Is a Memoryless Motion Detection Truly Relevant for Background Generation with LaBGen?

Benjamin Laugraud and Marc Van Droogenbroeck

Montefiore Institute, Department of Electrical Engineering and Computer Science
University of Liège, Belgium
`{blaugraud, m.vandroogenbroeck}@ulg.ac.be`

Abstract. The stationary background generation problem consists in generating a unique image representing the stationary background of a given video sequence. The LaBGen background generation method combines a pixel-wise median filter and a patch selection mechanism based on a motion detection performed by a background subtraction algorithm. In our previous works related to LaBGen, we have shown that, surprisingly, the frame difference algorithm provides the most effective motion detection on average. Compared to other background subtraction algorithms, it detects motion between two frames without relying on additional past frames, and is therefore *memoryless*. In this paper, we experimentally check whether the memoryless property is truly relevant for LaBGen, and whether the effective motion detection provided by the frame difference is not an isolated case. For this purpose, we introduce LaBGen-OF, a variant of LaBGen leverages memoryless dense optical flow algorithms for motion detection. Our experiments show that using a memoryless motion detector is an adequate choice for our background generation framework, and that LaBGen-OF outperforms LaBGen on the SBMnet dataset. We further provide an open-source C++ implementation of both methods at `http://www.telecom.ulg.ac.be/labgen`.

Keywords: Background generation, background initialization, motion detection, optical flow, median filter, SBI, SBMnet.

1 Introduction

Given a video sequence acquired from a fixed viewpoint, the *stationary background generation problem* consists in generating a unique image representing the stationary background (*i.e.* the set of elements that are motionless during the whole sequence). The generation of a background image is useful for many applications such as surveillance, segmentation, compression, inpainting, privacy protection, and computational photography, as stated in [11].

One of the simplest background generation method consists in applying a pixel-wise temporal median filter on the input sequence. It makes the strong assumption that the background is seen more than half of the time for each pixel, although it is hardly met for highly cluttered sequences. To tackle the different challenges raised by complex scenes (*e.g.* intermittent object motions, dynamic backgrounds, illumination changes, etc), several methods have been presented over the years [7,3,11]. Our method, called LaBGen, was ranked number one during the IEEE Scene Background Modeling Contest (SBMC 2016)[1]. In short, LaBGen works as follows. For a given input video sequence, it relies on motion detection to select the patches with the smallest quantities of motion. Then, the background image is generated by blending the selected patches with a pixel-wise median filter. In LaBGen, the motion detection is carried out by a background subtraction algorithm which indicates, for each pixel, whether it belongs to the background or not.

In our previous works related to LaBGen [10], we have shown that, surprisingly, the frame difference background subtraction algorithm is the most effective motion detection, on average, for our background generation framework. Compared to other background subtraction algorithms, the main specificity of the frame difference algorithm is that it detects motion between two frames without relying on additional past frames. Therefore, it is *memoryless* and cannot be influenced by past errors.

In this paper, we experimentally check whether the memoryless property is truly relevant for LaBGen, and whether the effective motion detection provided by the frame difference is not an isolated case. First, we model the amount of memory being used by the motion detection component to measure its impact on the performance achieved by LaBGen. Then, we compare the contribution of several motion detection algorithms with, or without memory, on the performance achieved by LaBGen. For this purpose, we propose a new variant of LaBGen, named LaBGen-OF hereafter, that leverages memoryless *dense optical flow* algorithms able to compute a velocity vector between two frames for each pixel. As LaBGen-OF performs better than LaBGen, we evaluate it in depth on the SBMnet dataset, and compare it to LaBGen and other state-of-the-art background generation methods.

This paper is organized as follows. Section 2 describes the principles of LaBGen, and its new variant, LaBGen-OF. Section 3 presents our experiments and results. Finally, section 4 concludes this paper.

2 The LaBGen Background Generation Framework

In addition to be computationally efficient[2], LaBGen offers a flexible mechanism for incorporating and testing new motion detection algorithms. Thus, in this section, we first describe the principles of LaBGen (see Section 2.1), and then explain how optical flow algorithms have been integrated into LaBGen to build the new LaBGen-OF variant (see Section 2.2).

[1] `http://pione.dinf.usherbrooke.ca/sbmc2016`

[2] LaBGen, with the set of parameters proposed in [9], runs at a frame rate of more than 1300 fps for a VGA video sequence on a Core i7-4790K.

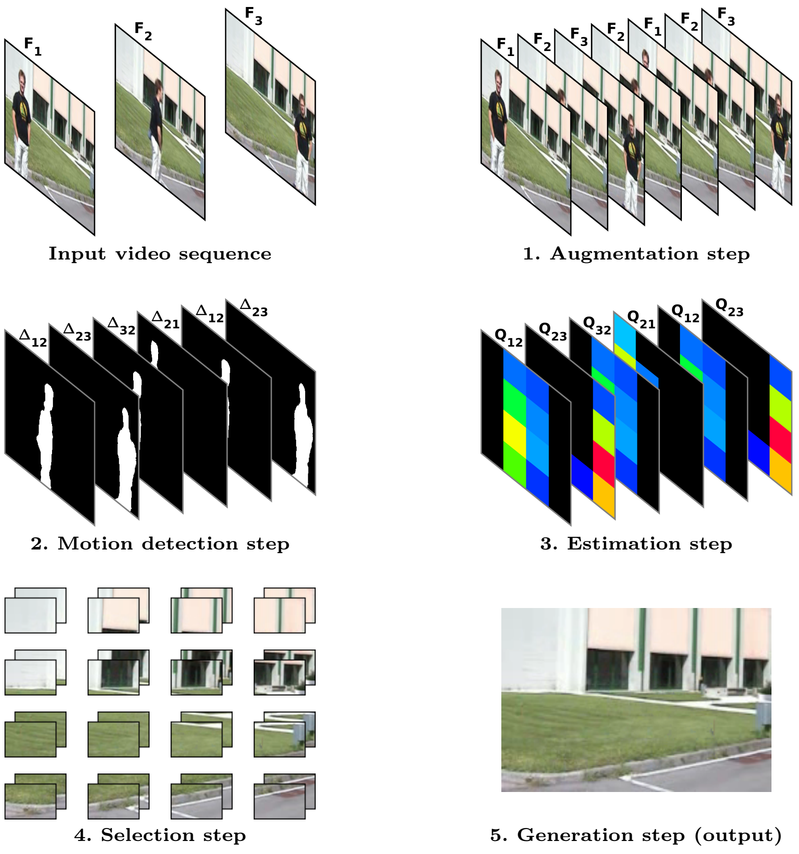

Fig. 1. Illustration of the different steps of LaBGen applied on a video sequence composed of three frames with $\mathcal{P} = 3$, $\mathcal{N} = 4$, $\mathcal{S} = 2$, and $\mathcal{A}$ = Ground-truth. Quantities of motion are represented by the hue component.

2.1 Short Description of the Principles of LaBGen

The LaBGen background generation method, extensively described in [10], combines a pixel-wise median filter and a patch selection mechanism based on a motion detection performed by a background subtraction algorithm. It is composed of the five following steps, which are illustrated in Fig. 1:

1. An *augmentation step* increases the length of the input video sequence according to a parameter $\mathcal{P}$. This is performed by duplicating the frames of the sequence in the forwards and backwards chronological orders alternatively.
2. For each frame of the augmented sequence, a *motion detection step* determines which pixels belong to the background using a background subtraction algorithm $\mathcal{A}$. The classification results are put in a binary *segmentation map*.

3. To estimate *quantities of motion* spatially, an *estimation step* divides the image plane into $\mathcal{N} \times \mathcal{N}$ spatial areas. Then, for each patch, quantities of motion are estimated by counting the number of pixels classified as foreground.
4. For each spatial area, a *selection step* builds a subset of $\mathcal{S}$ patches. The elements belonging to this subset are the $\mathcal{S}$ patches associated to the smallest quantities of motion.
5. Finally, a *generation step* builds the background image by applying a pixel-wise median filter on each subset built during the previous step.

2.2 LaBGen-OF: A Variant of LaBGen Incorporating Optical Flow

The LaBGen-OF variant of LaBGen consists in replacing background subtraction by optical flow in the motion detection step (see Fig. 2). As a motion estimation must be available for all pixels, we only use optical flow algorithms that output dense vector fields. For a video sequence, such an algorithm can estimate, at time t, the displacement of each pixel between the current frame $F(t)$ and the next frame $F(t+1)$, at time $t+1$. More specifically, for a given pixel $p(x, y, t)$ (x and y being its coordinates in the image plane), it computes a velocity vector $\mathbf{v}(x, y, t) = (v_x(x, y, t), v_y(x, y, t))$ that validates the following correspondence:

$$p(x, y, t) \;\leftrightarrow\; p(x + v_x(x, y, t), y + v_y(x, y, t), t + 1) . \tag{1}$$

As LaBGen expects a binary segmentation map as output from the motion detection step, we have to transform the vector field $\mathbf{v}(t)$ into such a map. By design, LaBGen relies on the presence of motion rather than on its direction. Thus, to build a binary segmentation map $m(t)$, we apply a hard threshold τ on the spatially normalized ℓ^2-norms of the velocity vectors belonging to the vector field $\mathbf{v}(t)$. The spatial normalization n, applied on the ℓ^2-norms, aims at reducing the sensibility to video scaling without the need to change the value chosen for τ; it works as follows:

$$n(\mathbf{v}(x, y, t)) = \frac{\|\mathbf{v}(x, y, t)\|_2}{\max\limits_{x,y} \|\mathbf{v}(x, y, t)\|_2} . \tag{2}$$

In short, to build the binary segmentation map $m(t)$, we use the following rule:

$$m(x, y, t) = \begin{cases} \text{foreground} & \text{if } n(\mathbf{v}(x, y, t)) > \tau, \\ \text{background} & \text{if } n(\mathbf{v}(x, y, t)) \leq \tau. \end{cases} \tag{3}$$

While a background subtraction algorithm applied between two frames $F(t)$ and $F(t+1)$ classifies each pixel of frame $F(t+1)$ as belonging to the background or not, an optical flow algorithm highlights the displacement of each pixel from frame $F(t)$ to $F(t+1)$. Consequently, a binary segmentation map $m(t+1)$ constructed from an optical flow should be used to compute the quantities of motion associated to the patches extracted from frame $F(t)$. For this reason, instead of skipping the first frame of a given video sequence, where no motion is temporally observable by a background subtraction algorithm, we skip the last one.

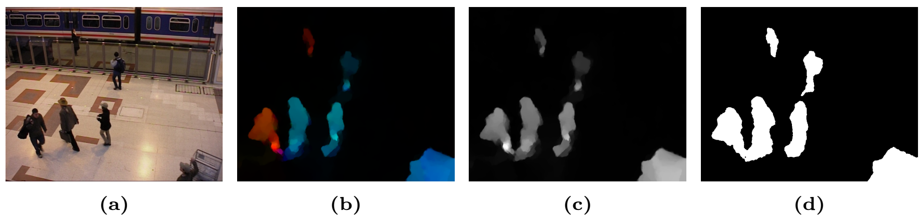

Fig. 2. The motion detection step of LaBGen-OF. The application of a dense optical flow algorithm on the frame (a) and its successor produces a vector field such as (b) (the hue and saturation components represent the angle and magnitude, respectively). Then, the spatially normalized ℓ^2-norms are computed (c), and thresholded (d), in order to get a binary segmentation map.

3 Experiments

3.1 Experimental Setup

Our experimental setup consists of 2 datasets, and 6 different metrics. The SBI 2016[3] [12] dataset gathers 14 video sequences, all provided with ground-truth. They are composed of 6 to 740 frames, and their resolution varies from 144×144 to 800×600 pixels. The SBMnet 2016 dataset[4] gathers 79 video sequences composed of 6 to 9370 frames and whose resolution varies from 240×240 to 800×600 pixels. The sequences are scattered through 8 categories, a category being associated to a specific challenge: Basic, Intermittent Motion, Clutter, Jitter, Illumination Changes, Background Motion, Very Long, and Very Short. The ground-truth is provided for only 13 sequences distributed among the categories. It should be noted that this dataset is associated to a reference web platform which aims at ranking background generation methods by assessing them on the overall dataset. In several experiments, we simultaneously consider the video sequences of SBI, and the ones of SBMnet provided with ground-truth. This special set of video sequences will be referred to as the SBI+SBMnet-GT dataset.

Six metrics, described in [12], are considered to assess quantitatively our results. The ones to minimize (resp. maximize) are followed by a $\downarrow$ (resp. $\uparrow$):

- Average Gray-level Error (AGE, $\downarrow$, $\in [0, 255]$)
- Percentage of Error Pixels (pEPs, $\downarrow$, in %)
- Percentage of Clustered Error Pixels (pCEPs, $\downarrow$, in %)
- Multi-Scale Structural Similarity Index (MS-SSIM, $\uparrow$, $\in [-1, 1]$)
- Peak-Signal-to-Noise-Ratio (PSNR, $\uparrow$, in dB)
- Color image Quality Measure (CQM, $\uparrow$, in dB)

Except for the CQM metric which works with RGB images, all the metrics are computed on the luminance Y channel.

[3] `http://sbmi2015.na.icar.cnr.it/SBIdataset.html`
[4] `http://scenebackgroundmodeling.net`

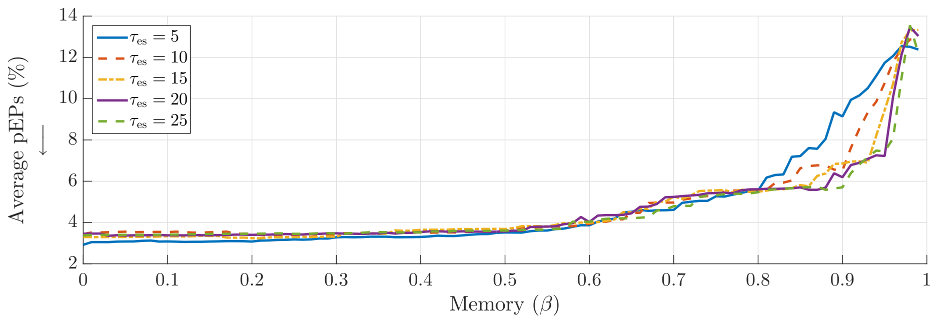

Fig. 3. Average performance of LaBGen embedded with the exponential smoothing algorithm as a function of the amount of memory (modeled by the weight β) used for motion detection.

3.2 Impact of the Motion Detection Memory on the Performance

Our first experiment consists in modeling the amount of memory being used by the motion detection component to measure its impact on the performance of LaBGen. For that purpose, we use an *exponential smoothing* based background subtraction algorithm [14] as a motion detector in LaBGen. Let $I\left(x, y, t\right)$ be the intensity of pixel $p\left(x, y, t\right)$ in the current input frame $F\left(t\right)$, and $B\left(x, y, t\right)$ the intensity of the same pixel in the background model of the exponential smoothing algorithm. At time t, each pixel of the background model is updated as follows:

$$B\left(x, y, t\right) = \left(1 - \beta\right) \cdot I\left(x, y, t\right) + \beta \cdot B\left(x, y, t - 1\right), \qquad (4)$$

with $\beta \in [0, 1]$. The segmentation map $m\left(t\right)$ is built by applying a hard threshold τ_{es} on the absolute difference of intensities between the current input frame $F\left(t\right)$ and the background model $B\left(t\right)$. For $\beta = 0$, the exponential smoothing algorithm is equivalent to the frame difference. The more β increases, the more importance is given to the past input frames in the background model update. In other words, the parameter β allows us to model and tune the amount of memory being used to detect motion.

To measure its impact on the performance of LaBGen, background images have been produced for each sequence of the SBI+SBMnet-GT dataset with 100 values of β linearly distributed in the interval $[0, 0.99]$. These images have been generated by LaBGen embedded with the exponential smoothing algorithm using the set of parameters proposed in [9] for the SBMnet dataset ($\mathcal{P} = 1$, $\mathcal{N} = 3$, $\mathcal{S} = 19$). Fig. 3 presents the *average performance* (*i.e.* the performance averaged over the considered sequences) achieved for each value of β and 5 different thresholds τ_{es}. One can observe that it decreases as the amount of memory increases. Furthermore, we see that the average performance achieved when $\beta = 0$ (*i.e.* when the exponential smoothing algorithm has no memory and is equivalent to the frame difference) is the best one, or close to the best one. Although our memory model is simple, this experiment is a first indication that using no memory to detect motion is an appropriate choice for LaBGen. In the next section, we further experiment the role of memory by using several motion detectors built upon more complex models.

3.3 Comparison of Motion Detectors With or Without Memory

In this section, we provide a comprehensive evaluation of the performance achieved by the LaBGen background generation framework when it incorporates a motion detection algorithm with, or without memory. More specifically, we compare the average performance achieved by LaBGen embedded with the memoryless frame difference and other background subtraction algorithms with memory, and LaBGen-OF with memoryless optical flow algorithms. To perform a fair comparison, we produced a background image for each sequence of the SBI dataset using LaBGen and LaBGen-OF with each combination of several parameter values. The parameter values being considered for LaBGen are $\mathcal{P} = 1, 3, 5$; $\mathcal{N} = 1, 2, \ldots, 10$; $\mathcal{S} = 1, 3, \ldots, 201$; and $\mathcal{A}$ with the frame difference, and 11 other background subtraction algorithms enumerated and described in [10]. The parameter values being considered for LaBGen-OF are the same for $\mathcal{P}$, $\mathcal{N}$ and $\mathcal{S}$; $\tau = 0.02, 0.03, \ldots, 0.1$; and $\mathcal{A}$ with 6 optical flow algorithms: a fast sparse-to-dense pyramidal Lucas-Kanade [2], Farnebäck [5], Dual TV-L^1 [21], SimpleFlow [18], DeepFlow [20], and DISFlow [8]. Their implementations have been found in the OpenCV 3.2 library[5], and they are used with their default or recommended sets of parameters. When they require gray-scale images, we arbitrarily chose to convert pixels from RGB images as follows:

$$I(x, y, t) = 0.299 \cdot r(x, y, t) + 0.587 \cdot g(x, y, t) + 0.114 \cdot b(x, y, t), \tag{5}$$

with $r(x, y, t)$, $g(x, y, t)$, and $b(x, y, t)$ being respectively the red, green, and blue components of pixel $p(x, y, t)$. Note that, regarding its controversial contribution, no denoising pre-processing has been used. Indeed, on the one hand, recent optical flow algorithms take noise into consideration in their model. On the other hand, pre-processing can degrade their performances [17].

Fig. 4 provides the average performance achieved by LaBGen-OF and LaBGen using motion detection algorithms with, or without memory. One can observe that the average performance achieved by LaBGen-OF embedded with any optical flow algorithm is always better than the one with any background subtraction algorithm with memory. Even though the average performance achieved by LaBGen-OF embedded with Farnebäck and Lucas-Kanade is slightly worse than the one achieved by LaBGen with the memoryless frame difference, the contributions of the other optical flow algorithms improve the average performance of our background generation framework on the SBI dataset ($\mathit{\Delta}$pEPs $\simeq 0.49\%$ for SimpleFlow). This suggests that, on average, a memoryless motion detection algorithm in the LaBGen background generation framework enables to reach the best achievable performances.

[5] `http://opencv.org`

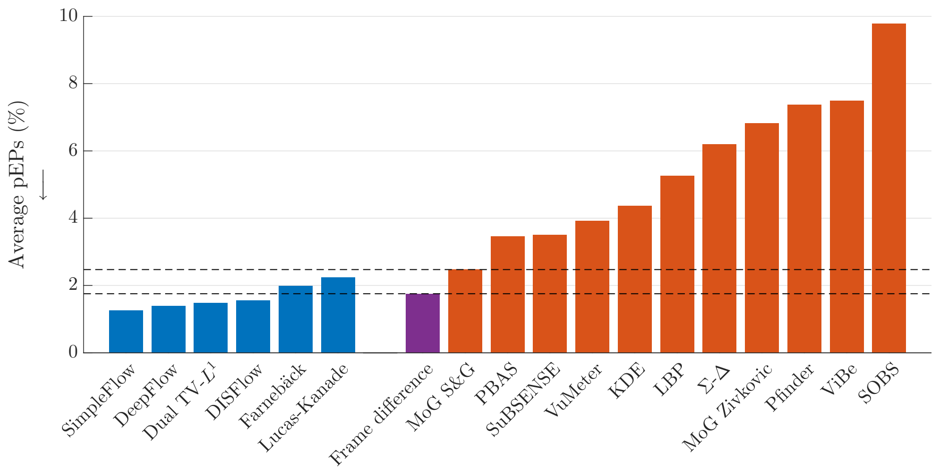

Fig. 4. Best average performance achieved on the SBI dataset by LaBGen-OF embedded with different memoryless optical flow algorithms 🟦, and LaBGen with the memoryless frame difference 🟪 and other background subtraction algorithms 🟧.

3.4 Average Performance of LaBGen-OF on SBMnet

As LaBGen-OF performs better than LaBGen on the SBI dataset (see Fig. 4), we decided to assess it on the state-of-the-art SBMnet dataset, and to send its results on the associated web platform. We first tuned the parameters over the same combinations of values described in Section 3.3 to achieve the best average performance. Table 1 provides the best set of $(\mathcal{P}, \mathcal{N}, \mathcal{S}, \tau)$ parameters for a given optical flow algorithm $\mathcal{A}$, being found by maximizing the CQM score averaged over the sequences of the SBI+SBMnet-GT dataset. As the following set of parameters:

$$(\mathcal{A}, \mathcal{P}, \mathcal{N}, \mathcal{S}, \tau)_{\text{default}} = (\text{DeepFlow}, 3, 8, 119, 0.04), \qquad (6)$$

leads to the best average performance, we decided to recommend it as a *default set of parameters*. Although there is no consensus, from reading the metrics, on which optical flow algorithm provides the second best contribution to the LaBGen-OF average performance, they agree to a large extent that the contribution of DeepFlow is the most valuable. According to the same metrics, the rank of LaBGen embedded with the frame difference is never better than 5 over 7. This suggests that LaBGen-OF performs better than LaBGen, and that most optical flow algorithms are adequate memoryless substitutes for the frame difference algorithm in our background generation framework. An analysis of the run times reported in Table 1 shows that using optical flow algorithms slows down LaBGen, especially when a state-of-the-art algorithm such as DeepFlow is used. Thus, LaBGen-OF is less suitable for real-time applications because it favors the quality of the estimated background at the price of an increase of the run time.

Table 1. Best average performance achieved by LaBGen-OF with respect to each optical flow algorithm on the SBI+SBMnet-GT dataset. The reported performances are colored according to their rank from white (best) to black (worse). The last column provides the run time expressed in frames per second (fps), when a VGA video sequence is processed with an Intel Core i7-4790K CPU. Note that the results of LaBGen have been computed with the set of parameters proposed in [9] ($\mathcal{P} = 1$, $\mathcal{N} = 3$, $\mathcal{S} = 19$).

	Best parameters				**Averaged metrics**						**Run time**
$\mathcal{A}$	$\mathcal{P}$	$\mathcal{N}$	$\mathcal{S}$	τ	**AGE** ↓	**pEPs** ↓	**pCEPs** ↓	**MS-SSIM** ↑	**PSNR** ↑	**CQM** ↑	**(fps)** ↑
DeepFlow	3	8	119	0.04	4.0621	2.60%	1.20%	0.9708	33.1041	33.7400	5
Lucas-Kanade	1	6	63	0.03	4.2515	3.02%	1.60%	0.9718	32.8775	33.5150	44
DISFlow	1	3	57	0.02	4.7678	3.91%	2.15%	0.9520	32.3675	32.9819	124
Farnebäck	3	3	83	0.05	4.5974	3.42%	1.70%	0.9521	32.2720	32.8954	13
SimpleFlow	3	6	49	0.06	4.4212	2.94%	1.36%	0.9596	31.9124	32.5169	5
Dual TV-L^1	5	10	75	0.06	4.3821	2.90%	1.41%	0.9669	31.8324	32.4793	2
LaBGen + Frame difference					4.5863	3.31%	1.63%	0.9464	31.8394	32.4585	1312

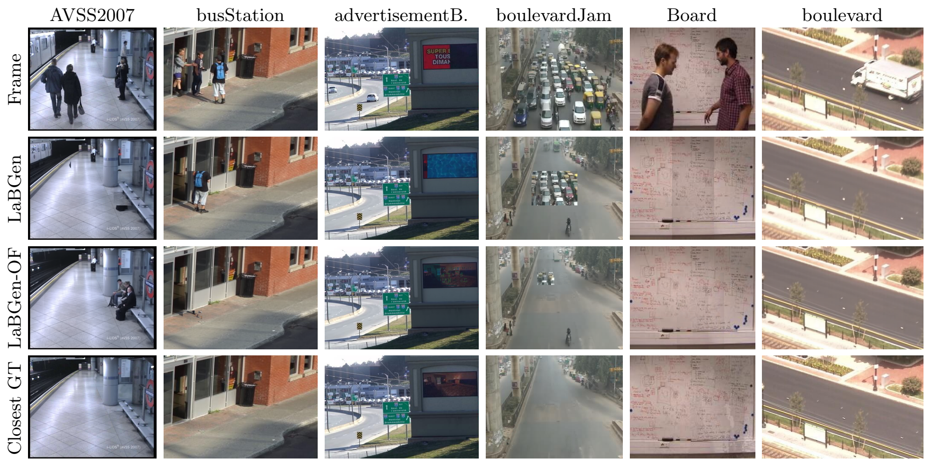

Fig. 5. Background generated for some SBMnet sequences provided with ground-truth (the 1st row contains a randomly selected frame) by LaBGen [9] (2nd row), and by LaBGen-OF with its default set of parameters given in Eq. 6 (3rd row). The 4th row contains the closest ground-truths to the results of LaBGen-OF. Only sequences where a difference is visually noticeable are shown.

Fig. 5 shows background image estimates for some SBMnet sequences provided with ground-truth by LaBGen and LaBGen-OF. It is easy to observe that LaBGen-OF improves most of the results produced by LaBGen. For instance, several errors are removed from the busStation, boulevardJam, and Board sequences. For a sequence such as AVSS2007, although the amount of errors made by both methods is almost similar, the errors are different. In the background estimated by LaBGen-OF, there is no incomplete objects, and the metro passing through the scene is totally removed. We can also observe that the background produced by LaBGen-OF for the boulevard sequence is less blurred, which suggests that this method is less sensitive to camera jitter.

Table 2. Top-8 reported on the SBMnet web platform the March 8th, 2017 in which the performance achieved by LaBGen-OF has been inserted (in blue).

Method	**Average ranking ↓**	**A. r. across categories ↓**	**Average AGE ↓**	**Average pEPs ↓**	**Average pCEPs ↓**	**Average MS-SSIM ↑**	**Average PSNR ↑**	**Average CQM ↑**	**Run time (fps) ↑**
MSCL (anon.)	**1.17**	4.75	**5.9545**	**5.24%**	**1.71%**	0.9410	**30.8952**	**31.7049**	unknown
LaBGen-OF	2.00	**4.25**	6.1897	5.66%	2.32%	**0.9412**	29.8957	30.7006	5
BEWiS [4]	4.17	5.63	6.7094	5.92%	2.66%	0.9282	28.7728	29.6342	3
LaBGen [10]	4.67	7.25	6.7090	6.31%	2.65%	0.9266	28.6396	29.4668	1312
LaBGen-P [9]	5.83	8.00	7.0738	7.06%	3.19%	0.9278	28.4660	29.3196	126
Photomontage [1]	6.33	10.38	7.1950	6.86%	2.57%	0.9189	28.0113	28.8719	unknown
SC-SOBS-C4 [13]	7.33	8.88	7.5183	7.11%	2.42%	0.9160	27.6533	28.5601	unknown
MAGRPCA [6]	8.67	8.88	8.3132	9.94%	5.67%	0.9401	28.4556	29.3152	2.5
Temporal median	10.33	8.25	8.2731	9.84%	5.46%	0.9130	27.5364	28.4434	100

3.5 Comparison to Other Background Generation Methods

Table 2 presents the Top-8 of background generation methods evaluated on the overall SBMnet dataset using the associated web platform, in which the results of LaBGen-OF have been inserted. The methods presented in this table are based on principles different from that of LaBGen and its variants. BEWiS [4] uses a weightless neural network, called WiSARD^{rp}, made up of memory nodes, each composed of several cells. For each pixel, the stimulated cells are increased by a reward, while the other ones are decreased by a punishment. The cells associated to the largest values compose the background. SC-SOBS-C4 [13] builds a neural map which associates each pixel to several HSV weight vectors. The final background is built by comparing the neural map to a reference background (the one given by Photomontage in [13]). Precisely, for each pixel, SC-SOBS-C4 keeps the weight vector with the smallest intensity difference with respect to the reference background. Unlike neural network based methods, Photomontage [1] combines a series of images with respect to an objective function optimized by graph-cuts. For background generation, the chosen objective function is the maximum likelihood. Finally, MAGRPCA [6] is a subspace learning method in which an optimization problem is formulated to decompose the matrix formed by the input frames into a low-rank (background) and a sparse (foreground) matrix. A motion-aware regularization of graphs on the low-rank component is used to exploit the similarity and locality information available among frames.

Note that other categories of methods exist [3]. For instance, WS2006 [19] looks for the most reliable temporal subsequence of stable intensities for each pixel, and takes its mean as the background value. RSL2011 [15] estimates the background in a Markov random field framework, where the optimal labeling solution is computed iteratively. In [16], Sobral et al. apply a matrix or tensor completion algorithm on areas that are considered in motion.

According to the average ranking across categories reported in Table 2, LaBGen-OF is now ranked first, although it is ranked second according to the average ranking. In the review written by Bouwmans et al. [3], the authors conclude that the best average performances are generally achieved by neural network based methods. However, LaBGen-OF, which is considered as a method based on subsequences of stable intensity, now outperforms BEWiS [4], the main competitor of LaBGen. Regarding the run time, LaBGen clearly outperforms its competitors by reaching 1312 fps on VGA sequences. Although the processing speed of LaBGen-OF with a good optical flow algorithm is reduced to 5 fps, it remains slightly above the speed achieved by most state-of-the-art methods.

4 Conclusion

The stationary background generation is a challenging problem which consists in generating a unique image representing the stationary background of a given video sequence. The LaBGen method, designed to perform this task, leverages a motion detection step driven by background subtraction algorithms. As the memoryless frame difference background subtraction algorithm provides, on average, the best contribution to the performance of LaBGen, we decided to check the importance of the memoryless property for our background generation framework. Our experiments confirm that a memoryless motion detection helps reaching the best achievable average performances. Moreover, using LaBGen-OF, a new variant of LaBGen using optical flow algorithms for motion detection, we also learned that a memoryless motion detection algorithm allows to achieve a better performance than several popular background subtraction algorithms with memory. Further investigations show that LaBGen-OF outperforms LaBGen embedded with the frame difference and almost all state-of-the-art background generation methods on the SBMnet dataset. Based on these results, we can conclude that a memoryless motion detection is an appropriate choice for the LaBGen background generation framework. Please note that an open-source C++ implementation of LaBGen-OF and LaBGen is available at `http://www.telecom.ulg.ac.be/labgen`.

Acknowledgments. We are thankful to Pr. Pierre-Marc Jodoin for letting us use the SBMnet web platform to assess some variants of LaBGen.

References

1. Agarwala, A., Dontcheva, M., Agrawala, M., Drucker, S., Colburn, A., Curless, B., Salesin, D., Cohen, M.: Interactive digital photomontage. ACM Transactions on Graphics 23(3), 294–302 (August 2004)
2. Bouguet, J.Y.: Pyramidal implementation of the Lucas Kanade feature tracker - Description of the algorithm. Intel Corporation 5(1-10), 4 (2001)
3. Bouwmans, T., Maddalena, L., Petrosino, A.: Scene background initialization: a taxonomy. Pattern Recognition Letters 96, 3–11 (2017)
4. De Gregorio, M., Giordano, M.: Background estimation by weightless neural networks. Pattern Recognition Letters 96, 55–65 (2017)
5. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Scandinavian Conference on Image Analysis (SCIA). Lecture Notes in Computer Science, vol. 2749, pp. 363–370. Springer (2003)

6. Javed, S., Mahmmod, A., Bouwmans, T., Jung, S.K.: Motion-aware graph regularized RPCA for background modeling of complex scene. In: IEEE International Conference on Pattern Recognition (ICPR), IEEE Scene Background Modeling Contest (SBMC). pp. 120–125. Cancún, Mexico (December 2016)
7. Jodoin, P.M., Maddalena, L., Petrosino, A., Wang, Y.: Extensive benchmark and survey of modeling methods for scene background initialization. IEEE Transactions on Image Processing 26(11), 5244–5256 (November 2017)
8. Kroeger, T., Timofte, R., Dai, D., Van Gool, L.J.: Fast optical flow using dense inverse search. CoRR abs/1603.03590 (2016)
9. Laugraud, B., Piérard, S., Van Droogenbroeck, M.: LaBGen-P: A pixel-level stationary background generation method based on LaBGen. In: IEEE International Conference on Pattern Recognition (ICPR), IEEE Scene Background Modeling Contest (SBMC). pp. 107–113. Cancún, Mexico (December 2016)
10. Laugraud, B., Piérard, S., Van Droogenbroeck, M.: LaBGen: A method based on motion detection for generating the background of a scene. Pattern Recognition Letters 96, 12–21 (2017)
11. Maddalena, L., Petrosino, A.: Background model initialization for static cameras. In: Background Modeling and Foreground Detection for Video Surveillance, chap. 3, pp. 3.1–3.16. Chapman and Hall/CRC (2014)
12. Maddalena, L., Petrosino, A.: Towards benchmarking scene background initialization. In: International Conference on Image Analysis and Processing Workshops (ICIAP Workshops). Lecture Notes in Computer Science, vol. 9281, pp. 469–476 (September 2015)
13. Maddalena, L., Petrosino, A.: Extracting a background image by a multi-modal scene background model. In: IEEE International Conference on Pattern Recognition (ICPR), IEEE Scene Background Modeling Contest (SBMC). pp. 143–148. Cancún, Mexico (December 2016)
14. McIvor, A.: Background subtraction techniques. In: Proc. of Image and Vision Computing. Auckland, New Zealand (November 2000)
15. Reddy, V., Sanderson, C., Lovell, B.: A low-complexity algorithm for static background estimation from cluttered image sequences in surveillance contexts. EURASIP Journal on Image and Video Processing 2011, 164956 (2011)
16. Sobral, A., Zahzah, E.H.: Matrix and tensor completion algorithms for background model initialization: A comparative evaluation. Pattern Recognition Letters 96, 22–33 (2017)
17. Sun, D., Roth, S., Black, M.J.: A quantitative analysis of current practices in optical flow estimation and the principles behind them. International Journal of Computer Vision 106(2), 115–137 (2014)
18. Tao, M.W., Bai, J., Kohli, P., Paris, S.: SimpleFlow: A non-iterative, sublinear optical flow algorithm. Computer Graphics Forum (Eurographics 2012) 31(2), 345–353 (May 2012)
19. Wang, H., Suter, D.: A novel robust statistical method for background initialization and visual surveillance. In: Asian Conference on Computer Vision (ACCV). Lecture Notes in Computer Science, vol. 3851, pp. 328–337. Berlin, Heidelberg (January 2006)
20. Weinzaepfel, P., Revaud, J., Harchaoui, Z., Schmid, C.: DeepFlow: Large displacement optical flow with deep matching. In: International Conference on Computer Vision (ICCV). pp. 1385–1392. Sydney, Australia (December 2013)
21. Zach, C., Pock, T., Bischof, H.: A duality based approach for realtime TV-L^1 optical flow. In: Joint Pattern Recognition Symposium (JPRS). Lecture Notes in Computer Science, vol. 4713, pp. 214–223. Springer (2007)