than the threshold and will start increasing the value in $A_t$. When the sum of the accumulator values for the MVO exceeds the threshold $T_{ghost}$ the MVO is forced into the background.

## 3. EXPERIMENTAL RESULTS ON VSSN '06 DATASET

We made an initial evaluation of the proposed background suppression approach on the VSSN '06 dataset provided at the web address `http://mmc36.informatik.uni-augsburg.de/VSSN06_OSAC/`, whose composition is reported in Table 1. We have not worked on Video 1 since we were unable to use the provided ground truth. Fig. 1 shows the results of the proposed method compared with those achieved with a mixture of Gaussians (MoG), in terms of false misses and false alarms at pixel level. It is clear that our method achieves lower overall falses, even though it is outperformed by MoG in terms of false misses.

| Video | Resolution | # frames | Description |
|---|---|---|---|
| Video 1 | 384×240 | 302 | indoor |
| Video 2 | 384×240 | 750 | indoor |
| Video 3 | 384×240 | 900 | outdoor,<br>vacillating bkg |
| Video 4 | 384×240 | 820 | outdoor,<br>vacillating bkg |
| Video 5 | 384×240 | 750 | indoor |
| Video 6 | 384×240 | 750 | indoor, no boots. |
| Video 7 | 384×240 | 750 | outdoor,<br>vacillating bkg |
| Video 8 | 384×240 | 1192 | indoor, bootstr.,<br>local ill. change |

**Table 1: VSSN '06 dataset**

The dataset contains also other videos, but no ground truth is provided. However, we tested also our system also on them and example snapshots of the visual results (wrt MoG) are reported in Fig. 2.

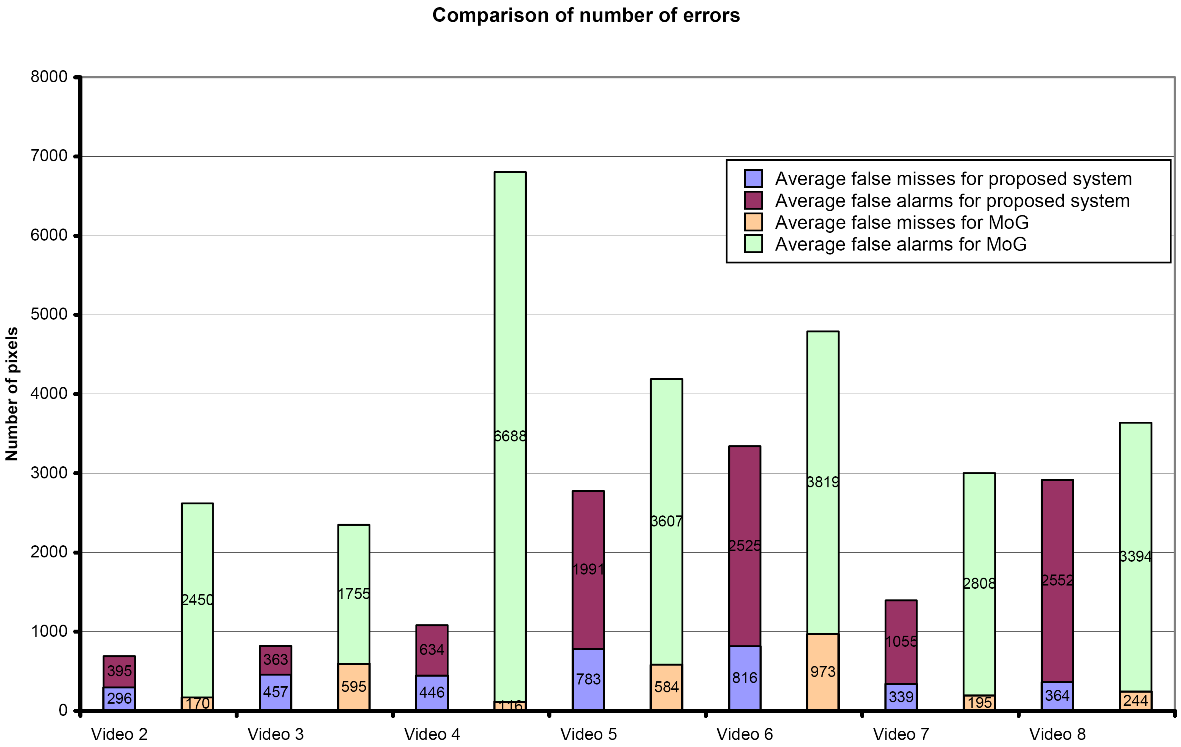
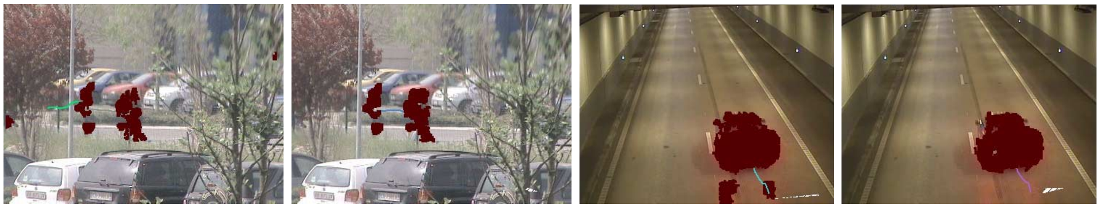


**Figure 1: Comparative results with respect to mixture of Gaussians (MoG) over the VSSN '06 dataset.**



(a) Result on Video 9 with MoG

(b) Result on Video 9 with our system

(c) Result on Tunnel video with MoG

(d) Result on Tunnel video with our system

**Figure 2: Visual results of segmentation on Video 9 and Tunnel videos.**

## 4. REFERENCES

[1] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts and shadows in video streams. *IEEE Trans. on PAMI*, 25(10):1337–1342, Oct. 2003.

[2] R. Duda, P. Hart, and D. Stork. *Pattern Classification (2nd ed.)*. Wiley Interscience, 2004.

[3] I. Haritaoglu, D. Harwood, and L. Davis. W4: real-time surveillance of people and their activities. *IEEE Trans. on PAMI*, 22(8):809–830, Aug. 2000.

[4] S. McKenna, S. Jabri, Z. Duric, A. Rosenfeld, and H. Wechsler. Tracking groups of people. *Computer Vision and Image Understanding*, 80(1):42–56, Oct. 2000.

[5] N. Ohta. A statistical approach to background suppression for surveillance systems. In *Proc. of IEEE Intl Conference on Computer Vision*, pages 481–486, 2001.

[6] A. Shio and J. Sklansky. Segmentation of people in motion. In *Proceedings of IEEE Workshop on Visual Motion*, pages 325–332, 1991.

[7] C. Stauffer and W. Grimson. Learning patterns of activity using real-time tracking. *IEEE Trans. on PAMI*, 22(8):747–757, Aug. 2000.

[8] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: real-time tracking of the human body. *IEEE Trans. on PAMI*, 19(7):780–785, July 1997.

[9] Q. Zhou and J. Aggarwal. Tracking and classifying moving objects from videos. In *Proceedings of IEEE Workshop on Performance Evaluation of Tracking and Surveillance*, 2001.