Detection and segmentation of moving objects in complex scenes

Aurélie Bugeau and Patrick Pérez

INRIA, Centre Rennes - Bretagne Atlantique
Campus de Beaulieu
35 042 Rennes Cedex, France
{abugea,perez}@irisa.fr

Abstract

In this paper, we address the difficult task of detecting and segmenting foreground moving objects in complex scenes. The sequences we consider exhibit highly dynamic backgrounds, illumination changes and low contrasts, and can have been shot by a moving camera. Three main steps compose the proposed method. First, a set of moving points is selected within a sub-grid of image pixels. A multi-cue descriptor is associated to each of these points. Clusters of points are then formed using a variable bandwidth mean shift technique with automatic bandwidth selection. Finally, segmentation of the object associated to a given cluster is performed using graph cuts. Experiments and comparisons to other motion detection methods on challenging sequences demonstrate the performance of the proposed method for video analysis in complex scenes.

Key words: motion detection, segmentation, mean shift clustering, graph cuts

1 Introduction

Detection of moving objects in sequences is an essential step for video analysis. It becomes a very difficult task in the presence of camera mouvement and dynamic background. We are interested in such challenging sequences, possibly shot by a moving camera, and containing complex, and sometimes large, motions in the background. In addition, the contrast between the background and interesting objects can be low.

1.1 Existing methods

Different kinds of methods exist to solve the problem of motion detection and motion segmentation. Good but incomplete reviews on motion detection methods can be found in [1, 2]. Here, we divide these methods into four categories: thresholding, background modeling, layers extraction and, finally, saliency based methods.

Thresholding methods First works on motion detection were based on adjacent frames difference [3]. The most obvious algorithm is to simply threshold the difference image. The choice of this threshold highly depends on the sequence, its noise, its motion. Furthermore there is no reason for this value to be constant on the whole image. Indeed, different objects and motions give different variations of intensity. Many methods have been developed to decide whether or not a pixel has moved. The decision can be made independently on each pixel [4] or on small blocks of pixels [5]. With independent pixel-wise detections, detection maps are usually corrupted by holes in the mask of moving objects and false detections due to noise. These errors can be attenuated using regularization constraints and contextual information via Markov Random Fields (MRFs) [6]. In [7], MRFs are applied after a step of motion compensation. This step is described in section 2. This method is well adapted to videos taken by a moving camera. The decision rule in many change detection algorithms is cast as a statistical hypotheses test [8]. More complex methods are proposed in [9] for modeling the spatial distribution of either noise or signal and selecting the appropriate threshold. In [10] an *a contrario* method is proposed. It is based on a perceptual grouping principle named the Helmholtz principle. It consists in defining an image model in the absence of moving objects instead of modeling the moving objects.

Background modeling and subtraction methods Methods based on adjacent frame difference are very sensitive to noise and to illuminations changes. When the number of frames in the sequence is large and there is little change between consecutive frames, another solution to motion detection is background modeling. This technique is routinely used in the context of surveillance applications, when the camera is fixed. Background modeling methods can be classified as predictive or non predictive methods. Non-predictive methods build a probability density function of the intensity at an individual pixel. In static environment, the statistical distribution of a pixel can be represented by a single Gaussian [11, 12, 13]. The foreground pixels are determined as those for which the intensity value is far from the mean background model. Then clustering allows the detection and the segmentation of foreground objects. A variable number of Gaussian distributions corresponding to each different foreground object can be added. It was used by [14] for generic objects detection and by [15] for people tracking. In the presence of dynamic background the use of a single Gaussian becomes inappropriate and a

mixture of several Gaussians is preferred to model the background [16, 17]. When changes in the background are too fast, the variance of the Gaussians becomes too large and non parametric approaches are more suited. In [18], Gaussian kernels calculated on the past frames are used to model the intensity distribution at a particular pixel. Contrary to previous approaches, this method addresses the uncertainty of spatial location. Until recently, methods were almost all based on photometric properties. However, a lot of outdoor scenes exhibit a persistent background motion. In such cases, a non-parametric approach that combines color and optical flow has been proposed recently [19]. As optical flow can not be computed when there is no intensity gradient, the authors have chosen to use kernels with variable bandwidth. In [20] the authors extended a statistical background modeling technique to cope with non stationary camera. The current image is registered to the estimated background image using an affine or projective transformation. The foreground information can also be used as in [21] in which the background and foreground maps are assumed to be MRFs. All these pixel-wise approaches allow an accurate detection of moving objects but are memory and possibly computationally expensive. Also, they can be sensitive to noise and they do not take into account spatial correlation. Spatial consistency can be added [22] with a MAP-MRF modeling of both foreground and background. This method has been extended to novelty detection in [23].

Predictive methods use a dynamical model to predict the intensity of a pixel from previous observations. In case of linear Gaussian dynamic model, a Kalman filter can be used to estimate the posterior distribution at a given location and time instant [24, 25]. In [26] an algorithm called wallflower is described. It uses a simpler version of the Kalman filter called *Weiner filter* to predict a pixel's current intensity from its k previous intensities. Pixels whose prediction error is high are classified as foreground pixels. Recent methods are based on more complicated models. For example, in [27] and [28], an autoregressive model was proposed to capture background properties.

Background subtraction and tresholding methods are a preliminary step to moving object detection and subsequent processing is necessary to get the masks of moving objects.

Layer approaches Motion segmentation can be seen as the problem of fitting a collection of motion models to spatio-temporal image data. This leads to the layer approach [29] that aims at fitting a mixture of motion models to the entire image. Layers are then found by associating each pixel to the model it belongs to. In a number of techniques [30, 31, 32, 33], a mixture of probabilistic models is iteratively built with an Expectation-Maximization algorithm (EM). A major drawback of such approaches is that they are very sensitive to the initialization and are computationally expensive. In [34], graph cuts have been used to extract these models or layers. After a number of seed regions are determined using two frames corre-

spondences, these seed regions are first extended thanks to a graph cut segmentation method. The resulting initial regions are then merged into layers according to motion similarities. This method relies on the restrictive assumption that the scene can be approximated by a set of planar regions having an apparent affine motion. Other approaches aim at fitting a polynomial model to all the image measurements and then factorizing this polynomial to obtain the parameters of each 2D motion models (multi-body factorization) [35]. It was adapted to both static and dynamic scenes in [36]. Recently, in [37], an incremental approach to layer extraction has been introduced. Feature points are detected, tracked and then merged into groups based on their motion. Objects are detected incrementally when enough evidence allows them to be distinguished from their background.

In [38], a combination of background modeling with a layer technique is proposed. The layers are called short-term backgrounds. The idea is to assign a layer to each moving object and to keep this object as a single layer even when it stops moving.

Saliency based methods A last approach is to define moving objects as areas having salient motion. In [39], salient motion was defined as motion that is likely to result from a typical surveillance target (*e.g.* person, vehicle). This definition was used in [40] and [41] to detect moving objects. The assumption made is that an object with salient motion moves in an approximate consistent direction during a time period. Therefore moving objects are searched as localized image regions that have moved in the same direction during a time period. In [41], the accumulation of flow motions is done during 10 frames. A fusion of background modeling and saliency was proposed in [42]. A specificity in this paper is that the background is only sparsely modeled on corners, and moving objects are then found by the clustering of foreground corner trajectories.

1.2 Contribution of the paper and description of the videos

In this paper, we aim at extracting the foreground moving objects from a video sequence. We define these objects as groups of pixels that are salient for both motion and color: *A moving object is a set of moving pixels (local mouvements at these locations are different from the one of the background) that have a nearly constant photometry and motion*.

Conversely, we will call "background" the set of all the pixels of the image either that exhibit no motion relative to the background or that have a color and/or motion substantially different from the ones of its neighbors.

Contrary to most existing methods presented in this introduction, we address the problem of moving object detection in the context of complex dynamic scenes.

Indeed, we are interested in challenging sequences containing complex motions, with possible high amplitude, and sudden changes in the background. For example, in the context of driver surveillance, the motions visible through the windows are often hard to characterize because of motion blur and extremely large apparent motion. The “background” is composed of both the passenger compartment (non moving pixels) and what is behind the windows (irrelevant moving areas). Furthermore, contrast between background and interesting objects (face, hands) can be low. Also, the motion of the "interesting" moving objects can be close to the one of the moving background. For example, on driver sequences, depending on the speed of the car, an arm going from the steering wheel to the face can have the same speed as some trees behind the window. Finally, the sequences we consider can be shot by a moving camera. Few frames of such a sequence are shown in figure 1. Most existing methods would fail to detect only the moving arm because of all the motions present behind the window and the low contrast between the arm and the highway guardrail.

$t = 15$ $t = 16$ $t = 17$

Figure 1. Exemple of a driver sequence

In this paper we propose a method that directly finds the areas of the image that matches our definition of a moving object. It combines some of the advantages of the existing methods presented previously.

1.3 Analysis of the existing methods

We now propose a quick analysis of the existing methods and explain why none of them is well adapted for the sequences addressed in this paper.

Thresholding methods are fast and usually easy to implement. They handle well changes in the background but are not robust to sudden illumination changes. Furthermore, they are likely to fail if the contrast between the moving objects and the background is low. Finally, these methods do not return the mask of each object but a binary map and a subsequent processing is needed to separate the objects. To conclude, thresholding methods could not detect by themselves the interesting moving objects in the dynamic complex sequences considered in this paper. Nevertheless,

they permit to know which pixels are not moving (and therefore do not belong to moving objects). As such, they can be used as a preprocessing step to get a more robust and fast detector.

Background modeling methods are dedicated to specific video sequences, acquired by a static camera, in which the background does not vary too much. These constraints are not met by our videos. However, we can retain the method of Mittal and Paragios [19] in which a combination of color and motion features is used. Combining these two types of features can allow the algorithm to handle the small contrast (on color or on motion) between the objects and the background.

Layer approaches permit to directly extract the mask of each object. The requirement is that the scene can be decomposed is several different areas characterized by a well defined motion. In most of our sequences, the dynamic parts of the background (see for example the trees behind the window of the driver) can not be characterized by a parametric motion. Furthermore, using only motion might not be sufficient. Nevertheless, an interesting idea is the definition of an object as a layer that has nearly constant (or salient) characteristics.

Saliency-based approaches only aim at detecting the salient objects without trying to separate all the layers of the video. They are only based on some points trajectories (*i.e.* on their motion) and are then not sufficient when motion contrast between the objects and the background is too low. Furthermore, the direction of the motion vectors is only meaningful when the amplitude of the vectors is large enough. The restriction of the detection on characteristics points [42, 37] allows an important reduction of the computational cost while giving promising results. Of course, it does not return the complete segmentation of each object and a post-processing segmentation step must be added.

1.4 Overview of the paper

As we just explained, in the type of complex dynamic scenes we are interested in, modeling the background or decomposing the images in several layers is really difficult. However, some of the advantages of existing methods can be combined into a procedure that will allow a robust detection. The proposed method thus relies on several classical tools of computer vision.

The algorithm can be divided in three main steps. First, the camera motion is computed and a sub-grid of "moving" pixels, *i.e* not belonging to camera motion, is selected (section 2). This first step has to main goals. On the one hand, it finds the non moving pixels (*i.e.* a part of the background), which do not need to be further processed since they are likely not to belong to moving objects. On the other hand, it gives the possibility to work on a smaller number of pixels which accelerates the all algorithm. Indeed, we want an algorithm that do not require more than few

seconds per frame. At the end of this first step, a descriptor (based on color and motion) is defined to characterize each points of this subset of moving pixels.
The second step of the algorithm aims at extracting from this pixel subset the areas having a nearly constant photometry and motion. Therefore, selected pixels are merged into clusters consistent for both color and motion (section 3), using a mean shift filtering algorithm [43] and a method for automatic kernel bandwidth selection.
These clusters are not completely satisfactory since they do not correspond to the complete pixel-wise segmentation of moving objects. This segmentation is obtained by minimizing an energy function in a MAP-MRF framework (section 4).
Experimental results of the method are presented in section 5. The validity of the whole process will be qualitatively demonstrated on grayscale and color complex dynamical sequences.

1.5 Notations

Let us introduce now the notations. In all the paper, $\mathcal{P}$ will denote the set of N pixels of the frame I_t at time t from an input sequence of images. To each pixel $s = (x, y)$ of $\mathcal{P}$ is associated a feature vector $\mathbf{z}_t(s)$. In case of color sequences, $\mathbf{z}_t(s) = (\mathbf{z}_t^{(G)}(s), \mathbf{z}_t^{(C)}(s), \mathbf{z}_t^{(M)}(s))$, where $\mathbf{z}_t^{(G)}(s)$ is the scalar grayscale value, $\mathbf{z}_t^{(C)}(s)$ a 3-dimensional vector of color values and $\mathbf{z}_t^{(M)}(s)$ a 2-dimensional vector characterizing the apparent motion. The computation of $\mathbf{z}_t^{(M)}(s)$ is explained in section 2.3.1 and the one of $\mathbf{z}_t^{(C)}(s)$ in section 2.3.2. In case of grayscale sequences, the feature vector only contains two elements: $\mathbf{z}_t(s) = (\mathbf{z}_t^{(G)}(s), \mathbf{z}_t^{(M)}(s))$.

2 Point selection and description

The first step of our system consists in building and describing a sub-grid of moving pixels. As we only aim at extracting moving objects, the pixels that are static do not need to be processed. Furthermore, similarly to [40] and [41], we have chosen to perform detection only on several points and their neighborhood. We have preferred the computational lightness and the robustness to noise of these methods to the accuracy of the pixel-wise approaches. This section is organized as follows. First we explain what are the moving pixels. Next we discuss the construction of the grid of points and finally we present the selected descriptors.

2.1 Sensor motion and non-moving pixels

Since moving pixels are the pixels that don't comply with the apparent motion induced by the physical motion of the camera, we have first to estimate this apparent motion. We assume that it is dominant in the image (among various movements in the image field is the one that concerns the larger number of pixels) and that it can be well approximated by a 2D affine model. The parametric flow vector $\mathbf{w}_\theta(s)$ at location $s = (x, y)$ reads:

$$\mathbf{w}_\theta(s) = \begin{pmatrix} a_1 \\ a_4 \end{pmatrix} + \begin{pmatrix} a_2 & a_3 \\ a_5 & a_6 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad . \tag{1}$$

Note that if the camera is static, this vector should be equal to zero. Different methods are available to estimate the parameters of such a model [44, 45, 46]. We use the robust real-time multiresolution algorithm described in [45]. The parameter vector $\theta = (a_1, a_2, a_3, a_4, a_5, a_6)$ is estimated between two consecutive frames I_{t+1} and I_t as follows:

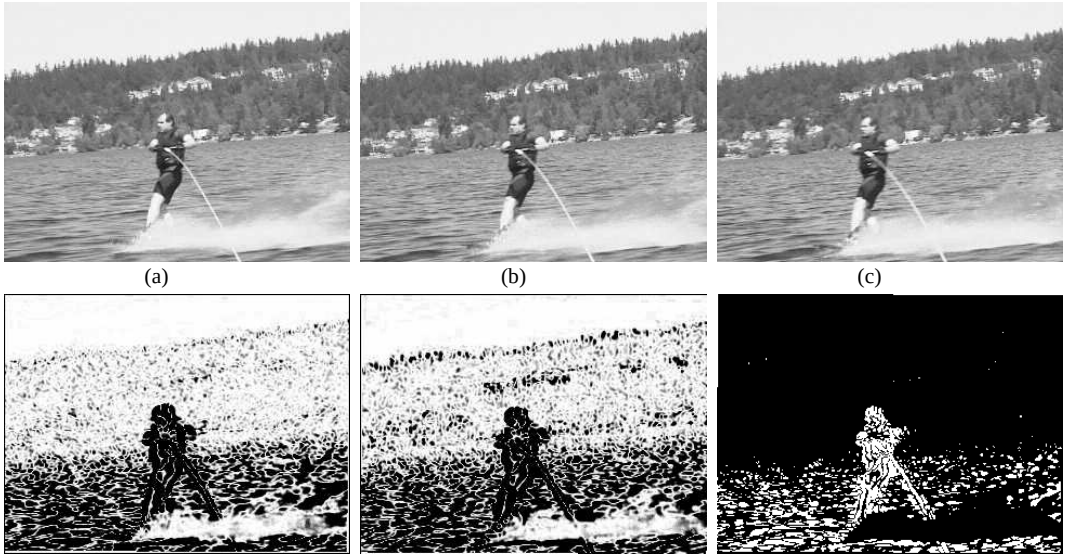
$$\hat{\theta} = \arg\min_\theta \sum_s \rho\Big(\mathbf{z}_{t+1}^{(G)}(s + \mathbf{w}_\theta(s)) - \mathbf{z}_t^{(G)}(s) + \zeta_t\Big) \quad , \tag{2}$$

where $\rho(x)$ is an M-estimator and ζ_t is a global intensity shift that accounts for global illumination changes. The minimization is done through a multiscale Gauss-Newton method that yields a succession of reweighted least-squares problems. As in [45], for all our experiments, 4 scales are used. The auxiliary weight map of the M-estimator will be denoted as W_t ($W_t(s) \in [0, 1]$). The final map indicates if a pixel participates to the final robust motion estimate ($W_t(s)$ close to 1) or is more considered as an outlier ($W_t(s)$ close to 0). A simple pixel-wise motion detector can be built using these maps. A pixel is considered as "moving" at time t if it is an outlier to the dominant motion at times t and $t - 1$:

$$M_t(s) = \begin{cases} 1 \text{ if } W_t(s + \mathbf{w}_{\theta,t-1}(s))) + W_{t-1}(s) < \tau, \\ 0 \text{ else} \quad , \end{cases} \tag{3}$$

where τ is a threshold that we set equal to 0.02. If $M_t(s) = 0$, pixel s is considered as a motionless pixel, and it will not be used for the clustering step of the algorithm. The choice of pure outliers to dominant motion for moving pixels can seem drastic. However experiments have shown that nearly no moving objects are lost using this method. This choice has been made to avoid false labeling of pixels. Furthermore, it permits to deal with occlusion and disocclusion of the scene background. Figure 2 shows a result of this pixel-wise motion detector. It is easy to see that this preliminary step is not sufficient to detect interesting moving objects. In this sequence,

we are willing to detect the water skier. The motion in the water is complex, not repetitive, and so can not be estimated directly. Another drawback of dominant motion estimation can appear in the presence of large uniform objects moving slowly. Inner portions of such objects often appear to belong to the dominant motion (see the person on figure 3). Our algorithm will handle this loss of moving pixels thanks to the final segmentation step where the whole pixel grid is considered (see section 4).



(a) (b) (c)

(d) (e) (f)

Figure 2. Motion maps for the water skier sequence. (a)-(c) Initial grayscale frames 107 to 109. (d) Displaced frame difference at time t. (e) Displaced frame difference at time $t + 1$. (f) Result M_t of the pixel-wise motion detector.

(a) (b)

Figure 3. Motion maps for the person walking in front of water sequence. (a) Initial grayscale frame (b) Result M_t of the pixel-wise motion detector.

2.2 *Pixel subset selection*

The goal of the algorithm is to build groups of pixels consistent both for motion and for some photometric/colorimetric features. These groups must correspond to interesting moving objects. As we already mentioned in the introduction, we also want to design a system that does not require more than few seconds per frame. Unfortunately, the clustering step to be described later is computationally expensive if all the "moving" pixels of the image are used. Therefore, a smaller set of points must be selected.

In [42], moving objects are found using corners, detected with the Harris corner detector. The authors justify the use of corners by claiming that a moving object contains a large number of corners. In our experiments, we have observed that the number of corners belonging to a moving object can be much lower than the number of corners belonging to the background (figure 4). Besides, if variations in the background are fast and if parallax changes, the number of corners and their neighborhoods can be significantly different from one frame to the other. Finally, corner detection adds one stage of calculation and requires the setting of two thresholds.

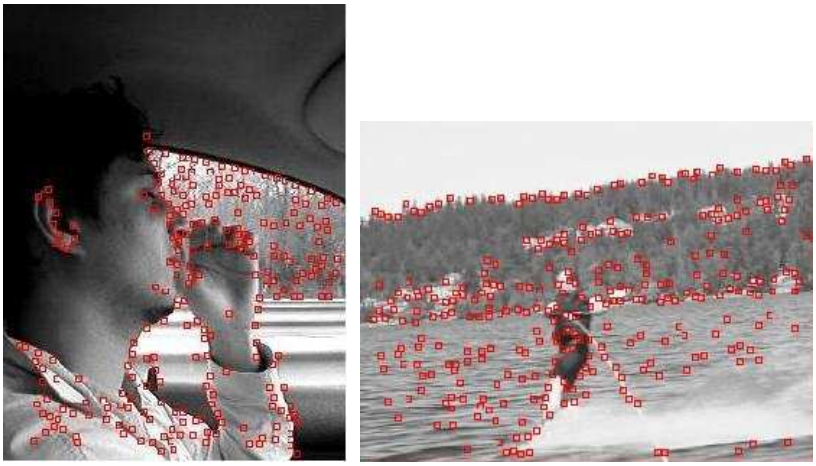


Figure 4. Result of the Harris corner detector on frame 16 of the driver sequence and frame 108 of the water skier sequence.

As no *a priori* information is assumed on the shape and texture of objects, we have chosen to use points of arbitrary type. We only require that their spatial density is roughly constant over the whole image. Hence, we only consider a sub-grid of pixels and prune it thanks to the simple pixel-wise motion detector from previous subsection. At this stage we focus on the following sub-set of pixels:

$$\mathcal{G} = \left\{ s = \left(\frac{k.w}{N_{\mathcal{G}}}, \frac{l.h}{N_{\mathcal{G}}} \right), k = 0 \cdots N_{\mathcal{G}} \ , \ l = 0 \cdots N_{\mathcal{G}} \mid M_t(s) = 1 \right\} \ , \qquad (4)$$

where w and h are the dimensions of the image and $N_{\mathcal{G}}^2$ the size of the grid before

pruning. We have arbitrarily chosen to use the same number of points along x and y axes. The value of the parameter $N_{\mathcal{G}}$ is important. It controls the balance between computational cost (regional methods) and accuracy (local methods). The next step of the system can become computationally expensive if the number of points on the grid is too large. An important thing to note is that $N_{\mathcal{G}}$ may depend on the number $m = \sum_{s \in \mathcal{P}} M_t(s)$ of "moving" pixels in the image. To limit the computational cost for clusters creation, we fix the approximate number of moving points to $n_{\mathcal{G}}$ (500 in our experiments) that are retained at this stage. The size $N_{\mathcal{G}}$ of the grid is then set as $N_{\mathcal{G}} = \lfloor \sqrt{w * h * n_{\mathcal{G}}/m} \rfloor$, where $\lfloor \bullet \rfloor$ is the integer part of $\bullet$.

2.3 Description of the selected points

Now that the subset of pixel locations is chosen, the features that will be used to create clusters corresponding to objects need to be defined. It is necessary to chose only few discriminant features. Indeed, clustering methods can become very computationally expensive in high dimensional feature spaces. Intensity or color is often not sufficient to extract foreground objects because the contrast between an object and the background can be small, nor is optical flow in case of similar motion between an object and the background. Hence the descriptor is formed by three different groups of features. The first group is composed of the coordinates of the point. The second group contains its motion, and the last one contains discriminant photometric features.

2.3.1 Motion features

Several types of methods exist for computing optical flow. Good reviews are given in [47, 48]. We concentrate on the Lucas and Kanade algorithm [49], with an incremental multiscale implementation (as for the dominant motion computation, here 4 scales are used). The reasons for such a choice are that we want an efficient approach and that we do not aim at computing the flow on the whole image but only on several points (which are not particular feature points). However, the 2x2 linear system to be solved at each KLT iteration is regular and well conditioned only for neighborhoods having a non constant luminance. Therefore we will not use the points for which the intensity gradient is very low in all the region. These points will not be considered in next steps of the algorithm and are therefore discarded from the pixel set $\mathcal{G}$, which now reads:

$$\mathcal{G} = \left\{ s = \left(\frac{k.w}{N_{\mathcal{G}}}, \frac{l.h}{N_{\mathcal{G}}} \right) \mid M_t(s) = 1 \,\&\, \exists (x_i, y_i) \in \mathcal{V}\left(\frac{k.w}{N_{\mathcal{G}}}, \frac{l.h}{N_{\mathcal{G}}} \right), |\nabla I(x_i, y_i)| \neq 0 \right\} , \tag{5}$$

where $\mathcal{V}(s)$ defines the neighborhood of the pixel s.

The whole procedure can be applied, independently, to the different locations of any

pixel set. No spatial consistency is enforced over estimates at neighboring locations. We could instead have used Horn and Schunk algorithm [50], which resorts to a smoothness term to regularize over the whole image, or the robust estimation of Black and Anandan [51] to get a better estimation. However these algorithms are more computationally expensive and we do not aim at computing a dense and precise motion estimation over the whole image.

The computation of optical flow using gradient based methods fails when the brightness constancy is not satisfied and when a point does not move as its close neighbors. Instead of using a more robust method to compute optical flow, we have chosen to keep Lucas and Kanade algorithm, which is easy to implement, not expensive, robust to noise, and we verify afterwards if the flow vectors are good or not. To do so, a comparison is done between the neighborhood of pixel $s = (x, y)$ in image at time t (data sample X), and the neighborhood of the corresponding point $s' = (x + d_x, y + d_y)$ at time $t + 1$ (data sample Y), where (d_x, d_y) is the motion vector estimated at location s. The linear relationship between intensity values of X and Y is often estimated by computing the normalized cross correlation γ. This coefficient is also known as Pearson product-moment correlation coefficient by statisticians. Unfortunately, the correlation does not take into account the individual distributions of X and Y. A statistical test related to the linear correlation is the so-called "p-value" (or observed significance level). A small p-value means that the null hypothesis is false and that the two data samples are in fact correlated. For more information on the theory and the implementation of the p-value, we refer to [52]. If the p-value obtained for a point s is larger than 0.05, the motion vector at this point is considered as non valid. The point s will then not be processed in next steps of the algorithm. Finally, recalling that $s' = (x + d_x, y + d_y)$, the processed subset of pixels further reduces to

$$\mathcal{G} = \Big\{ s = \Big(\frac{k.w}{N_\mathcal{G}}, \frac{l.h}{N_\mathcal{G}}\Big) \mid M_t(s) = 1 \;\&\; \exists (x_i, y_i) \in \mathcal{V}\Big(\frac{k.w}{N_\mathcal{G}}, \frac{l.h}{N_\mathcal{G}}\Big), |\nabla I(x_i, y_i)| \neq 0 \;\&\; \text{p-value}(s, s') < 0.05 \Big\} \tag{6}$$

with a valid motion vector $\mathbf{z}_t^{(M)}(s) = (d_x, d_y)$ associated to each of its point s. The size of this set $\mathcal{G}$ will be denoted as $|\mathcal{G}|$.

The influence of the p-value is shown on figure 5. The first row shows the original image and the set of moving pixels. The second row shows the final pixel subsets $\mathcal{G}$ obtained without using a validation step (Figure 5c), when keeping only vector flows for which the correlation coefficient is higher than 0.5 (Figure 5d) and the pixel subset obtained with the p-value test (Figure 5e). The third row presents the associated motion fields, as estimated by multiscale Lucas-Kanade technique at each point of the various grids. The parameters used in these examples are the following: the number of moving pixels m is 10845, the size of the image is 240x320 and the parameter $N_\mathcal{G}$ is equal to 5. Pixel subset $\mathcal{G}$ finally contains 420 pixels when

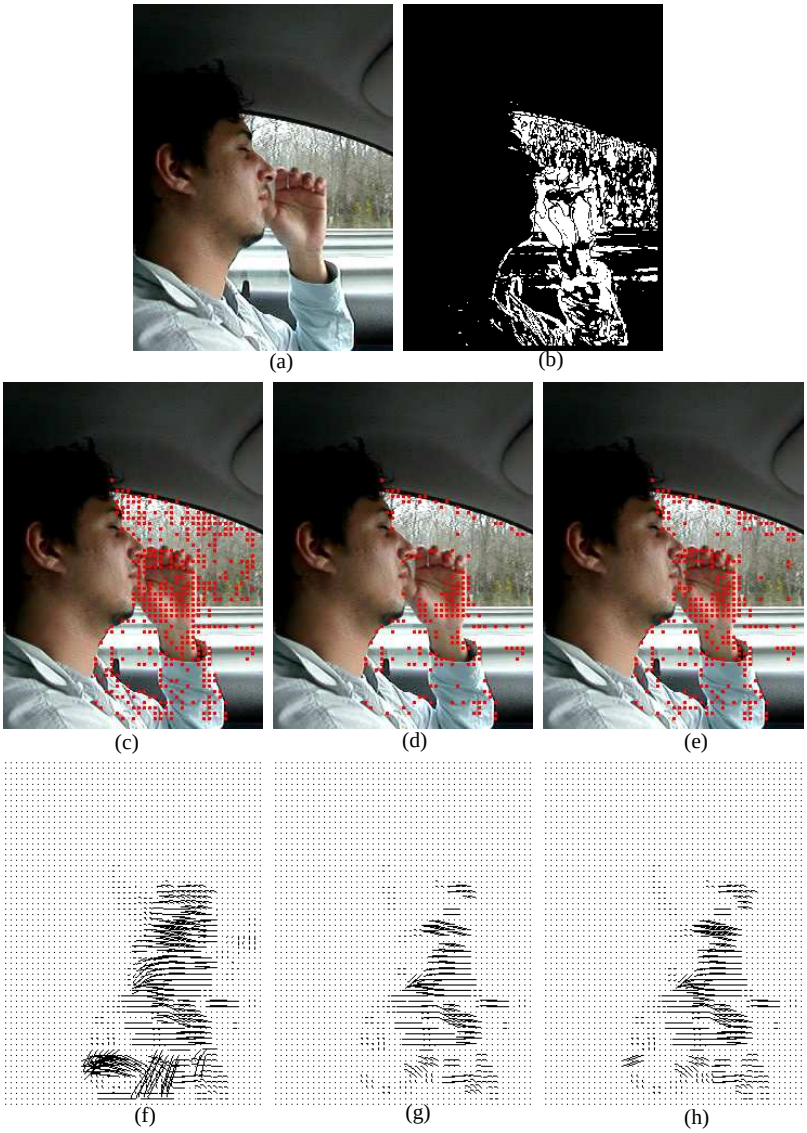


Figure 5. Result of pixel selection (second row) and associated motion fields (third row) on frame 16 of the driver sequence. (a) Original image. (b) Moving pixels. (c) Final subset without validation of motion vectors (associated motion field is shown on image (f)). (d) Final subset obtained by a correlation validation with correlation threshold at 0.5. (associated motion field is shown on image (g)). (e) Final subset obtained by using the p-value (associated motion field is shown on image (h)).

no validation is used, 184 pixels for a correlation test and 277 with the use of p-value. The p-value enables to keep more points than the correlation, especially important points on the arm of the driver.

2.3.2 Photometric features

The last features concern the photometry at selected locations. Many photometric descriptors have been proposed in the literature, including celebrated SIFT descrip-

tor (Scale Invariant Feature Transform) [53]. SIFT descriptor is invariant to certain amounts of illumination changes and to geometric transformations. However, as it is a vector of dimension 128, it is not the best choice in the prospect of clustering.

The photometric descriptor we will use for clustering must be discriminative while not being of too high dimension. We designed two different descriptors depending on whether sequences are with colors or simply intensities.

Grayscale sequences:
In case of grayscale sequences, the first photometric feature is the intensity itself, and more precisely the mean $\overline{\mathbf{z}_t^{(G)}}(s)$ of the luminance on a 3x3 window around the point $s = (x, y)$. The reason for using the mean is to be more robust to noise. The choice of a 3x3 window is somewhat arbitrary. Empirically, this choice has proved effective to improve the robustness of the approach while keeping modest the corruption by irrelevant information in textured areas and at locations on object's boundaries. Experiments have shown that this feature is not sufficient to separate the objects from the background when the contrast is low. Instead of adding numerous dimensions related to the intensity gradients, we have decided to use an information on the texture. Many definitions and descriptions of textures exist. Here, in order to keep the dimension of the descriptor low, the texture is simply associated to the quantity or the strength of edges present in the studied area. The feature that we introduce to capture the texture is the standard deviation $\sigma_{\Delta \mathbf{z}_t^{(G)}(s)}$ to the mean of the Laplacian of intensity on a 3x3 window around the point.

To include some simple temporal consistency, we add image intensity and texture values at time $t + 1$ for the displaced point $s' = s + (d_x, d_y)$. The idea behind this feature is that if two points are merged in the same cluster at time t, they should also be merged at time $t + 1$.

Finally, for grayscale sequences, the descriptor at each individual valid point $s = (x, y)$ of $\mathcal{G}$, indexed by i ($i = 1 \dots |\mathcal{G}|$), is:

$$\mathbf{x}^{(i)} = (\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \mathbf{x}_3^{(i)}), \tag{7}$$

where

$$\mathbf{x}_1^{(i)} = (x, y)\ ,\ \mathbf{x}_2^{(i)} = (d_x, d_y)\ ,\ \mathbf{x}_3^{(i)} = (\overline{\mathbf{z}_t^{(G)}}(s), \sigma_{\Delta \mathbf{z}_t^{(G)}(s)}, \overline{\mathbf{z}_{t+1}^{(G)}}(s'), \sigma_{\Delta \mathbf{z}_{t+1}^{(G)}(s')})\ ,$$

and $s' = (x + d_x, y + d_y)$.

Color sequences:
For color sequences, the information given by the three channels have appeared to be sufficient during our experiments. Hence, no texture information is considered. However, we observed that RGB color channels do not provide the best representation of our images since they are highly correlated and sensitive to illumination

changes. Therefore it is better to separate the luminance and the chrominance informations. Furthermore, most of our test sequences contain human skin, which has a specific signature in the space of chrominance [54, 55]. Therefore, it is interesting to use a color system where chrominance is explicitly expressed. Most chrominance spaces give the same results on skin detection [56]. We use the YUV color space, which proved appropriate for various types of sequences. As for grayscale sequences, we also include some simple temporal consistency by adding image $t+1$ chrominances at the corresponding location. Finally, for color sequences, the descriptor at each individual valid point indexed by i ($i = 1 \dots |\mathcal{G}|$) is:

$$\mathbf{x}^{(i)} = (\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \mathbf{x}_3^{(i)}), \tag{8}$$

where

$$\mathbf{x}_1^{(i)} = (x, y)\ ,\ \mathbf{x}_2^{(i)} = (d_x, d_y)\ ,\ \mathbf{x}_3^{(i)} = (\overline{\mathbf{z}_t^{(C)}}(s), \overline{\mathbf{z}_{t+1}^{(C)}}(s'))\ \ ,$$

where the 3-dimensional color feature vector is: $\mathbf{z}_t^{(C)}(s) = (Y_t(s), U_t(s), V_t(s))$.

3 Clustering points: mean shift for mixed feature spaces

Now that the subset of valid points has been chosen and described, we address the problem of grouping them into clusters. Many data clustering methods have been described in the literature. A good review on classic clustering techniques can be found in [57]. An appealing technique to extract the clusters is the mean shift algorithm, which does not require to fix the (maximum) number of clusters and to assume prior knowledge on the shape of the clusters. On the other hand the kernel bandwidth and its shape have to be chosen or estimated for each dimension.

3.1 Mean shift partitioning

Mean shift is an iterative gradient ascent method used to locate the density modes of a cloud of points, *i.e.* the local maxima of its density. This technique is well described in [43]. Usual mean shift is based on a fixed bandwidth kernel density estimator. Given a set of n points $\{\mathbf{x}^{(i)}\}_{i=1..n}$ in the d-dimensional space $\mathcal{R}^d$, the non-parametric density estimation at each point $\mathbf{x}$ is given by:

$$\begin{aligned} \widehat{f}(\mathbf{x}) &= \frac{1}{n|\mathbf{H}|^{1/2}} \sum_{i=1}^{n} K(\mathbf{H}^{-1/2}(\mathbf{x} - \mathbf{x}^{(i)})) \\ &= \frac{c_k}{n|\mathbf{H}|^{1/2}} \sum_{i=1}^{n} k(\|\mathbf{H}^{-1/2}(\mathbf{x} - \mathbf{x}^{(i)})\|^2) \end{aligned} \tag{9}$$

where K is a kernel with associated profile k, $\mathbf{H}$ is the bandwidth matrix (also simply called bandwidth) and c_k is a strictly positive normalization constant which makes $K(\mathbf{x})$ integrate to one.

Variable bandwidths are essential when local characteristics of the data vary significantly across the feature space [58]. In [59] the authors have demonstrated that the "balloon estimator" [60] is better adapted than the fixed kernel density estimator (Eq. 9) for data of dimension larger than 3. Here the dimension of our data is 10 for color sequences and 8 for grayscale ones. Hence, we have chosen to use a mean shift based on this estimator, which makes the bandwidth vary at each estimation point. It is defined as

$$\begin{aligned} \widehat{f}(\mathbf{x}) &= \frac{1}{n}\sum_{i=1}^{n} K_{\mathbf{H}(\mathbf{x})}(\mathbf{x}-\mathbf{x}^{(i)}) \\ &= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{|\mathbf{H}(\mathbf{x})|^{1/2}} k(\mathbf{H}(\mathbf{x})^{-1/2}(\mathbf{x}-\mathbf{x}^{(i)})) \quad . \end{aligned} \tag{10}$$

The mean shift technique using the balloon estimator has been introduced in [61] and we use this method to partition our set of points.

3.2 Bandwidth selection for mean shift

A mode seeking algorithm can be derived by iteratively computing and applying the so-called mean shift vector (obtained after differentiating the kernel estimator). The partition of the feature space is obtained by grouping together all the data points whose associated mean shift procedures converged to the same mode. The quality of the results highly depends on the choice of the bandwidth matrix $\mathbf{H}$.

The bandwidth selection can be based either on statistical analysis or task-oriented. Statistical methods compute the best bandwidth by balancing the bias and the variance of the density estimate. Task-oriented methods rely on the stability of the feature space partitioning. In [62] a method that combines the two different approaches has been introduced. This method is based on the maximization of the normalized mean shift vector. However, there is no theoretical result permitting to relate formally this maximization to the quality of the clustering. In [61], a method based on the task oriented step of [62] has been developed. It is dedicated to the variable bandwidth selection in mixed multi-dimensional spaces. The main idea underlying this bandwidth selection method is that if one cluster can be represented by a normal distribution, then the best cluster is the one for which the normal distribution is the most stable. If few points are added to the cluster or if some are left apart, the associated distribution should not change. Therefore by partitioning the data for several predefined bandwidths, one can identify the best bandwidth as the one that gives the most stable partitions.

This method provides a way to select bandwidths for heterogeneous data of high dimension. However it relies on the delicate choice of candidate bandwidths. One could simply use a large range of candidate bandwidths but this would be very expensive as the number of mean shift partitioning procedures would be very large. In our application of moving object detection the data space is divided in $P = 3$ feature spaces: the position, the motion and the color. In all our experiments we have selected the same predefined bandwidth matrices based on empirical choices that are explained below. For all the feature spaces ($\rho = 1 \dots P$) we have selected $B_\rho = 9$ different bandwidths. For the position they depend on the size of the grid. Indeed, the larger the spacing between two consecutive points of the grid is, the larger the predefined bandwidths for the position should be. They are defined as:

$$\mathbf{H}_1^{(b)} = \frac{w}{N_\mathcal{G}}(1 + \frac{3b}{B_1 - 1}) \quad \mathbf{H}_2^{(b)} = \frac{h}{N_\mathcal{G}}(1 + \frac{3b}{B_1 - 1}) \ , b = 1 \dots B_1 \ , \tag{11}$$

where B_1 is the number of predefined bandwidths for the first feature space (here $B_1 = 9$). The range of predefined matrices for color and motion is directly computed from image noises. We define color, respectively motion, noises as the standard deviation over the whole grid of the color values difference, respectively motion, between two neighboring points of the grid. Introducing $\mathcal{V}_\mathcal{G}$ the set of pairs of neighboring points of the pixel set $\mathcal{G}$, $|\mathcal{V}_\mathcal{G}|$ its cardinal, the mean and standard deviation vectors are:

$$\alpha_\rho = \frac{1}{|\mathcal{V}_\mathcal{G}|} \sum_{(i,j) \in \mathcal{V}_\mathcal{G}} |\mathbf{x}_\rho^{(i)} - \mathbf{x}_\rho^{(j)}| \tag{12}$$

and

$$\beta_\rho = \sqrt{\frac{1}{|\mathcal{V}_\mathcal{G}|} \sum_{(i,j) \in \mathcal{V}_\mathcal{G}} (|\mathbf{x}_\rho^{(i)} - \mathbf{x}_\rho^{(j)}| - \alpha_\rho)^2} \ . \tag{13}$$

Some experimentations have shown that the range of predefined matrices for color ($\rho = 2$) and motion ($\rho = 3$) can be defined as

$$\mathbf{H}_\rho^{(b)} = \beta_\rho(0.5 + \frac{1.5b}{B_\rho - 1})\mathbf{I}_{d_\rho}, b = 1 \dots B_\rho \ , \tag{14}$$

where $\mathbf{I}_{d_\rho}$ is the identity matrix of size d_ρ.

The clustering step decomposes the data into several clusters, each corresponding to a moving object or a moving part. We retain only large enough clusters (*e.g.*, with more than 15 valid points). Finally we obtain k_t clusters $C_{u,t}, u = 1 \dots k_t$.

4 Segmenting the objects

In order to get the complete masks of objects, each corresponding to one cluster, a final step is necessary. Numerous segmentation methods exist. Some reviews are [63, 64, 65] and the recent book [66]. Here we need a method that has the following properties. First, the number of final segmented regions is known beforehand. The method must take this information into account. Secondly, it must handle high dimensional data, while running fast, since we will rely on both color and motion measurements. Another wanted property is the possibility to force certain pixels to belong to one of the segmented regions. The reason is that we want that the pixels belonging to one cluster at the end of the previous stage belong to the corresponding segmented object. The last and probably most important property concerns motion features. Indeed, the segmentation concerns all the pixels, whether they belong or not to pixel subset $\mathcal{G}$. If pixel s at time t belongs to $\mathcal{G}$, its associated feature vector is $\mathbf{z}_t(s) = (\mathbf{z}_t^{(C)}(s), \mathbf{z}_t^{(M)}(s))$ (in case of color sequences). The component $\mathbf{z}_t^{(C)}(s)$ is a 3-dimensional color vector in the YUV chrominance space and $\mathbf{z}_t^{(M)}(s)$ is a 2-dimensional motion estimate. As explained in section 2.3.1, motion vectors are only computed and validated on the pixels belonging to the set defined in (6). Hence, if $s \notin \mathcal{G}$, motion vector is not defined at location s whose feature vector reduces to color only. As a result, pixel measurements are of variable dimension.

Given these requirements, graph cut techniques appear as the natural candidate to construct the segmentation. Segmentation with mean-shift for instance could not fulfill the different prerequisites.

Segmenting the object associated to a given cluster amounts to assigning a label l_s, either "background" or "object", to each pixel s of the image. This problem can be reformulated into the graph cut framework as a bi-partitioning problem. In recent years graph cuts have been increasingly used in image segmentation. The reason for such a popularity is that the exact maximum *a posteriori* (MAP) of a two label pairwise Markov Random Field (MRF) can be computed in polynomial time using min-cut/max-flow algorithms [67]. In seminal paper [68], Boykov *et al.* introduce an iterative foreground/background segmentation system based on this principle, using hard constraints provided by the user. Here we can directly learn some properties of the object from the points belonging to its cluster. These points are called inliers. The energy function is defined so that its minimum should correspond to a good segmentation, in the sense that it is guided both by the motion and color of the inliers (observed foreground) and by distributions of color and motion built on the whole image. Also, boundaries of the segmentations should preferably coincide with large photometric gradients. These various specifications are captured by the

following objective function:

$$E_t(L) = -\gamma_c \sum_{s\in\mathcal{P}} \ln(\Pr(\mathbf{z}_t^{(C)}(s)|l_s)) - \gamma_m \sum_{s\in\mathcal{G}} \ln(\Pr(\mathbf{z}_t^{(M)}(s)|l_s)) + \lambda \sum_{(s,r)\in\mathcal{V}} \exp\Big(-\frac{\|\mathbf{z}_t^{(G)}(s) - \mathbf{z}_t^{(G)}(r)\|^2}{\sigma^2}\Big)\frac{1}{\mathrm{dist}(s,r)}\delta(l_s,l_r) \tag{15}$$

where L is the set of all the labels l_s, $s \in \mathcal{P}$, dist is a distance measure, $\mathcal{V}$ is the set of unordered pairs (s,r) of neighboring elements of $\mathcal{P}$, and δ is the function defined by:

$$\delta(l_s,l_r) = \begin{cases} 1 \text{ if } l_s \neq l_r \\ 0 \text{ else.} \end{cases} \tag{16}$$

The parameters γ_m, γ_c, λ are some weight constants discussed below. Since we are segmenting each object independently, there is one such energy function to be minimized per cluster.

The two first terms of the cost function are based on pixel-wise modeling of color and motion features distributions. As previously explained, motion term only concerns a subset of points. For both, color and motion, the object distribution is a mixture of Gaussians on the inliers. For the background, the mixture is built as follows. For color it is computed on the whole image whereas for motion it is only computed on the grid. In [69], authors have shown that it is possible to force some pixels to belong to the object or to the background. Here we force inliers to belong to the object. This is done by rewriting the energy function as:

$$\begin{aligned} E_t(L) = &-\gamma_c \sum_{s\in\mathcal{P}\setminus\mathcal{C}_{u,t}} \ln\Big(\Pr(\mathbf{z}_t^{(C)}(s)|l_s)\Big) - \gamma_m \sum_{s\in\mathcal{G}\setminus\mathcal{C}_{u,t}} \ln\Big(\Pr(\mathbf{z}_t^{(M)}(s)|l_s)\Big) - \\ &\sum_{s\in\mathcal{C}_{u,t}} \Big[\Big(1 + \max_{s'\in\mathcal{P}} \lambda \sum_{r|(s',r)\in\mathcal{V}} V_{\{s',r\}}\Big)\delta(l_s, \text{"object"})\Big] + \\ &\lambda \sum_{(s,r)\in\mathcal{V}} V_{\{s,r\}}\delta(l_s,l_r) \end{aligned} \tag{17}$$

where u is the object or cluster we are trying to segment and $V_{\{s,r\}} = \exp\big(-\frac{\|\mathbf{z}_t^{(G)}(s)-\mathbf{z}_t^{(G)}(r)\|^2}{\sigma^2}\big)\frac{1}{\mathrm{dist}(s,r)}$.

Because motion only concerns pixel subset $\mathcal{G}$, we chose to set the parameters γ_c and γ_m such that $\gamma_c = 1$ and

$$\gamma_m = N_{\mathcal{G}}^2. \tag{18}$$

This choice permits to give more influence to the points having a valid motion. If γ_m and γ_c were equal, the motion would have only a very small influence on the segmentation result.

The parameter σ in the third energy term can be related to noise [70]:

$$\sigma = 2 * \langle(\mathbf{z}_t^{(G)}(s) - \mathbf{z}_t^{(G)}(r))^2\rangle \tag{19}$$

where $\langle . \rangle$ denotes expectation over the whole image. The value of parameter λ has not been really studied in literature. To avoid a possible saturation of all binary edges in the max-flow procedure, we fix here its value as:

$$\lambda = \frac{1}{N}\Big(-\gamma_c \sum_{s\in\mathcal{P}} \sum_{l_s="bkg","obj"} \ln(\Pr(\mathbf{z}_t^{(C)}(s)|l_s)) - \gamma_m \sum_{s\in\mathcal{G}} \sum_{l_s="bkg","obj"} \ln(\Pr(\mathbf{z}_t^{(M)}(s)|l_s))\Big) \ . \tag{20}$$

After the minimization, all pixels labeled as "object" form the final mask of the moving object.

5 Experimental results

This section presents several experimental results on different kinds of sequences. Some of these videos have been shot by a non-moving camera, and they all exhibit rather different types of motions. Also we tested our method on both grayscale and color sequences. For all the tests, exactly the same parameters were used. The size of the grid as well as the predefined matrices for bandwidths selection are set as explained in previous sections.

This section is decomposed in two parts. First we present the results obtained for both the clustering and the segmentation step. In a second part we see the final output (set of all segmented objects) of our algorithm as a motion detection mask in order to compare our approach with other methods mainly based on background modeling.

5.1 Sequences taken by a moving camera

We start by showing the results of the clustering and the segmentation on three different video sequences. For visualization purpose, an arbitrary individual color is assigned in each frame to each segmented object. This color only depends on the arbitrary order in which the objects were handled. Hence there is no temporal consistency and the same object can be represented by different colors along the sequence.

The two following results are on color sequences shot by a moving camera. The first one is the water skier sequence already discussed in section 2. This sequence is hard because a lot of moving pixels are present in the water. On figure 6 we show the original images (first column), the moving pixels maps (second column),

the results of the clustering steps (third column) and finally the segmented moving objects (fourth column). The water skier is most of the time well detected and segmented while no objects are found on the water. On frame 124, the skier is not detected since, at this instant, his apparent motion is similar to the dominant motion estimated on the scene (see the mask of moving pixels). Note also that on the same frame, part of the water was detected as a moving object at the end of the clustering step. This happened several times during the sequence. However most of the time there is no corresponding moving object. The reason is that the water occupies a large part of the image, making the probability to belong to the background almost the same as the one to belong to the object. At the end of the energy minimization, all the pixels in the water are labeled as background, except for the inliers that were forced to belong to the object. These inliers are only visible by zooming on the segmentation image. In frame 214, the cluster covers both the skin (arm, face, legs) and the body of the skier. This comes from the automatic bandwidth selection for the mean shift partitioning. The chrominance values corresponding to these different parts are very close. However it is not the case for the Y channel, corresponding to the intensity. The range of predefined bandwidths we are using is large, and the preferred bandwidth for the intensity turned out to be large.

The next sequence is the difficult driver sequence presented in the introduction. In this sequence we have to deal with the high dynamic background (behind the window), the low contrast between the shirt and the guardrail, some drastic illumination changes. Furthermore the motion of the hand and the trees behind the window are sometimes rather similar. The clusters corresponding to the moving objects and the segmentations can be seen on figure 7. Despite the number of moving pixels found behind the window, no objects are detected in this part of the image. The hand is very well detected in the three frames shown. However the arm is not detected in the last frame because not enough moving pixels with similar motion are found. Unfortunately a part of the passenger compartment and the guardrail are detected as moving objects. The clusters found there were small but as there is no strong enough contours, the segmentation spills over. We believe that detecting this area is not such a big problem. Indeed, it is not moving during several successive frames. Hence, adding some temporal consistency and/or tracking that would reject static objects would probably allow the removal of such spurious detections. On figure 8, we show another part of this sequence with a different background. The results obtained, although not perfect, are rather encouraging for such a difficult sequence. The processing time for each of these 240x320 images is about 1 second on a standard computer and with a non optimized code.

The last results presented in this subsection were obtained on a grayscale sequence with a non moving camera showing some cars and pedestrians moving in a street. The difficulty here comes from the high level of noise and the low contrast present along the sequence. Furthermore, there is a large number of small moving objects.

Figure 6. Results on the water skier sequence for frames 74, 124, 144, 214, 232, 236 and 242

If, for these reasons, the segmentation (not shown) are disappointing, the detected clusters are nonetheless interesting. Figure 9 shows a result on one frame of the sequence. As can be seen on the moving pixel map, the amount of noise is important. Another difficulty faced by the segmentation part lies in the small number of pixels available in each cluster for learning the foreground distributions used in the energy functions. In figure 10, we show the bounding boxes corresponding to these

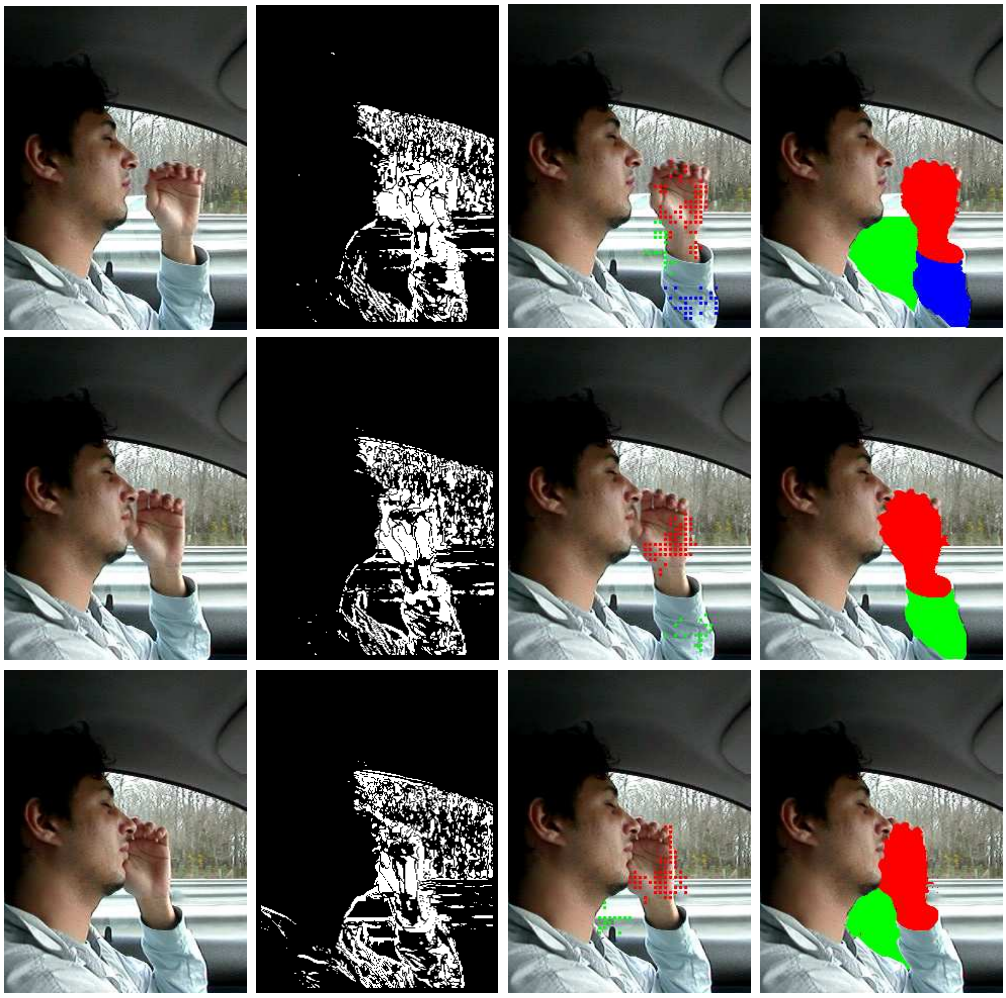


Figure 7. Results on the driver sequence for frames 15, 16, 17

clusters. Most of the largest pedestrians are detected. The one on the right and the middle is not, most of the time, because there are not enough moving pixels with valid optical flow vectors on him. Also, some clusters corresponding to noise are sometimes detected. However, as there is not any constancy in their detection, once again, a temporal consistency or a tracking scheme would probably flag these objects as false detections.

In this subsection we have shown promising results on three different kinds of sequences. To extend the validation of our method, comparisons with other motion detection methods are presented in the sequel.

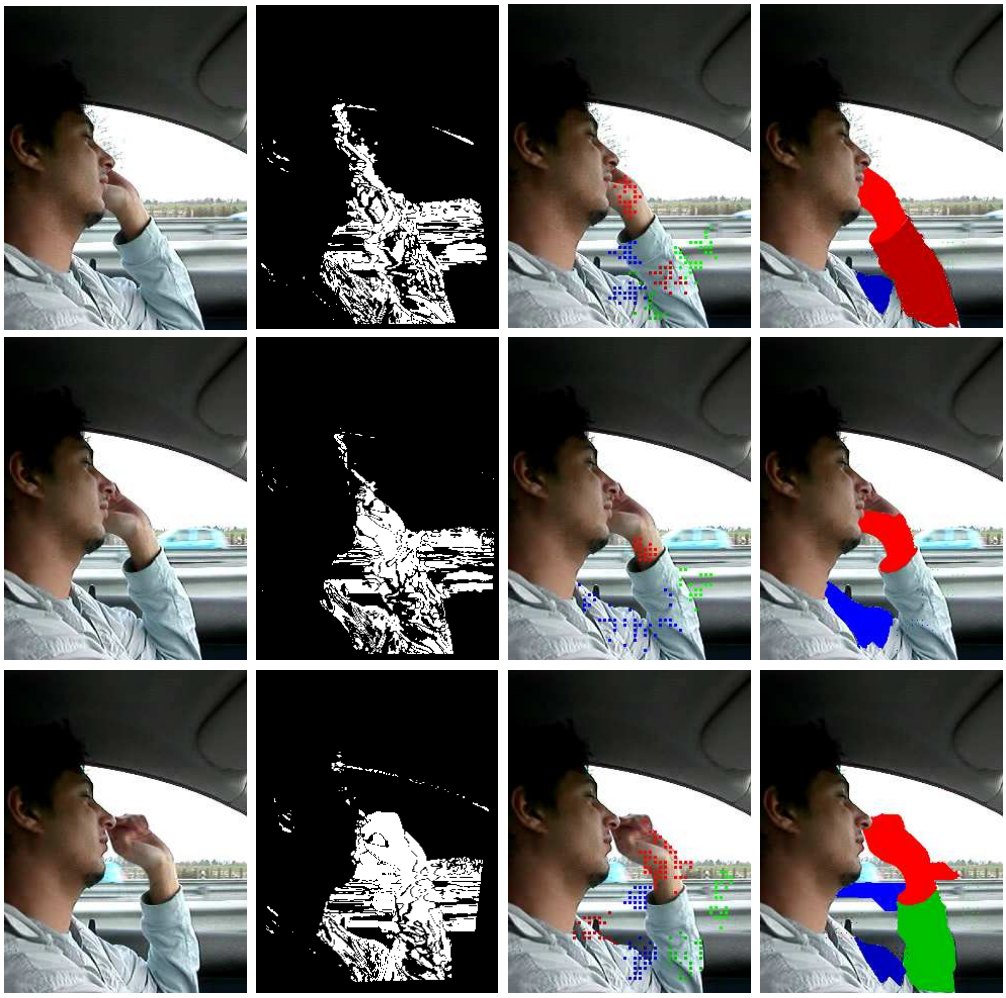


Figure 8. Results on the driver sequence for frames 48, 49, 50

5.2 Comparison with other motion detection methods

We now compare our algorithm to the background modeling method of Grimson and Stauffer [17] (with a history of 50 frames and mixtures of three Gaussians)and the non parametric background modeling of Elgammal *et al.* [18] (with a long term model, a selective update and $th = 10^{-6}$). We also show the masks of moving pixels resulting from the robust estimation and the compensation of the dominant motion with the technique of Odobez and Bouthemy [45]. In order to get motion detection results as binary maps, all the objects segmented by our approach are set as the moving areas of the image (white pixels). Also note that the sequences used here were taken by a fixed camera so that the background modeling methods can be applied.

The first sequence is a grayscale sequence of a pedestrian walking in front of water

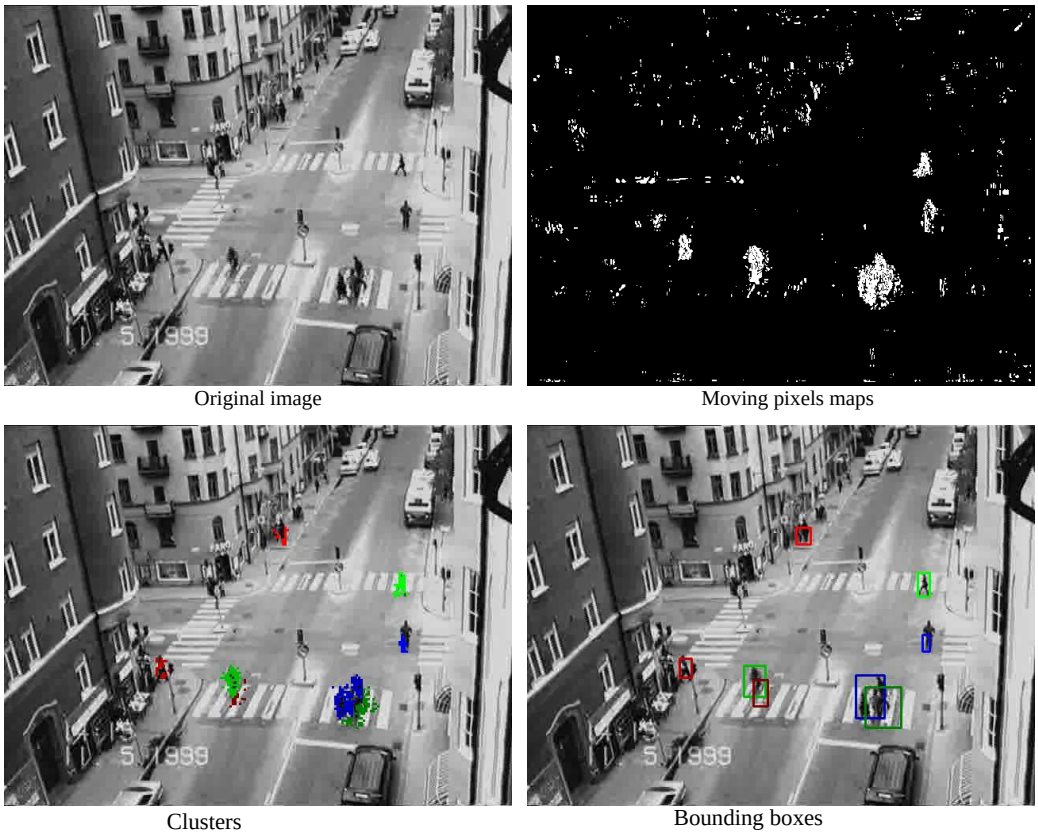


Figure 9. Results on frame 7 of the traffic sequence.

(figure 11). This is not a very difficult sequence for our algorithm as there are not many moving pixels detected in the water (second row of figure 11). While our method detects and segments well the person, the bike and the small cars moving on the bridge, the other methods detect several moving pixels in the water while there are many holes on the person mask. The problem of these false negatives in the detection mask (moving pixels) was mentioned in section 2. Here these false negatives result from the fact that the coat of the person is not highly textured and the person is walking slowly. Despite this lack of motion information at places, the segmentation step enables to recover the whole person as one unique moving object. As one can see, this comparison is not totally fair since our algorithm includes in the segmentation step a spatial regularization while the other methods do not incorporate spatial consistency. On figure 12 we show the results obtained with the other methods when a post processing step of regularization is added. This regularization is obtained by minimizing, with min-cut/max-flow, the following energy:

$$E(L) = \sum_{s \in \mathcal{P}} \delta(l_s^0, l_s) + \sum_{(s,r) \in \mathcal{V}} V_{\{s,r\}} \delta(l_s, l_r), \tag{21}$$

where l_s^0 is the initial label of pixel s. The first term of previous equation indicates that the cost of changing the initial label of a pixel is 1 and the second term is the smoothness term of (17). This figure highlights the holes in the masks of the objects for all the other methods, the quality of our results, and validates the en-

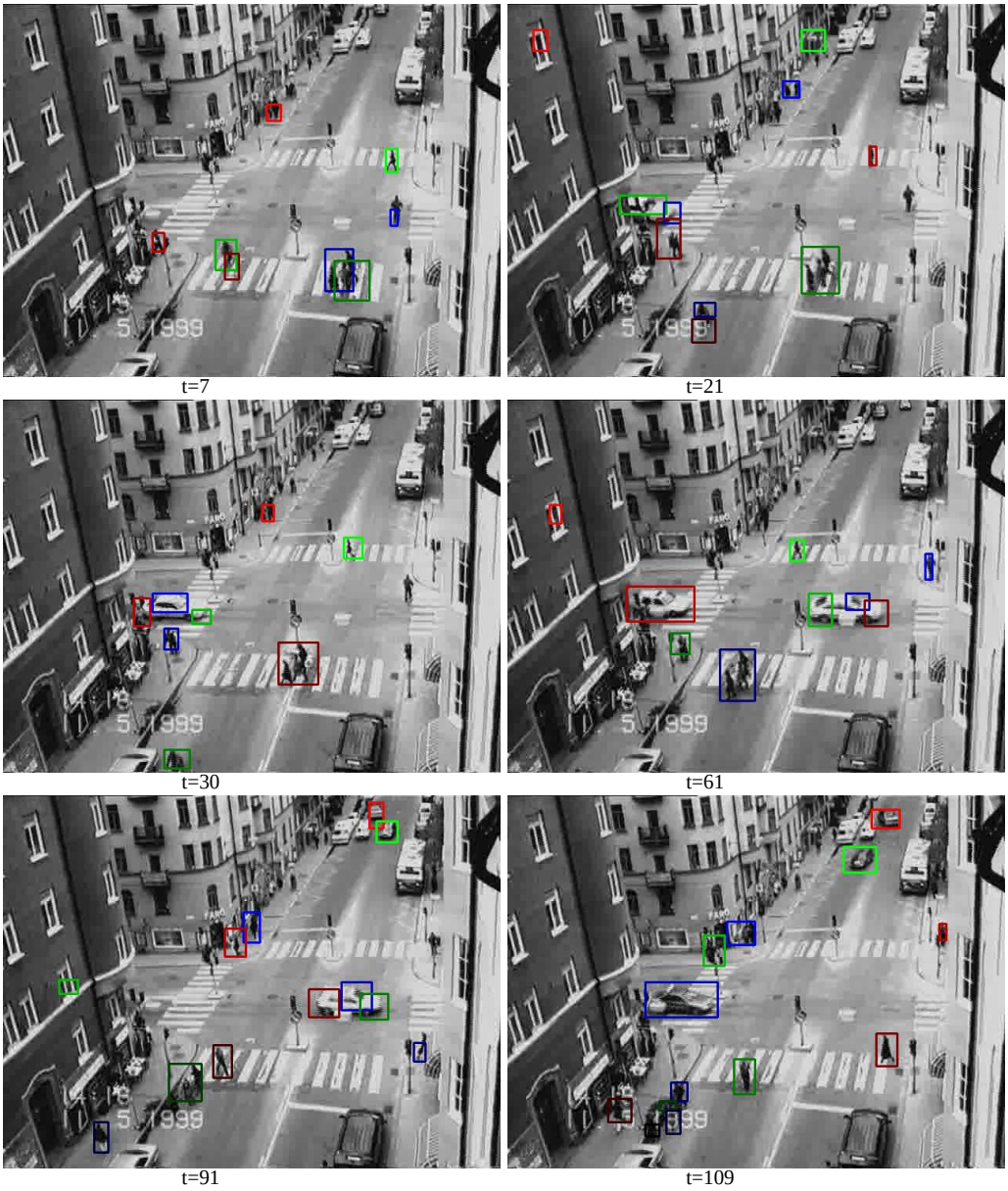


Figure 10. Results on the traffic sequence. Bounding boxes of detected clusters are shown.

ergy function used for our segmentation. We could not use the same energy for the other methods since it relies on some distributions of each objects (which are not delineated with the other methods). Furthermore, this energy forces some pixels to belong the segmented objects which is not possible for the binary maps resulting from the other methods.

The second sequence on which we compare our algorithm to others is a color sequence of two pedestrians walking behind some waving trees. This sequence is hard because of the complex movements of the trees and the frequent occlusions of the two persons. A spatial regularization has been added to the other methods such that a comparison is more complete. Grimson and Stauffer's method does not permit to

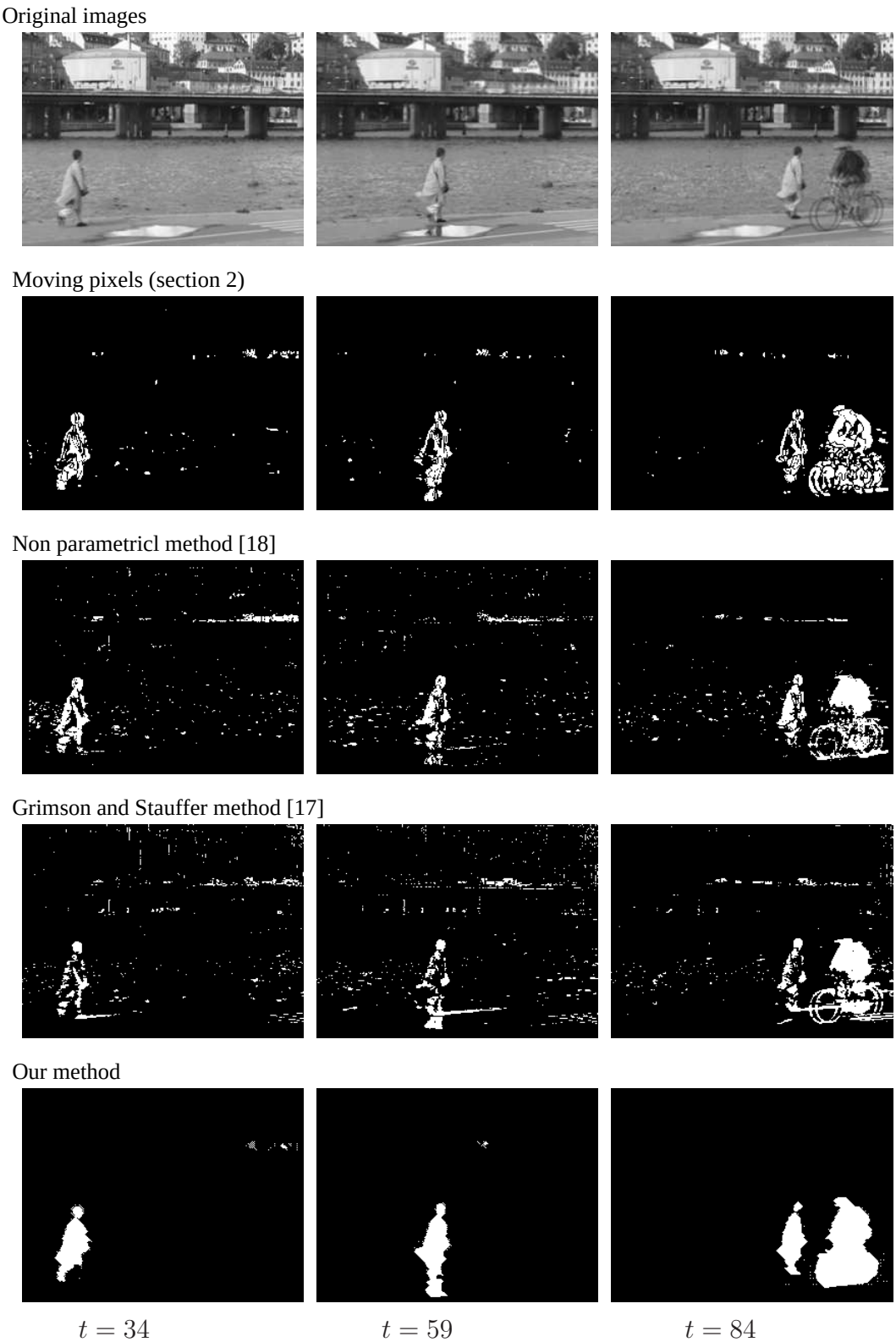


Figure 11. Detection masks on the person walking in front of water sequence for different methods.

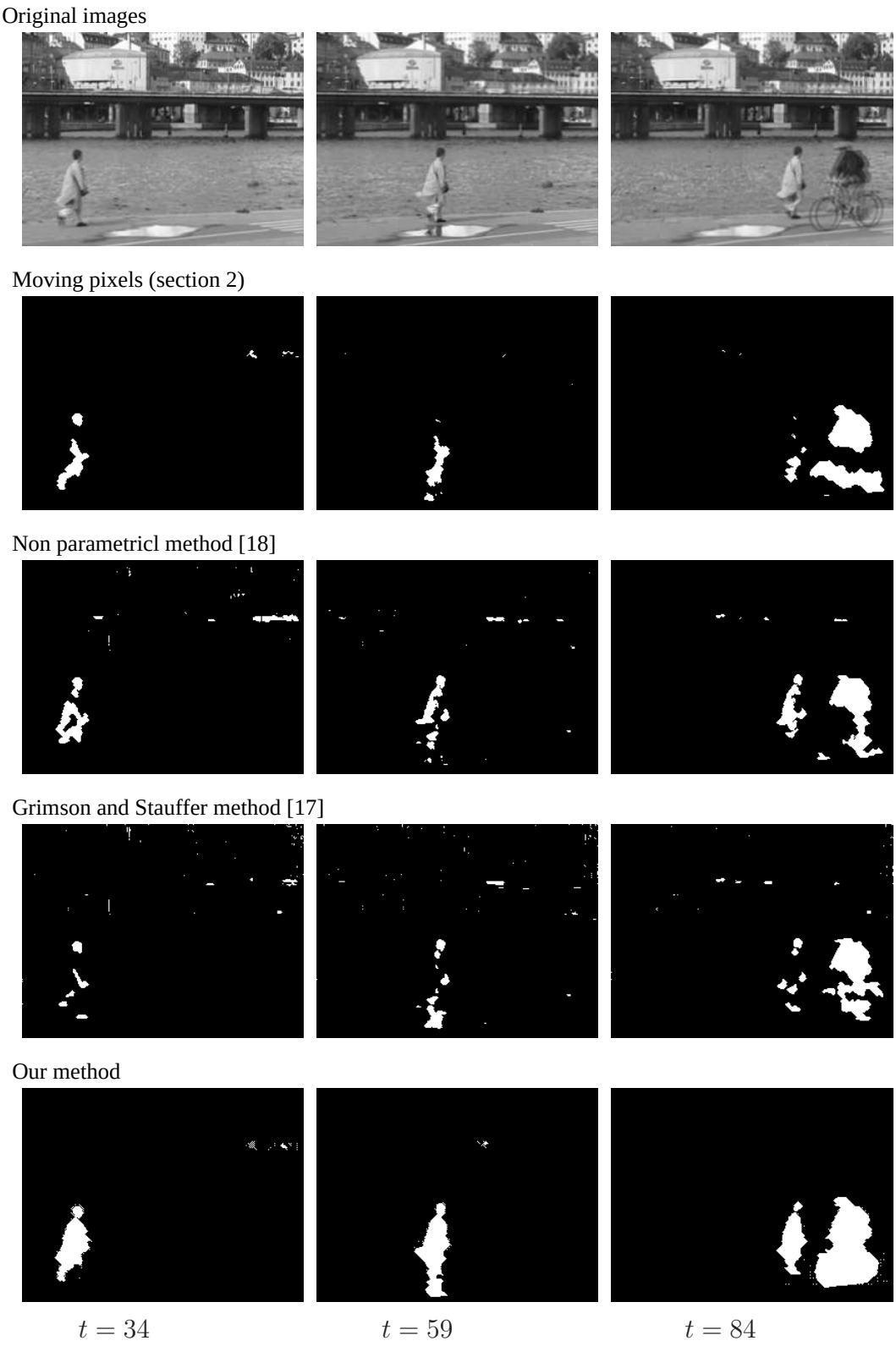


Figure 12. Detection masks on the person walking in front of water sequence for different methods all containing a spatial regularization.

distinguish the trees from the building behind. This mainly results from the use of a short history in the models. The results obtained by the non parametric method and the detection of moving pixels method are similar. These two methods do not detect the buildings but find few spurious motions within the trees. Our method also detects few objects within the trees but the segmentation permits to discard most of them. The masks of objects obtained by our algorithm are not perfect but seem more complete than the ones obtain with a non parametric approach or the detection of moving pixels approaches. The partial occlusions lead to many strong contours on the pedestrians which stop the segmentation flow. Also, due to these occlusions, the clusters are usually very small, which does not permit to construct good enough distributions of the interesting objects. Still the results obtained by our method, as compared to other motion detection techniques, are very promising.

6 Conclusion

A new technique to detect and segment moving objects in complex dynamic scenes shot by possibly moving cameras has been presented in this paper. The algorithm can be divided into three main steps. First, a set of points are selected and described in terms of color and motion, then these points are clustered according to their descriptors and finally a segmentation is obtained from these clusters. Each cluster corresponds to a moving area of the image. Several salient aspects of the contribution can be emphasized. First we only work on "moving" pixels belonging to a grid regularly spread on the whole image and with a motion estimate that passed a statistical test. Second, we use position, color and motion to describe each point. Third, we use a variable bandwidth mean shift using the balloon estimator and an automatic bandwidth selection to create the clusters. And finally, we use sparse motion data in an optimization framework to get the final segmentations of moving objects. A number of experiments on a large variety of complex videos demonstrate the validity and the potential of our approach.

It is worth emphasizing that the parameters involved in the preliminary motion computations (optic flow and parametric dominant motion) are fixed to the same values in all experiments, while the other parameters (for clustering and segmentation) are automatically selected.

Several directions can be investigated to extend and possibly improve our system. In the current system objects are segmented independently. Segmenting all the objects jointly by minimizing a multilabel energy function using the α-expansion algorithm [71, 68] could be envisaged.

Also, the proposed method does not make use of any temporal consistency. The introduction of such a consistency could be obtained either on a frame-to-frame

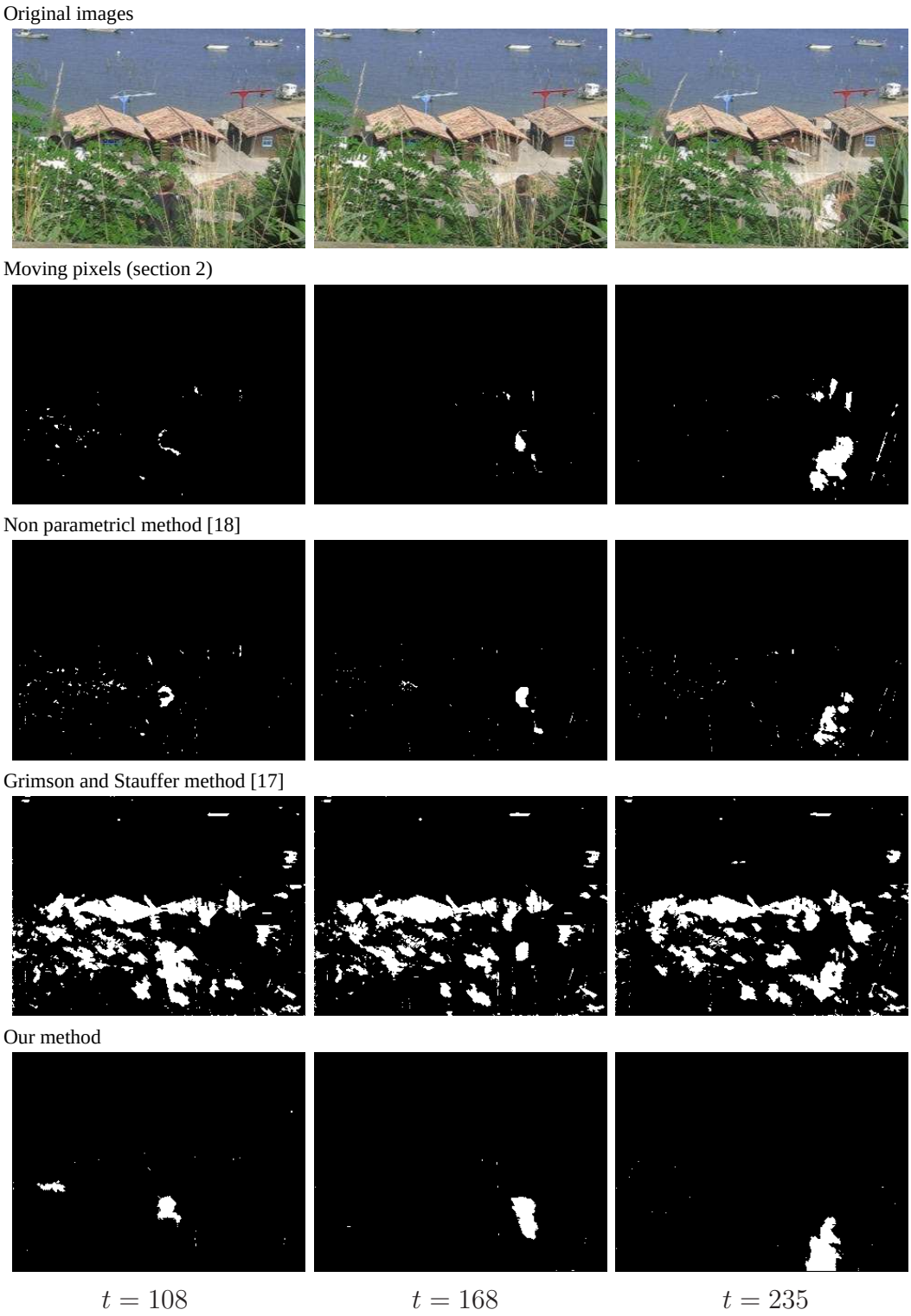


Figure 13. Detection masks on the waving trees sequence for different methods all containing a spatial regularization.

basis or within a tracker whose (re)initialization would rely on detection maps. To that end, we proposed in [72] a first method to combine the tracking and the

segmentation phase.

References

[1] A. Mitiche and P. Bouthemy. Computation and analysis of image motion: a synopsis of current problems and methods. *Int. J. Computer Vision*, 19(1):29–55, 1996.

[2] R.J. Radke, S. Andra, O. Al-Kofahi, and B. Roysam. Image change detection algorithms: a systematic survey. *IEEE Transactions on Image Processing*, 14(3):294–307, 2005.

[3] R. Jain and H.H. Nagel. On the analysis of accumulative difference pictures from image sequence of real world scenes. *IEEE Trans. Pattern Anal. Machine Intell.*, 1(2):206–214, 1979.

[4] J. Konrad. *Handbook of image and Video processing*. Academic press, 2000.

[5] R. Fablet, P. Bouthemy, and M. Gelgon. Moving object detection in color image sequences using region-level graph labeling. *Proc. Int. Conf. Image Processing*, October 1999.

[6] T. Aach and A. Kaup. Bayesian algorithms for change detection in image sequences using markov random fields. *Signal Processing: Image Communication*, 7(2):147–160, 1995.

[7] J.-M. Odobez and P. Bouthemy. *Separation of moving regions from background in an image sequence acquired with a mobile camera*. Kluwer Academic Publisher, 1997.

[8] Y. Hsu, H. Nagel, and G. Rekers. New likelihood test methods for change detection in image sequences. *Comput. Vision, Graphics, Image Proc.*, 26(1):73–106, 1984.

[9] P. Rosin. Thresholding for change detection. *Proc. Int. Conf. Computer Vision*, pages 274–279, 1998.

[10] T. Veit, F. Cao, and P. Bouthemy. A maximality principle applied to a contrario motion detection. *Proc. Int. Conf. Image Processing*, 2005.

[11] T. Kanade, R. Collins, A. Lipton, P. Burt, and L. Wixson. Advances in cooperative multi-sensor video surveillance, 1998.

[12] A. Cavallaro and T. Ebrahimi. Video object extraction based on adaptive background and statistical change detection. *in Proc. of SPIE VCIP*, 2000.

[13] S. Huwer and H. Niemann. Adaptive change detection for real-time surveillance applications. In *Third IEEE International Workshop on Visual Surveillance*, Dublin, 2000.

[14] M. Hötter, R. Mester, and M. Meyer. Detection of moving objects using a robust displacement estimation including a statistical error analysis. *Proc. Conf. Comp. Vision Pattern Rec.*, 1996.

[15] C.R. Wren, A. Azarbayejani, T. Darrell, and A. Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Anal. Machine Intell.*, 19(7):780–785, 1997.

[16] N. Friedman and S. Russell. Image segmentation in video sequences: A probabilistic approach. *Uncertainty in Artificial Intelligence*, pages 175–181, 1997.

[17] Y. Grimson, C. Stauffer, R. Romano, and L. Lee. Using adaptive tracking to classify and monitor activities in a site. *Proc. Conf. Comp. Vision Pattern Rec.*, 1998.

[18] A. Elgammal, D. Harwood, and L. Davis. Non-parametric model for background subtraction. *Proc. Europ. Conf. Computer Vision*, 2000.

[19] A. Mittal and N. Paragios. Motion-based background subtraction using adaptative kernel density estimation. *Proc. Conf. Comp. Vision Pattern Rec.*, 2004.

[20] Y. Ren, C. Chua, and Y. Ho. Statistical background modeling for non-stationary camera. *Pattern Recognition Letters*, 24(1-3):183–196, January 2003.

[21] N. Paragios and G. Tziritas. Adaptive detection and localization of moving objects in image sequences. *Signal Processing: Image Communication*, 14:277–296, 1999.

[22] Y. Sheikh and M. Sha h. Bayesian modeling of dynamic scenes for object detection. *IEEE Trans. Pattern Anal. Machine Intell.*, 27(11):603–619, 2005.

[23] S. Mahamud. Comparing belief propagation and graph cuts for novelty detection. *Proc. Conf. Comp. Vision Pattern Rec.*, 2006.

[24] K. Karmann and A. Brand. *Time-varying image processing and moving object recognition*. Elsevier Science Publish., 1990.

[25] D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In *Proc. Europ. Conf. Computer Vision*, 1994.

[26] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *Proc. Int. Conf. Computer Vision*, 1999.

[27] G. Doretto, A. Chiuso, Y.N. Wu, and S. Soatto. Dynamic textures. *Int. J. Computer Vision*, 51(2):91–109, 2003.

[28] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. *Proc. Int. Conf. Computer Vision*, 2003.

[29] T. Darrel and A. Pentland. Robust estimation of a multi-layered motion representation. *IEEE Workshop on Visual Motion*, 1991.

[30] S. Ayer and H. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding. *Proc. Int. Conf. Computer Vision*, 1995.

[31] A. Jepson and M. Black. Mixture models for optical flow computation. *Proc. Conf. Comp. Vision Pattern Rec.*, 1993.

[32] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Trans. on Image Processing Special Issue*, 3(5):625–638, 1994.

[33] Y. Weiss. Smoothness in layers: Motion segmentation using nonparametric mixture estimation. *Proc. Conf. Comp. Vision Pattern Rec.*, 1997.

[34] J. Xiao and M. Shah. Accurate motion layer segmentation and matting. *Proc. Conf. Comp. Vision Pattern Rec.*, 2005.

[35] R. Vidal and Y. Ma. A unified algebraic approach to 2-d and 3-d motion segmentation. *Proc. Europ. Conf. Computer Vision*, 2004.

[36] R. Vidal and D. Singaraju. A closed form solution to direct motion segmentation. *Proc. Conf. Comp. Vision Pattern Rec.*, 2005.

[37] S. Pundlik and S. Birchfield. Motion segmentation at any speed. *Proc. of the British Machine Vision Conf.*, 2006.

[38] K. Kim, D. Harwood, and L. Davis. Background updating for visual surveillance. *Int. Symposium on Visual Computing.*, 2005.

[39] R. Wildes. A measure of motion salience for surveillance applications. *Proc. Int. Conf. Image Processing*, 1998.

[40] L. Wixson. Detecting salient motion by accumulating directionally-consistent flow. *IEEE Trans. Pattern Anal. Machine Intell.*, 22(8):774–780, 2000.

[41] Y.L. Tian and A. Hampapur. Robust salient motion detection with complex background for real-time video surveillance. *Workshop on Motion and Video Computing*, 2005.

[42] S. Zhu, Q. Avidan and K.-T. Cheng. Learning a sparse, corner-based representation for time-varying background modeling. *Proc. Int. Conf. Computer Vision*, 2005.

[43] D. Comaniciu and P. Meer. Mean shift: A robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Machine Intell.*, 24(5):603–619, 2002.

[44] M.J. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, 1996.

[45] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *J. Visual Com. and Image Representation*, 6(4):348–365, December 1995.

[46] H. Sawhney and S. Ayer. Compact representations of videos through dominant and multiple motion estimation. *IEEE Trans. Pattern Anal. Machine Intell.*, 18(8):814–830, 1996.

[47] J. Barron, D. Fleet, S. Beauchemin, and T. Burkitt. Performance of optical flow techniques. *CVPR*, 1992.

[48] S. Beauchemin and J. Barron. The computation of optical flow. *ACM Computing Surveys*, 27(3):433–467, 1995.

[49] B.D. Lucas and T. Kanade. An iterative technique of image registration and its application to stereo. *Proc. Int. Joint Conf. on Artificial Intelligence*, 1981.

[50] B. Horn and B. Schunck. Determining optical flow. *Artif. Intell.*, 17(1-3):185–203, 1981.

[51] M.J. Black and P. Anandan. A framework for the robust estimation of optical flow. *Proc. Int. Conf. Computer Vision*, 1993.

[52] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992.

[53] D.G. Lowe. Object recognition from local scale-invariant features. *Proc. Int. Conf. Computer Vision*, 1999.

[54] R. Kjeldsen and J. Kender. Finding skin in color images. *International Conference on Automatic Face and Gesture Recognition*, 1996.

[55] S. Singh, D. Chauhan, M. Vatsa, and R. Singh. A robust skin color based face detection algorithm. *Tamkang Journal of Science and Engineering*, 6(4):227–234, 2003.

[56] J. Terrillon and S. Akamatsu. Comparative performance of different chrominance spaces for color segmentation and detection of human faces in complex scene images. *International Conference on Automatic Face and Gesture Recognition*, 2000.

[57] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM Comput. Surv.*, 31(3):264–323, 1999.

[58] D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and Data-Driven scale selection. *Proc. Int. Conf. Computer Vision*, 1, 2001.

[59] G. Terrell and D. Scott. Variable kernel density estimation. *The Annals of Statistics*, pages 1236–1265, 1992.

[60] D. Loftsgaarden and C. Quesenberry. A nonparametric estimate of a multivariate density function. *Annals of Mathematical Statistics*, 36:1049–1051, 1965.

[61] A. Bugeau and P. Pérez. Bandwidth selection for kernel estimation in mixed multi-dimensional spaces. *Technical report, IRISA*, (PI 1852), 2007.

[62] D. Comaniciu. An algorithm for data-driven bandwidth selection. *IEEE Trans. Pattern Anal. Machine Intell.*, 25(2):281–288, 2003.

[63] N. Pal and S. Pal. A review on image segmentation techniques. *Pattern Recognition*, 26(9):1277–1294, 1993.

[64] L. Lucchese and S. Mitra. Color image segmentation: A state-of-the-art survey. *Proc. of the Indian National Science Academy*, 67:207–221, 2001.

[65] A. Meziane. Digital images segmentation: a state of art of the different methods. *Revue d'Information Scientifique et Technique*, 12(1):105–120, 2002.

[66] Y. Zhang. Advances in image and video segmentation. In *IRM Press*, 2006.

[67] D. Greig, B. Porteous, and A. Seheult. Exact maximum a posteriori estimation for binary images. *J. Royal Statist. Soc.*, 51(2):271–279, 1989.

[68] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Anal. Machine Intell.*, 23(11):1222–1239, 2001.

[69] Y. Boykov and M. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images. *Proc. Int. Conf. Computer Vision*, 2001.

[70] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[71] Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *Proc. Conf. Comp. Vision Pattern Rec.*, 1998.

[72] A. Bugeau and P. Pérez. Joint tracking and segmentation of objects using graph cuts. *Advanced Concepts for Intelligent Vision Systems.*, 2007.