Detection and segmentation of moving objects in complex scenes

Aurelie Bugeau, Patrick Pérez

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Detection and segmentation of moving objects in complex scenes

Aurélie Bugeau — Patrick Pérez

N° 6282

September 2007

Thèmes COM et COG

Rapport de recherche

ISSN 0249-6399 ISRN INRIA/RR--6282--FR+ENG

Detection and segmentation of moving objects in complex scenes

Aurélie Bugeau , Patrick Pérez

Thèmes COM et COG — Systèmes communicants et Systèmes cognitifs
Projet Vista

Rapport de recherche n° 6282 — September 2007 — 50 pages

Abstract: Detecting and segmenting moving objects in dynamic scenes is a hard but essential task in a large number of applications such as surveillance. Most existing methods only give good results in the case of persistent or slowly changing background, or if both the objects and the background can be characterized by simple parametric motions. This paper aims at detecting and segmenting foreground moving objects in the absence of such constraints. The sequences we consider have 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, moving points are selected within a sub-grid of image pixels. A descriptor is associated to each of these points. Clusters of points are then formed using a variable bandwidth mean shift with automatic bandwidth selection. Finally, segmentation of the object associated to a given cluster is performed using Graph cuts. Experiments and comparison to other motion detection methods on challenging sequences show the performance of the proposed method and its utility for video analysis in complex scenes.

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

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 Rennes Cedex (France)
Téléphone : +33 2 99 84 71 00 — Télécopie : +33 2 99 84 71 71

Détection et segmentation d'objets en mouvement dans des scènes complexes

Résumé : De nombreuses applications en vision par ordinateur et en surveillance nécessitent la détection et la segmentation des objets en mouvement. La plupart des méthodes existantes ne donnent de bons résultats que pour des fonds statiques ou peu changeants, ou si le fond et les objets sont rigides et ont un mouvement affine 2D. Le but de ce papier est de directement détecter les objets en mouvement dans des séquences complexes n'ayant pas ces caractéristiques. Les vidéos considérées ici ont un fond dynamique, avec de forts changements d'illumination et de faibles contrastes, et peuvent avoir été prises par une caméra en mouvement. La méthode proposée se divise en trois étapes principales. Tout d'abord un ensemble de points en mouvement est sélectionné parmi une grille de pixels uniformément répartis sur toute l'image. Tous ces points sont associés à un descripteur. La deuxième étape consiste à former des groupes de ces points représentant chacun un objet en mouvement. Ces partitions sont obtenues par un algorithme mean shift à noyau variable avec une sélection automatique de la taille du noyau. Enfin, à partir de ces groupes de points, la segmentation des objets est donnée en minimisant une énergie par coupure de graphe. Des résultats et comparaisons avec d'autres méthodes de segmentation de mouvement montrent l'efficacité de la méthode proposée.

Mots-clés : détection de mouvement, segmentation, partitionnement mean shift, coupure de graphe

Contents

1 Introduction

Detection of moving objects in sequences is an essential step for video analysis. It becomes a difficult task in the presence of a dynamic background. We are interested in very challenging sequences containing a complex motion in the background. This motion can also have a high amplitude. Furthermore, contrast between background and interesting objects can be small. Like in many applications in video analysis, the algorithms have to be robust to illumination and point of view changes. The last particularity about our sequences is that the camera taking the scene can move.

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 [39, 47]. Here, we divide these methods into four categories: thresholding, background modeling, layers extraction and finally

saliency based methods.

Tresholding methods First works on motion detection were based on adjacent frames difference [28]. 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 luminosity. Many methods have been developed to decide whether or not a pixel has moved. The decision can be directly made independently on each pixel [35] or on small blocks of pixels [18]. 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 [1]. In [43], Markov Random fields 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 test hypotheses [25]. More complex methods are proposed in [49] for modeling the spatial distribution of either noise or signal and selecting the appropriate threshold. In [58] 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 mostly sensitive to noise and to illuminations changes. When the number of frames in the sequence is high and there is not much 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 [12] [26] [30]. The foreground pixels are determined as those for which the intensity value is far from the mean background model and are clustered into objects. A variable number of Gaussian distributions corresponding to each different foreground object can be added. It was used by [24] for generic objects and by [66] 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 [19, 21]. When changes in the background are too fast, the variance of the Gaussians becomes too large and non parametric approaches are more suited. In [17], Gaussian kernels calculated on the past frames are used to model the density 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. A lot of outdoor scenes exhibit a persistent

motion which is well modeled by optical flow and a non parametric algorithm that combines color and flow features can be used [40]. As optical flow can not be computed when there is no intensity gradient, the authors have chosen to use kernels with variable bandwidth. In [48] 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 [44] in which the background and foreground maps are forced to be a Markov random field. 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 [52] with a MAP-MRF modeling of both foreground and background. This method has been extended to novelty detection in [38].

Predictive methods use a dynamical model to predict the value of a pixel from previous observations. A Kalman filter based approach that models the dynamics of the intensity at a particular pixel can be used [31, 34]. In [57] an algorithm called wallflower is described. It uses a simpler version of the Kalman filter called *Weiner filter* to predict a pixel's current value from its k previous values. Pixels whose prediction error is high are classified as changed pixels. Recent methods are based on more complicated models. For example, in [16] and [68], 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 the spatio-temporal image data. This leads to the layer approach [15] that tries to fit a mixture of motion models to the entire image. Layers are then found by associating each pixel to the model it belongs to. In many papers [2, 29, 62, 63] 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 [67] graph cuts have been used to extract these models or layers. After a number of seed regions are determined using two frames correspondences, these seed regions are first extended thanks to a graph cuts segmentation method. The resulting initial regions are then merged into layers according to motion similarities. This method requires the scene to be induced by multiple 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) [59]. It was adapted to both static and dynamic scenes [60]. Recently, in [46], 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 [32] 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 [64], 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 [65] and [56] 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 [56], the accumulation of flow motions is done during 10 frames. A fusion of background modeling and saliency was proposed in [69]. 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 Overview of the paper

In this paper, 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. The “background” is composed of both the passenger compartment and what is behind the windows. 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 harm and the guardrail. Our work does not aim at modeling the background or at finding every layer but only at detecting moving foreground objects. We define these objects as groups of pixels salient for both motion and color. Our algorithm can be divided in three main steps. First, in section 2, the camera motion is computed and a subgrid of “moving” pixels, *i.e* not belonging to camera motion, is selected. A descriptor is defined to characterize each points of this grid. They are then merged into clusters consistent for both color and motion (section 3), using mean shift filtering algorithm [14]. From the clusters, the complete pixel-wise segmentation of moving objects is found using a MAP-MRF framework (section 4). Finally, section 5 presents some experimental results.

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

Figure 1: Exemple of a driver sequence

2 Point selection and description

In this paper, similarly to [65] and [56], the moving objects detection will only be performed on several points and their neighborhood. We have preferred the computational lightness and the noise robustness of these methods to the accuracy of the pixel-wise approaches. The first step of our algorithm is then to build a sub-grid of moving pixels and to select a descriptor for these pixels. 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.

First, let us introduce 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 a one dimensional vector of 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: $\mathbf{z}_t(s) = (\mathbf{z}_t^{(G)}(s), \mathbf{z}_t^{(M)}(s))$.

2.1 Sensor motion

Moving pixels are the pixels not belonging to the camera motion. We only work on moving pixels because we aim at detecting moving objects. We assume that the apparent motion induced by the physical motion of the camera is dominant in the image (among various movements in the image field, it is the one that concerns the larger number of pixels) and is well approximated by a 2D affine motion field. The parametric flow vector $\mathbf{w}_\theta(s)$ at location $s = (x, y)$ reads:

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

Different methods are available to estimate the parameters of such a model [6, 42, 51]. We use the robust real-time multiresolution algorithm described in [42]. 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:

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

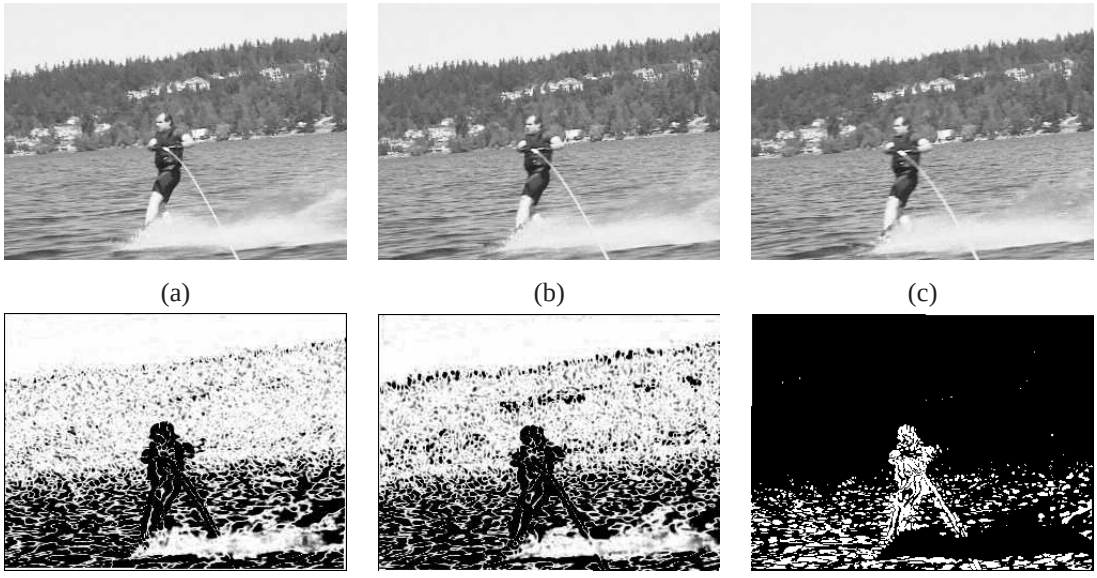
where $\rho(x)$ is an M-estimator and ζ_t is a global intensity shift that accounts for global illumination changes. The minimization is done through a multiscale Gauss-Newton method that yields a succession of reweighted least-squares problems. The auxiliary weight maps of the M-estimator will be denoted 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 this map. A pixel is considered as "moving" at time t if it is an outlier to the dominant motion at times t and $t-1$:

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

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 no moving object is lost using such 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. A 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 segmentation step where the whole pixel grid is considered (see section 4).

2.2 Grid construction

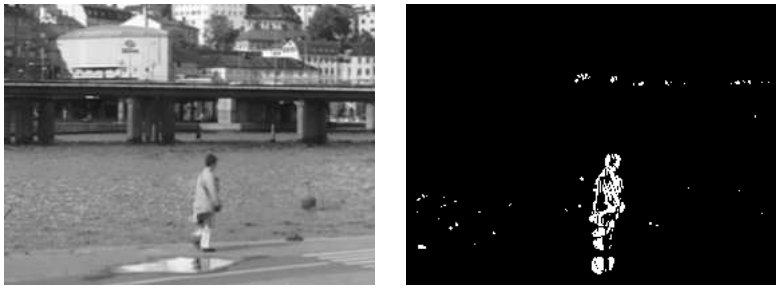
The goal of the algorithm is to build groups of pixels consistent both for motion and for some photometric or colorimetric features. These groups must correspond to interesting moving objects. In [69], moving objects are found using corners, detected with the Harris corner detector. The authors justify 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 signicatively different from one frame to the other. Finally, corner detection adds one stage of calculation and requires two thresholds.



(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.

As no *a priori* is assumed on the shape and texture of objects, we have chosen to use points of arbitrary type. Hence, we only use a grid of points regularly spread on the image. As the purpose is to detect moving objects, the simple pixel-wise motion detector from previous subsection is used to restrict this step to the grid subset:

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

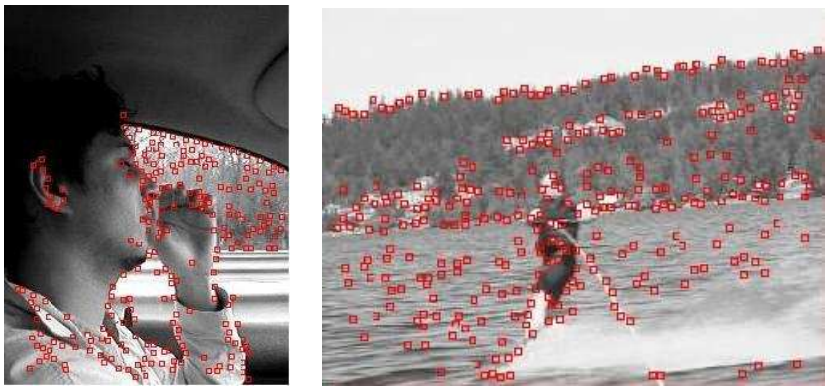


Figure 4: Result of the Harris corner detector on frame 16 of the driver sequence and frame 108 of the water skier sequence.

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 in the 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). Next step of the algorithm can become computationally expensive if the number of points of the grid is too large. An important thing to note is that $N_{\mathcal{G}}$ may depend on the number m of "moving" pixels in the image, $m = \sum_{s \in \mathcal{P}} M_t(s)$. To limit the computational cost for clusters creation, we fix the number of moving points to $n_{\mathcal{G}}$ (500 in our experiments) that will be approximatively kept in further steps of the algorithm. 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 points are 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. An object is defined as a moving and compact area over which the values of displacement and photometry are nearly constant. Color is not sufficient because the contrast between an object and the background can be small, nor is 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

As we try to detect moving objects, an essential feature is the displacement of the selected points. Several types of methods exist for computing optical flow. Good reviews are given in [3, 4]. We concentrate on the Lucas and Kanade algorithm [37], with an incremental multiscale implementa-

tion. 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). In order to use classical notations, in this subsection I denotes the intensity function, $I(x, y, t)$ is the intensity value at point $s = (x, y)$ and at time t, and d_x and d_y are the horizontal and vertical components of the apparent displacement between instants t and $t + 1$. The initial assumptions are that image intensities are approximatively constant under motion for at least a short period of time and that motions are small:

$$I(x + d_x, y + d_y, t + 1) \simeq I(x, y, t) \ . \tag{5}$$

Using a Taylor approximation, previous equation leads to the brightness constancy equation:

$$\frac{\partial I}{\partial x} d_x + \frac{\partial I}{\partial y} d_y + \frac{\partial I}{\partial t} = 0 \ . \tag{6}$$

This equation has two unknowns and is thus insufficiently constrained. This is the aperture problem: edges parallel to the motion do not convey motion information. Only the motion component in the direction of the local gradient of the intensity function can be computed. To find the optical flow some additional constraint must be added. The solution of Lucas and Kanade is to use a local neighborhood $\mathcal{V}(x, y)$ and to assume that the flow is constant in this small region of size n. This leads to an overdetermined system of equations solved using a least squares method. The flow is then obtained by solving:

$$\text{argmin}_{(d_x,d_y)} \sum_{(x_i,y_i)\in\mathcal{V}(x,y)} \Big[\frac{\partial I}{\partial x}(x_i, y_i, t)d_x + \frac{\partial I}{\partial y}(x_i, y_i, t)d_y + \frac{\partial I}{\partial t}(x_i, y_i, t)\Big]^2 \ . \tag{7}$$

Unfortunately the associated 2x2 linear system is regular and well conditioned only for "textured" neighborhoods. 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 left apart from the grid:

$$\mathcal{G} = \Big\{ s = \Big(\frac{k.w}{N_\mathcal{G}}, \frac{l.h}{N_\mathcal{G}}\Big) \mid M_t(s) = 1 \ \& \ \exists (x_i, y_i) \in \mathcal{V}\big(\frac{k.w}{N_\mathcal{G}}, \frac{l.h}{N_\mathcal{G}}\big), |\nabla I(x_i, y_i)| \neq 0 \Big\} \ . \tag{8}$$

The brightness constancy equation is only valid for small motion. In order to access large motions, an incremental multiscale approach can be used. A Gaussian pyramid is derived from each image. The optical flow constraint holds for coarsest scales of the pyramids even when the motion is large. Estimated motion for large scales are used as initial estimates for lower scales.

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 [23] that adds a smoothness term to regularize over the whole image or the robust estimation of Black and Anandan [5] to get a better estimation. However these algorithms are more computationally expensive and we do not aim at having a dense and very precise estimation over the whole image.

The computation of flow using gradient based methods fails when the brightness constancy is not satisfied and when a point does not move as its close neighbors. Indeed, the two basic assumptions are not valid in such cases. 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 validate values of displacement, 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). 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. A value near zero indicates that the two samples are uncorrelated. Unfortunately, the correlation does not take into account the individual distributions of X and Y. Hence it is a poor statistics for deciding whether or not two distributions are really correlated.

In statistics, a result is called significant if it is unlikely to have occurred by chance. Statistical tests exist to assess this correlation. They are based on two principal notions: the null hypotheses and the alpha risk (or fixed level testing). In 1928, Neyman and Pearson [41] argued that rather than having a single hypothesis that should be rejected or not, it is better to choose between two hypotheses. One of them is the null hypothesis and the other one is the alternative hypothesis. The goal of statistical tests is then to decide if the null hypothesis should be rejected. An alpha risk can be attached to the null hypothesis. This risk is defined as the risk of rejecting the null hypothesis when in fact it is true. It is stated in terms of probability and corresponds to the confidence level of a statistical test. In our case, the null hypothesis H_0 asserts that the two data samples X and Y are uncorrelated.

A statistical test related to the linear correlation is the so-called "p-value" (or observed significance level). To get the p-value, the T-statistics (or *Student's statistics*) has to be computed:

$$T = \gamma \sqrt{\frac{n-2}{1-\gamma^2}}. \tag{9}$$

The parameter n is the size of the sample X and Y and γ the normalized cross correlation. A T-statistics near zero is evidence under the null hypothesis that there is no correlation between the two samples. Assuming that H_0 is true, T is well represented by a Student's distribution defined by:

$$A(x) = \frac{1}{\sqrt{n-2}\, B(\frac{1}{2}, \frac{n-2}{2})} \int_{-x}^{x} \left(1 + \frac{y^2}{n-2}\right)^{\frac{3-n}{2}} dy \ , \tag{10}$$

where B is the beta function given by:

$$B(a, b) = B(b, a) = \int_0^1 x^{a-1}(1-x)^{b-1} dx \ . \tag{11}$$

The p-value is finally the probability, when the null hypothesis is true, that the absolute value of the T-statistics would exceed the alpha risk. It is equal to $A(|T|)$. A small p-value means that the null hypothesis is false and that the two data samples are in fact correlated. If one wants to limit to 5% the alpha risk, then data are assumed correlated if the p-value is lower than 0.05. For more information on the theory and the implementation of the p-value, we refer to [45].

If the p-value obtained for a point s is larger than 0.05, the flow for this point is considered as non valid. The point s will then not be processed in next steps of the algorithm. Finally, recalling that $s' = (x + d_x, y + d_y)$, a new grid

$$\mathcal{G} = \Big\{ s = \big(\frac{k.w}{N_\mathcal{G}}, \frac{l.h}{N_\mathcal{G}}\big) \mid M_t(s) = 1 \,\&\, \exists (x_i, y_i) \in \mathcal{V}\big(\frac{k.w}{N_\mathcal{G}}, \frac{l.h}{N_\mathcal{G}}\big), |\nabla I(x_i, y_i)| \neq 0 \,\&\, \text{p-value}(s, s') < 0.05 \Big\} \tag{12}$$

is obtained with a flow vector $\mathbf{z}_t^{(M)}(s) = (d_x, d_y)$ associated to each of its point s. The size of the grid $\mathcal{G}$ will be denoted as $|\mathcal{G}|$.

The influence of the p-value is shown on figure 5.

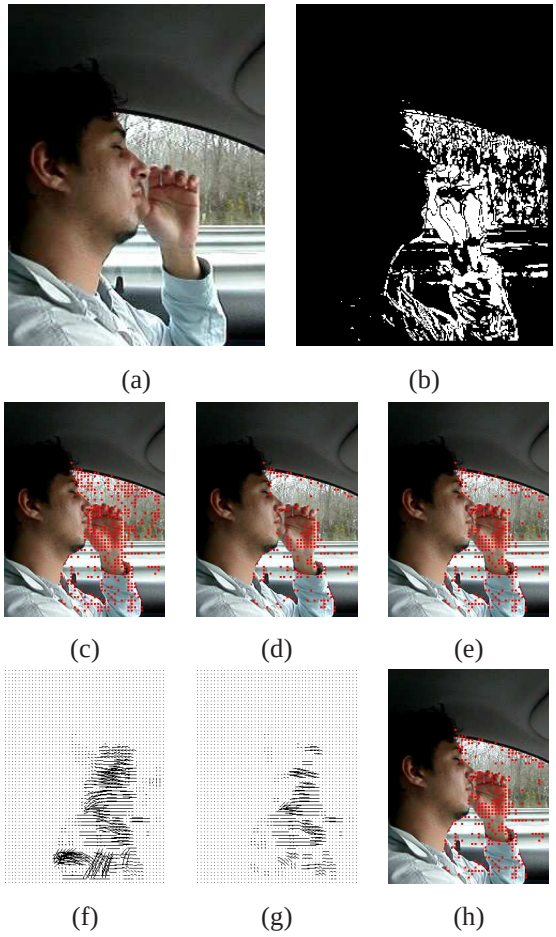


Figure 5: Result of grid construction (second row) and associated motion fields (third row) on frame 16 of the driver sequence. (a) Original image. (b) Moving pixels. (c) Final grid without flow vectors validation (associated motion field is shown on image (f)). (d) Final grid obtained by a correlation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

The first row shows the original image and the set of moving pixels. The second row shows the final grid 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 final grid 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 that yield these grids 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. The grid finally contains 420 pixels when 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. They are different for grayscale and color sequences. To be robust to noise, the features are computed over the neighborhood of each point of the grid defined in previous subsection.

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 $s = (x, y)$. As the contrast between the object and the background can be small, this feature is not sufficient. Another interesting information is the texture. A lot of definitions and characterizations of texture can be found in literature. Here it will be 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 $s' = s + (d_x, d_y)$. Finally, for grayscale sequences, the descriptor at each individual valid point $s = (x, y)$ of the grid, indexed by i ($i = 1 \dots |\mathcal{G}|$), is:

$$\mathbf{x}^{(i)} = (\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \mathbf{x}_3^{(i)}), \tag{13}$$

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 will be added. However, we observed that the three color channels RGB do not give the best representation of our images. Indeed, they are highly correlated and this representation is 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 [33, 53]. Therefore, it is interesting to use a color system representing the chrominance instead of the classical RGB system. Most chrominance spaces give the same results on skin detection [55]. In this paper we use the YUV color space. This choice proves appropriate for various types of sequences. Here again, to include some simple temporal consistency, we add image $t+1$ chrominance values of the corresponding point. Finally, for color sequences, the descriptor at each individual valid point indexed by i ($i = 1 \dots |\mathcal{G}|$) of the grid is:

$$\mathbf{x}^{(i)} = (\mathbf{x}_1^{(i)}, \mathbf{x}_2^{(i)}, \mathbf{x}_3^{(i)}), \tag{14}$$

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 a grid of valid points has been chosen and described, we address the problem of grouping the points into clusters. Many data clustering methods have been described in the literature. A good review on classic clustering techniques can be found in [27]. An appealing technique to extract the clusters is the mean shift algorithm, which does not require to fix the (maximum) number of clusters. On the other hand the kernel bandwidth and its shape have to be chosen or estimated for each dimension.

3.1 Fixed bandwidth 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 maximum of its density. This technique, which we now summarize, is well described in [14].

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:

$$\begin{aligned}\widehat{f}(\mathbf{x}) &= \frac{1}{n|\mathbf{H}|^{1/2}} \sum_{i=1}^{n} K(\mathbf{H}^{-1/2}(\mathbf{x}-\mathbf{x}^{(i)})) \\ &= \frac{c_k}{n|\mathbf{H}|^{1/2}} \sum_{i=1}^{n} k(\|\mathbf{H}^{-1/2}(\mathbf{x}-\mathbf{x}^{(i)})\|^2)\end{aligned} \tag{15}$$

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

$$g(\mathbf{x}) = -k'(\mathbf{x})$$

the density gradient reads:

$$\nabla\widehat{f}(\mathbf{x}) = \mathbf{H}^{-1}\ \widehat{f}(\mathbf{x})\ \mathbf{m}(\mathbf{x}) \tag{16}$$

where $\mathbf{m}$ is the "mean shift" vector,

$$\mathbf{m}(\mathbf{x}) = \frac{\sum_{i=1}^{n} \mathbf{x}^{(i)}\ g\big(\|\mathbf{H}^{-1/2}(\mathbf{x}-\mathbf{x}^{(i)})\|^2\big)}{\sum_{i=1}^{n} g\big(\|\mathbf{H}^{-1/2}(\mathbf{x}-\mathbf{x}^{(i)})\|^2\big)}\ -\ \mathbf{x}\quad . \tag{17}$$

Using exactly this displacement vector at each step of an iterative search guaranties convergence to the local maximum of the density [14]. With a *d*-variate Gaussian kernel, equation (17) becomes

$$\mathbf{m}(\mathbf{x}) = \frac{\sum_{i=1}^{n} \mathbf{x}^{(i)}\ \exp(-\frac{1}{2}D^2(\mathbf{x},\mathbf{x}^{(i)},\mathbf{H}))}{\sum_{i=1}^{n} \exp(-\frac{1}{2}D^2(\mathbf{x},\mathbf{x}^{(i)},\mathbf{H}))}\ -\ \mathbf{x} \tag{18}$$

where

$$D^2(\mathbf{x},\mathbf{x}^{(i)},\mathbf{H}) \equiv (\mathbf{x}-\mathbf{x}^{(i)})^T\mathbf{H}^{-1}(\mathbf{x}-\mathbf{x}^{(i)}) \tag{19}$$

is the Mahalanobis distance from $\mathbf{x}$ to $\mathbf{x}^{(i)}$.

A mode seeking algorithm, or mean shift filtering can be derived by iteratively computing the mean shift vector. The final partition of the feature space is obtained by grouping together all the data points that converged to the same mode.

3.2 Variable bandwidth mean shift

Usual mean shift procedures use a fixed bandwidth for all the data set. However, variable bandwidths are essential when local characteristics of the data vary significantly across the feature space. Two density kernel estimators have been introduced to take into account a bandwidth that is not fixed for each point [61]. The first one is called "balloon estimator" and makes the bandwidth vary at each estimation point. It was first introduced by Loftsgaarden and Quensberry [36]. It is defined as:

$$\begin{aligned}\widehat{f}(\mathbf{x}) &= \frac{1}{n}\sum_{i=1}^{n} K_{\mathbf{H}(\mathbf{x})}(\mathbf{x}-\mathbf{x}^{(i)}) \\ &= \frac{1}{n}\sum_{i=1}^{n} \frac{1}{|\mathbf{H}(\mathbf{x})|^{1/2}} K(\mathbf{H}(\mathbf{x})^{-1/2}(\mathbf{x}-\mathbf{x}^{(i)}))\end{aligned} \tag{20}$$

where $\mathbf{H}(\mathbf{x})$ is the bandiwth matrix varying at each estimation point. When applied globally, this estimator typically does not integrate to 1 and thus is usually not itself a density, even when K is. In [54] the authors have investigated the degree of improvement that this estimator allows over fixed kernel estimates. For data up to 3 dimensions, the improvement seems to be very modest. However the balloon estimator becomes very efficient as soon as the number of dimensions becomes larger than 3.

The second density kernel estimator is the "sample point estimator" that makes the bandwidth vary at each data point. In [54] the advantages and drawbacks of this estimator have been studied. The major advantages are that it is a density and that a particular choice of the bandwidth can considerably reduce the bias [22]. However, finding this value for multivariate data is a hard problem not yet solved. A disadvantage is that the estimate at a point may be influenced by observations very far away and not just by points nearby. In [54] simulations have shown that this estimator has a very good behavior for small-to-moderate sample sizes, but deteriorates in performance compared to fixed estimates as the sample size grows.

In this paper we are working on data having more than 3 dimensions. Therefore, we will concentrate only on the balloon estimator. The mean shift technique using the balloon estimator has been introduced in [10] for univariate and multivariate data.

3.3 Bandwidth selection for mean shift

A mode seeking algorithm can be derived by iteratively computing and applying the mean shift vector. 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 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 [13] 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 [10], a method based on the task oriented step of [13] has been developed. It is dedicated to the variable bandwidth selection in mixed multi-dimensional spaces. Here we briefly remind its principle.

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 gave the most stable partitions.

Assume that the d-dimensional data can be decomposed as the Cartesian product of P independent spaces associated to different types of information (*e.g.* position, color), also called feature spaces or domains, with dimension $d_\rho, \rho = 1 \dots P$ (where $\sum_{\rho=1}^{P} d_\rho = d$). The method in [10] proposes to find iteratively for each feature space ρ the best bandwidth $\Upsilon_\rho^{(i)}$ for each estimation point i. The iterative estimation of bandwidths is shown in algorithm 1. The final partition of the data is obtained by applying a last time the mean shift partitioning using the balloon estimator with the selected variable bandwidths $\Upsilon^{(i)} = \mathrm{diag}[\Upsilon_\rho^{(i)}, \rho = 1 \dots B]$.

The last point to discuss is how to choose the range of predefined bandwidths for our application of moving object detection. It is obvious that this choice influences the results. One could use a large sample of 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. For all the feature spaces we have taken $B_\rho = 9$ bandwidths. For the position they depend on the size of the grid:

$$\mathbf{H}_1^{(b)} = \frac{w}{N_\mathcal{G}}(1 + \frac{3b}{B_1 - 1}) \quad \mathbf{H}_2^{(b)} = \frac{h}{N_\mathcal{G}}(1 + \frac{3b}{B_1 - 1}) \ , b = 1 \dots B_1 \ , \tag{26}$$

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 images 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 grid $\mathcal{G}$, $|\mathcal{V}_\mathcal{G}|$ its cardinal, the mean and standard deviation vectors are:

$$\alpha_\rho = \frac{1}{|\mathcal{V}_\mathcal{G}|} \sum_{(i,j) \in \mathcal{V}_\mathcal{G}} |\mathbf{x}_\rho^{(i)} - \mathbf{x}_\rho^{(j)}| \tag{27}$$

and

$$\beta_\rho = \sqrt{\frac{1}{|\mathcal{V}_\mathcal{G}|} \sum_{(i,j) \in \mathcal{V}_\mathcal{G}} (|\mathbf{x}_\rho^{(i)} - \mathbf{x}_\rho^{(j)}| - \alpha_\rho)^2} \ . \tag{28}$$

Algorithm 1 Iterative estimation of mean shift bandwidths

Given n data points $\mathbf{x}^{(i)} = (\mathbf{x}_\rho^{(i)})_{\rho=1\dots P}$, $i = 1\dots n$. Given a set of B_ρ predefined bandwidths $\{\mathbf{H}_\rho^{(b)}, b = 1\dots B\}$ for each feature space ρ, the bandwidth selection is as follows.
For $\rho' = 1, \dots, P$

- Evaluate the bandwidth at the partition level: For all $b = 1, \dots, B$

  1. For all $\rho = 1, \dots, P, \rho \neq \rho'$, $i = 1, \dots, n$ compute $\tilde{\mathbf{H}}_\rho^{(i)}$:

$$\tilde{\mathbf{H}}_\rho^{(i)} = \begin{cases} \frac{1}{B_\rho} \sum_{b=1}^{B_\rho} \mathbf{H}_\rho^{(b)} & \text{if } \rho > \rho' \\ \Upsilon_\rho^{(i)} & \text{else .} \end{cases} \tag{21}$$

  2. Define $\{\tilde{\mathbf{H}}^{(b,i)} = \text{diag}[\tilde{\mathbf{H}}_1^{(i)}, \dots, \tilde{\mathbf{H}}_{\rho'-1}^{(i)} \mathbf{H}_{\rho'}^{(b)} \tilde{\mathbf{H}}_{\rho'+1}^{(i)}, \dots, \tilde{\mathbf{H}}_P^{(i)}], b = 1, \dots, B_{\rho'}\}$.
  3. Partition the data using the balloon mean shift partitioning. The result is $k^{(b)}$ clusters denoted as $\mathcal{C}_u^{(b)}, u = 1 \dots k^{(b)}$. We introduce the function c that associates the $i-th$ data point to its cluster: $c(i, b) = u \;\Leftrightarrow \mathbf{x}^{(i)} \in \mathcal{C}_u^{(b)}$.
  4. Compute the normal representation $\mathcal{N}(\mu_u^{(b)}, \mathbf{\Sigma}_u^{(b)})$ of each cluster using:

$$\mu_u^{(b)} = \left(\mu_{u,\rho}^{(b)}\right)_{\rho=1\dots P} \text{ with } \mu_{u,\rho}^{(b)} = \frac{1}{|\mathcal{C}_u^{(b)}|} \sum_{i|c(i,b)=u} \mathbf{x}_\rho^{(i)} \ , \tag{22}$$

     and

$$\mathbf{\Sigma}_u^{(b)} = \left(\mathbf{\Sigma}_{u,\rho}^{(b)}\right)_{\rho=1\dots P} \text{ with } \mathbf{\Sigma}_{u,\rho}^{(b)} = \frac{1}{|\mathcal{C}_u^{(b)}|} \sum_{i|c(i,b)=u} (\mathbf{x}_\rho^{(i)} - \mu_{u,\rho}^{(b)})(\mathbf{x}_\rho^{(i)} - \mu_{u,\rho}^{(b)})^T. \tag{23}$$

  5. Associate to each point $\mathbf{x}^{(i)}$ the mean $\mu_{c(i,b)}^{(b)}$ and covariance $\mathbf{\Sigma}_{c(i,b)}^{(b)}$ of the cluster it belongs to. Denote $p^{(i,b)}$ the corresponding normal distribution.

- Evaluate the bandwidth at the data level: For each point $\mathbf{x}^{(i)}$

  1. Select the scale b' giving the most stable normal distribution by solving:

$$b' = \text{argmin}_{r=2,\dots,B-1} \text{JS}(p^{(i,r-1)}, p^{(i,r)}, p^{(i,r+1)}) \tag{24}$$

     where JS is the Jensen Shanon divergence defined by:

$$\begin{aligned} \text{JS}(p^{(i,r-1)}, p^{(i,r)}, p^{(i,r+1)}) &= \frac{1}{2} \log \frac{|\frac{1}{3} \sum_{b=r-1}^{r+1} \mathbf{\Sigma}_{c(i,b)}^{(b)}|}{\sqrt[3]{\prod_{b=r-1}^{r+1} |\mathbf{\Sigma}_{c(i,b)}^{(b)}|}} \\ &+ \frac{1}{2} \sum_{b=r-1}^{r+1} (\mu_{c(i,b)}^{(b)} - \frac{1}{3} \sum_{b=r-1}^{r+1} \mu_{c(i,b)}^{(b)})^T (\sum_{b=r-1}^{r+1} \mathbf{\Sigma}_{c(i,b)}^{(b)})^{-1} (\mu_{c(i,b)}^{(b)} - \frac{1}{3} \sum_{b=r-1}^{r+1} \mu_{c(i,b)}^{(b)}) \ . \end{aligned} \tag{25}$$

  2. The best bandwidth $\Upsilon_{\rho'}^{(i)}$ is $\mathbf{H}_{\rho'}^{(b')}$.

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

$$\mathbf{H}_{\rho}^{(b)} = \beta_{\rho}(0.5 + \frac{1.5b}{B_{\rho} - 1})\mathbf{I}_{d_{\rho}}, b = 1 \dots B_{\rho} \quad , \tag{29}$$

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 grid 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, a final step is necessary. 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 [20]. In seminal paper [9], 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:

$$\begin{aligned} E_t(L) = &- \gamma_c \sum_{s\in\mathcal{P}} \ln(\Pr(\mathbf{z}_t^{(C)}(s)|l_s)) - \gamma_m \sum_{s\in\mathcal{G}} \ln(\Pr(\mathbf{z}_t^{(M)}(s)|l_s)) + \\ &\lambda \sum_{(s,r)\in\mathcal{V}} \exp\Big(-\frac{\|\mathbf{z}_t^{(G)}(s) - \mathbf{z}_t^{(G)}(r)\|^2}{\sigma^2}\Big) \frac{1}{\mathrm{dist}(s,r)} (1-\delta(l_s,l_r)) \end{aligned} \tag{30}$$

where L is the set of all the labels l_s, $s \in \mathcal{P}$, dist is a distance measure, and $\mathcal{V}$ is the set of unordered pairs (s,r) of neighboring elements of $\mathcal{P}$. 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. Motion term only concerns the points of the grid. 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 [7], 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:

$$\begin{aligned} E_t(L) = &- \gamma_c \sum_{s\in\mathcal{P}\setminus\mathcal{C}_{u,t}} \ln(\Pr(\mathbf{z}_t^{(C)}(s)|l_s)) - \gamma_m \sum_{s\in\mathcal{G}\setminus\mathcal{C}_{u,t}} \ln(\Pr(\mathbf{z}_t^{(M)}(s)|l_s)) - \\ &\sum_{s\in\mathcal{C}_{u,t}} \Big(1 + \max_{s'\in\mathcal{P}} \lambda \sum_{r|(s',r)\in\mathcal{V}} V_{\{s',r\}}\Big)\delta(l_s, \text{"object"}) + \lambda \sum_{(s,r)\in\mathcal{V}} V_{\{s,r\}}(1-\delta(l_s,l_r)) \end{aligned} \tag{31}$$

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 for motion we only consider points of the grid, we chose to set the parameters γ_c and γ_m such that $\gamma_c = 1$ and

$$\gamma_m = N_{\mathcal{G}}^2. \tag{32}$$

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 to the segmentation result.

The parameter σ in the third energy term can be related to noise [50]:

$$\sigma = 2 * \langle (\mathbf{z}_t^{(G)}(s) - \mathbf{z}_t^{(G)}(r))^2 \rangle \tag{33}$$

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:

$$\lambda = \frac{1}{N}\Big(- \gamma_c \sum_{s \in \mathcal{P}} \sum_{l_s = "bkg","obj"} \ln(\Pr(\mathbf{z}_t^{(C)}(s)|l_s)) - \gamma_m \sum_{s \in \mathcal{G}} \sum_{l_s = "bkg","obj"} \ln(\Pr(\mathbf{z}_t^{(M)}(s)|l_s))\Big) \ . \tag{34}$$

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. These videos have been shot by a moving or a non moving camera, and they 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 detailed in the paper. 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 the frame at time $t = 124$, 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 number 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 detection. 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 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. If, for these reasons, the segmentation (not shown) are disappointing, the detected clusters are nonetheless interesting. In figure 9 we show the bounding boxes corresponding to these clusters. 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. 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.

5.2 Comparison with other motion detection methods

We now compare our algorithm to the background modeling method of Grimson and Stauffer [21] and the non parametric background modeling of Elgammal *et al.* [17]. 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 [42]. 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 10). 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 10). 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 masks. The problem of holes in the detection mask (moving pixels) was mentioned in section 2. Here they 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.

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. Grimson and Stauffer's method does not permit to distinguish the trees from the building behind. 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 many spurious motions within the trees. Our method also detects few objects within the trees but the segmentation permits to drop most of them. The masks of objects obtained by our algorithm are not perfect though. The partial occlusions lead to many high contours on the pedestrians which stop the flow. Also, due to these occlusions, the clusters are usually very small, which does not permit to construct enough good 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.

Until now, we have segmented each object independently. We are now trying to segment all the objects jointly by using a multilabel energy function that could be minimized using the α-expansion algorithm [8, 9].

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.

Finally, the proposed method does not make use of any temporal consistency. We are now studying the introduction of such a consistency either on a frame-to-frame basis or within a tracker whose (re)initialization would rely on detection maps. To that end, we proposed in [11] a first method to combine the tracking and the segmentation phase.

References

[1] 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.

[2] 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.

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

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

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

[6] 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.

[7] 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.

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

[9] 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.

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

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

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

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

[14] 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.

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

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

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

[18] 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.

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

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

[21] 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.

[22] P. Hall, T. Hui, and J. Marron. Improved variable window kernel estimates of probability densities. *The Annals of Statistics*, 23(1):1–10, 1995.

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

[24] 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.

[25] 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.

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

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

[28] 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.

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

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

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

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

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

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

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

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

[37] 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.

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

[39] 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.

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

[41] J. Neyman and E. Pearson. On the use and interpretation of certain test criteria for purposes of statistical inference. *Biometrika*, 20-A:175–247 and 264–299, 1928.

[42] 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.

[43] 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.

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

[45] 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.

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

[47] 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.

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

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

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

[51] 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.

[52] 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.

[53] 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.

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

[55] 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.

[56] 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.

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

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

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

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

[61] M. P. Wand and M. C. Jones. *Kernel Smoothing*. Chapman & Hall, London, U.K., 1995.

[62] 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.

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

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

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

[66] 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.

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

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

[69] 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.

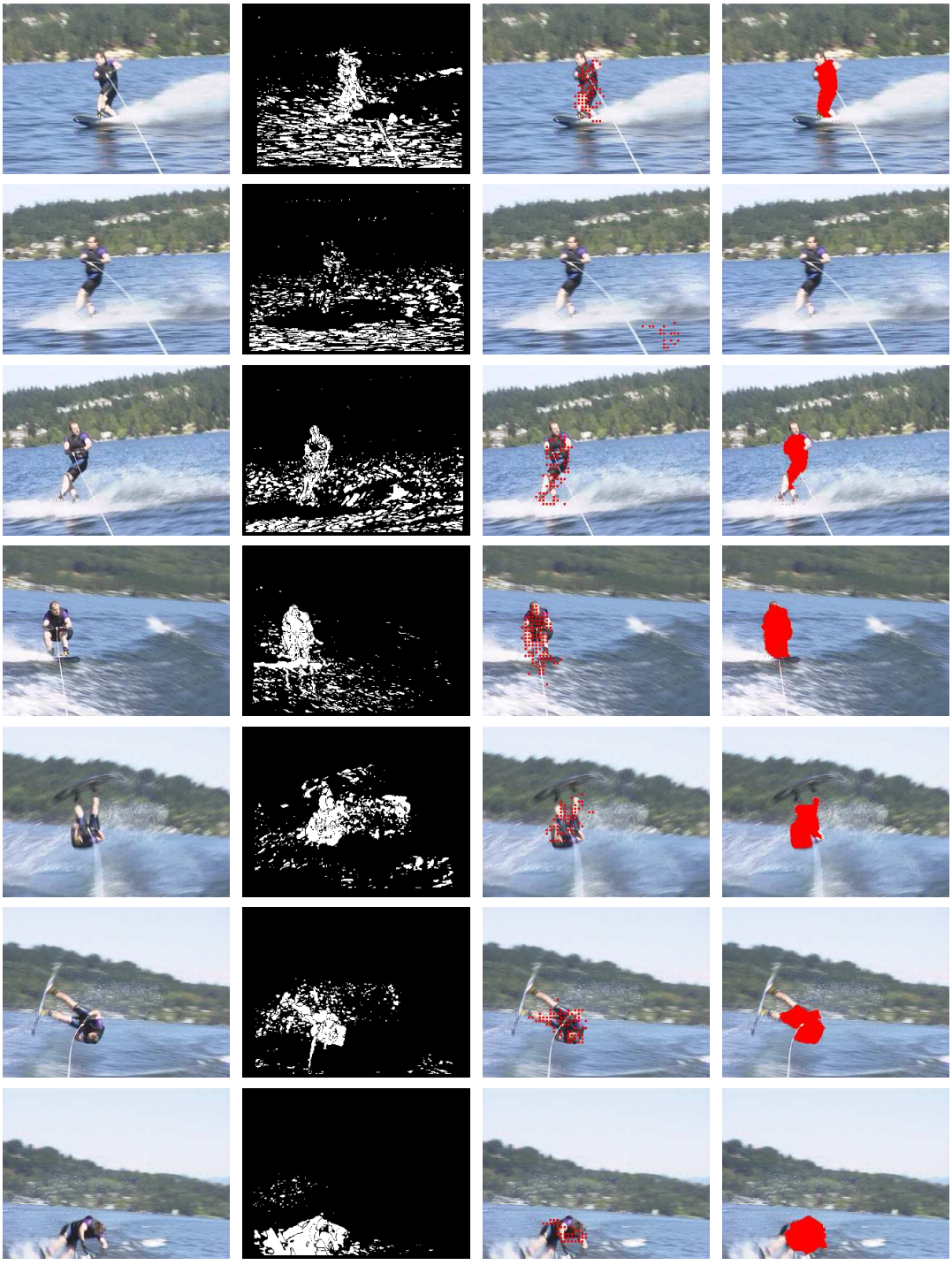


Figure 6: Results on the water skier sequence for frames 74, 124, 144, 214, 232, 236 and 242

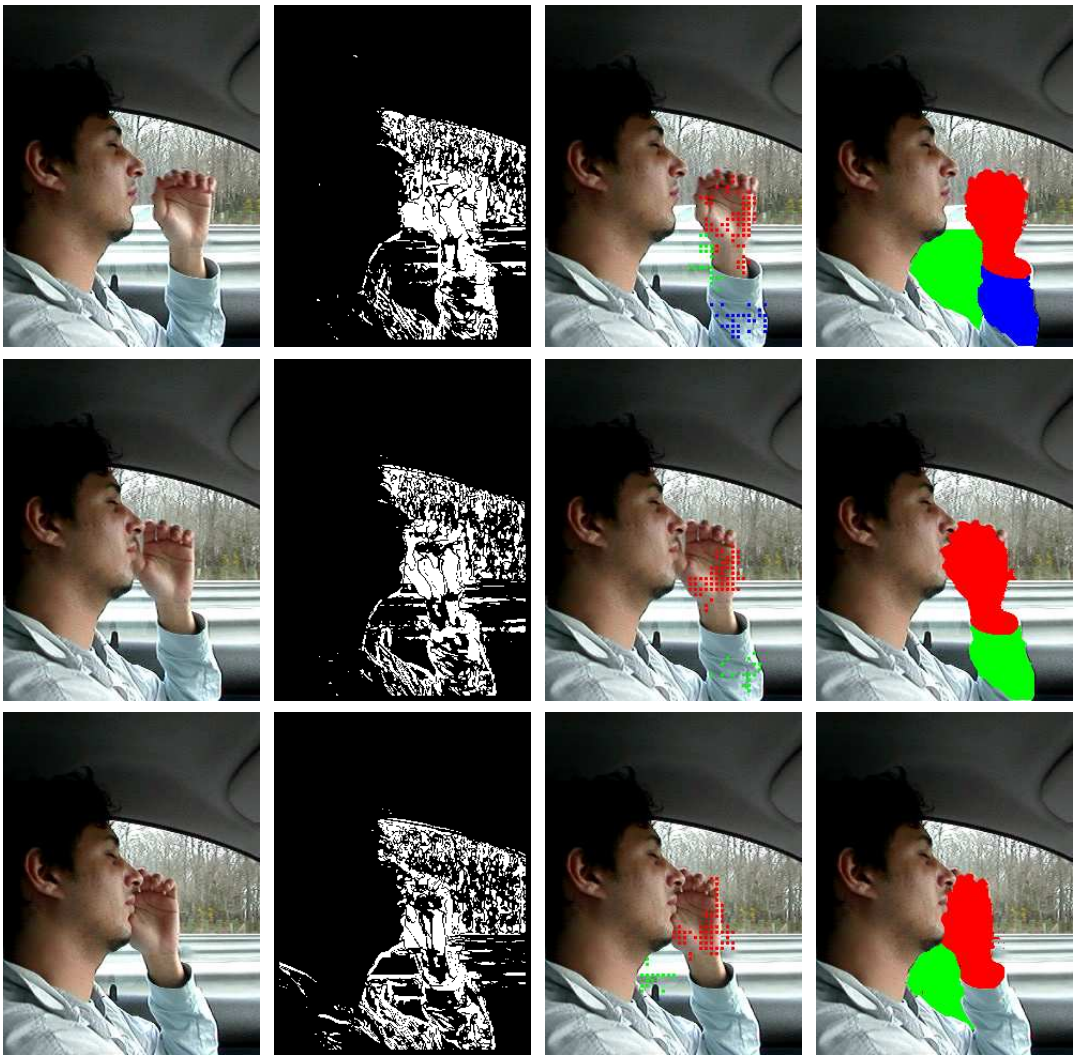


Figure 7: Results on the driver sequence for frames 15, 16, 17

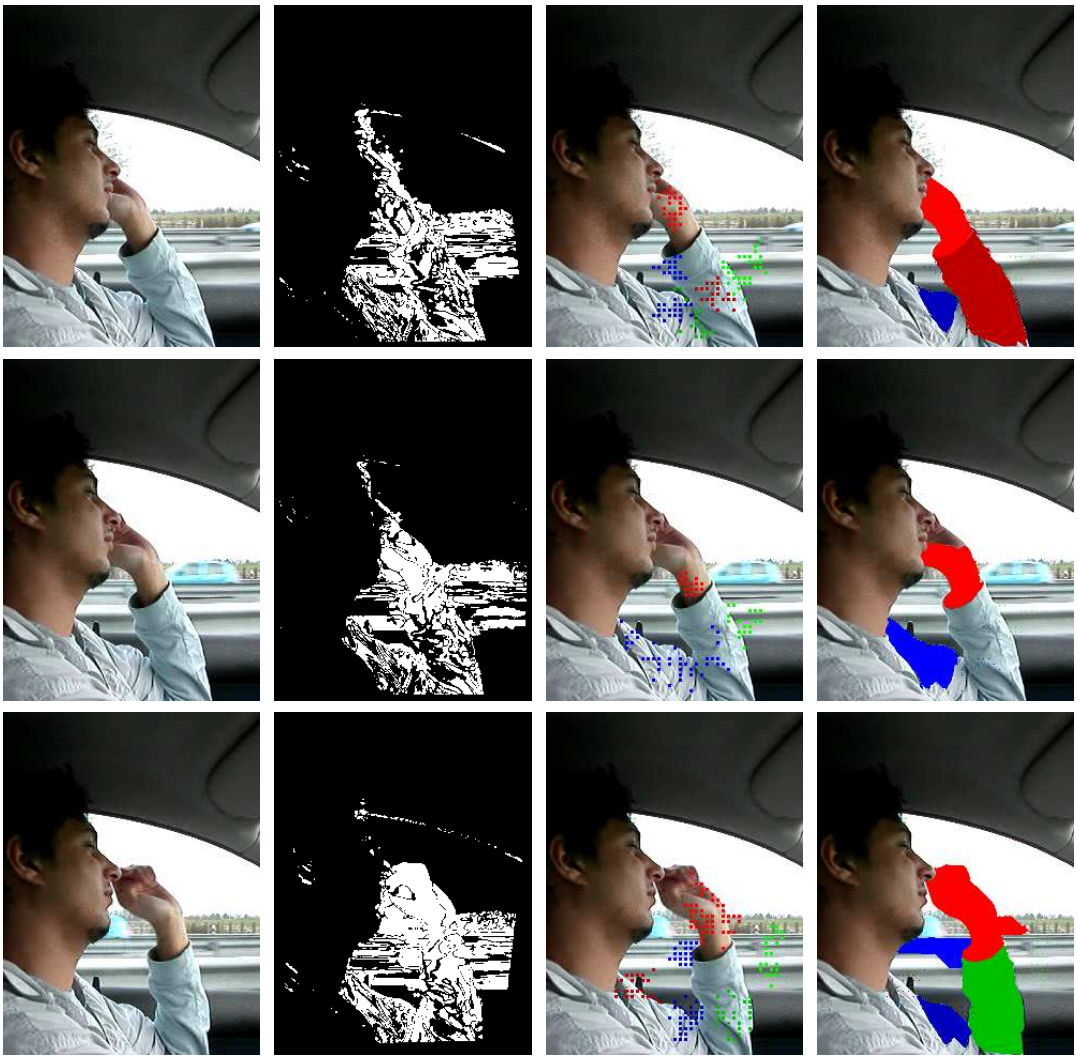
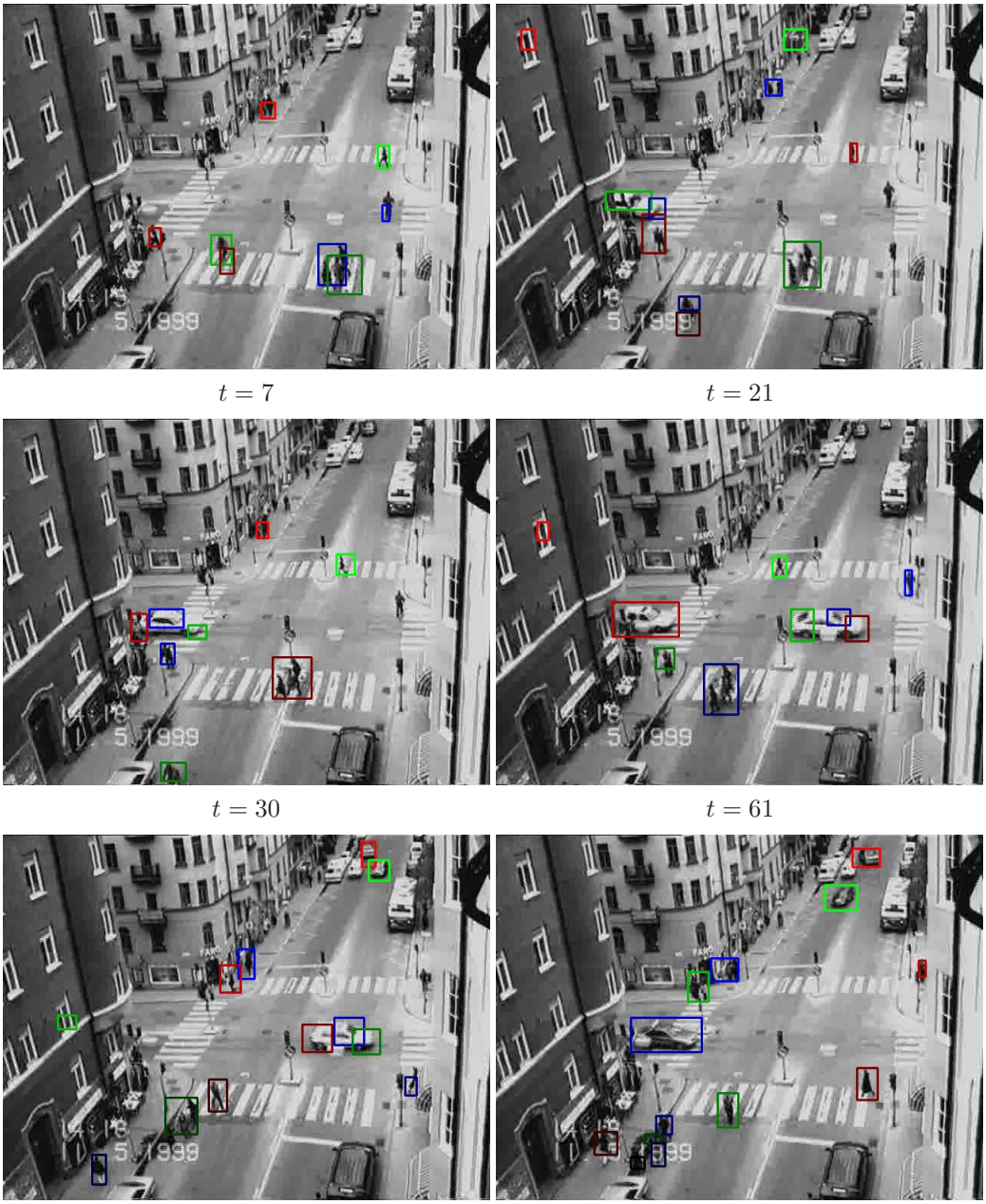


Figure 8: Results on the driver sequence for frames 48, 49, 50



$t = 7$ $t = 21$

$t = 30$ $t = 61$

$t = 91$ $t = 109$

Figure 9: Results on the traffic sequence. Bounding boxes of detected clusters are shown.

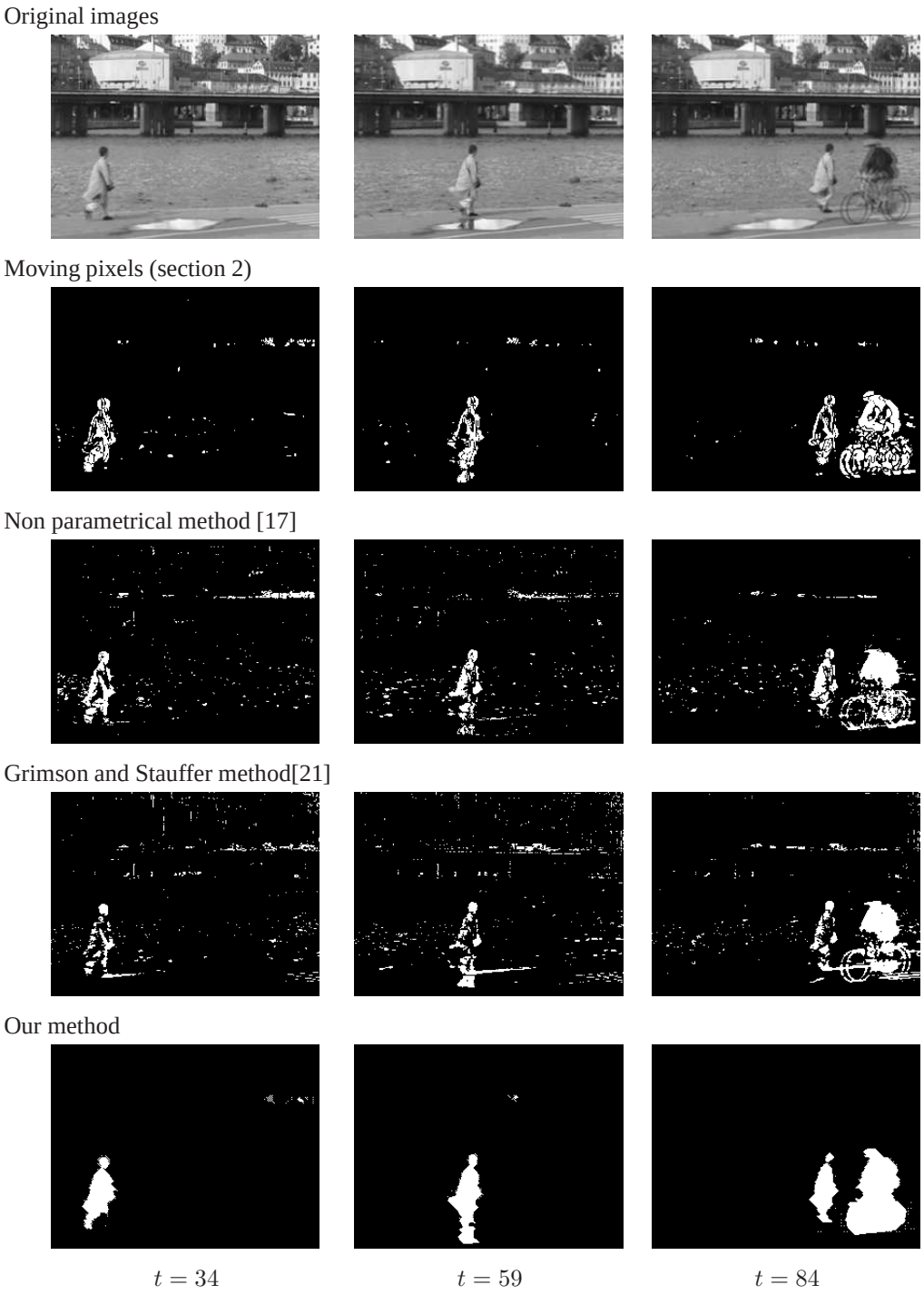


Figure 10: Detection masks on the person walking in front of water sequence for different methods.

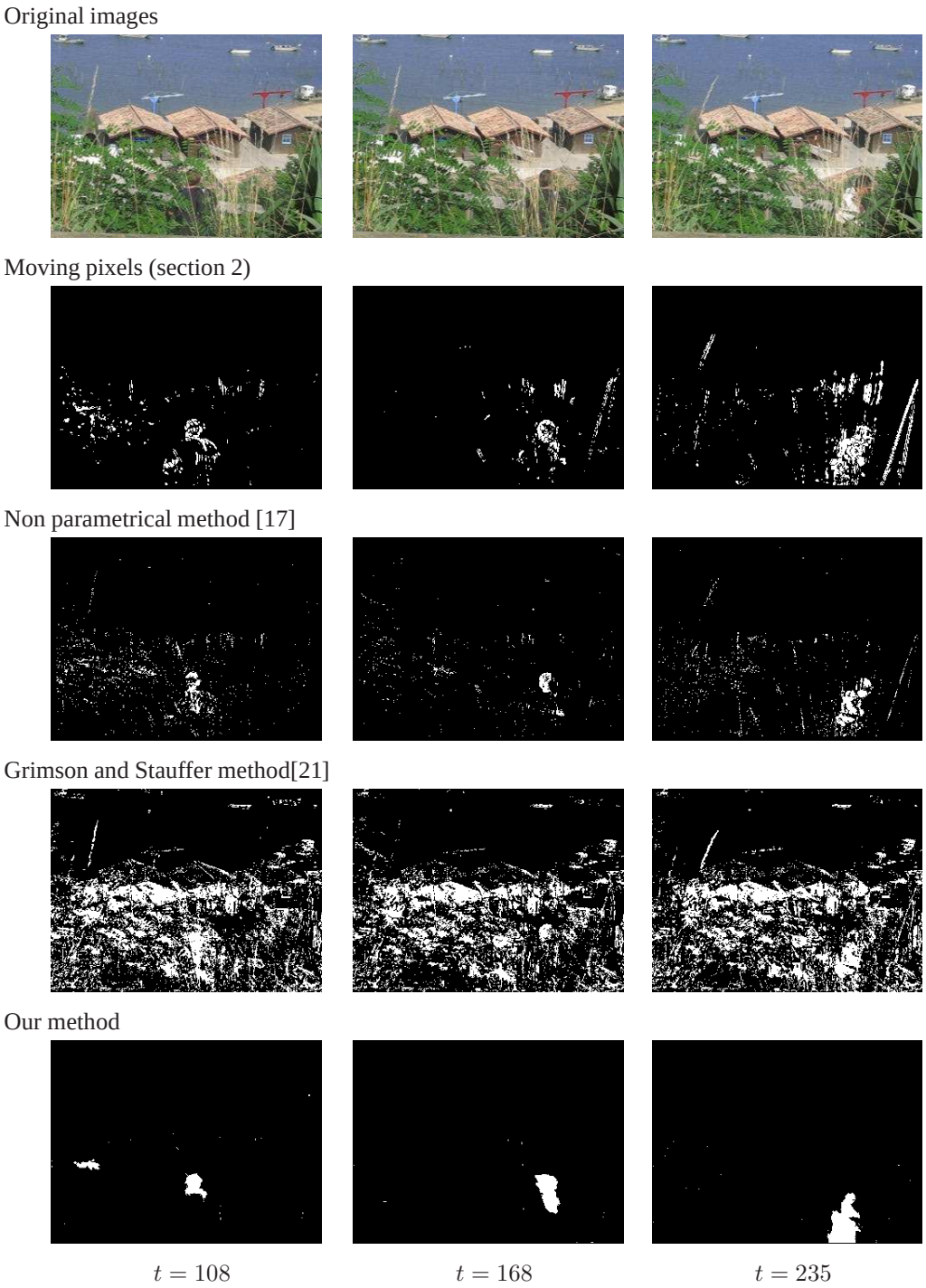


Figure 11: Detection masks on the waving trees sequence for different methods.

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex (France)
Unité de recherche INRIA Futurs : Parc Club Orsay Université - ZAC des Vignes
4, rue Jacques Monod - 91893 ORSAY Cedex (France)
Unité de recherche INRIA Lorraine : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex (France)
Unité de recherche INRIA Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)
Unité de recherche INRIA Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex (France)
Unité de recherche INRIA Sophia Antipolis : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex (France)

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399