Noname manuscript No.
(will be inserted by the editor)

The effect of downsampling-upsampling strategy on foreground detection algorithms

Miguel A. Molina-Cabello · Jorge García-González · Rafael M. Luque-Baena · Ezequiel López-Rubio

Received: date / Accepted: date

Abstract In video surveillance systems which incorporate stationary cameras, the first phase of movement object detection is crucial for the correct modelling of the behavior of these objects, as well as being the most complex in terms of execution time. There are many algorithms that provide a reliable and adequate segmentation mask, obtaining real-time ratios for reduced image sizes. However, due to the increased performance of camera hardware, the application of previous methods to sequences with higher resolutions (from 640x480 to 1920x1080) is not carried out in real time, compromising their use in real video surveillance systems. In this paper we propose a methodology to reduce the computational requirements of the algorithms, consisting of a reduction of the input frame and, subsequently, an interpolation of the segmentation mask of each method to recover the original frame size. In addition, the viability of this meta-model is analyzed together with the different selected algorithms, evaluating the quality of the resulting segmentation and its gain in terms of computation time.

Keywords Foreground detection · video size downsampling · interpolation techniques

1 Introduction

The detection of foreground objects in video sequences is a well-known issue. It is a low level task of paramount importance in current automated video surveillance systems which has been addressed by several works (Wren et al., 1997;

Miguel A. Molina-Cabello
Department of Computer Languages and Computer Science
University of Málaga
Bulevar Louis Pasteur, 35.
29071 Málaga. Spain.
E-mail: miguelangel@lcc.uma.es

López-Rubio et al., 2011; López-Rubio et al., 2018a). The advent of inexpensive high resolution surveillance cameras provides higher quality video to be supplied to the foreground detection algorithms. However, these algorithms are typically heavy in computational load terms. Therefore, high resolution video frames cannot be processed by many algorithms in real time, which hampers their application to practical situations. This calls for methods to reduce the computational complexity of foreground detection methods when they have to process high resolution video frames.

A possible strategy to address this issue involves reducing the spatial resolution of the incoming video data. Despite the information loss, if the quality of the output of the foreground detection algorithm is similar and the final complexity time is reduced, this input image size reduction is worthwhile. This combination has already been considered in several previous papers, not only in the field of foreground detection (Xue et al., 2013; Cvetkovic et al., 2006) but also in a more general scope such as saliency detection (Lowe, 2004; Guan, 2010; Zhang et al., 2017; Yubing et al., 2011; Yan et al., 2013). However, in previous literature the image downsampling is integrated into each proposed method, but its effect has not been studied as a preprocessing method in the field of foreground detection.

In our previous work we have studied the degradation of the performance of foreground detection systems under noise conditions (López-Rubio et al., 2018b); now we aim to study the effect of spatial resolution reduction in the performance of the foreground detection algorithms. This way, we propose a general methodology to reduce the computational requirements of a background model by downsampling the original input video frames. A subsequent upsampling is carried out in order to yield a foreground mask of the same size of the original frame. Furthermore, several relevant foreground detection methods have been considered to carry out a systematic study of the influence of the downsampling-upsampling strategy on this type of algorithms. These selected algorithms stand out for their high impact and number of citations. For example, *Grimson* (Stauffer and Grimson, 1999) has been one of the most used methods in video surveillance in recent years and is a reference within the existing bibliography. On the other hand, *SuBSENSE* (St-Charles et al., 2014), *LOBSTER* (St-Charles and Bilodeau, 2014), and *PAWCS* (St-Charles et al., 2016) are more recent and stand out for being the most effective in several data sets. In addition, a public repository formed by 31 video sequences with more than 88000 annotated frames has been analyzed, because it offers a representative vision of the behavior of the selected foreground detection algorithms by applying the proposed methodology in different categories. Our proposal extends our earlier results (Molina-Cabello et al., 2016), which were restricted to a particular background model.

The rest of the paper is organized as follows. Section 2 outlines the related works in the field of background modelling, 3 sets out the methodology of this approach by describing the probabilistic model applied to each tiling and how to compute the foreground mask. Section 4 shows several experimental results over some well-known public video surveillance sequences and finally Section 5 concludes the article.

2 Related Works

The field of the foreground detection in video sequences, also called background modelling or background subtraction, has been widely studied by the scientific community in recent years. Numerous works have been published in the last two decades, highlighting the diversity of techniques applied to solve a problem that is still considered open due to its difficulty. The key idea is to distinguish between objects (or pixels) that are in motion from those that belong to the background of the scene. Despite its simplicity from the human point of view, there are many key issues that make difficult to provide an adequate foreground segmentation mask for each frame (Bouwmans, 2014; Hu et al., 2004). The most traditional proposals have applied probabilistic distributions for modelling the background of the scene, with satisfactory results in first generation video surveillance systems (Wren et al., 1997; Stauffer and Grimson, 1999; Zivkovic, 2004). Progressively, more complex models have been incorporated, such as kernel distributions (Elgammal et al., 2002; Mao and Shi, 2005; Mao et al., 2012), self-organized neural networks (Maddalena and Petrosino, 2008, 2012; López-Rubio et al., 2011) or algorithms for dimensionality reduction (Javed et al., 2018). All of them try to generate a robust background model that adapts to the scene and face the main issues that still remain open.

Among them, it is worth noting the problem of illumination changes, both gradual and sudden, that must be handled by any robust method of motion detection. The use of different color spaces has been effective in reducing irregular lighting variations and sudden changes in the brightness of scene (Yong-Beom Lee et al., 2002). In addition, many works have shown that the use of color spaces with less correlation between channels (Lab, Luv, HSV) has improved background detection and reduced the influence of cast shadows by moving objects (Benedek and Szirányi, 2007; López-Rubio and López-Rubio, 2015). On the other hand, the combination of color and texture has also been used in many proposals, since it is possible to extract robust information from the input frames that is not sensitive to lighting variations (Shen et al., 2006).

A multiresolution method is considered in Wu et al. (2006) in order to effectively determine an adaptive threshold with hysteresis to separate the foreground from the background. A multiresolution strategy is also used in Zhao and Taubin (2011) in order to progressively reduce the spatial resolution of the incoming video, so that reliable background models can be learned while the details of the scene are preserved in the higher resolution stages of the multiresolution pipeline.

An approach which extracts some hundreds of clusters which are called superpixels is proposed in Motamed (2017). Background models are learned for such superpixels, which reduces the computational burden of adapting the models online. The technique is designed for underwater videos. Another possibility is explored in Mukherjee et al. (2013), where a wavelet multiresolution decomposition of the incoming frame is carried out, followed by a modeling of the wavelet coefficients by Gaussian Mixture Models (GMMs).

On the other hand, recent papers use local binary patterns (LBP) obtained from the local neighborhood around each pixel. These patterns compare the intensity in the current frame and in the background model. With its inclusion as part of a foreground detection technique, the tolerance to lighting variations and robustness in multimodal background regions and shadows has been improved (Heikkilä and Pietikäinen, 2006). This idea has been extended in the so-called LBSP (*local binary similarity patterns*) with very interesting results in recent proposals (St-Charles et al., 2015; St-Charles et al., 2016).

Image interpolation techniques have also been applied effectively to improve the efficiency and complexity of foreground detection techniques (Xue et al., 2013). Among them, we can highlight the most usual ones such as bilinear or bicubic interpolation (Kok and Tam, 2019), which are used both for downsampling and upsampling the image. In (Jiang et al., 2013), a set of foreground sub-images, with different sizes and resolutions, has been considered to reduce the computational cost and improve the pedestrian detector which they propose. In the case of upsampling methods, super-resolution techniques might be useful (Dong et al., 2015), although its temporal complexity makes it currently unfeasible for real-time performance in foreground detection algorithms.

Nowadays, the emergence of deep learning networks has allowed its application in the field of foreground detection with more than satisfactory results (Bouwmans et al., 2019; Babaee et al., 2018; Minematsu et al., 2018). However, most proposals involve supervised training using the segmentation mask or ground truth of the scene in some frames, which differs from previous papers, which mostly have an unsupervised behavior. In this work we have only used unsupervised models for the evaluation of the proposed strategy.

3 Methodology

Next we define our proposed method for background modeling of high resolution video sequences. Let us consider a frame size of $N \times M$ pixels, so that the foreground object detection is meant to be carried out at such resolution. In order to reduce the computational complexity of the background model learning, we propose to maintain a model with a smaller spatial resolution. To this end, let us consider a feature function which maps each point in the input frame to a feature vector of size D:

$$\psi : [1, N] \times [1, M] \to \mathbb{R}^D \tag{1}$$

$$\mathbf{z} = \psi\left(\mathbf{x}\right) \tag{2}$$

where the values ψ are only known at the points with integer pixel coordinates, $\mathbf{x} \in \{1, ..., N\} \times \{1, ..., M\}$.

In order to reduce the spatial resolution a downsampling procedure must be carried out. This means that a background model is learned only at the following coordinates:

$$\mathcal{H} =$$

$$\left\{ \left(1 + i\frac{N-1}{n-1}, 1 + j\frac{M-1}{m-1} \right) \mid i \in \{0, ..., n-1\}, j \in \{0, ..., m-1\} \right\} \quad (3)$$

so that the background model contains $n \times m$ points, with $n < N$, $m < M$.

In order to estimate the feature vector at non integer pixel coordinates, an interpolation procedure is required. The simplest one is the nearest neighbor approach:

$$\psi_{NN}(\mathbf{x}) = \psi(\mathrm{round}(\mathbf{x})) \quad (4)$$

where round is the rounding function, applied componentwise to the coordinate vector $\mathbf{x}$.

Another option is window averaging over blocks of size $W \times W$ pixels:

$$\psi_{AVG}(\mathbf{x}) = \frac{1}{W^2} \sum_{\mathbf{y} \in \mathcal{N}(\mathbf{x})} \psi(\mathbf{y}) \quad (5)$$

where $\mathcal{N}(\mathbf{x})$ stands for the block of size $W \times W$ pixels which point $\mathbf{x}$ belongs to.

Other options include bilinear and bicubic interpolation:

$$\psi_{LIN}(\mathbf{x}) = \sum_{p,q \in \{0,1\}} a_{pq} x_1^p x_2^q \quad (6)$$

$$\psi_{CUB}(\mathbf{x}) = \sum_{p,q \in \{0,1,2,3\}} b_{pq} x_1^p x_2^q \quad (7)$$

where a_{pq} and b_{pq} are suitable bilinear and bicubic interpolation coefficients, respectively. More details can be found in (Keys, 1981; Press et al., 1992).

The estimate feature vectors at the points in $\mathcal{H}$ are then used to learn the background model at those points. The background model outputs the probabilities to belong to the background at those points:

$$\rho : [1, N] \times [1, M] \to [0, 1] \quad (8)$$

$$\rho(\mathbf{x}) = P(Back \mid \mathbf{x}) \quad (9)$$

where $\rho(\mathbf{x})$ stands for the probability to belong to the background at point $\mathbf{x}$, which is known for $\mathbf{x} \in \mathcal{H}$.

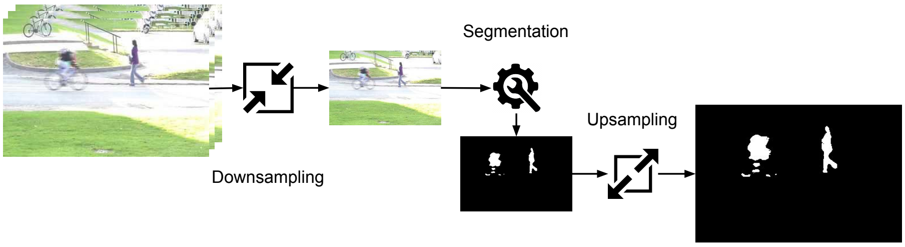


Fig. 1 Schema of the proposed methodology. Given an input frame, a downsampling process is carried out in order to obtain that frame resized with a lower frame size. After that, the execution of a segmentation method provides a foreground mask with the same frame size than the resized frame. Finally, an upsampling process resizes the foreground mask to achieve the same frame size than the original input frame.

Finally, an upsampling procedure is carried out to estimate the values of $\rho(\mathbf{x})$ for integer pixel coordinates, $\mathbf{x} \in \{1,...,N\} \times \{1,...,M\}$. For this purpose, bicubic interpolation is always used, since it yields the most accurate results:

$$\rho_{CUB}(\mathbf{x}) = \sum_{p,q \in \{0,1,2,3\}} c_{pq} x_1^p x_2^q \tag{10}$$

where c_{pq} are suitable bicubic interpolation coefficients. Please see (Keys, 1981; Press et al., 1992) for more details. Other less detailed interpolation methods like nearest neighbor and bilinear are not employed for the upsampling process because the final upsampled foreground mask must have the highest possible quality, and bicubic interpolation is the best option for this goal. The overall results are less dependent on the exactness of the downsampling process, so less detailed options like nearest neighbor and bilinear are considered.

Please note that our proposed procedure can be applied to any background model learning algorithm. Therefore, a different foreground detection algorithm is obtained by applying our proposal to each possible background model learning method.

The operation of the proposed methodology is described in Figure 1. The theoretical justification for our proposal can be outlined as follows. Natural images and videos have a high spatial redundancy, i.e. the observed pixel values are similar at nearby pixel locations most of the time. This means that the downsampling process does not remove a large amount of color information, so that the background models learned at a lower frame size are still relevant for the original frame. Moreover, the downsampling combines the information of neighboring pixels, which can have the beneficial side effect of removing some of the noise of the original video frame.

4 Experimental results

The experiments that have been carried out and their results are presented in this section. First of all, Subsection 4.1 details the software and the hardware used in the experiments and details the competitor methods. Then, Subsection 4.2 exhibits the tested video sequences. After that, the parameter selection is described in Subsection 4.3. Finally the results are reported in Subsection 4.4.

4.1 Methods

Regarding to the proposed methodology, several downsampling methods are considered, namely Nearest neighbor (NN), Bicubic interpolation (CUB), Bilinear interpolation (LIN) and Blockwise average (AVG). All of these methods are implemented in Matlab.

To test our approach, some well-known unsupervised segmentation methods have been considered. These selected methods, which are representative unsupervised algorithms with different kinds of background models, are: Wren (Wren et al., 1997), Grimson (Stauffer and Grimson, 1999), Zivkovic (Zivkovic, 2004), SOBS (Maddalena and Petrosino, 2008), SOBS_CF (Maddalena and Petrosino, 2010), SuBSENSE (St-Charles et al., 2014), LOBSTER (St-Charles and Bilodeau, 2014), PAWCS (St-Charles et al., 2016) and MFBM (López-Rubio and López-Rubio, 2015). All (except MFBM[1]) are implemented in BGS library[2] (Sobral, 2013).

The reported experiments have been carried out on a 64-bit Personal Computer with two Intel E5-2670 CPU with eight cores, 2.60 GHz per core, 32 GB RAM and standard hardware. The implementation of our method does not use any GPU resources, so it does not require any specific graphics hardware.

4.2 Sequences

The ChangeDetection.net (CDnet) 2012 dataset (Goyette et al., 2012) has been selected in order to test the proposal. It is composed by 31 videos organised into categories (Baseline (B), Dynamic Background (DB), Camera Jitter (CJ), Intermittent Object Motion (IOM), Shadow (S) and Thermal (TH)) in order to cover a wide range of different difficulties. This dataset is available in its website[3].

Additionally, the CAMO_UOW Dataset (Li et al., 2018) is used in the experiments. This dataset is formed by 10 high resolution videos captured in real scenes including both indoor and outdoor cases. 6 of its videos have a framesize of 1600 1200 pixels and they are recorded in the grayscale format; while the remaining 4 videos have a framesize of 1920 1080 pixels and they are recorded in the RGB format. It can be downloaded in its website[4].

[1] http://www.lcc.uma.es/~ezeqlr/index-en.html

[2] https://github.com/andrewssobral/bgslibrary

[3] http://changedetection.net/

[4] https://documents.uow.edu.au/~wanqing/#Datasets

Parameter		Values
Downsampling method	=	{NN, CUB, LIN, AVG}
Resize factor	=	$\{0.875, 0.75, 0.625, 0.5, 0.375, 0.25, 0.125\}$
Segmentation method	=	{MFBM, Wren, Grimson, Zivkovic, SOBS, SOBS_CF, SuBSENSE, LOBSTER, PAWCS}

Table 1 Considered parameter values which form the set of tuned configurations.

4.3 Parameter selection

The value of the different parameters of each competitor segmentation method is those recommended by their authors or the default value presented in the BGS library. Regarding to the parameters of the proposed approach, so that, the downsampling method and the resize factor, we have considered a wide range of values. These values are shown in Table 1. Additionally, we note as ORIG (or resize factor equals 1) when our proposal is not applied, so that, the considered frame size is the original size.

4.4 Results

The goal of this work is to study how the different downsampling methods and the selected resize factor affect to the foreground mask generated by the segmentation method. The employed memory and the execution time are also studied.

4.4.1 CDnet 2012

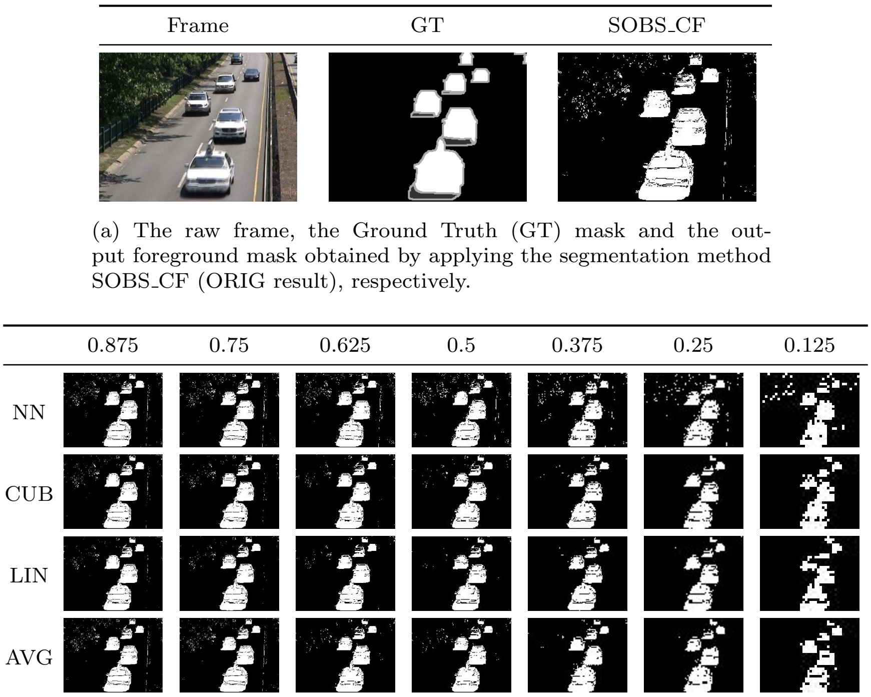
This subsection depicts the obtained results considering the whole CDnet 2012 dataset.

First of all, we have compared of the results from a qualitative point of view. Some of the obtained results are shown in Figures 2 and 3. The most relevant information is that the lower the resize factor the lower the result is detailed. It is shown how the detected foreground objects assume a squared shape with a low resize factor, even they can be removed. However, the false positives rate is reduced.

In addition, a comparison has been carried out from a quantitative point of view. We have considered the F-measure (F-m) as the measure in order to compare the quality of the foreground masks. This measure considers both the precision (PR) and the recall (RC, also called sensitivity) which are computed from the true positives (TP), false positives (FP) and false negatives (FN). The F-measure is commonly used in literature and it provides a value in the range between 0 and 1, where higher is better. Its definition is as follows:

$$\text{F-measure} = 2 * \frac{PR * RC}{PR + RC} \tag{11}$$

Fig. 2 Qualitative results for a benchmark scene corresponding to the frame 1642 of the video *Highway* by applying the segmentation method SOBS_CF.



(a) The raw frame, the Ground Truth (GT) mask and the output foreground mask obtained by applying the segmentation method SOBS_CF (ORIG result), respectively.

(b) Output masks generated by the approach after applying the segmentation method SOBS_CF considering the different tuned configurations for the downsampling method (first column) and the resize factor parameter (first row).

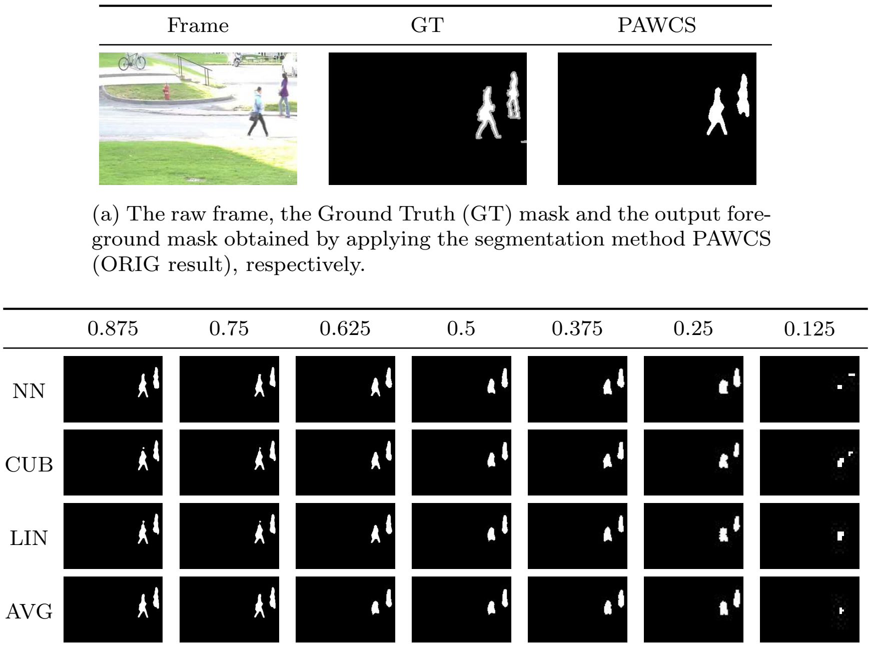
where

$$RC = \frac{TP}{TP + FN} \qquad PR = \frac{TP}{TP + FP} \tag{12}$$

Another considered measure in this study is the execution time. In this case, we have used the frames per second rate (fps), which is a positive real number, and higher is better. The maximum memory employed (in KBytes) by each approach is also considered. It is a positive number where lower is better.

Figure 4 shows the average F-measure achieved by each segmentation method considering the different tuned configurations. It can be observed that the conclusions of the qualitative comparison are further confirmed in this figure: the lower the resize factor the lower the foreground detection performance. It is interesting to see how the quality of the result of several segmentation methods degrades gently as the resize factor decreases. For example, this is the case of the segmentation methods MFBM or Zivkovic. On the other hand, other segmentation methods such as PAWCS or SuBSENSE are heavily affected when a low resize factor is used.

Fig. 3 Qualitative results for a benchmark scene corresponding to the frame 1642 of the video *Pedestrians* by applying the segmentation method PAWCS.



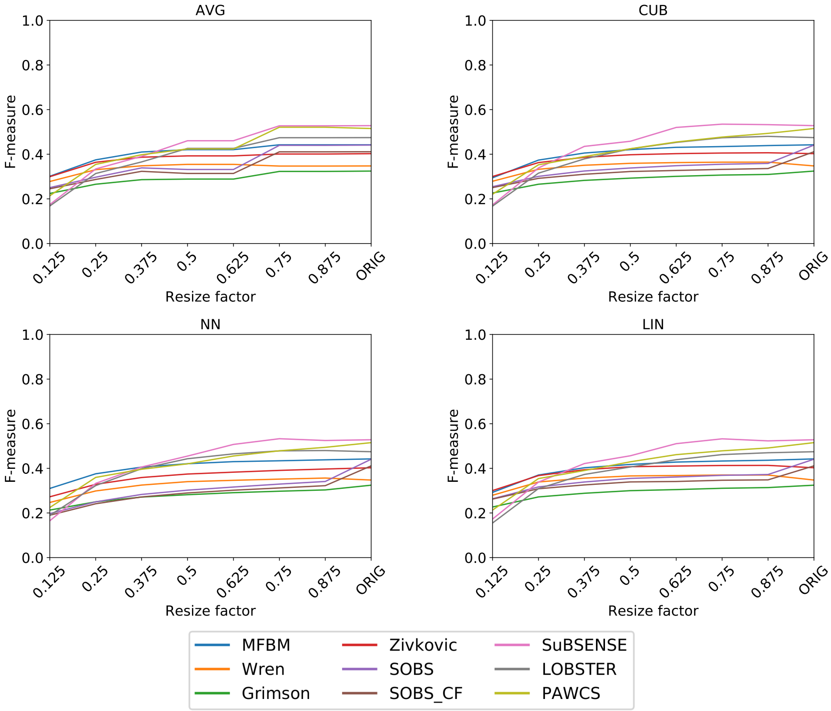
(a) The raw frame, the Ground Truth (GT) mask and the output foreground mask obtained by applying the segmentation method PAWCS (ORIG result), respectively.

(b) Output masks generated by the approach after applying the segmentation method PAWCS considering the different tuned configurations for the downsampling method (first column) and the resize factor parameter (first row).

The average fps is shown in Figure 5. In general, the use of the downsampling-upsampling process is faster than the traditional schema; only two methods (Wren and Zivkovic) are slower by applying the proposed methodology. The explanation of the behaviour of these two methods is that their complexity is very low (and much lower than the other selected methods) so they are very fast. Because of this, the reduction of the execution time of both methods with the downsampled frames is not significant respecting to the ORIG execution time. However, the addition of the proposed downsampling-upsampling methodology has a time consumption that increases the whole execution time over the ORIG time. In addition, the lower the resize factor the faster the proposal.

On the other hand, the average maximum memory used is reported in Figure 6. In general, ORIG configurations yield a better memory performance due to the memory requirements of the downsampling process. Furthermore, from a certain value of the resize factor parameter to lower values, the memory used by all the segmentation methods remains constantly. Again, the downsampling process fixes the memory used.

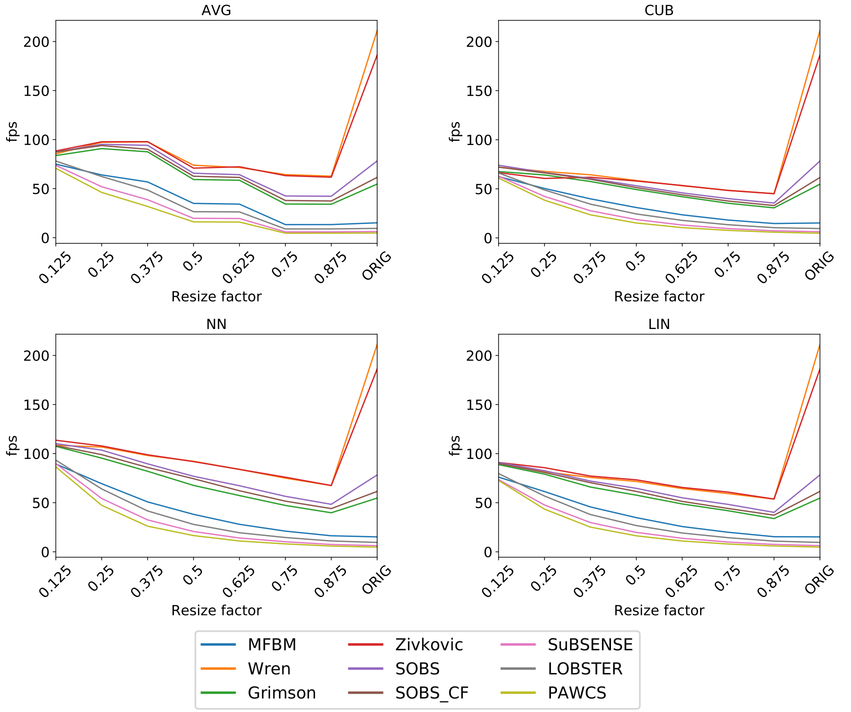
Fig. 4 Average F-measure in the whole dataset for each method (the higher, the better). Note that the values of each method are connected between them with lines to better compare the results, but this does not mean that the results are related.



In order to compare the quality of each considered downsampling method, the average of the performance of all tested segmentation methods are depicted in Figure 7. The best downsampling method according to its yielded F-measure (top left image) is LIN, while NN obtains the worst F-measure performance. Additionally, the use of a resize factor value lower than 0.5 provides similar performances with LIN, CUB and AVG. Regarding to the fps measure (top right image), LIN is the fastest, while CUB is the slowest. From the used memory point of view, NN, LIN and CUB have a similar waste of memory. On the other hand, AVG employs more memory when the value of the resize factor parameter is high, while its memory requirements are similar when the resize factor parameter is low.

The results can be analyzed in more detail by selecting a particular video. Table 2 exhibits the F-measure and fps performances yielded by different foreground methods in the video PETS2006 from the category Baseline. This category contains simple videos with no serious difficulties and the peculiarity of this video is that its frame size is the highest of the CDnet 2012 dataset (576x720 pixels). For each foreground method and resize factor, the mean performance achieved by the downsampling methods is shown. The variation of

Fig. 5 Average fps in the whole dataset for each method (the higher, the better). Note that the values of each method are connected between them with lines to better compare the results, but this does not mean that the results are related.



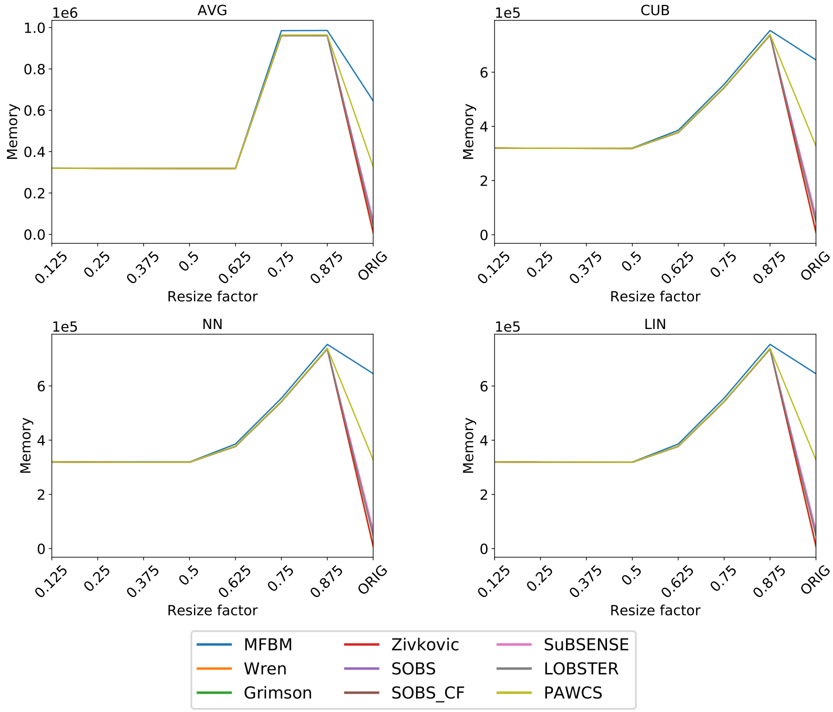
F-measure and the variation of the fps yielded by each tuned configuration (columns F-M Var and fps Var) have been calculated relative to the obtained by the ORIG configuration. The definition of the variation (where the higher the better) is as follows: given the performance values (for example, for the execution time) achieved by ORIG and the tuned configuration p_{ORIG} and p_{DSU}, respectively, the variation is:

$$\text{Variation} = 100\frac{p_{DSU} - p_{ORIG}}{p_{ORIG}} \tag{13}$$

Note that a positive value for the variation indicates that the performance of the tuned configuration is better than to the ORIG performance, while a negative variation means a worse performance than the achieved by ORIG.

Most foreground algorithms improve their fps performance by employing the proposed downsampling-upsampling methodology while they deteriorate their F-measure. Additionally, the increase in fps is more pronounced than the F-measure loss in the more recent algorithms such as PAWCS or LOBSTER. This can be seen by observing the columns of the variation in Table 2. Again, it is interesting to observe the performance achieved by Wren and

Fig. 6 Average maximum memory used (KBytes) in the whole dataset for each method (the lower, the better). Note that the values of each method are connected between them with lines to better compare the results, but this does not mean that the results are related.



Zivkovic. The use of the proposed methodology affects negatively to their fps. However, their F-measure is improved. As we have previously indicated, these two foreground methods are not so complex. In addition, the false positives rate is reduced when the proposed methodology is applied. This way, the downsampling-upsampling strategy adopts a postprocessing technique role for non complex foreground detection algorithms.

Moreover, Table 3 shows the total execution time and the execution time per step (downsampling, segmentation and upsampling) yielded by different foreground methods in the video PETS2006. As it was previously indicated, the time reduction is heavily pronounced in the more recent algorithms such as PAWCS or LOBSTER. It is interesting to observe how the lower the resize factor, the higher the percentage of the downsampling step and the lower the percentage of the segmentation step respecting to the whole execution time.

It must be highlighted that the execution time includes the time required for the downsampling, plus the segmentation and upsampling times. The upsampling process is carried out in order to obtain a foreground mask with the same size than the original input frame. Please note that the purpose of this upsampling process is only to establish a fair comparison between the

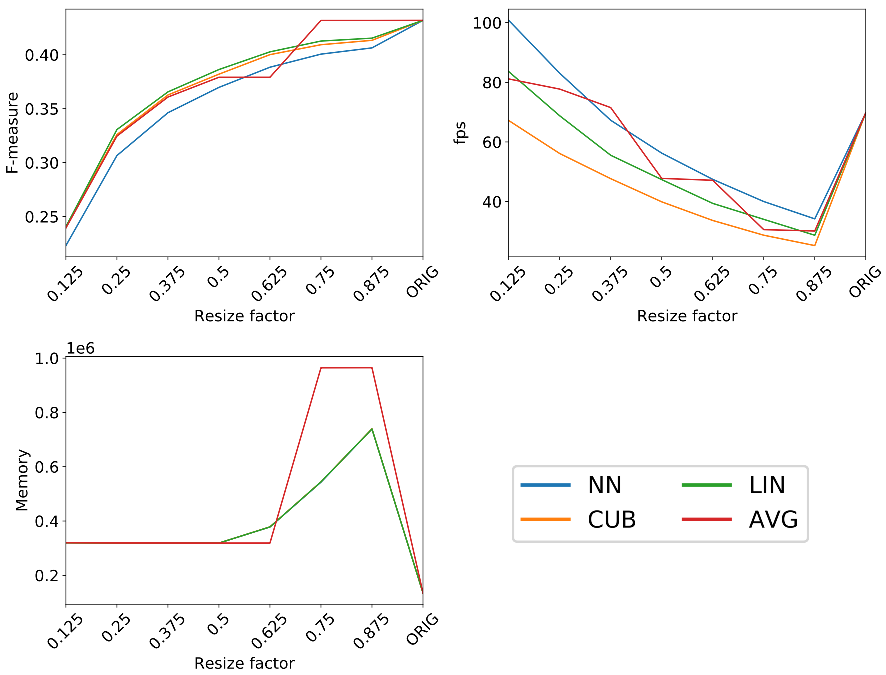
Method	RF	F-M	F-M Var	fps	fps Var
MFBM	ORIG	0.83	-	3.82	-
	0.875	0.77	-7.35	3.93	2.72
	0.750	0.77	-7.56	4.80	25.52
	0.625	0.74	-11.40	7.65	100.05
	0.500	0.72	-13.38	10.01	161.66
	0.375	0.70	-16.23	15.09	294.50
	0.250	0.62	-24.81	21.78	469.49
	0.125	0.49	-41.17	29.63	674.72
Wren	ORIG	0.65	-	50.71	-
	0.875	0.74	13.12	18.61	-63.30
	0.750	0.74	13.06	19.72	-61.10
	0.625	0.76	15.54	22.92	-54.80
	0.500	0.74	13.40	26.53	-47.67
	0.375	0.70	7.49	31.86	-37.16
	0.250	0.64	-1.89	35.15	-30.67
	0.125	0.48	-26.38	34.84	-31.29
Grimson	ORIG	0.55	-	12.96	-
	0.875	0.56	1.07	9.65	-25.54
	0.750	0.55	-0.62	11.07	-14.62
	0.625	0.51	-7.19	15.38	18.62
	0.500	0.49	-11.86	18.78	44.85
	0.375	0.45	-18.84	24.87	91.90
	0.250	0.39	-30.14	30.54	135.64
	0.125	0.28	-48.48	32.91	153.90
Zivkovic	ORIG	0.81	-	44.44	-
	0.875	0.85	5.98	17.82	-59.91
	0.750	0.85	5.21	19.32	-56.53
	0.625	0.84	3.95	22.70	-48.93
	0.500	0.82	1.54	26.27	-40.89
	0.375	0.77	-4.47	31.63	-28.84
	0.250	0.69	-14.31	33.77	-24.01
	0.125	0.51	-36.12	34.30	-22.83
SOBS	ORIG	0.85	-	18.15	-
	0.875	0.80	-6.40	12.12	-33.24
	0.750	0.78	-7.80	13.55	-25.36
	0.625	0.72	-15.34	17.72	-2.40
	0.500	0.68	-19.51	21.40	17.87
	0.375	0.64	-24.83	27.43	51.12
	0.250	0.53	-37.58	32.41	78.55
	0.125	0.37	-55.99	34.05	87.57
SOBS_CF	ORIG	0.84	-	15.40	-
	0.875	0.80	-4.45	10.74	-30.23
	0.750	0.79	-6.24	12.07	-21.65
	0.625	0.73	-12.48	16.45	6.84
	0.500	0.71	-15.31	19.88	29.09
	0.375	0.66	-20.82	26.31	70.87
	0.250	0.57	-31.55	31.59	105.13
	0.125	0.42	-50.23	33.67	118.67
SuBSENSE	ORIG	0.95	-	1.38	-
	0.875	0.91	-4.08	1.68	22.07
	0.750	0.90	-5.79	2.14	55.04
	0.625	0.84	-11.65	3.70	168.34
	0.500	0.81	-15.22	4.98	261.35
	0.375	0.76	-20.67	8.51	517.73
	0.250	0.69	-28.13	14.37	943.43
	0.125	0.18	-81.48	25.45	1747.69
LOBSTER	ORIG	0.93	-	2.15	-
	0.875	0.92	-1.01	2.45	14.19
	0.750	0.91	-1.85	3.11	44.76
	0.625	0.88	-5.28	5.23	143.59
	0.500	0.86	-7.17	6.96	223.88
	0.375	0.82	-11.67	11.14	418.47
	0.250	0.72	-23.20	17.63	720.58
	0.125	0.20	-78.94	28.07	1206.61
PAWCS	ORIG	0.94	-	0.96	-
	0.875	0.91	-3.16	1.20	24.94
	0.750	0.90	-4.62	1.58	65.58
	0.625	0.85	-9.63	2.93	205.75
	0.500	0.82	-12.33	4.00	318.07
	0.375	0.77	-17.89	7.08	640.16
	0.250	0.67	-28.96	12.40	1195.71
	0.125	0.19	-80.08	24.24	2433.81

Table 2 Average performances for each segmentation method in the video PETS2006 from category Baseline. Each column shows the selected foreground method (Method), the applied resize factor (RF), the F-measure (F-M), the variation of F-measure (F-M Var) respecting to the obtained by ORIG, the fps and the variation of fps (fps Var) respecting to the obtained by ORIG, respectively, where the higher the better. Each row shows the average performance achieved by the downsampling methods.

Method	RF	Total Time	t_D	t_S	t_U	% t_D	% t_S	% t_U
MFBM	ORIG	313.78	0.00	313.78	0.00	0.00	100.00	0.00
	0.875	307.70	35.68	258.53	13.49	11.60	84.02	4.38
	0.750	260.34	33.61	211.42	15.30	12.91	81.21	5.88
	0.625	161.06	33.96	114.02	13.08	21.09	70.80	8.12
	0.500	120.81	31.21	79.57	10.03	25.84	65.86	8.30
	0.375	81.38	27.80	45.00	8.57	34.17	55.30	10.53
	0.250	56.08	25.73	21.66	8.69	45.88	38.62	15.50
	0.125	41.69	27.00	7.22	7.47	64.77	17.31	17.92
Wren	ORIG	23.67	0.00	23.67	0.00	0.00	100.00	0.00
	0.875	67.04	35.68	17.90	13.46	53.22	26.70	20.08
	0.750	63.56	33.61	15.07	14.88	52.88	23.71	23.41
	0.625	55.21	33.96	8.28	12.97	61.52	15.00	23.49
	0.500	47.34	31.21	6.42	9.71	65.93	13.55	20.52
	0.375	40.47	27.80	4.05	8.62	68.70	10.01	21.30
	0.250	36.03	25.73	2.52	7.78	71.41	7.00	21.59
	0.125	35.99	27.00	1.84	7.14	75.03	5.12	19.85
Grimson	ORIG	92.58	0.00	92.58	0.00	0.00	100.00	0.00
	0.875	125.74	35.68	76.26	13.80	28.38	60.65	10.98
	0.750	111.77	33.61	63.21	14.94	30.07	56.56	13.37
	0.625	80.22	33.96	33.70	12.56	42.33	42.01	15.66
	0.500	65.37	31.21	24.65	9.51	47.75	37.71	14.54
	0.375	50.47	27.80	14.00	8.67	55.09	27.74	17.17
	0.250	40.78	25.73	7.15	7.91	63.09	17.52	19.39
	0.125	37.79	27.00	2.88	7.90	71.46	7.63	20.92
Zivkovic	ORIG	27.00	0.00	27.00	0.00	0.00	100.00	0.00
	0.875	69.79	35.68	20.49	13.61	51.13	29.36	19.51
	0.750	65.36	33.61	17.00	14.75	51.43	26.00	22.57
	0.625	55.58	33.96	9.39	12.23	61.10	16.89	22.01
	0.500	47.99	31.21	7.19	9.60	65.03	14.97	20.00
	0.375	40.77	27.80	4.33	8.63	68.20	10.63	21.17
	0.250	37.83	25.73	3.04	9.07	68.01	8.02	23.97
	0.125	37.05	27.00	1.61	8.43	72.89	4.34	22.77
SOBS	ORIG	66.11	0.00	66.11	0.00	0.00	100.00	0.00
	0.875	100.61	35.68	51.91	13.02	35.47	51.60	12.94
	0.750	91.22	33.61	42.78	14.83	36.85	46.90	16.26
	0.625	70.16	33.96	23.37	12.82	48.41	33.32	18.28
	0.500	57.92	31.21	17.20	9.51	53.89	29.69	16.42
	0.375	46.30	27.80	10.08	8.42	60.05	21.78	18.18
	0.250	38.69	25.73	4.99	7.97	66.50	12.89	20.61
	0.125	36.63	27.00	2.51	7.12	73.72	6.84	19.44
SOBS_CF	ORIG	77.93	0.00	77.93	0.00	0.00	100.00	0.00
	0.875	113.34	35.68	64.64	13.01	31.48	57.04	11.48
	0.750	102.23	33.61	53.56	15.06	32.88	52.39	14.73
	0.625	75.44	33.96	29.19	12.29	45.02	38.70	16.29
	0.500	61.96	31.21	21.21	9.54	50.37	34.23	15.39
	0.375	48.05	27.80	11.89	8.35	57.87	24.75	17.39
	0.250	39.58	25.73	6.08	7.77	65.00	15.36	19.64
	0.125	37.14	27.00	2.82	7.32	72.70	7.58	19.71
SuBSENSE	ORIG	871.22	0.00	871.22	0.00	0.00	100.00	0.00
	0.875	720.05	35.68	668.67	15.70	4.96	92.86	2.18
	0.750	589.96	33.61	541.50	14.85	5.70	91.79	2.52
	0.625	335.22	33.96	288.47	12.79	10.13	86.05	3.82
	0.500	241.61	31.21	201.05	9.34	12.92	83.21	3.87
	0.375	143.34	27.80	107.03	8.51	19.40	74.67	5.93
	0.250	84.33	25.73	50.82	7.78	30.51	60.26	9.22
	0.125	48.42	27.00	13.35	8.06	55.77	27.58	16.65
LOBSTER	ORIG	558.64	0.00	558.64	0.00	0.00	100.00	0.00
	0.875	492.44	35.68	443.61	13.15	7.25	90.08	2.67
	0.750	402.24	33.61	353.43	15.21	8.36	87.86	3.78
	0.625	235.63	33.96	189.32	12.34	14.41	80.35	5.24
	0.500	172.90	31.21	132.20	9.49	18.05	76.46	5.49
	0.375	109.54	27.80	73.23	8.50	25.38	66.86	7.76
	0.250	69.00	25.73	35.23	8.04	37.29	51.06	11.65
	0.125	43.92	27.00	9.83	7.09	61.48	22.38	16.14
PAWCS	ORIG	1254.13	0.00	1254.13	0.00	0.00	100.00	0.00
	0.875	1016.73	35.68	967.67	13.38	3.51	95.17	1.32
	0.750	824.41	33.61	776.28	14.52	4.08	94.16	1.76
	0.625	426.39	33.96	380.22	12.20	7.96	89.17	2.86
	0.500	300.23	31.21	259.64	9.38	10.40	86.48	3.13
	0.375	172.13	27.80	135.99	8.33	16.15	79.00	4.84
	0.250	97.45	25.73	63.88	7.84	26.40	65.55	8.05
	0.125	50.50	27.00	16.44	7.06	53.47	32.56	13.97

Table 3 Average performances for each segmentation method in the video PETS2006 from category Baseline. Each column shows the selected foreground method (Method), the applied resize factor (RF), the total execution time (Time, in seconds, where the lower the better), the execution time of each step (in seconds, downsampling t_D, segmentation t_S and upsampling t_U, where the lower the better) and its relation in percentage to the total execution time ($\%t_D$, $\%t_S$ and $\%t_U$), respectively. Each row shows the average performance achieved by the downsampling methods.

Fig. 7 Average performance of all considered segmentation methods in the whole dataset for each downsampling method. Note that the values of each method are connected together with lines to better compare the results, but this does not mean that the results are related.



groundtruth frame provided by the CDnet dataset and the foreground mask generated by each segmentation method. Thus, the upsampling method could be removed from the proposed methodology in order to obtain a faster system.

In addition, it should be pointed out the high number of experiments that we have run: each segmentation method (we have selected 9) has been executed 28 times according to the different tuned configurations (4 downsampling methods and 7 different values for the resize factor parameter) across the whole CDnet 2012 dataset (which is formed by 31 videos).

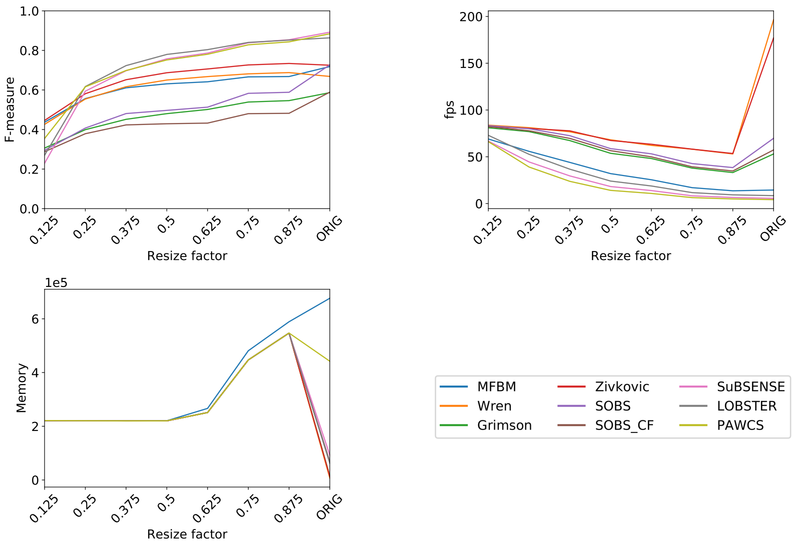
4.4.2 Baseline and Baseline High Resolution

Moreover, we have studied in more detail the results obtained by considering only those videos that belong to the category Baseline.

As it can be observed in Figure 8, the F-measure performance is higher than the presented in the whole dataset, the fps remains practically with the same performance while the memory employed by all tested configurations is lower.

However, the issue of the CDnet 2012 dataset videos is that their frame sizes are quite smaller than the size of the frames which belong to high resolution

Fig. 8 Average performances in the category Baseline for each segmentation method. First row exhibits the F-measure and the fps (the higher, the better), while second row shows the maximum memory used (in KBytes, where the lower the better). Note that the values of each method are connected between them with lines to better compare the results, but this does not mean that the results are related.

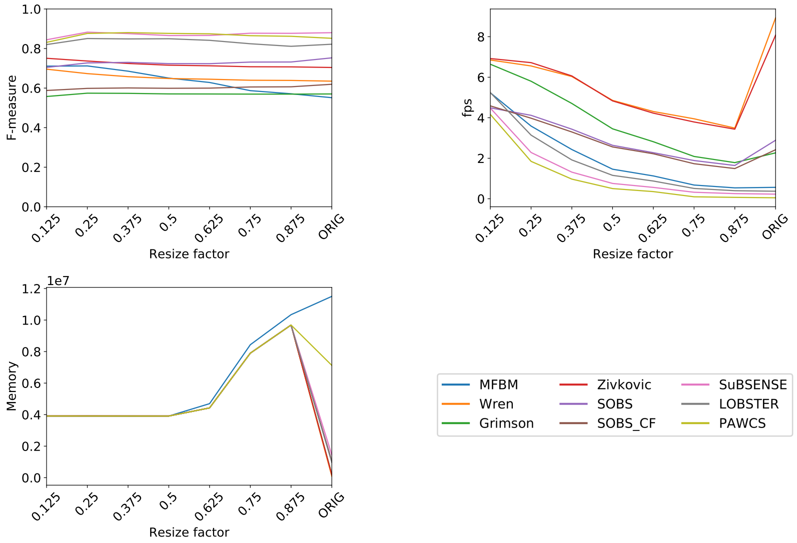


videos. In order to carry out the proposed methodology with larger frame size videos, we have built a high resolution version of the videos which compose the Baseline category. The high resolution frames corresponding to the input and the groundtruth videos are obtained by employing a superresolution (SR) method. The selected SR method is those presented in (López-Rubio, 2016). The four Baseline High Resolution videos generated after the application of the SR process are: Highway (1440x1920 pixels), Office (1280x1920), pedestrians (1280x1920) and PETS2006 (1536x1920).

The average performances of the experiments that we have carried out are shown in Figure 9. It is remarkable to see how our proposal, in general, yields a similar performance than ORIG. Regarding to the fps measure, in this case we do not have included the upsampling execution time. In general, the proposal obtains a lower execution time than ORIG. Nevertheless, the proposal wastes a higher quantity of memory than ORIG due to the effort that it employs in the downsampling process.

An overview of the performance of each segmentation method according to its F-measure and its frame rate can be observed in Figure 10. The figure shows how our methodology is able to improve the frames per second rate for almost all tested methods (all but Wren and Zivkovic) and F-measure is also improved for PAWCS, LOBSTER, SuBSENSE, Zivkovic, Wren and MFBM methods.

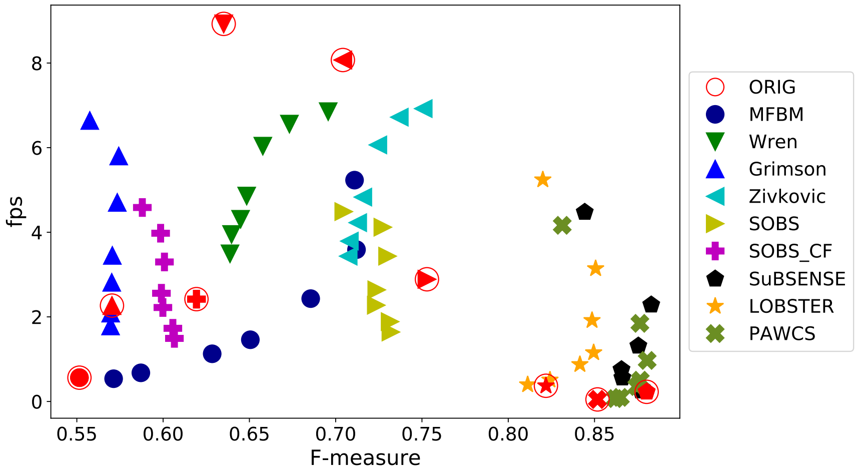
Fig. 9 Average performances in the category Baseline High Resolution for each segmentation method. First row exhibits the F-measure and the fps (the higher, the better), while second row shows the maximum memory used (in KBytes, where the lower the better). Note that the values of each method are connected between them with lines to better compare the results, but this does not mean that the results are related.



A comparison of these obtained results in the category Baseline High Resolution with the results of the category Baseline is done. It can be said that a high resolution frame can be downsampled by using the proposed methodology and the quality of the generated foreground mask will be similar to ORIG except for the smallest resize factors, where the proposal offers a poor performance. The obtained fps rate depends on the foreground detection algorithm. For the simplest foreground detection algorithms, like Wren and Zivkovic, the downsampled fps is lower (worse) when the original input frame is high resolution format. Even though the segmentation method receives a resized frame as input and it takes a low execution time, the downsampling process needs a large amount of computation time to reduce the size of the high resolution frame, as compared to the execution time of those simple foreground segmentation algorithms. On the other hand, for the more complex and more accurate foreground segmentation methods like SuBSENSE and PAWCS, our downsampling proposal yields a higher (better) fps than processing the original video frames. This is because these more complex foreground segmentation methods require a much higher computational load than the downsampling procedure. In terms of the used memory, it can be said that the downsampling process requires a large amount of memory.

It is interesting to go deeper into this comparison between the categories Baseline and Baseline High Resolution. According to the results obtained and the subsequent analysis in the CDnet 2012 dataset comparative, we have ex-

Fig. 10 Average performance for Baseline High Resolution with each segmentation method (better while closer to the upper right corner). Horizontal axis shows F-measure and vertical axis shows the frame rate. Each symbol form represents a segmentation method. There are as many symbols with the same form as configurations of our proposal (different resize factors). Symbols with red color represents the performance of the ORIG configuration (raw segmentation method).



amined the performances of foreground detection methods PAWCS, MFBM (both methods with a high complexity) and Wren (low complexity). Figures 12, 13 and 14 report their F-measure and execution time performances. In Figures 12 and 13 it can be observed how the use of the proposed methodology does not require heavy computations, and how the overall CPU time is reduced drastically. Although the F-measure is heavily affected in the category Baseline according as the applied downsampling, the performances of both methods remain similar in Baseline High Resolution to the obtained in Baseline by ORIG. On the other hand, a foreground method with low complexity such as Wren does not improve its execution time by using the proposed methodology. This behavior is shown in Figure 14.

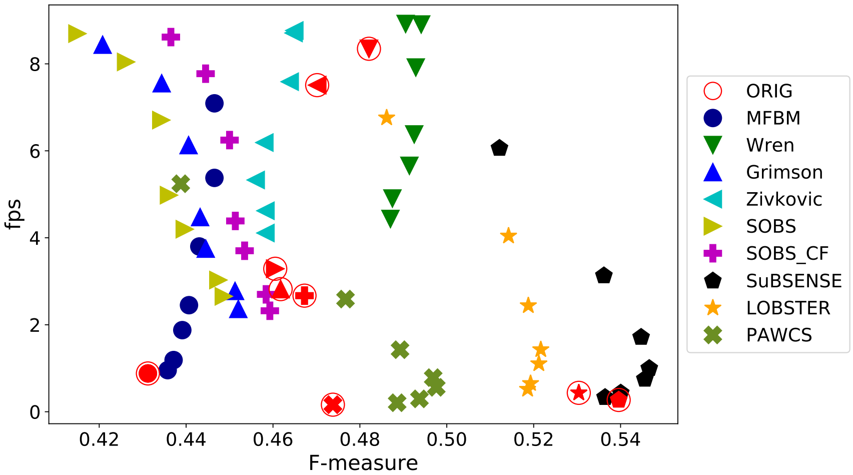
4.4.3 CAMO_UOW

This subsection depicts the obtained results considering the whole CAMO_UOW dataset, which is composed by high resolution videos.

The experiments carried out with this dataset are quite similar to the obtained ones in the previous subsection with the other high resolution videos considered.

Figure 11 reports the average F-measure and fps performances of each method in CAMO_UOW dataset. Most foreground algorithms enhance their fps performance when they apply the downsampling-upsampling strategy; however, the F-measure remains with practically the same performance than

Fig. 11 Average performance for CAMO_UOW with each segmentation method (better while closer to the upper right corner). Horizontal axis shows F-measure and vertical axis shows the frame rate. Each symbol form represents a segmentation method. There are as many symbols with the same form as configurations of our proposal (different resize factors). Symbols with red color represents the performance of the ORIG configuration (raw segmentation method).



ORIG. The symbols of a segmentation method can be considered as a Pareto front. Thus, it is possible to observe that for some methods such as MFBM, Wren or PAWCS, the original performance (in red) is dominated by some Pareto front symbols, which implies that our proposal clearly improves the performance in high resolution videos. These performances can be analyzed in more detail as it was done in previous subsections. For example, Tables 4 and 5 report the F-measure and fps performances and execution time of each step of the applied methodology for each foreground method when video10 is carried out, respectively. As it can be observed, the execution time of recent foreground algorithms is strongly reduced.

Additionally, Figure 15 exhibits a comparison between a high complexity foreground method (MFBM in left column) and a low complexity method (Wren in right column) in the CAMO_UOW dataset. It can be observed how the high complexity method is benefited by the application of the proposed methodology. The reduction of its execution time is drastically pronounced while its accuracy is very similar to ORIG. Nevertheless, the CPU time is not enhanced when the strategy is applied to a low complexity method.

5 Conclusions

In this paper a metamodel for the enhancement of motion detection algorithms is presented, with the aim of improving execution in real time for high

Method	RF	F-M	F-M Var	fps	fps Var
MFBM	1.000	0.37	-	0.80	-
	0.875	0.37	1.32	0.85	6.07
	0.750	0.38	2.87	1.04	29.90
	0.625	0.39	6.09	1.66	106.95
	0.500	0.40	7.91	2.18	172.10
	0.375	0.41	11.12	3.37	320.44
	0.250	0.42	14.76	4.87	508.15
	0.125	0.43	16.26	6.28	683.49
Wren	1.000	0.46	-	8.52	-
	0.875	0.46	1.11	4.32	-49.32
	0.750	0.46	1.20	4.84	-43.24
	0.625	0.46	1.81	5.14	-39.72
	0.500	0.47	2.05	6.08	-28.61
	0.375	0.46	1.74	7.78	-8.75
	0.250	0.46	1.85	8.60	0.89
	0.125	0.46	0.51	8.21	-3.69
Grimson	1.000	0.42	-	2.91	-
	0.875	0.40	-2.95	2.36	-18.68
	0.750	0.40	-3.22	2.72	-6.31
	0.625	0.40	-4.79	3.53	21.53
	0.500	0.39	-5.33	4.35	49.81
	0.375	0.39	-5.95	5.99	106.08
	0.250	0.38	-7.71	7.27	150.09
	0.125	0.37	-10.90	7.81	168.79
Zivkovic	1.000	0.47	-	7.48	-
	0.875	0.47	-0.10	3.99	-46.60
	0.750	0.47	-0.07	4.58	-38.76
	0.625	0.47	0.07	4.86	-35.00
	0.500	0.47	0.41	5.88	-21.43
	0.375	0.47	0.57	7.29	-2.50
	0.250	0.47	0.26	8.34	11.42
	0.125	0.47	-1.27	8.06	7.69
SOBS	1.000	0.45	-	3.20	-
	0.875	0.44	-0.75	2.63	-17.65
	0.750	0.44	-0.85	3.08	-3.49
	0.625	0.44	-1.47	3.92	22.66
	0.500	0.44	-1.70	4.86	52.23
	0.375	0.44	-2.16	6.53	104.51
	0.250	0.43	-3.36	7.79	143.70
	0.125	0.42	-6.00	8.12	154.14
SOBS_CF	1.000	0.45	-	2.54	-
	0.875	0.45	-0.39	2.27	-10.82
	0.750	0.45	-0.44	2.65	4.17
	0.625	0.45	-0.78	3.63	42.80
	0.500	0.45	-0.91	4.42	73.75
	0.375	0.45	-1.40	5.95	134.22
	0.250	0.44	-2.45	7.50	194.96
	0.125	0.43	-4.45	7.83	208.22
SuBSENSE	1.000	0.50	-	0.28	-
	0.875	0.50	0.06	0.34	22.50
	0.750	0.50	0.00	0.43	56.62
	0.625	0.50	-0.34	0.72	162.52
	0.500	0.49	-0.67	0.98	254.65
	0.375	0.49	-1.43	1.66	504.40
	0.250	0.48	-2.96	3.00	989.23
	0.125	0.46	-7.09	5.67	1960.47
LOBSTER	1.000	0.49	-	0.41	-
	0.875	0.49	0.32	0.49	19.70
	0.750	0.49	0.41	0.63	52.68
	0.625	0.49	0.57	1.04	153.16
	0.500	0.49	0.48	1.39	237.85
	0.375	0.49	-0.51	2.32	463.02
	0.250	0.48	-1.29	3.83	829.99
	0.125	0.46	-5.37	6.20	1407.72
PAWCS	1.000	0.49	-	0.11	-
	0.875	0.48	-0.85	0.14	30.85
	0.750	0.49	-0.33	0.24	122.22
	0.625	0.49	0.81	0.51	381.99
	0.500	0.49	-0.10	0.69	550.70
	0.375	0.49	-0.05	1.32	1140.17
	0.250	0.48	-0.73	2.38	2131.97
	0.125	0.46	-5.11	4.87	4457.29

Table 4 Average performances for each segmentation method in the video10 from CAMO_UOW. Each column shows the selected foreground method (Method), the applied resize factor (RF), the F-measure (F-M), the variation of F-measure (F-M Var) respecting to the obtained by ORIG, the fps and the variation of fps (fps Var) respecting to the obtained by ORIG, respectively, where the higher the better. Each row shows the average performance achieved by the downsampling methods.

Method	RF	Total Time	t_D	t_S	t_U	$\%t_D$	$\%t_S$	$\%t_U$
MFBM	1.000	574.02	0.00	574.02	0.00	0.00	100.00	0.00
	0.875	544.21	60.55	459.24	24.42	11.13	84.39	4.49
	0.750	473.84	56.59	394.91	22.34	11.94	83.34	4.71
	0.625	282.73	62.10	197.01	23.62	21.96	69.68	8.36
	0.500	212.94	53.60	136.95	22.39	25.17	64.32	10.51
	0.375	140.19	45.73	72.58	21.88	32.62	51.77	15.61
	0.250	96.77	43.74	33.74	19.28	45.20	34.87	19.93
	0.125	76.78	49.78	9.26	17.74	64.84	12.06	23.11
Wren	1.000	53.99	0.00	53.99	0.00	0.00	100.00	0.00
	0.875	114.17	60.55	31.50	22.12	53.04	27.59	19.37
	0.750	103.85	56.59	25.94	21.32	54.49	24.98	20.53
	0.625	92.11	62.10	14.93	15.08	67.42	16.21	16.37
	0.500	81.19	53.60	11.14	16.46	66.02	13.72	20.27
	0.375	63.90	45.73	6.71	11.46	71.57	10.50	17.93
	0.250	57.34	43.74	3.69	9.91	76.28	6.43	17.29
	0.125	60.26	49.78	1.54	8.93	82.61	2.56	14.82
Grimson	1.000	158.29	0.00	158.29	0.00	0.00	100.00	0.00
	0.875	199.51	60.55	115.81	23.15	30.35	58.05	11.60
	0.750	175.66	56.59	97.94	21.12	32.22	55.76	12.03
	0.625	132.09	62.10	53.73	16.26	47.01	40.68	12.31
	0.500	109.15	53.60	37.74	17.82	49.10	34.57	16.32
	0.375	80.43	45.73	20.71	13.99	56.86	25.74	17.40
	0.250	66.72	43.74	10.77	12.21	65.56	16.15	18.30
	0.125	64.01	49.78	3.50	10.72	77.77	5.47	16.75
Zivkovic	1.000	61.49	0.00	61.49	0.00	0.00	100.00	0.00
	0.875	120.98	60.55	37.58	22.85	50.05	31.07	18.89
	0.750	107.57	56.59	30.94	20.04	52.61	28.76	18.63
	0.625	96.80	62.10	17.92	16.78	64.15	18.51	17.33
	0.500	84.06	53.60	13.34	17.12	63.76	15.87	20.37
	0.375	67.37	45.73	7.14	14.50	67.88	10.60	21.52
	0.250	58.85	43.74	4.02	11.09	74.32	6.84	18.84
	0.125	61.52	49.78	1.91	9.83	80.92	3.10	15.98
SOBS	1.000	143.96	0.00	143.96	0.00	0.00	100.00	0.00
	0.875	179.84	60.55	100.91	18.39	33.67	56.11	10.22
	0.750	155.31	56.59	81.66	17.06	36.44	52.58	10.98
	0.625	119.71	62.10	43.56	14.05	51.88	36.39	11.74
	0.500	99.21	53.60	31.44	14.17	54.02	31.69	14.28
	0.375	74.99	45.73	17.72	11.53	60.99	23.63	15.38
	0.250	62.81	43.74	8.91	10.16	69.64	14.18	16.18
	0.125	61.52	49.78	3.13	8.61	80.91	5.10	13.99
SOBS_CF	1.000	181.00	0.00	181.00	0.00	0.00	100.00	0.00
	0.875	205.41	60.55	126.88	17.98	29.48	61.77	8.75
	0.750	177.98	56.59	104.31	17.08	31.80	58.61	9.60
	0.625	128.76	62.10	53.33	13.33	48.23	41.42	10.35
	0.500	108.12	53.60	39.07	15.46	49.57	36.13	14.30
	0.375	81.67	45.73	23.01	12.93	56.00	28.17	15.83
	0.250	64.91	43.74	10.96	10.21	67.39	16.88	15.73
	0.125	62.76	49.78	3.98	9.00	79.32	6.34	14.34
SuBSENSE	1.000	1671.67	0.00	1671.67	0.00	0.00	100.00	0.00
	0.875	1380.24	60.55	1300.80	18.90	4.39	94.24	1.37
	0.750	1124.91	56.59	1049.31	19.00	5.03	93.28	1.69
	0.625	651.31	62.10	574.78	14.43	9.53	88.25	2.22
	0.500	472.19	53.60	403.42	15.17	11.35	85.44	3.21
	0.375	281.16	45.73	222.97	12.46	16.27	79.30	4.43
	0.250	154.45	43.74	101.16	9.55	28.32	65.50	6.18
	0.125	83.85	49.78	25.81	8.26	59.37	30.78	9.85
LOBSTER	1.000	1118.18	0.00	1118.18	0.00	0.00	100.00	0.00
	0.875	942.54	60.55	863.32	18.66	6.42	91.60	1.98
	0.750	765.14	56.59	690.32	18.23	7.40	90.22	2.38
	0.625	453.94	62.10	377.86	13.98	13.68	83.24	3.08
	0.500	332.11	53.60	264.32	14.19	16.14	79.59	4.27
	0.375	202.72	45.73	145.32	11.67	22.56	71.69	5.75
	0.250	122.21	43.74	68.14	10.33	35.79	55.76	8.46
	0.125	78.13	49.78	17.78	10.56	63.72	22.76	13.52
PAWCS	1.000	4307.61	0.00	4307.61	0.00	0.00	100.00	0.00
	0.875	3418.49	60.55	3325.39	32.55	1.77	97.28	0.95
	0.750	2324.85	56.59	2236.90	31.35	2.43	96.22	1.35
	0.625	940.73	62.10	852.79	25.84	6.60	90.65	2.75
	0.500	664.37	53.60	585.12	25.65	8.07	88.07	3.86
	0.375	353.84	45.73	288.71	19.40	12.93	81.59	5.48
	0.250	193.69	43.74	132.88	17.08	22.58	68.60	8.82
	0.125	96.72	49.78	33.02	13.93	51.47	34.14	14.40

Table 5 Average performances for each segmentation method in the video10 from CAMO_UOW. Each column shows the selected foreground method (Method), the applied resize factor (RF), the total execution time (Time, in seconds, where the lower the better), the execution time of each step (in seconds, downsampling t_D, segmentation t_S and upsampling t_U, where the lower the better) and its relation in percentage to the total execution time ($\%t_D$, $\%t_S$ and $\%t_U$), respectively. Each row shows the average performance achieved by the downsampling methods.

resolution videos with the minimum possible loss of quality. It consists in applying a downsampling technique, applying the foreground detection model, to later interpolate the segmentation output to the original size. Some of the most recent techniques of foreground detection with stationary cameras have been compared, along with some downsampling algorithms. Thus, it is possible to observe some stability of the F-measure, with reductions of up to 25% in the size of the scene (0.75 as a resize factor). Furthermore, depending on the downsampling method, it is possible to maintain the same efficiency in most motion detection methods with resizing factors of 0.5 (50% of the size of the scene), as observed in the NN and LIN downsampling methods. Except for a couple of methods in which the application of the metamodel does not yield any improvement (Wren and Zivkovic), in the remaining ones a decrease in the execution time with limited losses in the segmentation effectiveness is found. On the other hand, the increase of the frame rate in the case of high resolution videos could be considered limited (less than 10 fps), although it should be taken into account that we start from frame rates that are in the range of 0 to 3 fps. In general, it can be concluded that this proposal is really interesting when the analyzed videos are high resolution, since the increase in fps is relevant and the quality of the segmentation (F-Measure) is hardly affected.

Acknowledgments

This work is partially supported by the Ministry of Economy and Competitiveness of Spain under grants TIN2016-75097-P and PPIT.UMA.B1.2017. It is also partially supported by the Ministry of Science, Innovation and Universities of Spain [grant number RTI2018-094645-B-I00], project name Automated detection with low cost hardware of unusual activities in video sequences. It is also partially supported by the Autonomous Government of Andalusia (Spain) under project P12-TIC-657, project name Self-organizing systems and robust estimators for video surveillance. All of them include funds from the European Regional Development Fund (ERDF). The authors thankfully acknowledge the computer resources, technical expertise and assistance provided by the SCBI (Supercomputing and Bioinformatics) center of the University of Málaga. They also gratefully acknowledge the support of NVIDIA Corporation with the donation of two Titan X GPUs used for this research. The authors acknowledge the funding from the Universidad de Málaga.

References

Babaee M, Dinh D, Rigoll G (2018) A deep convolutional neural network for video sequence background subtraction. Pattern Recognition 76:635–649, DOI 10.1016/j.patcog.2017.09.040

Benedek C, Szirányi T (2007) Study on color space selection for detecting cast shadows in video surveillance. International Journal of Imaging Systems and Technology 17(3):190–201, DOI 10.1002/ima.20110

Bouwmans T (2014) Traditional and recent approaches in background modeling for foreground detection: An overview. Computer Science Review 11-12:31–66, DOI 10.1016/j.cosrev.2014.04.001

Bouwmans T, Javed S, Sultana M, Jung S (2019) Deep neural network concepts for background subtraction: A systematic review and comparative evaluation. Neural Networks 117:8–66, DOI 10.1016/j.neunet.2019.04.024

Cvetkovic S, Bakker P, Schirris J, De With P (2006) Background estimation and adaptation model with light-change removal for heavily down-sampled video surveillance signals. pp 1829–1832, DOI 10.1109/ICIP.2006.312602

Dong C, Loy CC, He K, Tang X (2015) Image super-resolution using deep convolutional networks. IEEE transactions on pattern analysis and machine intelligence 38(2):295–307

Elgammal A, Duraiswami R, Harwood D, Davis L (2002) Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. In: IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp 1151–1163

Goyette N, Jodoin PM, Porikli F, Konrad J, Ishwar P, et al. (2012) Changedetection. net: A new change detection benchmark dataset. In: CVPR Workshops, 2012, pp 1–8

Guan YP (2010) Spatio-temporal motion-based foreground segmentation and shadow suppression. IET Computer Vision 4(1):50–60, DOI 10.1049/iet-cvi.2008.0016

Heikkilä M, Pietikäinen M (2006) A texture-based method for modeling the background and detecting moving objects. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(4):657–662, DOI 10.1109/TPAMI.2006.68, URL http://ieeexplore.ieee.org/document/1597122/

Hu W, Tan T, Wang L, Maybank S (2004) A survey on visual surveillance of object motion and behaviors. IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews 34(3):334–352, DOI 10.1109/TSMCC.2004.829274, URL http://ieeexplore.ieee.org/document/1310448/

Javed S, Narayanamurthy P, Bouwmans T, Vaswani N (2018) Robust pca and robust subspace tracking: A comparative evaluation. pp 598–602, DOI 10.1109/SSP.2018.8450718

Jiang Z, Huynh D, Moran W, Challa S (2013) Combining background subtraction and temporal persistency in pedestrian detection from static videos. pp 4141–4145, DOI 10.1109/ICIP.2013.6738853

Keys R (1981) Cubic convolution interpolation for digital image processing. IEEE Transactions on Acoustics, Speech, and Signal Processing 29(6):1153–1160

Kok CW, Tam WS (2019) Digital Image Interpolation in Matlab, 1st edn. Wiley-IEEE Press

Li S, Florencio D, Li W, Zhao Y, Cook C (2018) A fusion framework for camouflaged moving foreground detection in the wavelet domain. IEEE Transactions on Image Processing 27(8):3918–3930

López-Rubio E (2016) Superresolution from a single noisy image by the median filter transform. SIAM Journal on Imaging Sciences 9(1):82–115

López-Rubio E, Luque-Baena RM, Domínguez E (2011) Foreground detection in video sequences with probabilistic self-organizing maps. International Journal of Neural Systems 21(3):225–246

López-Rubio E, Molina-Cabello MA, Luque-Baena RM, Domínguez E (2018a) Foreground detection by competitive learning for varying input distributions. International Journal of Neural Systems 28(05):1750056

López-Rubio FJ, López-Rubio E (2015) Features for stochastic approximation based foreground detection. Computer Vision and Image Understanding 133:30–50

López-Rubio FJ, López-Rubio E, Molina-Cabello MA, Luque-Baena RM, Palomo EJ, Dominguez E (2018b) The effect of noise on foreground detection algorithms. Artificial Intelligence Review 49(3):407–438

Lowe D (2004) Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision 60(2):91–110, DOI 10.1023/B:VISI.0000029664.99615.94

Maddalena L, Petrosino A (2008) A self-organizing approach to background subtraction for visual surveillance applications. Trans Img Proc 17(7):1168–1177, DOI 10.1109/TIP.2008.924285, URL http://dx.doi.org/10.1109/TIP.2008.924285

Maddalena L, Petrosino A (2010) A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection. Neural Computing and Applications 19(2):179–186

Maddalena L, Petrosino A (2012) The sobs algorithm: what are the limits? In: Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on, IEEE, pp 21–26

Mao Y, Chen M, Meng Q (2012) Improved kernel density background estimation with diversity sampling and neighbor information for traffic monitoring. Lecture Notes in Electrical Engineering 128 LNEE(VOL. 5):281–286, DOI 10.1007/978-3-642-25792-6-43

Mao YF, Shi PF (2005) Diversity sampling based kernel density estimation for background modeling. Journal of Shanghai University 9(6):506–509, DOI 10.1007/s11741-005-0008-z

Minematsu T, Shimada A, Uchiyama H, Taniguchi RI (2018) Analytics of deep neural network-based background subtraction. Journal of Imaging 4(6), DOI 10.3390/jimaging4060078

Molina-Cabello MA, López-Rubio E, Luque-Baena RM, Palomo EJ, Domínguez E (2016) Frame size reduction for foreground detection in video sequences. In: Luaces O, Gámez JA, Barrenechea E, Troncoso A, Galar M, Quintián H, Corchado E (eds) Advances in Artificial Intelligence, Springer International Publishing, Cham, pp 3–12

Motamed C (2017) Automatic fish detection from video sequences using an adaptive background modeling algorithm. ACM International Conference Proceeding Series F131372:20–23

Mukherjee D, Wu Q, Nguyen T (2013) Multiresolution based gaussian mixture model for background suppression. IEEE Transactions on Image Processing 22(12):5022–5035

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical recipes in C: the art of scientific computing, 2nd edn. Cambridge University Press

Shen C, Lin X, Shi Y (2006) Moving object tracking under varying illumination conditions. Pattern Recognition Letters 27(14):1632 – 1643, DOI https://doi.org/10.1016/j.patrec.2006.03.010

Sobral A (2013) BGSLibrary: An opencv c++ background subtraction library. In: IX Workshop de Viso Computacional (WVC'2013), Rio de Janeiro, Brazil, URL `https://github.com/andrewssobral/bgslibrary`

St-Charles P, Bilodeau G (2014) Improving background subtraction using local binary similarity patterns. In: IEEE Winter Conference on Applications of Computer Vision, pp 509–515

St-Charles P, Bilodeau G, Bergevin R (2014) Flexible background subtraction with self-balanced local sensitivity. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp 414–419

St-Charles P, Bilodeau G, Bergevin R (2016) Universal background subtraction using word consensus models. IEEE Transactions on Image Processing 25(10):4768–4781

St-Charles PL, Bilodeau GA, Bergevin R (2015) Subsense: A universal change detection method with local adaptive sensitivity. IEEE Transactions on Image Processing 24(1):359–373

Stauffer C, Grimson WEL (1999) Adaptive background mixture models for real-time tracking. In: cvpr, IEEE, p 2246

Wren C, Azarbayejani A, Darrell T, Pentl A (1997) Pfinder: Real-time tracking of the human body. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7):780–785

Wu HH, Chang JH, Weng PK, Wu YY (2006) Improved moving object segmentation by multiresolution and variable thresholding. Optical Engineering 45(11)

Xue K, Liu Y, Ogunmakin G, Chen J, Zhang J (2013) Panoramic gaussian mixture model and large-scale range background substraction method for ptz camera-based surveillance systems. Machine Vision and Applications 24(3):477–492, DOI 10.1007/s00138-012-0426-4

Yan Q, Xu L, Shi J, Jia J (2013) Hierarchical saliency detection. pp 1155–1162, DOI 10.1109/CVPR.2013.153

Yong-Beom Lee, Bum-Jae You, Seong-Whan Lee (2002) A real-time color-based object tracking robust to irregular illumination variations. In: Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation, vol 2, pp 1659–1664, DOI 10.1109/robot.2001.932849

Yubing T, Cheikh F, Guraya F, Konik H, Trmeau A (2011) A spatiotemporal saliency model for video surveillance. Cognitive Computation 3(1):241–263, DOI 10.1007/s12559-010-9094-8

Zhang L, Yang C, Lu H, Ruan X, Yang MH (2017) Ranking saliency. IEEE Transactions on Pattern Analysis and Machine Intelligence 39(9):1892–1904, DOI 10.1109/TPAMI.2016.2609426

Zhao Y, Taubin G (2011) Real-time stereo on GPGPU using progressive multiresolution adaptive windows. GPU Computing Gems Emerald Edition pp 473–495

Zivkovic Z (2004) Improved adaptive gaussian mixture model for background subtraction. In: Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on, IEEE, vol 2, pp 28–31

Fig. 12 Average performances for foreground method PAWCS in categories Baseline and Baseline High Resolution. First column shows category Baseline while second column shows category Baseline High Resolution. Each row exhibits the video Highway, Office, Pedestrians and PETS2006, respectively. Each chart reports execution time (in seconds, in stacked bars to represent the downsampling, segmentation and upsampling steps, where the lower the better) and F-measure (line, the higher the better) performances per tuned configuration. Note that the values of the F-measure are connected between them with lines to better compare the results, but this does not mean that the results are related.

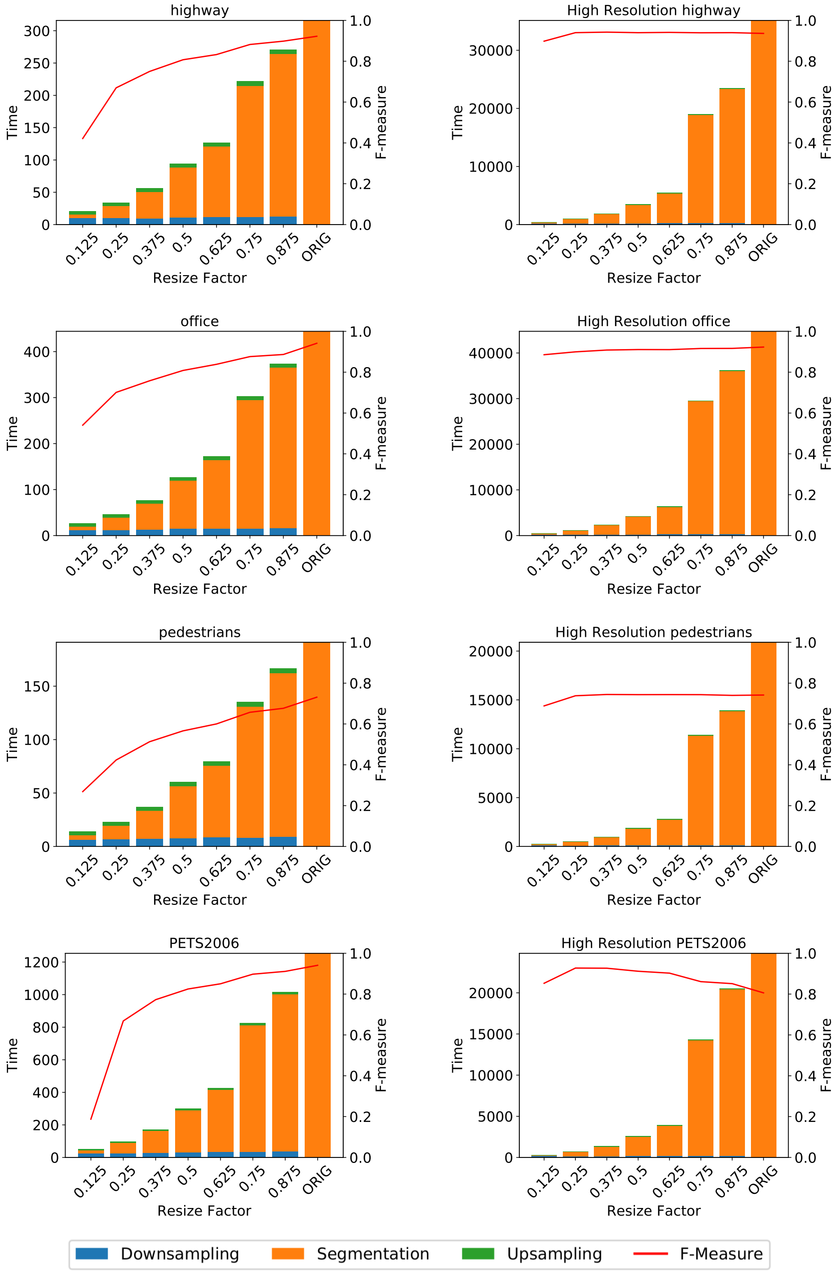


Fig. 13 Average performances for foreground method MFBM in categories Baseline and Baseline High Resolution. First column shows category Baseline while second column shows category Baseline High Resolution. Each row exhibits the video Highway, Office, Pedestrians and PETS2006, respectively. Each chart reports execution time (in seconds, in stacked bars to represent the downsampling, segmentation and upsampling steps, where the lower the better) and F-measure (line, the higher the better) performances per tuned configuration. Note that the values of the F-measure are connected between them with lines to better compare the results, but this does not mean that the results are related.

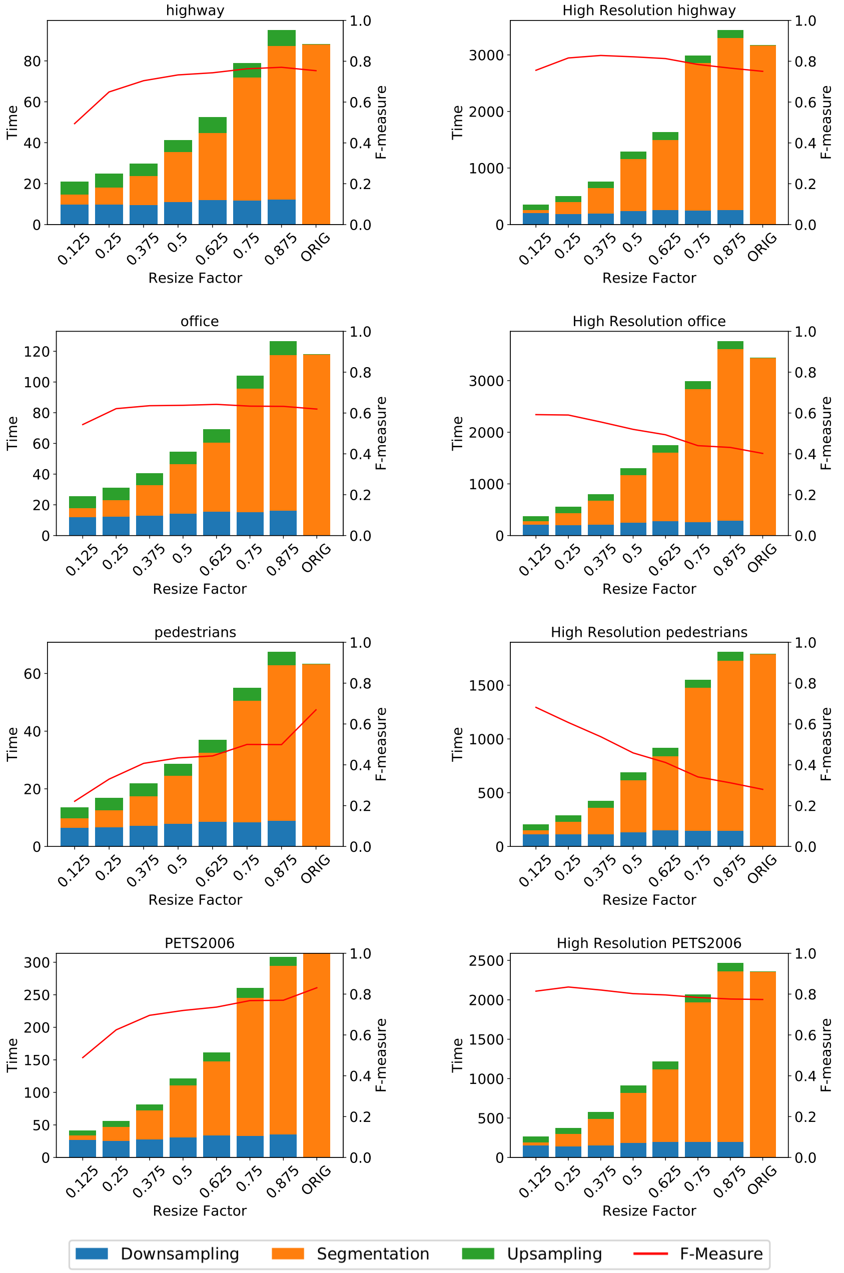


Fig. 14 Average performances for foreground method Wren in categories Baseline and Baseline High Resolution. First column shows category Baseline while second column shows category Baseline High Resolution. Each row exhibits the video Highway, Office, Pedestrians and PETS2006, respectively. Each chart reports execution time (in seconds, in stacked bars to represent the downsampling, segmentation and upsampling steps, where the lower the better) and F-measure (line, the higher the better) performances per tuned configuration. Note that the values of the F-measure are connected between them with lines to better compare the results, but this does not mean that the results are related.

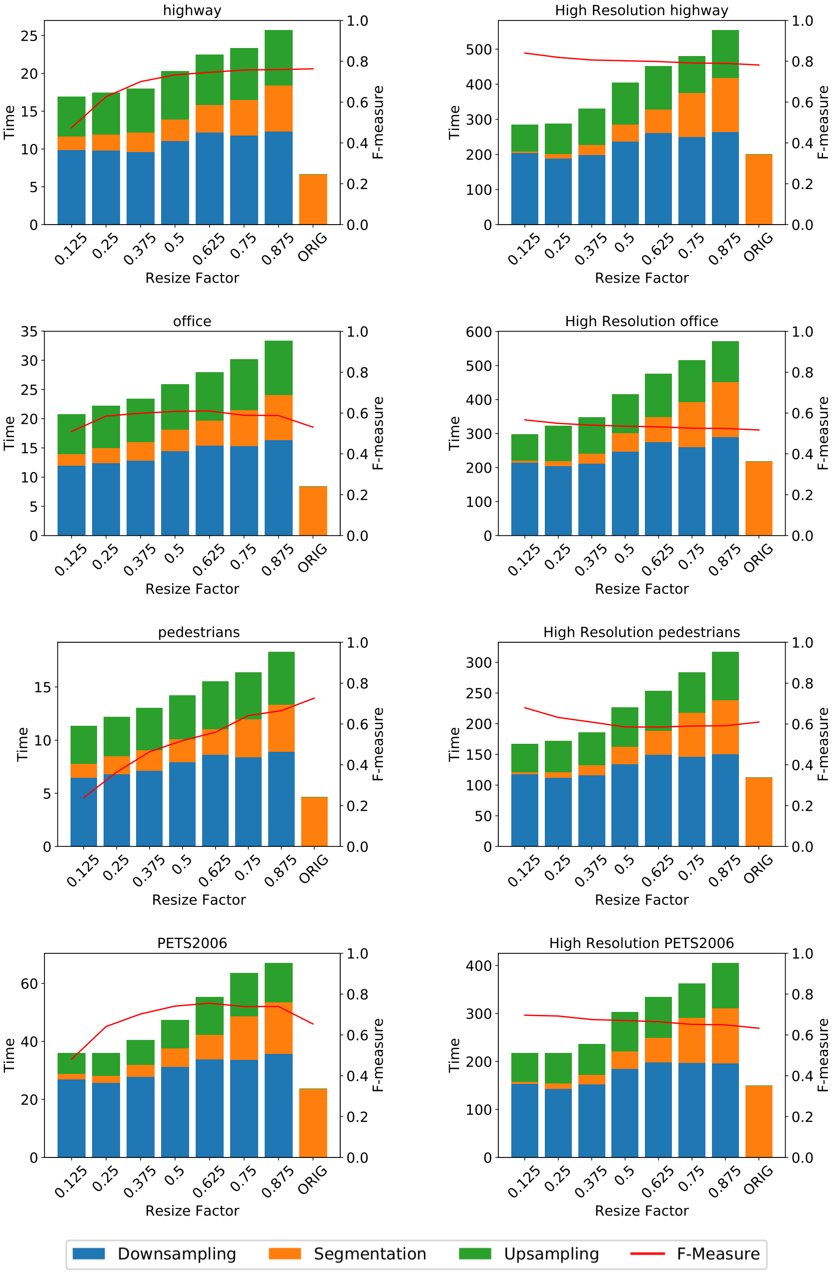
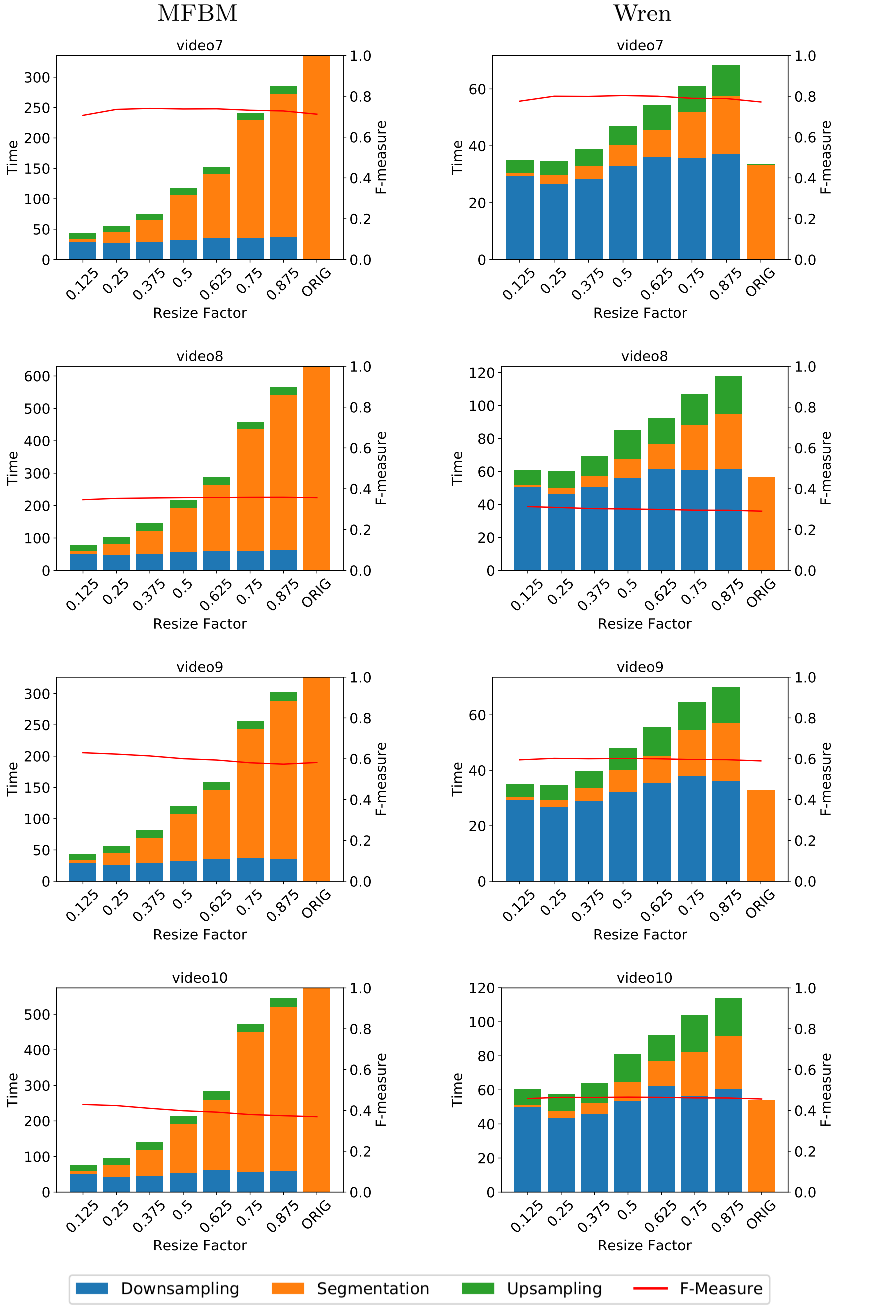


Fig. 15 Average performances for foreground method MFBM and Wren in video7, video8, video9 and video10 from CAMO_UOW dataset. First column shows method MFBM while second column shows method Wren. Each row exhibits a video (all shown videos have a 1920x1080 resolution). Each chart reports execution time (in seconds, in stacked bars to represent the downsampling, segmentation and upsampling steps, where the lower the better) and F-measure (line, the higher the better) performances per tuned configuration. Note that the values of the F-measure are connected between them with lines to better compare the results, but this does not mean that the results are related.


MFBM
Wren
video7
video8
video9
video10
Time
F-measure
Resize Factor
ORIG
Downsampling
Segmentation
Upsampling
F-Measure