The SOBS algorithm: what are the limits?

Lucia Maddalena
National Research Council, Institute for High-Performance Computing and Networking
Via P. Castellino 111, 80131 Naples, Italy
lucia.maddalena@cnr.it

Alfredo Petrosino
University of Naples Parthenope, Department of Applied Science
Centro Direzionale, Isola C/4, 80143 Naples, Italy
alfredo.petrosino@uniparthenope.it

Abstract

The Self-Organizing Background Subtraction (SOBS) algorithm implements an approach to moving object detection based on the neural background model automatically generated by a self-organizing method, without prior knowledge about the involved patterns. Such adaptive model can handle scenes containing moving backgrounds, gradual illumination variations and camouflage, can include into the background model shadows cast by moving objects, and achieves robust detection for different types of videos taken with stationary cameras. Moreover, the introduction of spatial coherence into the background update procedure leads to the so-called SC-SOBS algorithm, that provides further robustness against false detections. The paper includes extensive experimental results achieved by the SOBS and the SC-SOBS algorithms on the dataset made available for the Change Detection Challenge at the IEEE CVPR2012.

1. Introduction

Many computer vision applications, such as video surveillance or video compression, rely on the task of detecting moving objects in video streams, that provides the segmentation of the scene into background and foreground components.

Compared to other approaches for detecting moving objects, such as optical flow (e.g. [1]), background subtraction is computationally affordable for real-time applications. It consists in maintaining an up-to-date model of the background and detecting moving objects as those that deviate from such a model; thus, the main problem is its sensitivity to dynamic scene changes, and the consequent need for the background model adaptation via background maintenance. Such problem is known to be significant and difficult [12], and several well known issues in background maintenance have to be taken into account, such as *light changes*, *moving background*, *cast shadows*, *bootstrapping*, and *camouflage*. Due to its pervasiveness in various contexts, background subtraction has been afforded by several researchers (see surveys in [2, 5, 9, 10]).

In [7] the Self-Organizing Background Subtraction (SOBS) algorithm has been proposed, which implements an approach to change detection based on the neural background model automatically generated by a self-organizing method without prior knowledge about the involved patterns. Such adaptive model can handle scenes containing moving backgrounds, gradual illumination variations, and camouflage, can include into the background model shadows cast by moving objects, and achieves robust detection for different types of videos taken with stationary cameras. Moreover, the introduction of spatial coherence into the background update procedure, as proposed in [8], leads to the so-called SC-SOBS (Spatially Coherent Self-Organizing Background Subtraction) algorithm, that provides further robustness against false detections.

The paper reports an analysis of extensive experimental results achieved by the SOBS and SC-SOBS algorithms on the sequence dataset made available for the Change Detection Challenge at IEEE CVPR2012 [3].

In Section 2 we describe in a unified framework the neural background model adopted from [7] for change detection and its enhanced version adopted from [8], while in Section 3 we provide a thorough analysis of experimental results on the change detection dataset. Conclusions are drawn in Section 4.

2. Self-Organizing Background Subtraction

The background model constructed and maintained in SOBS algorithm [7] is based on the idea of building the

978-1-4673-1612-5/12/$31.00 ©2012 IEEE

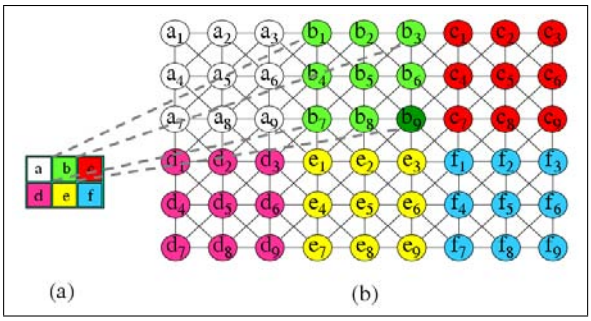

Figure 1. (a) A simple image I_t;(b) the modeling neuronal map B_t.

image sequence neural background model by learning in a self-organizing manner image sequence variations, seen as trajectories of pixels in time. The network behaves as a competitive neural network that implements a winner-take-all function, with an associated mechanism that modifies the local synaptic plasticity of neurons, allowing learning to be spatially restricted to the local neighborhood of the most active neurons. Therefore, the neural background model well adapts to scene changes and can capture the most persisting features of the image sequence.

2.1. Neural Model Representation

Given an image sequence $\{I_t\}$, for each pixel $\mathbf{p}$ in the image domain D, we build a neuronal map consisting of $n \times n$ weight vectors $m_t^{i,j}(\mathbf{p}), i, j$=0, …, n-1, which will be called a *model* for pixel $\mathbf{p}$ and will be indicated as $M_t(\mathbf{p})$:

$$M_t(\mathbf{p}) = \left\{ m_t^{i,j}(\mathbf{p}),\ i,j = 0,\ldots,n-1 \right\}. \qquad (1)$$

If every sequence frame has N rows and P columns, the complete set of models $M_t(\mathbf{p})$ for all pixels $\mathbf{p}$ of the t-th sequence frame I_t is organized as a 2D neuronal map B_t with $n \times N$ rows and $n \times P$ columns, where the weight vectors $m_t^{i,j}(\mathbf{p})$ for the generic pixel $\mathbf{p} = (x, y)$ are at neuronal map position $(n \times x + i, n \times y + j), i, j = 0, \ldots, n-1$:

$$B_t(n{\times}x{+}i, n{\times}y{+}j) = m_t^{i,j}(\mathbf{p}),\ i,j = 0,\ldots,n{-}1. \quad (2)$$

An example of such neuronal map structure for a simple image I_t with N=2 rows and M=3 columns obtained choosing n=3 is given in Fig. 1. The model $M_t(\mathbf{b}) = \{b_1, \ldots, b_9\}$ of the upper center pixel b of sequence frame I_t in Fig. 1-(a) is stored into the 3 × 3 elements of the upper center part of neuronal map B_t in Fig. 1-(b), and analogous relations exist for each pixel of I_t and corresponding weight vectors storage. This configuration of the whole neuronal map B_t allows us to easily take into account the spatial relationship among pixels and corresponding weight vectors, as we shall see in the following subsections.

Although the notation introduced in Eqs. (1) and (2) appears redundant, in the sequel we will adopt both notations:

- the model $M_t(\mathbf{p})$ for pixel $\mathbf{p}$ will be adopted in order to indicate the whole set of weight vectors for each single pixel, helping to focalize on the pixelwise representation of the background model;
- the neuronal map B_t will be adopted in order to refer to the whole background model for an image sequence, to highlight spatial relationships among weight vectors of adjacent pixels. Indeed, besides its role as a background model, the neuronal map B_t can be thought of as a 2D array that stores, at each time t, all the weight vectors of the background model needed for background subtraction.

2.2. Neural Model Initialization

In the case of our background modeling application, we have at our disposal a fairly good means of initializing the weight vectors of the network, because the first image of the sequence I_0 is indeed a good initial approximation of the background. Therefore, for each pixel $\mathbf{p}$, the corresponding weight vectors of the model $M_0(\mathbf{p})$ are initialized with the pixel brightness value at time $t = 0$:

$$m_0^{i,j}(\mathbf{p}) = I_0(\mathbf{p}), \quad i,j = 0,\ldots,n-1. \qquad (3)$$

Therefore, the resulting neuronal map B_0, obtained for all pixels $\mathbf{p}$ as in Eq. (2) with $t = 0$, using the values specified in Eq. (3), can be seen as an $n \times n$ enlarged version of the first sequence frame I_0.

2.3. Background Subtraction and Model Update

At each subsequent time step t, background subtraction is achieved by comparing each pixel $\mathbf{p}$ of the t-th sequence frame I_t with the current pixel model $M_{t-1}(\mathbf{p})$, in order to determine if there exists a best matching weight vector $BM(\mathbf{p})$ that is close enough to it. If no acceptable matching weight vector exists, $\mathbf{p}$ is detected as belonging to a moving object (foreground). Otherwise, if such weight vector is found, it means that $\mathbf{p}$ is a background pixel.

In case of background pixels, further learning of the neuronal map enables the adaptation of the background model to slight scene modifications. Such learning is achieved by updating the neural weights according to a visual attention mechanism of reinforcement, where the best matching weight vectors, together with their neighborhood, are reinforced into the neuronal map.

2.3.1 Finding the Best Match

Given the current pixel $\mathbf{p}$ at time t, the value $I_t(\mathbf{p})$ is compared to the current pixel model, given by $M_{t-1}(\mathbf{p})$, to determine the weight vector $BM(\mathbf{p})$ that best matches it:

$$d(BM(\mathbf{p}), I_t(\mathbf{p})) = \min_{i,j=0,\ldots,n-1} d(m_{t-1}^{i,j}(\mathbf{p}), I_t(\mathbf{p})), \quad (4)$$

where the metric $d(\cdot,\cdot)$ is suitably chosen according to the specific color space being considered. Example metrics could be the Euclidean distance in RGB color space, or the Euclidean distance of vectors in the HSV color hexcone, as suggested in [6]. For the experiments reported in Section 3 the latter has been adopted, that gives the distance between two HSV pixel values $I_t(\mathbf{p}) = (h_p, s_p, v_p)$ and $I_t(\mathbf{q}) = (h_q, s_q, v_q)$ as:

$$d(I_t(\mathbf{p}), I_t(\mathbf{q})) = \|(v_p s_p \cos(h_p), v_p s_p \sin(h_p), v_p) - (v_q s_q \cos(h_q), v_q s_q \sin(h_q), v_q)\|_2^2. \quad (5)$$

2.3.2 Updating the Model

In order to adapt the background model to scene modifications, including gradual light changes, the model for current pixel $\mathbf{p}$, given by $M_{t-1}(\mathbf{p})$, should be updated. To this end, the weight vectors of B_{t-1} in a neighborhood of $BM(\mathbf{p})$ are updated according to weighted running average. In details, if $BM(\mathbf{p})$ is found at position $\overline{\mathbf{p}}$ in B_{t-1}, then weight vectors of B_{t-1} are updated according to

$$B_t(\mathbf{q}) = (1 - \alpha_t B_{t-1}(\mathbf{q}) + \alpha_t I_t(\mathbf{p}) \quad \forall \mathbf{q} \in N_{\overline{\mathbf{p}}}, \quad (6)$$

where $N_{\overline{\mathbf{p}}} = \{\mathbf{q} : |\overline{\mathbf{p}} - \mathbf{q}| \leq k\}$ is a 2D spatial neighborhood of $\overline{\mathbf{p}}$ having width $(2k+1) \in \mathbb{N}$ (in the reported experiments k=1). Moreover,

$$\alpha_t = \gamma \cdot G(\mathbf{q} - \overline{\mathbf{p}}) \cdot (1 - D_t(\mathbf{p})), \quad (7)$$

where γ represents the learning rate, $G(\cdot) = \mathcal{N}(\cdot; \mathbf{0}, \sigma^2 I)$ is a 2D Gaussian low-pass filter with zero mean and $\sigma^2 I$ variance, and $D_t(\mathbf{p})$ indicates the background subtraction mask value for pixel $\mathbf{p}$.

In the SOBS algorithm [7], the background subtraction mask value $D_t(\mathbf{p})$ is computed as

$$D_t^{SOBS}(\mathbf{p}) = \begin{cases} 1 & \text{if } d(BM(\mathbf{p}), I_t(\mathbf{p})) > \varepsilon \\ 0 & \text{otherwise} \end{cases}, \quad (8)$$

where ε is a threshold enabling the distinction between foreground pixels (having value 1) and background pixels (having value 0). This implies that, if the best matching weight vector $BM(\mathbf{p})$ in the current background model is *close enough* to the pixel value $I_t(\mathbf{p})$, then $\mathbf{p}$ is detected as a background pixel; otherwise it is detected as a foreground pixel.

In [8], the notion of spatial coherence is introduced in the updating formula, and the background subtraction mask value $D_t(\mathbf{p})$ is computed as

$$D_t^{SC-SOBS}(\mathbf{p}) = \begin{cases} 1 & \text{if } NCF_t(\mathbf{p}) \leq 0.5 \\ 0 & \text{otherwise} \end{cases}, \quad (9)$$

where the *Neighborhood Coherence Factor* $NCF_t(\mathbf{p})$ is defined as [4]:

$$NCF_t(\mathbf{p}) = \frac{|\Omega_{\mathbf{p}}|}{|N_{\mathbf{p}}|}. \quad (10)$$

Here $|\cdot|$ refers to the set cardinality, $N_{\mathbf{p}} = \{\mathbf{q} : |\mathbf{p} - \mathbf{q}| \leq h\}$ is a 2D spatial neighborhood of $\mathbf{p}$ having width $(2h+1) \in \mathbb{N}$ (in the reported experiments h=2), and $\Omega_{\mathbf{p}}$ indicates the set of pixels $\mathbf{q}$ belonging to $N_{\mathbf{p}}$ that have in their background model a best match that is close enough to their value $I_t(\mathbf{q})$:

$$\Omega_{\mathbf{p}} = \{\mathbf{q} \in N_{\mathbf{p}} : d(BM(\mathbf{q}), I_t(\mathbf{q})) \leq \varepsilon\}, \quad (11)$$

where ε has the same role as in Eq. (8). The Neighborhood Coherence Factor gives a relative measure of the number of pixels belonging to the spatial neighborhood $N_{\mathbf{p}}$ of a given pixel $\mathbf{p}$ whose value is well represented by the background model. If $NCF_t(\mathbf{p}) > 0.5$, most of the pixels in such spatial neighborhood have values well represented by the background model, and this should imply that also the value of pixel $\mathbf{p}$ is well represented by the background model, that is, it can be considered a background pixel. It has been shown that the introduction of spatial coherence in the updating formula allows us to enhance robustness of the background subtraction algorithm against false detections [8].

Whichever is the choice of the background subtraction mask value $D_t(\mathbf{p})$, the α_t values in Eq. (7) are weights that allow us to smoothly take into account the spatial relationship between current pixel $\mathbf{p}$ (through its best matching weight vector found at position $\overline{\mathbf{p}}$) and its neighboring pixels in I_t (through weight vectors at position $\mathbf{q} \in N_{\overline{\mathbf{p}}}$), thus preserving topological properties of the input in the neural network update (close inputs correspond to close outputs). Moreover, both the thresholding functions $D_t(\mathbf{p})$ defined in Eqs. (8) and (9) ensure selectivity in the update of the background model, allowing it to adapt to scene modifications, but without introducing into it the contribution of pixels that do not belong to the background scene.

The above described spatially coherent background subtraction and update procedure for each pixel can be sketched as the SC-SOBS algorithm reported in Fig. 2. The original

SC-SOBS Algorithm

Input: value $I_t(\mathbf{p})$ of pixel $\mathbf{p}$ in frame I_t, $t = 1, \ldots, T$;
background model $M_{t-1}(\mathbf{p})$ for pixel $\mathbf{p}$ at time t-1.
Output: detection mask value $D_t(\mathbf{p})$ for pixel $\mathbf{p}$ at time t;
background model B_t updated at time t.

1. *Initialize background model* $M_0(\mathbf{p})$ as in Eq. (3)
2. **for** $t = 1, T$
3. *Find in* $M_{t-1}(\mathbf{p})$ *the best match to* $I_t(\mathbf{p})$ *as in Eq.* (4)
4. *Compute the mask value* $D_t(\mathbf{p})$ *as in Eq.* (9)
5. *Update the background model* B_t *as in Eq.* (6)
6. **endfor**

Figure 2. Spatially Coherent Self-Organizing Background Subtraction Algorithm.

SOBS algorithm is obtained by adopting Eq. (8) instead of Eq. (9) for the computation of the background subtraction mask value $D_t(\mathbf{p})$ at line 4.

In the usual case that a set of K initial sequence frames is available for training, the above described initialization and update procedures on the first $K-1$ sequence frames are adopted for training the neural network background model, to be used for detection and adaptation in all subsequent sequence frames. Therefore, what differentiates the training and the adaptation phases is the choice of parameters in Eqs. (8), (11) and (6).

The threshold ε in Eqs. (8) and (11) is chosen as

$$\varepsilon = \begin{cases} \varepsilon_1 & \text{if} \quad 0 < t < K \\ \varepsilon_2 & \text{if} \quad t \geq K, \end{cases} \tag{12}$$

with ε_1 and ε_2 small constants. The mathematical ground behind the choice of ε_1 and ε_2 is as follows. To obtain a (possibly rough) initial background model that includes several observed pixel intensity variations, the value for ε (ε_1 in Eq. (12)) within the initial training sequence frames should be high. On the other side, to obtain a more accurate background model in the detection and adaptation phase, the value for ε (ε_2 in Eq. (12)) should be lower within subsequent frames. Therefore, it should be $\varepsilon_2 \leq \varepsilon_1$.

The learning rate γ in Eq. (7) is chosen as:

$$\gamma = \begin{cases} \gamma_1 - t\dfrac{\gamma_1 - \gamma_2}{K} & \text{if} \quad 0 < t < K \\ \gamma_2 & \text{if} \quad t \geq K, \end{cases} \tag{13}$$

where γ_1 and γ_2 are predefined constants such that $\gamma_2 \leq \gamma_1$. Indeed, in order to ensure neural network convergence during the training phase, the learning factor γ (γ_1 in Eq. (13)) is chosen as a monotonically decreasing function of time t, while, during the subsequent adaptation phase, the learning factor γ (γ_2 in Eq. (13)) is chosen as a constant value, that depends on the scene variability. Large γ values enable the network to faster learn changes corresponding to the background, but also leading to false negatives, that is, inclusion into the background model of pixels belonging to foreground moving objects. On the contrary, lower learning rates make the network slower to adapt to rapid background changes, but making the model more tolerant to errors due to false negatives through self-organization. Indeed, weight vectors of false negative pixels are readily smoothed out by the learning process itself.

In order to have in (6) values for α_t that belong to $[0,1]$, γ_1 and γ_2 are chosen as

$$\gamma_1 = \frac{c_1}{\max_{\mathbf{q} \in N_{\overline{\mathbf{p}}}} G(\mathbf{q} - \overline{\mathbf{p}})}, \quad \gamma_2 = \frac{c_2}{\max_{\mathbf{q} \in N_{\overline{\mathbf{p}}}} G(\mathbf{q} - \overline{\mathbf{p}})}, \tag{14}$$

with c_1 and c_2 constants such that $0 \leq c_2 \leq c_1 \leq 1$.

3. Analysis of experimental results

Experimental results for change detection have been produced for all video categories of the Change Detection Challenge dataset using the SOBS and SC-SOBS softwares made available in the download section of http://cvprlab.uniparthenope.it. The parameter values for all the videos have been chosen as follows: the number n^2 of weight vectors used to model each pixel has been fixed to 9; the distance threshold ε in Eqs. (8) and (11) has been fixed to 1.0 during the training phase and to 0.008 during the testing phase; the learning rate γ in Eq. (7) has been fixed to 1 in the training phase and to 0.05 during the testing phase; the number K of sequence frames for training is chosen equal to the number of all initial frames that are not included into the temporal region of interest defined for the contest. Finally, shadow detection has been applied to all videos, choosing parameter values as in [7].

In Table 3 we report pixel-based accuracy results achieved by SOBS and SC-SOBS algrithms on all videos, as well as average accuracy results for each video category (boldface), in terms of F-measure. All other performance measures are available at [3].

Concerning the *baseline* category, pretty high accuracy values can be observed for all image sequences. In Figure 3 we show how SC-SOBS result (Figure 3-(b)) further improves that of SOBS algorithm (Figure 3-(b)), in terms of spatial coherence of the moving object mask.

Also bootstrapping is well handled by the two algorithms, even for those sequences (e.g., *highway* and *PETS2006*) where a sufficient number of "empty" images is not available for training the background model. Indeed, although moving objects are learned during the background training phase, and thus are included into the initial background model, the subsequent update of the background model during the online phase allows us to reduce their contribution to the model itself, as long as they are not anymore present in the background. This can be observed in Figure 3, where the initial background model constructed by the SC-SOBS algorithm (Figure 3-(d)), including information of moving cars learned during the training phase, better adapts to the true background model during the online phase (Figure 3-(e)). Moreover, the latter further improves the background model updated by the SOBS algorithm (Figure 3-(f)); indeed, the exploitation of spatial coherence in the update procedure helps avoiding the inclusion of spurious moving object pixels into the background model.

Concerning the *cameraJitter* category, accuracy values are much lower than those for the previous category. Indeed, the jittering causes many pixel variations being absorbed into the initial background model (e.g., see Figure 4-(a)). However, these variations are not true appearance variations of each single pixel; rather, they are mainly variations induced by different (more or less adjacent) pixels. This leads to a high number of false negative detections. In order to achieve higher accuracy, it would be necessary to lower the segmentation threshold ε_2. As an example, in Figure 4 we compare the SC-SOBS moving object mask obtained by setting such threshold equal to 0.008 (Figure 4-

Category / Sequence	SOBS	SC-SOBS
Baseline	**0,925144**	**0,933343**
highway	0,932718	0,945495
office	0,966277	0,970268
pedestrians	0,949023	0,949173
PETS2006	0,852558	0,868436
CameraJitter	**0,708596**	**0,705093**
badminton	0,880300	0,881782
boulevard	0,688668	0,696733
sidewalk	0,535792	0,535632
traffic	0,729625	0,706226
DynamicBckg	**0,643882**	**0,668649**
boats	0,808266	0,895743
canoe	0,938500	0,952459
fall	0,292153	0,277527
fountain01	0,112060	0,115833
fountain02	0,858133	0,886564
overpass	0,854177	0,883771
Int.Obj.Motion	**0,562772**	**0,591790**
abandonedBox	0,570762	0,567283
parking	0,366776	0,403411
sofa	0,621813	0,640270
streetLight	0,947814	0,972937
tramstop	0,741986	0,841761
winterDriveway	0,127483	0,125078
Shadow	**0,771554**	**0,778445**
backdoor	0,825559	0,840589
bungalows	0,780856	0,788709
busStation	0,822829	0,836779
copyMachine	0,572060	0,571774
cubicle	0,720493	0,72334
peopleInShade	0,907527	0,909481
Thermal	**0,683426**	**0,692349**
corridor	0,849352	0,857299
diningRoom	0,717265	0,739639
lakeSide	0,320059	0,326417
library	0,933410	0,940044
park	0,597045	0,598344

Table 1. F-measure results achieved by SOBS and SC-SOBS algorithms on all videos of the Change Detection Challenge dataset [3], and average F-measure for each video category (boldface).

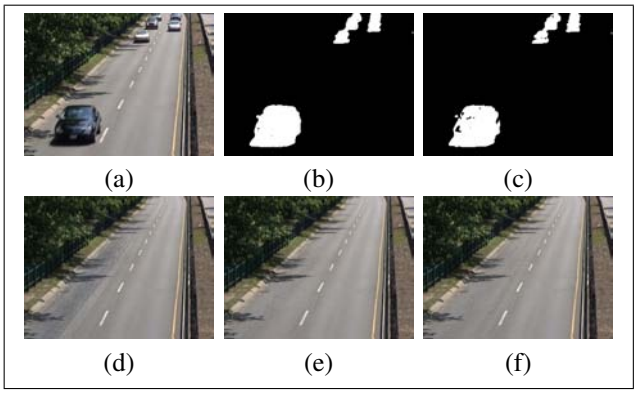

Figure 3. Sequence *highway*: (a) frame 690; (b) SC-SOBS moving object mask at frame 690; (c) SOBS moving object mask at frame 690; (d) SC-SOBS background model learned at frame 469; (e) SC-SOBS background model updated at frame 690; (f) SOBS background model updated at frame 690.

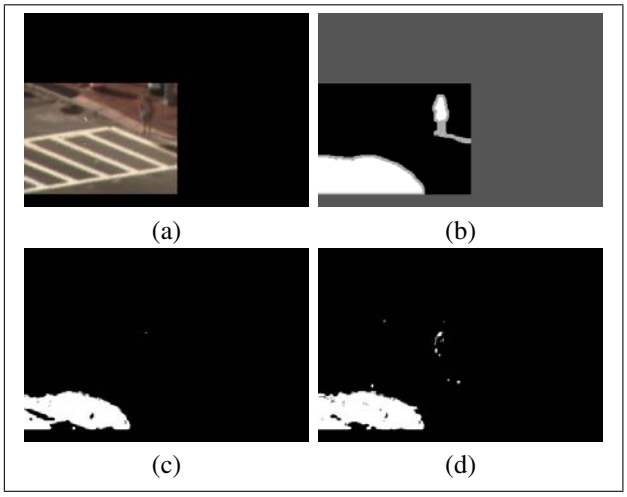

Figure 4. Frame 880 of sequence *sidewalk*: (a) SC-SOBS background model; (b) ground truth mask; (c) SC-SOBS moving object mask with ε_2=0.008; (d) SC-SOBS moving object mask with ε_2=0.002.

(c)) and to 0.003 (Figure 4-(d)) for one frame of sequence *sidewalk*, showing the consequent reduction of false negative pixels. Analogous results can be achieved by SOBS algorithm.

Dynamic backgrounds are well handled by the two algorithms in most of the sequences of such category. However, there are two sequences (namely, *fall* and *fountain01*) where accuracy results are low. In both sequences, background motion is due to "background moving objects" (leaves and water fountains, respectively) having considerable size, which cannot be handled at the pixel level (e.g., using a median filter). Slightly higher accuracy can be obtained by rising the number n of weight vectors adopted to model each pixel; indeed, a higher value of n allows us to save a wider number of pixel variations, thus reducing the number of false positive pixels (see Figure 5).

Widely varying accuracy values can be observed for the *IntermittentObjectMotion* sequences. Lower accuracy values are achieved for sequences (such as *parking* and *winterDriveway*) containing "removed objects", that is, stationary objects that were in the scene during the training phase and then move away. Indeed, when a background object leaves the scene, it uncovers a part of the background that has not

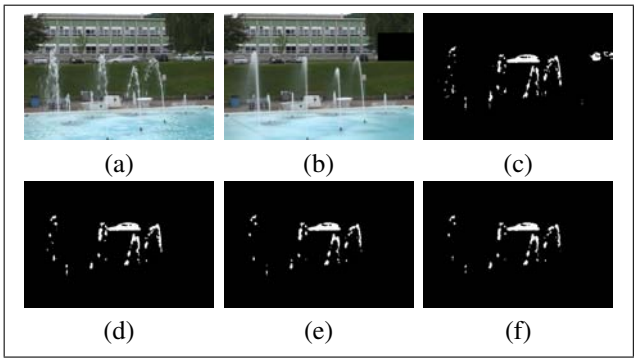

Figure 5. Frame 1130 of sequence *fountain01*: (a) input frame; (b) updated SC-SOBS background model; (c) SOBS moving object mask with n=3; SC-SOBS moving object masks with: (d) n=3, (e) n=5, and (f) n=7.

been previously learned. The selective update of our background model does not allow us to detect such situations, nor to include the "uncovered background" into the background model. Anyway, as usual, higher level analysis is needed to handle such cases and to classify foreground objects as either "moving" or "removed" objects, such as in [11]. On the other hand, the selective update does enable to properly handle "abandoned objects", that is, stationary objects that were not in the scene during the training phase. An example is given by the sequence *streetLight*, where the stationary car waiting for the red street light is perfectly detected as a foreground object during the whole sequence, thus allowing SOBS and SC-SOBS algorithms to reach very high accuracy values.

Shadows handling by SOBS and SC-SOBS algorithms can be considered quite good, even though there is room for improvement to better segment foreground into moving objects and shadows. As instance, an enhancement of our algorithms consists in including into the background model the contribution of shadow pixels (while in the present version they are not considered for the model update). We are experiencing this approach and expecting that it enables the maintenance of a more faithful background model, including not only shadows, but also strong lighting variations having an appearance similar to shadows.

Finally, for thermal sequences, average accuracy results achieved by SOBS and SC-SOBS algorithms can again be considered quite good. For such sequences two considerations are in order: 1) thermal sequences do not include shadows of foreground objects, and thus it makes no sense to apply shadow detection; 2) as they are greyscale sequences, it makes no sense to adopt sophisticated color spaces (and related time-consuming conversions), such as the HSV color space adopted for our experiments. Further experiments taking into account such considerations on the contest dataset could reserve pretty better results.

4. Conclusions

The reported analysis of experimental results achieved by the SOBS and SC-SOBS algorithms on the Change Detection Challenge dataset allowed us to show how they handle well-known issues in background maintenance, resulting robust to moving backgrounds, gradual illumination changes, and cast shadows, and how accuracy can still be improved by taking into account the scene characteristics. There is room for improvements in different directions and we expect to be able to solve some discrepancies from what could the best we may obtain.

References

[1] J. L. Barron, D. J. Fleet, and S. S. Beaucheminand. Performance of optical flow techniques. *Int. J. Comput. Vis.*, 12(1):42–77, 1994. 1

[2] S.-c. S. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. In S. Panchanathan and B. Vasudev, editors, *Proc. Visual Communications and Image Processing*, volume 5308, pages 881–892. SPIE, 2004. 1

[3] CVPR2012. IEEE Workshop on Change Detection, in conjunction with CVPR 2012, 2012. http://www.changedetection.net/. 1, 4, 5

[4] J. Ding, R. Ma, and S. Chen. A scale-based connected coherence tree algorithm for image segmentation. *IEEE Trans. Image Process.*, 17(2):204–216, 2008. 3

[5] S. Elhabian, K. El Sayed, and S. Ahmed. Moving object detection in spatial domain using background removal techniques: State-of-art. *Recent Patents on Computer Science*, 1(1):32–54, Jan. 2008. 1

[6] R. Fisher. Change detection in color images. http://homepages.inf.ed.ac.uk/rbf/PAPERS/iccv99.pdf. 3

[7] L. Maddalena and A. Petrosino. A self-organizing approach to background subtraction for visual surveillance applications. *IEEE Trans. Image Process.*, 17(7):1168–1177, July 2008. 1, 3, 4

[8] L. Maddalena and A. Petrosino. A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection. *Neural Comput. Appl.*, 19:179–186, March 2010. 1, 3

[9] M. Piccardi. Background subtraction techniques: a review. In *Proc. IEEE SMC*, volume 4, pages 3099–3104, 2004. 1

[10] R. J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: A systematic survey. *IEEE Trans. on Image Process.*, 14:294–307, 2005. 1

[11] Y. Tian, R. Feris, H. Liu, A. Hampapur, and M.-T. Sun. Robust detection of abandoned and removed objects in complex surveillance videos. *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, 41(5):565–576, Sept. 2011. 6

[12] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: principles and practice of background maintenance. In *Proc. ICCV*, volume 1, pages 255–261, 1999. 1