Review and evaluation of commonly-implemented background subtraction algorithms

Yannick Benezeth, Pierre-Marc Jodoin, Bruno Emile, Hélène Laurent, Christophe Rosenberger

Review and Evaluation of Commonly-Implemented Background Subtraction Algorithms *

Y. Benezeth[1] P.M. Jodoin[2] B. Emile[1] H. Laurent[1] C. Rosenberger[3]

[1]Institut PRISME
Université d'Orléans
88 boulevard Lahitolle
18020 Bourges Cedex, France

[2]MOIVRE
Université de Sherbrooke
2500 boulevard de l'Université
Sherbrooke, J1K 2R1, Canada

[3]GREYC, ENSICAEN
Université de Caen - CNRS
6 boulevard Maréchal Juin
14000 Caen, France

Abstract

Locating moving objects in a video sequence is the first step of many computer vision applications. Among the various motion-detection techniques, background subtraction methods are commonly implemented, especially for applications relying on a fixed camera. Since the basic inter-frame difference with global threshold is often a too simplistic method, more elaborate (and often probabilistic) methods have been proposed. These methods often aim at making the detection process more robust to noise, background motion and camera jitter. In this paper, we present commonly-implemented background subtraction algorithms and we evaluate them quantitatively. In order to gauge performances of each method, tests are performed on a wide range of real, synthetic and semi-synthetic video sequences representing different challenges.

1. Introduction

For various computer vision applications, background subtraction (BS) is a "quick and dirty" way of localizing moving objects. Based on the assumption that a moving object is made of colors which differ from those in the background, typical BS methods label "in motion" every pixel at time t whose color is significantly different from the ones in the background [10]. Unfortunately, a simple interframe difference with global threshold reveals itself as being sensitive to phenomena that violate the basic assumptions of BS, *i.e.* a rigorously fixed camera with a static noise-free background [4]. In real-life scenarios, the illumination can change (gradually or suddenly), the background may contain moving objects (waves on the water, trees shaken by the wind), the camera can jitter and so forth. In order to deal with those challenges, numerous background models and distance measures have been proposed. Those methods are (at least in theory) more robust to background instability than the basic ones. But are they really? And if they are, how much better are they? In this paper, we review commonly-implemented BS methods which we compare on various real, synthetic and semi-synthetic sequences. In section 2, five commonly-implemented motion detection methods are described. The video dataset used to compare those methods is described in section 3 while results and conclusion are presented in sections 4 and 5.

*This work was realized with the financial help of the Regional Council of Le Centre and the French Industry Ministry within the Capthom project of the Competitiveness Pole S^2E^2

2. Background Subtraction Algorithms

Although different, most BS techniques share a common framework: they make the hypothesis that the observed video sequence I is made of a fixed background B in front of which moving objects are observed. With the assumption that a moving object at time t has a color (or a color distribution) different from the one observed in B, the principle of BS methods can be summarized by the following formula:

$$\mathcal{X}_t(s) = \begin{cases} 1 & \text{if } d(I_{s,t}, B_s) > \tau \\ 0 & \text{otherwise,} \end{cases} \quad (1)$$

where $\mathcal{X}_t$ is the motion label field at time t (also called *motion mask*), d is a distance between $I_{s,t}$ the video frame at time t at pixel s and B_s the background at pixel s; τ is a threshold. The main difference between most BS methods is how B is modeled and which distance metric d is being used. In the following subsections, various BS techniques are presented.

Basic Motion Detection *(Basic)*

The easiest way to model the background is with a grayscale/color image B. This image can be a picture taken in absence of moving objects and/or estimated via a temporal median filter [10]. In order to keep the background up to date, it can be iteratively updated as follows:

$$\boldsymbol{B}_{s,t+1} = (1-\alpha)\boldsymbol{B}_{s,t} + \alpha.\boldsymbol{I}_{s,t} \quad (2)$$

where α is an updating constant whose value ranges between 0 and 1. Foreground pixels can be detected by thresholding various distance metrics such as:

$$d_0 = |I_{s,t} - B_{s,t}| \quad (3)$$

$$d_1 = |I^R_{s,t} - B^R_{s,t}| + |I^G_{s,t} - B^G_{s,t}| + |I^B_{s,t} - B^B_{s,t}| \quad (4)$$

$$d_2 = (I^R_{s,t} - B^R_{s,t})^2 + (I^G_{s,t} - B^G_{s,t})^2 + (I^B_{s,t} - B^B_{s,t})^2 \quad (5)$$

$$d_\infty = \max\{|I^R_{s,t} - B^R_{s,t}|, |I^G_{s,t} - B^G_{s,t}|, |I^B_{s,t} - B^B_{s,t}|\} \quad (6)$$

where exponents R, G and B stand for the red, $green$ and $blue$ channels. Note that distance d_0 operates on grayscale images.

One Gaussian *(1-G)*

Many authors model each background pixel with a probability density function (PDF) learned over a set of training frames. In this case, the BS problem often becomes a PDF-thresholding problem. For instance, to account for noise, some authors [9] model every background pixel with a Gaussian distribution $\eta(\boldsymbol{\mu}_{s,t}, \boldsymbol{\Sigma}_{s,t})$ where $\boldsymbol{\mu}_{s,t}$ and $\boldsymbol{\Sigma}_{s,t}$ stand for the average background color and covariance matrix over pixel s at time t. In this context, the distance metric can be the log likelihood:

$$d_G = \frac{1}{2}\log((2\pi)^3|\boldsymbol{\Sigma}_{s,t}|) + \frac{1}{2}(\boldsymbol{I}_{s,t} - \boldsymbol{\mu}_{s,t})\boldsymbol{\Sigma}^{-1}_{\boldsymbol{s,t}}(\boldsymbol{I}_{s,t} - \boldsymbol{\mu}_{s,t})^T \quad (7)$$

or the Mahalanobis distance:

$$d_M = |\boldsymbol{I}_{s,t} - \boldsymbol{\mu}_{s,t}|\boldsymbol{\Sigma}^{-1}_{s,t}|\boldsymbol{I}_{s,t} - \boldsymbol{\mu}_{s,t}|^T \quad (8)$$

where $\mathbf{I}_{s,t}$ and $\boldsymbol{\mu}_{s,t}$ are RGB vectors and $\boldsymbol{\Sigma}_{s,t}$ is a covariance matrix. To account for illumination variations, the mean and covariance of each pixel can be iteratively updated as follows:

$$\boldsymbol{\mu}_{s,t+1} = (1-\alpha).\boldsymbol{\mu}_{s,t} + \alpha.\boldsymbol{I}_{s,t} \quad (9)$$

$$\boldsymbol{\Sigma}_{s,t+1} = (1-\alpha).\boldsymbol{\Sigma}_{s,t} + \alpha.(\boldsymbol{I}_{s,t} - \boldsymbol{\mu}_{s,t})(\boldsymbol{I}_{s,t} - \boldsymbol{\mu}_{s,t})^T \quad (10)$$

Note that the covariance matrix can be a 3×3 matrix or can be assumed to be diagonal to reduce processing costs.

Gaussian Mixture Model *(GMM)*

To account for backgrounds containing animated textures (such as waves on the water or trees shaken by the wind), multimodal PDFs have been proposed. For instance, Stauffer and Grimson [8] model every pixel with a mixture of K Gaussians. Thus, the probability of occurrence of a color at a given pixel s is represented as:

$$P(I_{s,t}) = \sum_{i=1}^{K} \omega_{i,s,t}.\eta(\boldsymbol{I}_{s,t}, \boldsymbol{\mu}_{i,s,t}, \boldsymbol{\Sigma}_{i,s,t}) \quad (11)$$

where $\eta(\boldsymbol{I}_{s,t}, \boldsymbol{\mu}_{i,s,t}, \boldsymbol{\Sigma}_{i,s,t})$ is the i^{th} Gaussian model and $\omega_{i,s,t}$ its weight. Note that for computational reasons, as suggested by Stauffer and Grimson, we assume that the covariance matrix $\boldsymbol{\Sigma}_{i,s,t}$ is diagonal. In their method, parameters of a matched component (*i.e.* the Gaussian model for which $I_{s,t}$ is within 2.5 standard deviations of its mean) are updated as follows:

$$\omega_{i,s,t} = (1-\alpha)\omega_{i,s,t-1} + \alpha \quad (12)$$

$$\mu_{i,s,t} = (1-\rho).\mu_{i,s,t-1} + \rho.I_{i,s,t} \quad (13)$$

$$\sigma^2_{i,s,t} = (1-\rho).\sigma^2_{i,s,t-1} + \rho.(I_{i,s,t} - \mu_{i,s,t})^2 \quad (14)$$

where α is a user-defined learning rate and ρ is a second learning rate defined as $\rho = \alpha\eta(\boldsymbol{I}_{s,t}, \boldsymbol{\mu}_{i,s,t}, \boldsymbol{\Sigma}_{i,s,t})$. The μ and σ parameters of unmatched distributions remain the same while their weight is reduced as follows: $\omega_{i,s,t} = (1-\alpha)\omega_{i,s,t-1}$ to achieve decay. Whenever no component matches $\boldsymbol{I}_{s,t}$, the one with lowest weight is replaced by a Gaussian with mean $\boldsymbol{I}_{s,t}$, a large initial variance σ_0 and a small weight ω_0. Once every Gaussian has been updated, the K weights $\omega_{i,s,t}$ are normalized so they sum up to 1. Then, the K distributions are ordered based on a fitness value $\omega_{i,s,t}/\sigma_{i,s,t}$ and only the H most reliable are chosen as part of the background :

$$H = \underset{h}{\operatorname{argmin}}(\sum_{i=1}^{h} \omega_i > \tau) \quad (15)$$

where τ is a threshold. Then, those pixels which are at more than 2.5 standard deviations away from any of those H distributions are labeled "in motion".

Kernel Density Estimation *(KDE)*

An unstructured approach can also be used to model a multimodal PDF. In this perspective, Elgammal *et al.* [5] proposed a Parzen-window estimate of every background pixel:

$$P(I_{s,t}) = \frac{1}{N}\sum_{i=t-N}^{t-1} K(I_{s,t} - I_{s,i}) \quad (16)$$

where K is a kernel (typically a Gaussian one) and N is the number of previous frames used to estimate $P(.)$. When dealing with color video frames, products of one-dimensional kernels can be used:

$$P(\boldsymbol{I}_{s,t}) = \frac{1}{N}\sum_{i=t-N}^{t-1}\prod_{j=\{R,G,B\}} K\left(\frac{(I^j_{s,t} - I^j_{s,i})}{\sigma_j}\right) \quad (17)$$

A pixel is labeled as foreground if it is unlikely to come from this distribution, *i.e.* when $P(I_t)$ is smaller than a predefined threshold. Note that σ_j can be fixed or pre-estimated following Elgammal *et al.*'s method [5].

Minimum, Maximum and Maximum Inter-Frame Difference *(MinMax)*

The W^4 videosurveillance system [6] uses a background model made of a minimum m_s, a maximum M_s, and a maximum of consecutive frames difference D_s. For *MinMax*, a pixel s belongs to the background if:

$$|M_s - I_{s,t}| < \tau d_\mu \quad \text{or} \quad |m_s - I_{s,t}| < \tau d_\mu \quad (18)$$

where τ is an user-defined threshold and d_μ is the mean of the largest interframe difference over pixels. Note that *MinMax* operates on grayscale videos only.

3. Video Dataset

In order to gauge performances, the five BS methods have been executed on a wide range of real, synthetic and semi-synthetic video sequences (some examples are given in figure 1). Our dataset is composed of 29 video sequences (15 real, 10 semi-synthetic and 4 synthetic). While some synthetic and semi-synthetic videos have been created by the authors, others were downloaded from the PETS2001 dataset [1], the IBM dataset [3] and the VSSN 2006 competition [2]. The semi-synthetic videos are made of synthetic foreground objects (people and cars) moving over a real background. Those videos represent both indoor (20 videos) and outdoor scenes (9 videos). Moreover, 6 videos contain animated background textures. Ground truths are easily obtained for synthetic and semi-synthetic video sequences. For real videos, the ground truth is only available on some reference images (manually annotated or provided with the dataset).

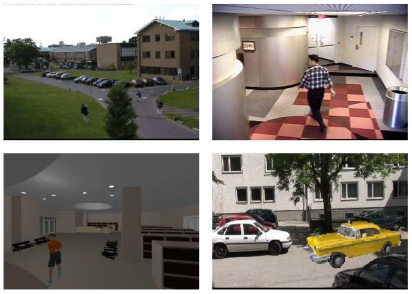

Figure 1. Snapshots from the dataset

4. Experimental Results

Since our dataset is made of videos representing different challenges, we divided those sequences into three categories: (1) the noise-free with perfectly fixed background sequences, (2) the multimodal sequences and (3) the noisy sequences. Each of these dataset is composed of real, synthetic and semi-synthetic video sequences. Those categories are used to evaluate and compare the different background models in section 4.2. In order to quantitatively evaluate each method, $precision$ and $recall$ average values are used [7]. Note that we use the *RGB* color space and a despeckle filter is applied on every motion mask in order to eliminate groups of 4-connected foreground pixels made of less than 8 pixels.

4.1. Influence of the Distance Measure

As mentioned in section 2, different distance measures can be implemented in BS methods. To quantify those distance measures, the *Basic* method has been executed on 15 videos taken from categories (1) and (3).

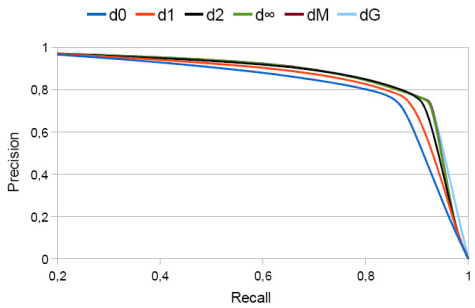

Figure 2. ROC Curves obtained for the six different distance measures

As can be seen in figure 2, the four distance measures d_2, d_∞, d_M and d_G globally produced the same results, while d_0 and d_1 seem slightly less precise. Table 1 shows results obtained with a fixed $precision$ of 0.75. A fixed precision means that every method has been tuned to produce the same amount of true positives, while the number of false positives is used to differentiate the methods. Again though, the uniform recall values underscore the fact that none of the 4 remaining distance measures outperforms the other ones.

	d_0	d_1	d_2	d_∞	d_M	d_G
Recall	0.88	0.90	**0.92**	**0.92**	**0.92**	**0.92**

Table 1. Recall values with fixed precision (0.75) for the six distance measures

4.2. Evaluation of Background Models

Every background models are compared on videos taken from the three categories previously introduced. Note that distance d_2 is used for algorithm *Basic*, d_M for *1-G*, $K = 3$ for *GMM*, $N = 100$ and σ_j is pre-estimated following Elgammal *et al.*'s method [5] for *KDE*, the learning rate α is fixed to 10^{-3} for *Basic*, *1-G* and *GMM*. Others parameters have been tuned to fix the precision value to 0.75 or 0.5.

Test 1: Evaluation on Noise-Free with Perfectly Static Background Videos

The BS algorithms presented in section 2 were executed on noise-free video sequences exhibiting a rigorously static background. A total of 15 videos (both indoor and outdoor) have been retained. Results with precision

fixed to 0.75 are presented in table 2. Interestingly, the results are globally homogeneous. Algorithms *Basic*, *1-G* and *GMM* are, to all practical purposes, equivalent while algorithms *KDE* and *MinMax* show less effectiveness. The *Basic* algorithm is thus well suited for sequences which do not contain specific difficulties. This is a rather interesting observation since the *Basic* approach is, by far, the fastest and simplest method of all.

Test 2: Evaluation on Multimodal Videos

The second test aims at evaluating the robustness of every BS methods to animated background textures. Here, 6 video sequences exhibiting strong background motion have been used. Results with precision fixed to 0.5 are presented in table 2.

As one would expect, the *Basic* and *MinMax* methods are strongly penalized by this test. On the other hand, results obtained with the *1-G* method are surprisingly good. This can be explained by the fact that the *1-G* threshold is locally weighted by a covariance matrix which compensates well some background instabilities. Thanks to their multimodal shape, the *KDE* and *GMM* methods produced the most accurate results. Figure 3 presents examples of detection on a semi-synthetic video with a strongly animated background.

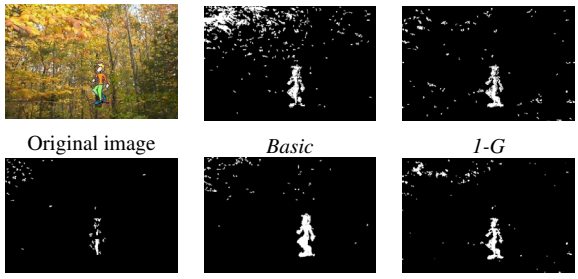

Figure 3. Results for different BS algorithms on a semi-synthetic multimodal video

Test 3: Evaluation on Noisy Videos

The third test aims at evaluating the influence of noise. Here, 15 videos corrupted with strong additive Gaussian noise are used for testing. Results with precision fixed to 0.75 are presented in table 2. As can be seen, methods *1-G*, *KDE* and *GMM* produced good, yet homogeneous recall values while the *MinMax* method does not seem to be well suited for noisy videos. This may be explained by the fact that the *MinMax* threshold (which is global) depends on the maximum interframe difference (which is large for noisy videos). Moreover, the *MinMax* method works on grayscale sequences and thus ignores color information. For the *Basic* method, its global threshold significantly penalizes the performances.

	Basic	1-G	KDE	MinMax	GMM
$test\ 1$	0.92	0.92	0.88	0.88	**0.93**
$test\ 2$	0.55	0.76	**0.84**	0.48	0.79
$test\ 3$	0.62	**0.77**	0.75	0.28	0.76

Table 2. Recall values for different video sequences. Precision is fixed to 0.75 for $test\ 1$ and $test\ 3$ and to 0.5 for $test\ 2$.

5. Conclusion

In this paper, we reviewed and evaluated commonly-implemented BS algorithms. Evaluation, performed on a large video dataset made of various real, synthetic and semi-synthetic video sequences, allows us to draw two conclusions. First, no method outperforms the other ones on every video category. Consequently, the choice of a BS method shall be motivated more by the content of the scene than the model itself. For instance, when dealing with videos respecting the fundamental BS assumption (a fixed camera with a static noise-free background), a basic BS implementation is well suited and no better results can be expected by other, yet more complicated, methods. Secondly, the *1-G*, *GMM* and *KDE* methods are the most reliable over noisy sequences. Thirdly, for videos containing significant background motion, methods with a global threshold (namely the *Basic* and *MinMax* methods) are significantly penalized while *GMM* and *KDE* show good robustness.

Our future work will focus on the robustness of these BS methods to bursty background motion and to corrupted training sequences. We will also compare the amount of memory and processing power those methods require.

References

[1] www.cvg.cs.rdg.ac.uk/PETS2001.

[2] http://imagelab.ing.unimore.it/vssn06.

[3] L. Brown, A. Senior, Y. Tian, J. Vonnel, A. Hampapur, C. Shu, H. Merkl, and M. Lu. Performance evaluation of surveillance systems under varying conditions. *in IEEE Proc. PETS Workshop*, 2005.

[4] S. Cheung and C. Kamath. Robust techniques for background subtraction in urban traffic video. *in Proc. of the VCIP*, 2004.

[5] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. *in ECCV*, 2000.

[6] I. Haritaoglu, D. Harwood, and L. Davis. W4-real time detection and tracking of people and their parts. *in IEEE Trans. PAMI*, 2000.

[7] A. Prati, I. Mikic, M. Trivedi, and R. Cucchiara. Detecting moving shadows: Algorithms and evaluation. *in IEEE Trans. PAMI*, 2003.

[8] C. Stauffer and W. Grimson. Adaptive background mixture models for real-time tracking. *in IEEE Proc. CVPR*, 1999.

[9] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *in IEEE Trans. PAMI*, 1997.

[10] Q. Zhou and J. Aggarwal. Tracking and classifying moving objects from video. *in IEEE Proc. PETS Workshop*, 2001.