A new video segmentation method of moving objects based on blob-level knowledge

Enrique J. Carmona[a*], Javier Martínez-Cantos[a], José Mira[a]

[a]*Dept. of Artificial Intelligence, ETSI Informática, UNED, Juan del Rosal 16, 28040, Madrid, Spain.*

{ecarmona, jmira}@dia.uned.es

javiermc@info-ab.uclm.es

*Corresponding author: ecarmona@dia.uned.es, Tfn: (34) 91 398 73 01, Fax : (34) 91 398 88 95.

Abstract

Variants of the background subtraction method are broadly used for the detection of moving objects in video sequences in different applications. In this work we propose a new approach to the background subtraction method which operates in the colour space and manages the colour information in the segmentation process to detect and eliminate noise. This new method is combined with blob-level knowledge associated with different types of blobs that may appear in the foreground. The idea is to process each pixel differently according to the category to which it belongs: real moving objects, shadows, ghosts, reflections, fluctuation or background noise. Thus, the foreground resulting from processing each image frame is refined selectively, applying at each instant the appropriate operator according to the type of noise blob we wish to eliminate. The approach proposed is adaptive, because it allows both the background model and threshold model to be updated. On the one hand, the results obtained confirm the robustness of the method proposed in a wide range of different sequences and, on the other hand, these results underline the importance of handling three colour components in the segmentation process rather than just the one grey-level component.

Keywords: Background subtraction, reflection detection, shadow detection, ghost detection, permanence memory, blob-level knowledge

1. Introduction

The detection of moving objects in video sequences is the first relevant step in the extraction of information in many computer vision applications including, for example, semantic video annotation, recognition pattern, video surveillance, traffic monitoring and people tracking. The quality of the results obtained by applying this stage is very important. The more reliable the shape and position of moving objects, the more reliable their identification is. In turn, this will guarantee greater success in subsequent tasks related to tracking and classification. Therefore, the crucial issues related to automatic video segmentation are to separate moving objects from the background and obtain accurate boundaries from this kind of objects.

There are different methods for detecting moving objects based, for example, on statistical methods (Horprasert et al, 1999; Lee, 2005; Stauffer and Grimson, 1999), fuzzy logic (Jadon et al, 2001), the subtraction of consecutive frames (Lipton et al, 1998), optical flow (Wang et al, 2003), genetic algorithms (Kim and Park, 2006; Carmona et al, 2006; Martínez-Cantos et al, 2006) or hybrid approaches (Collins et al, 2000; Dedeoglu, 2004) that combine some of the aforementioned techniques. Nevertheless, one of the most frequently used approaches with a fixed camera is based on *background subtraction method* and its multiple variants (Wren et al, 1997; Haritaoglu et al, 2000; Stauffer and Grimson, 2000; McKenna et al, 2000; Kim and Kim, 2003; Cucchiara et al, 2003; Xu et al, 2005; Leone and Distante, 2007), because of its speed and ease of implementation. Basically, this method enables moving regions to be detected by subtracting, pixel by pixel, the current image from a *background model* taken as a reference.

The outputs produced by the detection algorithms mentioned above, especially if working with real scenes, generally contain noise. The causes of noise are primarily due to the intrinsic noise of the camera itself, undesired reflections, objects whose total or partial colour coincides with the background and the existence of sudden shadows and changes (artificial or natural) in lighting. The total effect of these factors may be twofold: first, it may mean that areas that do not belong to

moving objects are incorporated into the foreground (foreground noise) and secondly, that determined areas belonging to moving objects no longer appear in the foreground (background noise). Specifically, in the set of approaches that use the background subtraction method to do segmentation, different proposals exist to detect and eliminate this noise. Thus, proposals exist that address the problem partially, for example, by only detecting the shadows (Xu et al, 2005; Leone et al, 2006; Leone and Distante, 2007). Other proposals, however, attack the problem globally, i.e., trying to differentiate and classify moving object blobs and different types of noise blobs (Cucchiara et al, 2003).

In this work we propose an adaptive segmentation method based on a new variant of the background subtraction method which uses relevant information from different types of blobs that may appear in the foreground as a result of processing the current frame. This information is closely related to different regions of interest that appear in the colour space by comparing the angle and module of the vector associated with each point of the image and the corresponding vector from the background model. Thus, it is possible to characterise and classify each point of the image according to the region to which it belongs: real moving object, shadow, ghost, reflection, fluctuation or background noise. The final aim is to use this new information based on blob-level knowledge to refine the foreground and update the background model to thereby achieve maximum precision during the segmentation. At all times, we will suppose that we are working with colour video sequences obtained from a fixed standard camera and, without losing generality, that we are working on the standard RGB colour space. The method could be applied in any other colour space.

The rest of this article is organised as follows: section 2 characterises different types of foreground blobs. Then, the segmentation method that we propose, called the *truncated cone method* (TCM) is described and broken down into its different stages. First (section 3), a transformation of the representation space of the problem is done, passing from a three-dimensional space (colour space) to another two-dimensional one (angle-module space). This enables us to define a rule, called *angle-module rule*, whose application produces a foreground map as an initial approach to the segmentation result. Second (section 4), this new representation space allows us not only to characterise the different noise blobs in a more operative way than section 2, but also to define a simple set of operators to eliminate them from the foreground. The arrangement of all these elements (rules and operators) in a suitable order will form the TCM (section 5). The section 6 shows the results of the different made experiments as well as its discussion. Finally, the conclusions of this work are described in section 7.

2. Blob characterisation: an initial approach

In this section, we make one first approach to the characterization of the different types of blobs that may appear in the foreground. Later, in section 4, we will refine this approach. Other proposals exist (Cucchiara et al, 2003), but ours consists of a set of entities and relations (see Figure 1) that are initially defined by comparing the current image and background model intensity levels:

- *Blob:* set of connected points.
- *Foreground*: binary image obtained from comparing the current image with the background model and applying a threshold value. In this image, theoretically, only the points associated with real moving objects appear.
- *Moving Visual Object* (MVO): foreground blob, b_{MVO}, associated with a real moving object.
- *Foreground Noise* (FN): blob that erroneously appears in the foreground but does not correspond to any real moving object.

- *Background Noise* (BN): blob that erroneously does not appear in the foreground (virtual blob) but corresponds to some real moving object. This type of noise usually appears when the colour of a part of a real moving object is similar to that area of the background located in the same position and the threshold used is not sufficiently tuned to segment correctly. Therefore, in first approach, this type of noise is characterised because all and each of the points, p, in this type of virtual blob, b_{BN}, have the property: $|I_t(x,y)-B_t(x,y)|<\delta$, $\forall(x,y)$ $p(x,y)\in b_{BN}$ and δ segmentation threshold.
- *Shadow*: a type of foreground noise. It is associated with any zone of the image covered by a real shadow. In first approach, it is characterised because all and each of the points in this type of blob, b_{Sh}, have the property: $[I_t(x,y)-B_t(x,y)]<0$, $\forall(x,y)$, $p(x,y)\in b_{Sh}$.
- *Reflection*: a type of foreground noise. It is associated with any zone of the image enhanced by a real reflection. In first approach, it is characterised because all and each of the points in this type of blob, b_{Rf}, have the property: $[I_t(x,y)-B_t(x,y)]>0$, $\forall(x,y)$, $p(x,y)\in b_{Rf}$.
- *Ghost*: a type of foreground noise. It is associated with the final position of a moving object that is stopped or the initial position of a stationary object that initiates its movement. In both instances, the difference $|I_t(x,y)-B_t(x,y)|$ is sufficiently great to make foreground blobs appear associated with the positions, final or initial, mentioned.
- *Fluctuation*: a type of foreground noise. With this term, we are talking about small variations in lighting that are caught between two consecutive frames. These variations can be produced because the optical sensors from a video camera, even in the absence of changes in lighting, do not register the light intensity values received in an exactly constant manner and/or because the very source of light (artificial or natural) can be subject to slight oscillations. Thus, the pixels in this type of blob, b_{Fl}, have the property: $|I_t(x,y)-I_{t-1}(x,y)|\approx 0$, $\forall(x,y)$ $p(x,y)\in b_{Fl}$, and therefore, in first approach, $|I_t(x,y)-B_t(x,y)|\approx 0$, $\forall(x,y)$ $p(x,y)\in b_{Fl}$. In other words, the difference is next to zero but is not rigorously null.

It is important to underline that all the properties mentioned previously establish necessary but not sufficient conditions. For example, if $|I_t(x,y)-B_t(x,y)|<\delta$, then point (x,y) could belong to background noise or fluctuation. This circumstance will have to be considered at the time of classifying each image point into one or several blob classes. In any case, the aim is to use the characterization each one of the entities defined above to facilitate identification of each type of blob in the scene and, in the last instance, do a more precise segmentation and more effective updating of the background model.

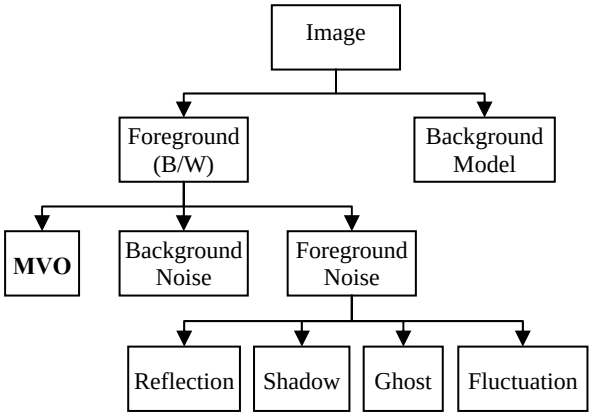

Figure 1. Classification of types of foreground blobs.

3. Segmentation: angle-module rule

We present here a transformation of the three-dimensional representation colour space to a new two-dimensional space called angle-module space that will be the reference frame of the segmentation method that we propose. The new resulting space allows us to define a segmentation rule whose application produces an initial approach to the final foreground map and constitutes the first stage of the truncated cone method. Thus, for each position, (x,y), of a pixel of a given image frame and at a moment of time, t, the relation existing between the image RGB vector associated with this position, $\mathbf{I_t}^{(x,y)}(r,g,b)$, and the background RGB vector, $\mathbf{B_t}^{(x,y)}(r,g,b)$, can be characterised with the value of the angle that they form, $\theta_t^{(x,y)}$, and the magnitude of difference of their modules in absolute value, $\Delta_t{}^{(x,y)}{}_{\text{mod}}$. From here onwards, to simplify the notation, we will use $\mathbf{M_t}(x,y)$ to refer to the RGB vector associated with the point (x,y) pertaining to map, $\mathbf{M}$, at moment t.

At each instant of time, the difference in modules, in absolute value and matrix form, $\mathbf{\Delta^t_{mod}}$, is computed as (1), where $|\mathbf{I_t}|$ and $|\mathbf{B_t}|$ are the image and background module matrix, respectively. The calculation of angle, in matrix notation, $\mathbf{\Theta_t}$, is calculated from (4), using the two alternative definitions of the scalar product, see eqs. (2) and (3). In (3), the operator $A \cdot *B$ denotes the product of the two matrices, A and B, element by element. In (4), the operator, (A/B), denotes the quotient between matrix A and B, element by element, the operator, $A \cdot B$, denotes the product between matrix A and B, element by element, and ε represents a very small value which avoids a possible division by zero.

$$\mathbf{\Delta^t_{mod}} = abs\left(|\mathbf{I_t}| - |\mathbf{B_t}|\right) \tag{1}$$

$$\mathbf{I_t} \bullet \mathbf{B_t} = |\mathbf{I_t}| \cdot |\mathbf{B_t}| \cdot \cos(\mathbf{\Theta_t}) \tag{2}$$

$$\mathbf{I_t} \bullet \mathbf{B_t} = \mathbf{I_t^R} \cdot * \mathbf{B_t^R} + \mathbf{I_t^G} \cdot * \mathbf{B_t^G} + \mathbf{I_t^B} \cdot * \mathbf{B_t^B} \tag{3}$$

$$\mathbf{\Theta_t} \approx \arccos\left(\frac{\mathbf{I_t} \bullet \mathbf{B_t}}{\left(|\mathbf{I_t}| \cdot |\mathbf{B_t}|\right) + \varepsilon}\right) \tag{4}$$

By adequately comparing the respective values of these two matrices, $\mathbf{\Theta_t}$ and $\mathbf{\Delta^t_{mod}}$, some interesting relations can be obtained. The idea is the following, for each point, (x,y), of the image, a revolution cone can be built in the RGB space (see Figure 2) by using as an axis the straight line that contains the point vector, (r_B,g_B,b_B), associated with position (x,y) of the background, $\mathbf{B}(x,y)$, and another straight line as a generatrix which, passing through the origin, forms an angle ω with the previous one. If we now trace three planes perpendicular to the vector $\mathbf{B}(x,y)$, one containing the point (r_B,g_B,b_B), and the other two, situated above and below this, at a distance h, they will delimit, along with the cone surface, two regions of interest: a truncated cone located on the upper part of the intermediate plane and another on the lower part.

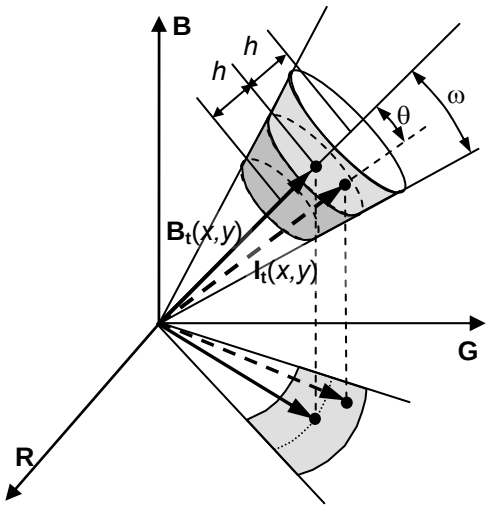

Figure 2. Truncated cones associated with a background point, (*x,y*), in the RGB space.

Once the size of the truncated cone region is determined by means of, for example, [ω_0, h_0], the point (r_I,g_I,b_I), associated with the position (*x,y*) in the image, $\mathbf{I}(x,y)$, will belong to the volume delimited by this region if and only if the two conditions, (5) and (6), are verified simultaneously, where ω_0 and h_0 are threshold values.

$$\mathbf{\Theta}(x,y) \le \omega_0 \tag{5}$$

$$\mathbf{\Delta}_{\mathbf{mod}}(x,y) \le h_0 \tag{6}$$

The advantage of using the angle-module space is twofold: on the one hand, it reduces the dimensionality, transforming the segmentation problem from a RGB three-dimensional to a two-dimensional space and, on the other hand, as will be seen in the following section, it will facilitate the characterization of the different types of blobs that may appear in the foreground, according to the fulfilment or non-fulfilment of these two conditions. In particular, it is quite intuitive that if we choose sufficiently small ω_0 and h_0, it is possible to establish as a condition of movement that every point (*r,g,b*) associated with the position (*x,y*) of the image, $\mathbf{I}(x,y)$, which is outside the truncated cone region defined by conditions (5) and (6), will belong to a real moving object. For this to happen, one of the two conditions will not be fulfilled. Thus, it is possible to define a rule, (7), to calculate approximately the first foreground map.

$$\mathbf{F}_t(x,y) = \begin{cases} 1, \text{if } \mathbf{\Theta}_t(x,y) > \omega_0 \text{ or } \mathbf{\Delta}^t_{\mathbf{mod}}(x,y) > h_0 \\ 0, \text{else} \end{cases} \tag{7}$$

Really, in the method that we propose, the angle threshold values, ω_i, and the module threshold values, h_i, are calculated according to the position (*x,y*) and the instant *t*, $\mathbf{\Omega}_t(x,y)$ and $\mathbf{H}_t(x,y)$, respectively. Consequently, equation (7) is transformed into a new rule, (8), that has been denominated *angle-module rule*.

$$\mathbf{F}_t(x,y) = \begin{cases} 1, \text{if } \mathbf{\Theta}_t(x,y) > \mathbf{\Omega}_t(x,y) \text{ or } \mathbf{\Delta}^t_{\mathbf{mod}}(x,y) > \mathbf{H}_t(x,y) \\ 0, \text{else} \end{cases} \tag{8}$$

Both the background model and the threshold matrices represent statistical properties of the pixel intensities observed in the image sequences from earlier moments $\{\mathbf{I_k}(x,y)\}$ for $k<t$. $\mathbf{B_1}(x,y)$ is

initialised with the first image (in which it is supposed that there are no moving objects), in other words, $\mathbf{B_1}(x,y)=\mathbf{I_1}(x,y)$, and $\mathbf{H_1}(x,y)$ and $\mathbf{\Omega_1}(x,y)$ are initialised with some predetermined value different from zero. The literature offers several ways of updating this type of matrices over time (Collins et al, 2000; Kim and Kim, 2003; Dedeoglu, 2004; Leone and Distante, 2007). Here, we propose the approach expressed in (9), (10) and (11), where $\alpha_B \in [0.0, 1.0]$ is a learning constant that specifies how much information from the incoming image is transferred to the background, α_Ω, $\beta_\Omega \in [0.0, 1.0]$ are learning constants that specify how much information from the angle matrix, weighted by the value of $\gamma_{\Omega i}$, $i=\{1,2\}$, is transferred to the angle threshold matrix, and α_H, $\beta_H \in [0.0, 1.0]$ are learning constants that specify how much information from the matrix of module differences, weighted by the value of γ_{Hi}, $i=\{1,2\}$, is transferred to the module threshold matrix. The values for α_i, $i \in \{B,\Omega,H\}$, β_j, $j \in \{\Omega,H\}$, $\gamma_{\Omega k}$, and γ_{Hk}, $k=\{1,2\}$, are adjusted according to experience based on the type of scene and the objectives to achieve at later stages in the segmentation process.

$$\mathbf{B}_{t+1}(x,y) = \begin{cases} \alpha_B \cdot \mathbf{B}_t(x,y) + (1-\alpha_B) \cdot \mathbf{I}_t(x,y), & \text{if } (x,y) \notin \mathbf{F}_t(x,y) \\ \mathbf{B}_t(x,y), & \text{if } (x,y) \in \mathbf{F}_t(x,y) \end{cases} \tag{9}$$

$$\mathbf{\Omega}_{t+1}(x,y) = \begin{cases} \alpha_\Omega \cdot \mathbf{\Omega}_t(x,y) + (1-\alpha_\Omega) \cdot \gamma_{\Omega 1} \cdot \mathbf{\Theta}_t(x,y), & \text{if } (x,y) \notin \mathbf{F}_t(x,y) \\ \beta_\Omega \cdot \mathbf{\Omega}_t(x,y) + (1-\beta_\Omega) \cdot \gamma_{\Omega 2} \cdot \mathbf{\Theta}_t(x,y), & \text{if } (x,y) \in \mathbf{F}_t(x,y) \end{cases} \tag{10}$$

$$\mathbf{H}_{t+1}(x,y) = \begin{cases} \alpha_H \cdot \mathbf{H}_t(x,y) + (1-\alpha_H) \cdot \gamma_{H1} \cdot \mathbf{\Delta}^t_{\mathbf{mod}}(x,y), & \text{if } (x,y) \notin \mathbf{F}_t(x,y) \\ \beta_H \cdot \mathbf{H}_t(x,y) + (1-\beta_H) \cdot \gamma_{H2} \cdot \mathbf{\Delta}^t_{\mathbf{mod}}(x,y), & \text{if } (x,y) \in \mathbf{F}_t(x,y) \end{cases} \tag{11}$$

It should be stressed that the $\mathbf{B_t}(x,y)$ value is only updated for non-foreground points. This could cause the appearance of foreground noise from ghost blobs (Dedeoglu, 2004). The solution to this problem is considered in the following section.

4. Noise detection

The aim of this section is to describe the characteristics of some notable regions that may be defined in the angle-module space, by combining the fulfilment or non-fulfilment of conditions (5) and (6), and properly choosing the values ω_0 and h_0 from some domain knowledge heuristic. The result is the characterisation of the different types of noise blobs according to whether or not they belong to one of these regions, allowing a more operative definition than the one in section 2. Finally, to a great extent, this characterisation will facilitate the construction of filtering operators that will selectively be applied to eliminate each type of noise blob.

4.1 Detecting shadows

One of the major causes of foreground noise are the shadows that objects project with highly undesirable effects (Horprasert et al, 1999; Salvador et al, 2004). Therefore, it is necessary to use a method to eliminate this type of noise because, otherwise, failure at a possible subsequent stage of tracking and/or classification is almost certain.

To characterise this type of noise, we make use of the fact that, for each pixel belonging to a shadow region, the associated image RGB vector is approximately in the same direction as the background RGB vector and the RGB vector module of the image pixel is always slightly less than the corresponding vector module of the background pixel (Horprasert et al, 1999). This means that all shadow noise image points will be confined in the lower truncated cone region of the RGB space. Thus, initially, the shadow map, $\mathbf{Sh_t}(x,y)$, is given by equation (12), where ϕ_{sh} and h_{sh} are thresholds.

$$\mathbf{Sh_t}(x,y) = \begin{cases} 1, \text{if } \boldsymbol{\Theta_t}(x,y) \le \phi_{sh} \text{ and } |\mathbf{B_t}(x,y)| \ge |\mathbf{I_t}(x,y)| \ge (|\mathbf{B_t}(x,y)| - h_{sh}) \\ 0, \text{else} \end{cases} \quad (12)$$

The shadow map thus defined also contains all those MVO pixels whose colour is confined in this small region and their elimination from the foreground produces background noise. In order to minimize the inclusion of this type of points in the shadow map, a second definition, (13), is proposed, where h_{shi} and ϕ_{shi}, i={1,2} are thresholds that fulfil $h_{sh1}>h_{sh2}$, $h_{sh1},h_{sh2}\in[0.0,1.0]$ and $\phi_{sh1}<\phi_{sh2}$. This allows greater flexibility to delimit the shadow region (see Figure 3a). Experimentally, it is observed that with $\phi_{sh1},\phi_{sh2}\in[0.0,6.0]^o$ and $h_{sh1},h_{sh2}\in[0.5,1.0]$ good results are obtained. Instead of working with absolute module threshold values, as proposed in (12), in the new definition of the shadow map, percentage values are calculated relative to the module of the background RGB vector. It is a way of normalising with respect to the module size. From a practical point of view, this allows better results to be obtained.

$$\mathbf{Sh_t}(x,y) = \begin{cases} 1, \text{if } \phi_{sh1} \le \boldsymbol{\Theta_t}(x,y) \le \phi_{sh2} \text{ and } (h_{sh1} \cdot |\mathrm{B_t}(x,y)|) \ge |\mathrm{I_t}(x,y)| \ge (h_{sh2} \cdot |\mathrm{B_t}(x,y)|) \\ 0, \text{else} \end{cases} \quad (13)$$

4.2 Detecting reflections

To characterise this type of noise, we applied the opposite reasoning to the one used for shadows. Thus, for each reflection pixel, the associated image RGB vector is approximately in the same direction as the background RGB vector and the RGB vector module of the image pixel is always slightly greater than the corresponding vector module of the background pixel. This means that all reflection noise image points will be confined in the upper truncated cone region of the RGB space (see Figure 3b). Therefore, the reflection map, $\mathbf{Rf_t}(x,y)$, is given by (14), where h_{rfi} and ϕ_{rfi}, i={1,2}, are thresholds that fulfil $h_{rf1}>h_{rf2}$, $h_{rf1},h_{rf2}\in[1,\infty)$, $\phi_{rf1}<\phi_{rf2}$. Experimentally, it is observed that with $\phi_{rf1},\phi_{rf2}\in[0.0,6.0]^o$ and $h_{rf1},h_{rf2}\in[1.0,2.0]$ the best results are obtained.

$$\mathbf{Rf_t}(x,y) = \begin{cases} 1, \text{if } \phi_{rf1} \le \boldsymbol{\Theta_t}(x,y) \le \phi_{rf2} \text{ and } (h_{rf1} \cdot |\mathrm{B_t}(x,y)|) \ge |\mathrm{I_t}(x,y)| \ge (h_{rf2} \cdot |\mathrm{B_t}(x,y)|) \\ 0, \text{else} \end{cases} \quad (14)$$

4.3 Detecting fluctuations

To characterise this type of noise in the context of our approach, it is necessary to consider that the fluctuation will be associated with small variations in module differences, $|\mathbf{I_t}|$ and $|\mathbf{B_t}|$, and small variations in the angles, $\boldsymbol{\Theta_t}$. Consequently, all points from the fluctuation noise image will be confined in the upper and lower truncated cone region of the RGB space, with very small h and ω (see Figure 3c). Thus, the fluctuation map, $\mathbf{Fl_t}(x,y)$, is given by (15), where h_{fl} and ϕ_{fl} are now absolute thresholds of small value.

$$\mathbf{Fl_t}(x,y) = \begin{cases} 1, \text{if } \boldsymbol{\Theta_t}(x,y) \le \phi_{fl} \text{ and } \mathrm{abs}(|\mathrm{I_t}(x,y)| - |\mathbf{B_t}(x,y)|) \le h_{fl} \\ 0, \text{else} \end{cases} \quad (15)$$

Nevertheless, note that the condition that makes it possible to delimit the fluctuation noise region, characterised by eq. (15), is exactly the opposite to the condition expressed by the angle-module rule, (8). This means that the segmentation obtained initially by applying just the angle-module rule excludes all the fluctuation noise points and, therefore, it is not necessary to use any operator to eliminate this type of noise.

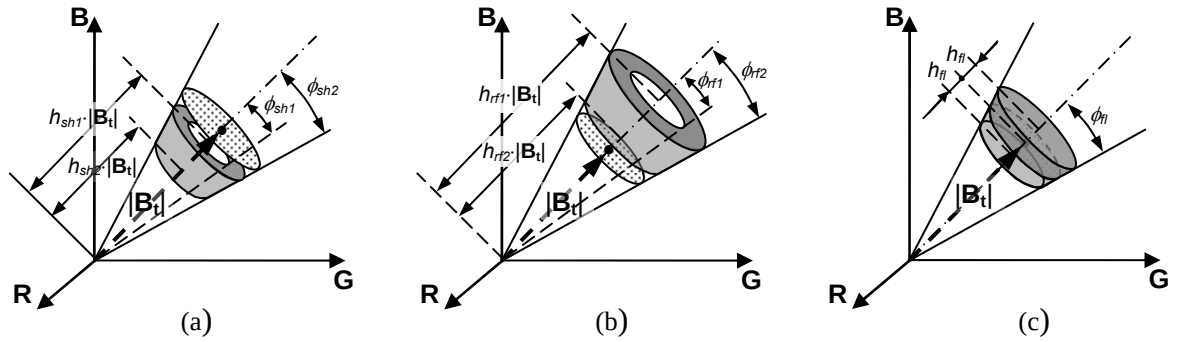

Figure 3. Noise blob characterisation based on the truncated cone method: (a) shadow region, (b) reflection region, (c) fluctuation region.

4.4 Detecting ghosts

The elimination of ghosts could have been done (Dedeoglu, 2004) if we had updated the background of all the foreground pixels in equation (9), i.e., just as it is indicated in expression (16), where α, $\beta \in [0.0,1.0]$.

$$\mathbf{B}_{t+1}(x,y) = \begin{cases} \alpha \cdot \mathbf{B}_t(x,y) + (1-\alpha)\cdot \mathbf{I}_t(x,y), & \text{if } (x,y) \notin \mathbf{F}_t(x,y) \\ \beta \cdot \mathbf{B}_t(x,y) + (1-\beta)\cdot \mathbf{I}_t(x,y), & \text{if } (x,y) \in \mathbf{F}_t(x,y) \end{cases} \tag{16}$$

The smaller the value of β, the faster the background updates and, therefore, the faster the ghost blobs are eliminated from the foreground. However, there is one disadvantage: if the value of β is too small, as well as eliminating the foreground ghosts in the scene, a kind of *wake-effect* can appear in the foreground in the opposite direction to the object's movement. This effect is more pronounced the larger the moving object, the slower it moves and the lower the value of β. Consequently, if we want the system to be highly reactive in eliminating the ghost, this strategy is not recommended. Thus, we propose a strategy for eliminating ghosts based on the *permanence memory* concept, PM, (Fernández et al, 2003). At each instant of time, *t*, permanence memory defines a map of data, and each $\mathbf{PM_t}(x,y)$ is obtained from its value in the previous instant, *t-1*, and from a binary reference input $\mathbf{Y_t}(x,y)$, just as is indicated in equation (17), where $\forall x,y$ $\mathbf{PM_0}=0$, C_0 (respectively, D_0) is a constant with which PM is incremented (decremented), C_{max} is the saturation value, and C_{min} is the minimum value that can be stored in PM.

$$\mathbf{PM}_t(x,y) = \begin{cases} \mathbf{PM}_{t-1}(x,y) + C_0 \text{ (up to } C_{max}), & \text{if } \mathbf{Y}_t(x,y) = 1 \\ \mathbf{PM}_{t-1}(x,y) - D_0 \text{ (down to } C_{min}), & \text{else} \end{cases} \tag{17}$$

In our approach, the permanence memory is a matrix the same size as the foreground matrix (binary input). Since every ghost pixel has the property of remaining constant at value 1 in the foreground, a constant charge increment will occur in the permanence memory position associated at this pixel and, consequently, the following condition can be imposed: if at a time instant, *t*, a pixel of the permanence memory, $\mathbf{PM_t}(x,y)$, reaches the saturation value, C_{max}, then the background pixel situated in the same position, $\mathbf{B_t}(x,y)$, is updated to the value of the current image at that instant, $\mathbf{I_t}(x,y)$. All those points (x,y) of the permanence memory that fulfil this condition will be associated with ghosts, and the automatic updating of the background to the current image in these positions will force this type of noise to disappear from the foreground at the following instant, *t+1*. The particularisation of the permanence memory concept that we propose here is given by the expression (18).

$$\mathbf{PM}_t(x,y) = \begin{cases} \mathbf{PM}_{t-1}(x,y)+1, & \text{if } \mathbf{F}_t(x,y)=1 \\ 0, & \text{else} \end{cases} \tag{18}$$

And, if the effect of charging this memory is considered, the background updating given in (9) is now transformed into equation (19).

$$\mathrm{B}_{t+1}(x,y) = \begin{cases} \alpha_B \cdot \mathrm{B}_t(x,y) + (1-\alpha_B) \cdot \mathrm{I}_t(x,y), & \text{if } (x,y) \notin \mathbf{F}_t(x,y) \\ \mathbf{I}_t(x,y), & \text{if } (x,y) \in \mathbf{F}_t(x,y) \text{ and } \mathbf{PM}_t(x,y) = C_{\max} \\ \mathrm{B}_t(x,y) & \text{if } (x,y) \in \mathbf{F}_t(x,y) \text{ and } \mathbf{PM}_t(x,y) < C_{\max} \end{cases} \tag{19}$$

Note that, in our case, it is not necessary to use the constant D_0 because every ghost pixel always increments the charging in PM. Any other cause (MVOs, noise different to the ghost) that makes the charging of other positions of PM increment, is not permanent and, when this cause ceases, these positions directly discharge the minimum value, C_{min}=0. Furthermore, the C_0 value is equal to 1. Thus, the value chosen for C_{max} will determine the number of frames that must pass to eliminate a ghost pixel from the instant in which this appears in the foreground. In fact, this last constant measures reactivity to eliminate ghost blobs: the smaller C_{max}, the greater reactivity. Nevertheless, the constant value will depend on the typical speeds of the moving objects in the scene and the objectives to be reached at later stages in the segmentation process. Thus, for example, in scenes where the typical speeds of the moving objects are low they will require greater values of C_{max} than if these speeds are high, otherwise, it could eliminate foreground pixels associated to slow MVOs. Moreover, for a specific application, it could be necessary to keep all the ghost blobs in the foreground, then, in this case, $C_{max} \to \infty$.

5. Truncated cone method

The final aim of any segmentation system for moving objects in the scene is that their detection is as precise as possible. Thus, the condition of movement expressed by the angle-module rule, see eq. (8), is not enough. Indeed, it implies the detection, not only of MVOs, but also reflection, shadow and ghost blobs. In order to filter other noise blobs and just have the MVOs in foreground, TCM applies a sequence of operators according to the flow chart shown in Figure 4.

Thus, after calculating the angle matrix, Θ_t, and the module difference matrix, Δ^t_{mod}, for the current image frame, we obtain an initial approach to the foreground map by applying the angle-module rule. Next, we proceed to eliminate reflections in this map by applying a reflection-filtering operator. The implementation of this operator requires two stages: one for detecting noise and another for filtering. The detection is done by calculating the reflection binary map from eq. (14). The filtering implies doing an operation *and* between the foreground obtained from the angle-module rule and the result of applying an operation *not* on the reflection map (see Figure 4). In particular, with the operation *and* it is possible to select those points that simultaneously belong to the foreground and, furthermore, do not belong to reflection noise (*not* operation). Next, we proceed to eliminate shadows by applying a shadow-filtering operator. The dynamics of this operator is identical to the reflection-filtering operator. Now, however, the shadow noise needs to be removed. Thus, the detection stage is done by calculating the shadow map from eq. (13). Furthermore, the filtering stage is done by applying an operation *and* between the foreground obtained after eliminating the reflection noise and the result of applying an operation *not* on the shadow map (see Figure 4). With the application of shadow-filtering operator, the foreground from the final stage of the segmentation process is obtained.

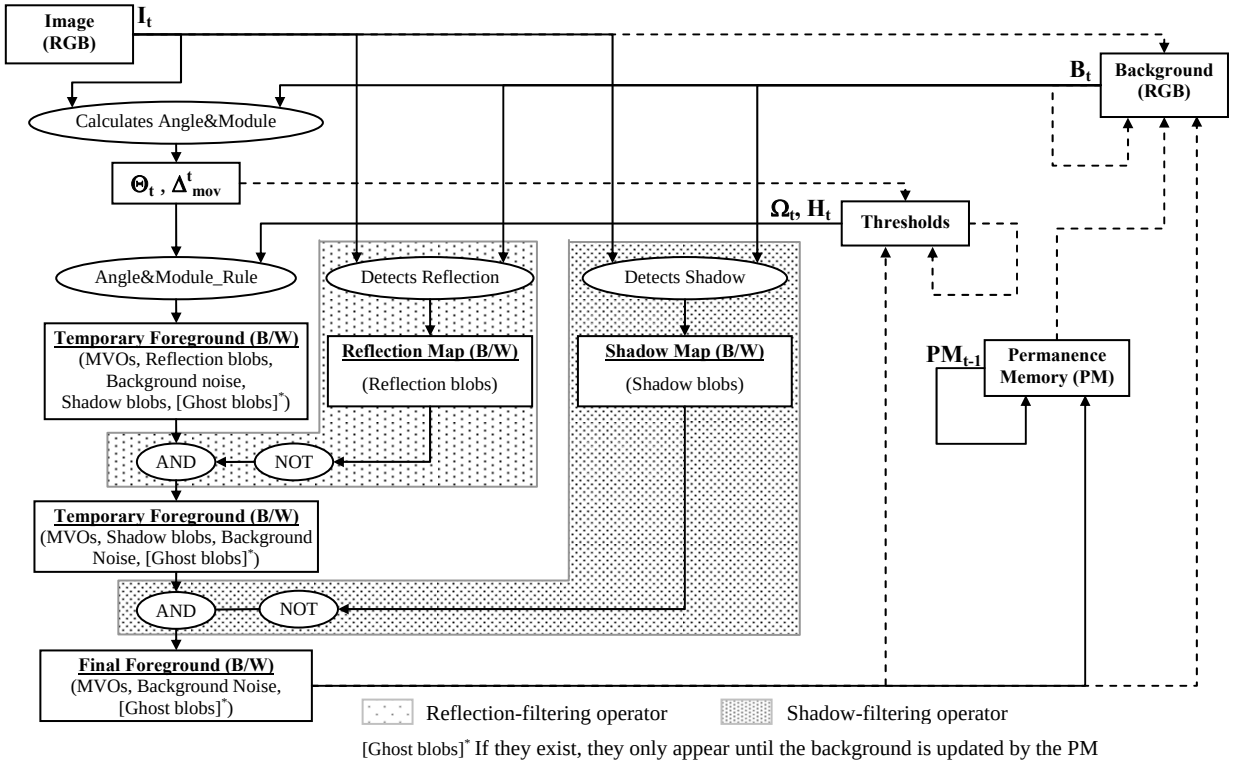

Figure 4. Flow chart of the truncated cone method. The broken lines represent updating mechanisms that are executed after obtaining the final foreground and the charging/discharging of permanence memory.

As indicated in Figure 4, the final segmentation, in addition to the MVOs, may contain ghosts and background noise. The appearance of this first type is temporary, i.e., the ghost pixels, if they exist, only appear during the time interval that the respective positions of the permanence memory take to charge to the value chosen for C_{max}. From this instant, the pixels associated with these positions disappear from the foreground. This is a direct consequence of how background is updated, see eq. (19). Conversely, the appearance of background noise will depend on the number of MVO pixels that are confined in the truncated cone regions shown in Figure 3. Nevertheless, given that the size of these regions is delimited by small angle and height values, their final volume will also be small and, therefore, the hypothesis is that the number of pixels missing in the foreground belonging to MVOs (background noise) will also be small. Otherwise, the solution is to apply morphological post-processing to the '*Final Foreground*' (see Figure 4) to recover approximately the missing pixels. Finally, Figure 5 complements the flow chart and it represents the pseudocode algorithm of the truncated cone segmentation method.

6. Experimental results

In this section we showed the results obtained after making several experiments and comparing our method with other approaches in different video sequences. Nevertheless, in the first place, an output example of each of the stages in the process from applying the TCM to a video sequence will be shown (see Figure 6). Thus, from the image frame in Figure 6a, with the application of the angle-module rule it is possible to obtain the map shown in Figure 6b where, as well as the MVO, other noise blobs can be seen. Immediately after, the application of the operator to eliminate reflections produces the map shown in Figure 6d. The map in Figure 6c shows all the undesired reflection noise that was filtered. Figure 6e shows the shadow noise in the scene and using the filtering operator to eliminate shadows gives the foreground map depicted in Figure 6f. As indicated in the previous section, to refine even more the quality of the foreground obtained, the post-processing

stage is applied, which here simply consists of applying a filtering operator of small isolated points. On other occasions, this filtering is usually followed by a morphological closing to compensate for the noise due to background noise. The post-processing result is the final foreground map that is shown in Figure 6g. Figure 6h shows the RGB segmentation of the original image using the post-processing foreground map as a mask. The other subfigures show the current state of the different dynamic maps used by the method. Thus Figure 6i shows the state of the background model, Figure 6j-k represents the map of the angle and module threshold matrix, respectively, and Figure 6l shows the state of the permanence memory.

Initially:
$B_0(x,y)=I_0(x,y)$
$\forall x,y$, $PM_0(x,y)=0$, $\Omega_1(x,y)=Cte_1$, and $H_1(x,y)=Cte_2$

FOR t=1 TO $t=t_{final}$

1. Computes module difference matrix, Δ^t_{mod}, and angle matrix, Θ_t: see eq. (1) and (4), respectively
2. Computes foreground, F_t: see eq. (8)
2. Computes reflection map, $Rf_t(x,y)$: see eq. (14)
3. Updates foreground, $F_t(x,y)= F_t(x,y) \,\&\, \neg\, Rf_t(x,y)$
4. Computes shadow map, $Sh_t(x,y)$: see eq. (13)
5. Computes final foreg., $F_t(x,y)= F_t(x,y) \,\&\, \neg\, Sh_t(x,y)$
6. Charges/discharges permanence memory, PM_t: see eq. (18)
7. Updates background, $B_{t+1}(x,y)$: see eq. (19)
8. Updates thresholds, $\Omega_{t+1}(x,y)$ and $H_{t+1}(x,y)$: see eq. (10) and (11), respectively.

ENDFOR

Figure 5. Algorithm in pseudocode of the truncated cone method.

To show the quantitative results of the TCM, we are going to work with two types of video sequence. The first sequence, which we have called *Human walks*, consists of colour image frames of 320x240 pixels. The main characteristic of this sequence is that, although it does not present a very complex background, it does contain mainly reflection and shadow noise. In this instance, a human appears on the left of the scene, carries a suitcase in one hand and a coat in the other, walks and, finally, disappears on the left. The second scene, *Hall Monitor*, is a 352x240 pixel colour image frame sequence, frequently used in benchmarking. This sequence presents a complex background, the moving objects are relatively small and, moreover include non-uniform lighting and noise which makes the segmentation process more difficult.

First, we are going to do an experiment to compare the quality of segmentation obtained by applying just the angle-module rule, i.e., not using the noise filtering operators, with that obtained by applying simple background subtraction. In both instances dynamic updating of the background model and thresholds is done. With the angle-module rule, since the colour management is done naturally, colour images will be used. This does not occur in background subtraction, so this will be applied to the same former images but after they have been transformed to the grey-level using a linear combination of the three RGB components. The aim of this experiment is to analyse the importance of working or not in colour and whether the need to update the two threshold maps that

the angle-module rule requires is justified, compared with updating just one threshold map in background subtraction. The comparison will be done in terms of sensitivity (Se=TP/TP+FN) and specificity (Sp=TN/TN+FP), with TP and FP, true and false positives, and TN and FN, true and false negatives, respectively. Sensitivity computes the percentage of the MVO pixels detected by the method compared with the real MVO pixels existing in the scene. If in a frame no MVOs exist and, after applying the segmentation method, TP=FN=0 is obtained, then the indetermination obtained from trying to calculate the sensitivity value, will be resolved doing Se=1. Moreover, specificity computes the percentage of background pixels detected by the method regarding the real background pixels in the scene. Sensitivity is directly related to background noise. Thus, the lower the number of holes and/or fractures in the segmented MVO silhouette, the greater the sensitivity. Specificity, on the other hand, is related to the remaining noise. In other words, when fewer shadows, reflections, fluctuation and ghosts are obtained in the segmentation result, the greater the specificity value. Thus, perfect segmentation would produce a sensitivity and specificity value equal to 1. Figure 7 shows the results of the comparison for the *Human walks* sequence. It can be seen that the angle-module rule produces greater sensitivity (Figure 7a) and specificity (Figure 7b) in virtually all the sequence frames. This supremacy of the angle-module rule is repeated in the relatively more complex *Hall Monitor* sequence (see Figure 8).

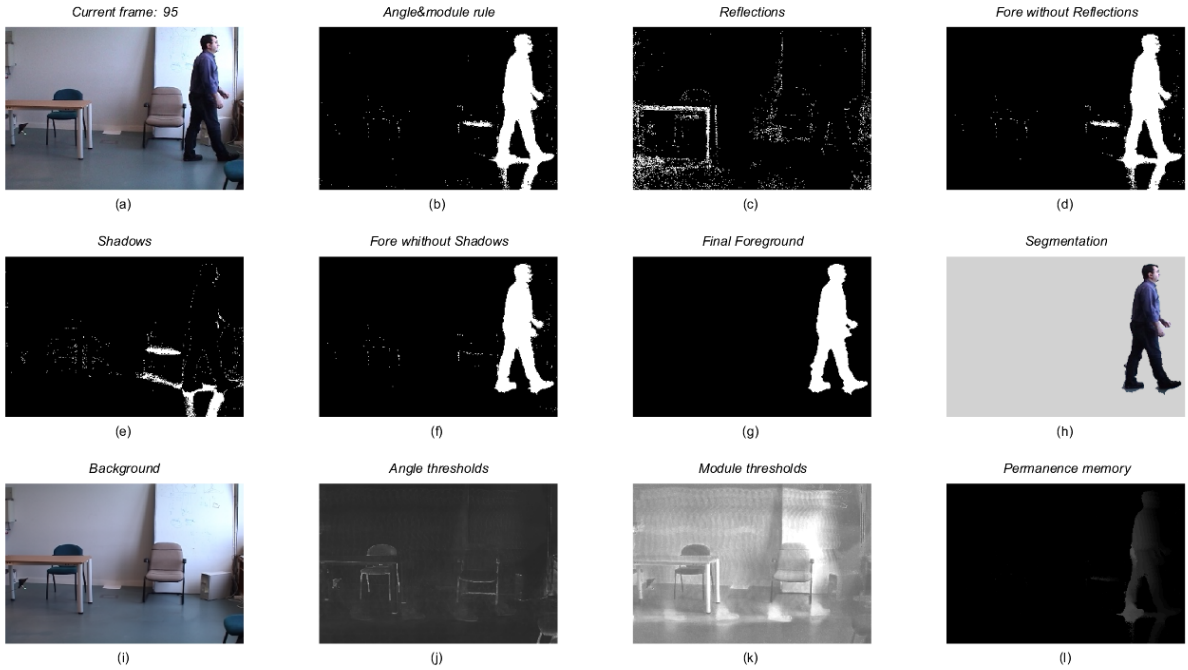

Figure 6. Output example of each of the stages in the truncated cone method applied to a video sequence frame.

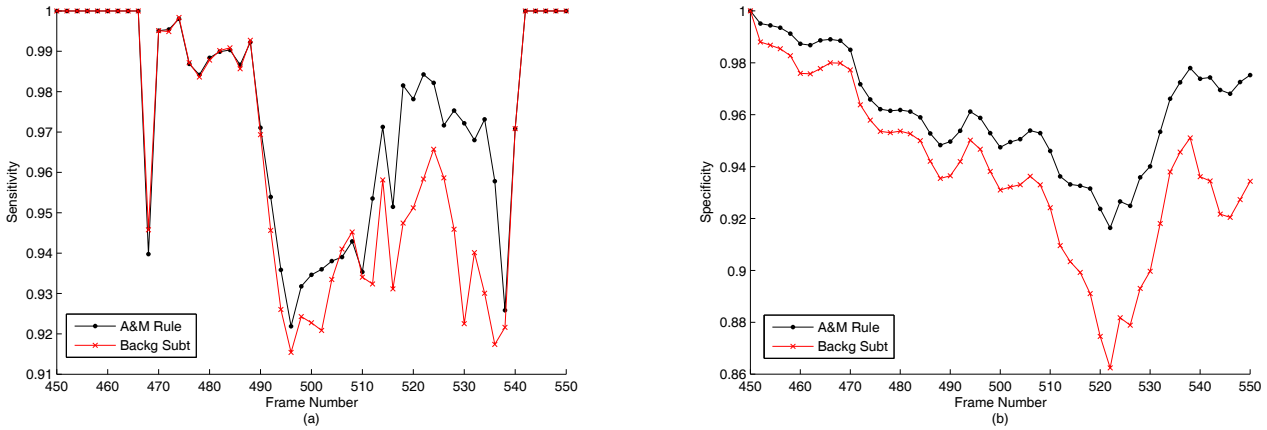

Figure 7. Comparison of the simple background subtraction (Backg. Subt.) and the angle-module rule (A&M rule) applied to the *Human Walks* sequence.(a) Sensitivity, (b) Specificity.

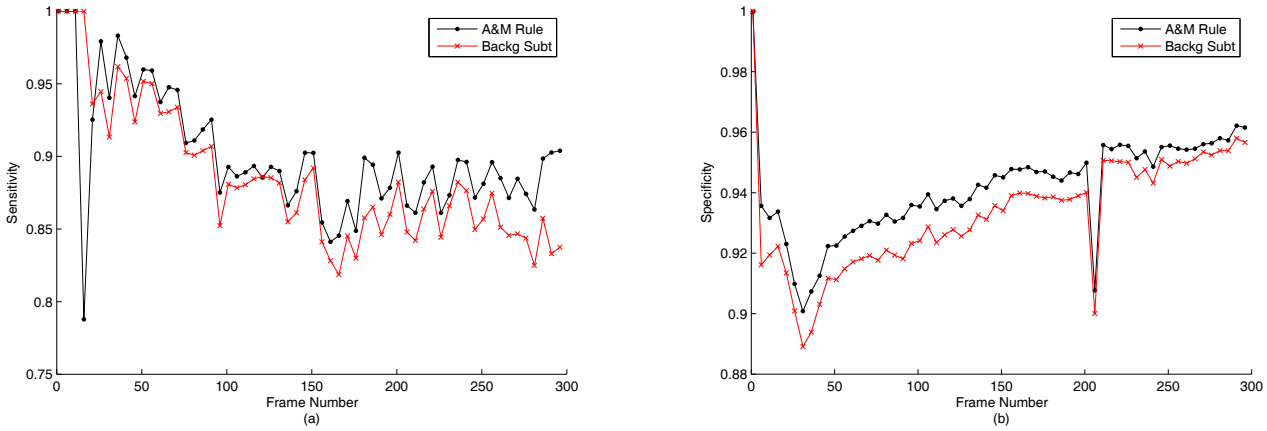

Figure 8. Comparison of the simple background subtraction (Backg. Subt.) and the angle-module rule (A&M Rule) applied to the *Hall Monitor* sequence. (a) Sensitivity, b) Specificity.

When the advantages of the angle-module rule had been analysed, the effect of the noise filtering operators on the two work sequences was studied which, along with this rule, compose the truncated cone method. The sensitivity analysis on the *Human Walks* sequence (Figure 9a) revealed that the effect of the noise filtering operators on the final segmentation was to slightly increase the background noise (decreased sensitivity) in comparison with the results obtained by applying the angle-module rule. This is because the filtrate operators not only eliminate pixels associated to noise but that also eliminate all those MVO pixels contained in the truncated cone regions associated to noise. This circumstance was described in section 5. However, it is possible to eliminate a large part of this noise by applying post-processing to the foreground obtained with the TCM. Specifically, in this instance morphological closing was applied and the final result improved sensitivity. However, where the TCM is highly competitive is the elimination of the remaining noise. This is evident in the high specificity value close to 1 (see Figure **9**b), obtained with the TCM for practically all the sequence frames. As was to be expected, the effect of morphological post-processing adds some false positives to the final segmentation and slightly decreases specificity compared with the TCM output without post-processing. Figure 10 shows the qualitative result of the segmentation process for some frames of this sequence. Similarly, the sensitivity (Figure 11a) and specificity results (Figure 11b) obtained for the *Hall Monitor* sequence are similar to the ones

mentioned for the previous sequence. Lastly, Figure 12 shows the final qualitative segmentation result for some of the frames in this sequence.

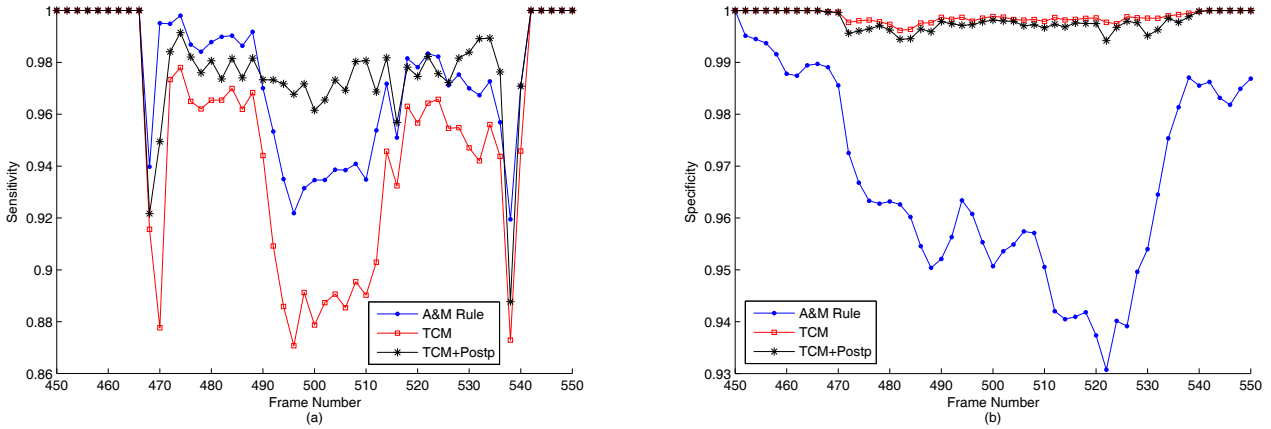

Figure 9. Comparison of the resulting segmentation from applying the angle-module rule (*A&M rule*), the truncated cone method without post-processing *(TCM)* and with processing (*TCM+Postp*) to the *HumanWalks* sequence. (a) Sensitivity, (b) Specificity.

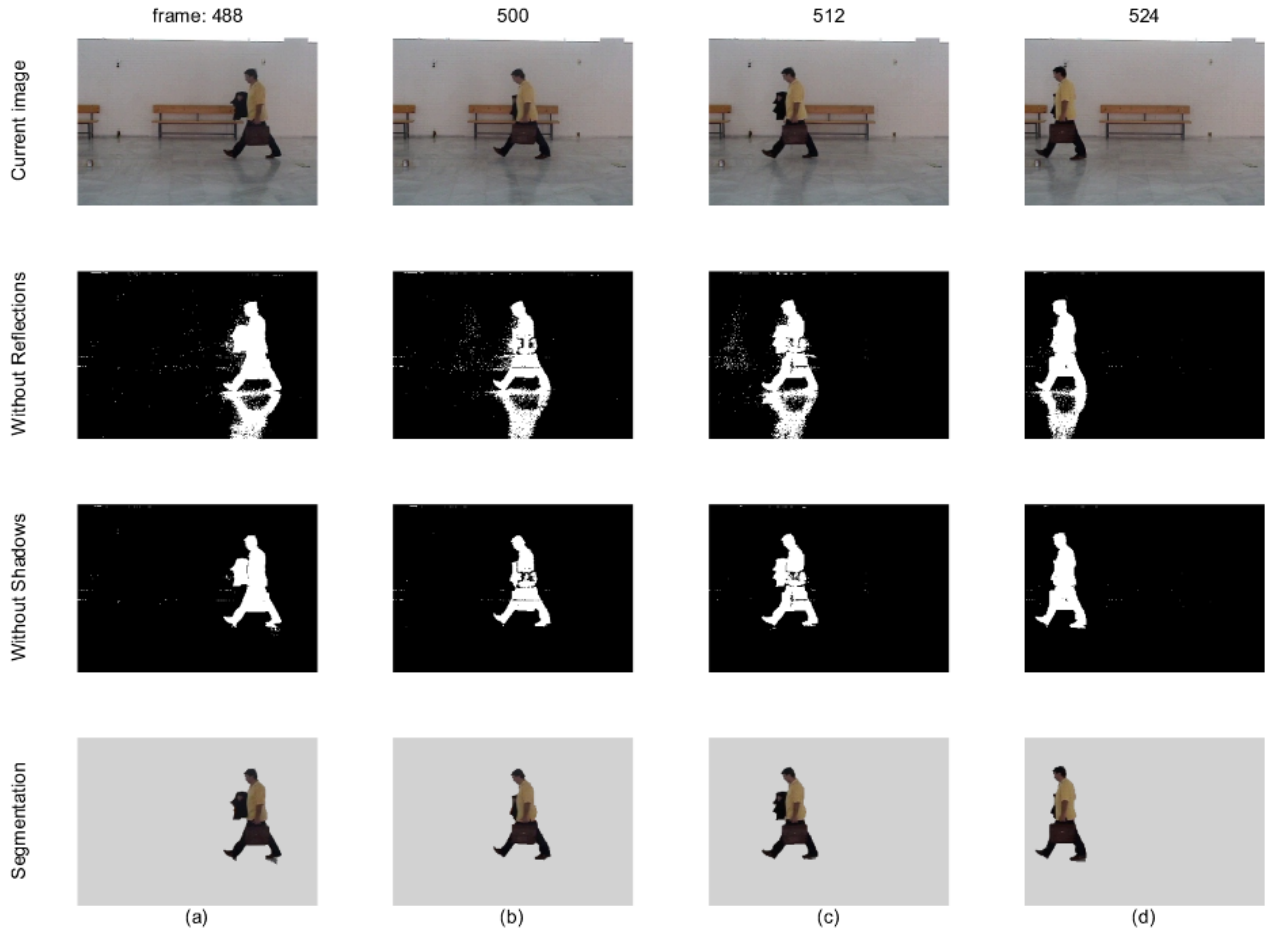

Figure 10. TCM segmentation results for some frames of the *HumanWalks* sequence. For each column from top to bottom, the first box contains the current image, the second and third show the foreground resulting from eliminating sequentially reflections and shadows, and the fourth shows the final segmentation after applying post-processing.

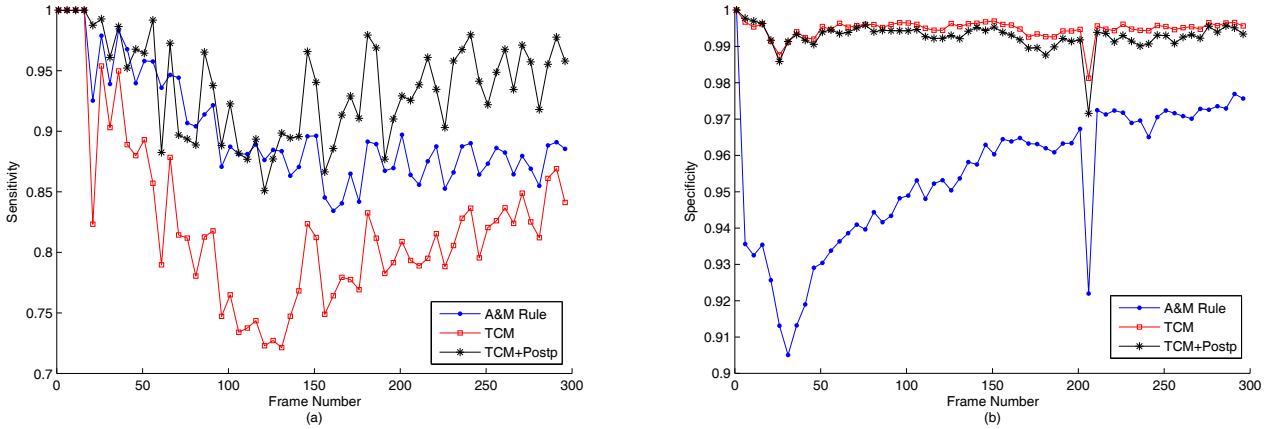

Figure 11. Comparison of the resulting segmentation from applying the angle-module rule (*A&M rule*), the truncated cone method without post-processing *(TCM*) and with processing (*TCM+Postp*) to the *Hall Monitor* sequence. (a) Sensitivity, (b) Specificity.

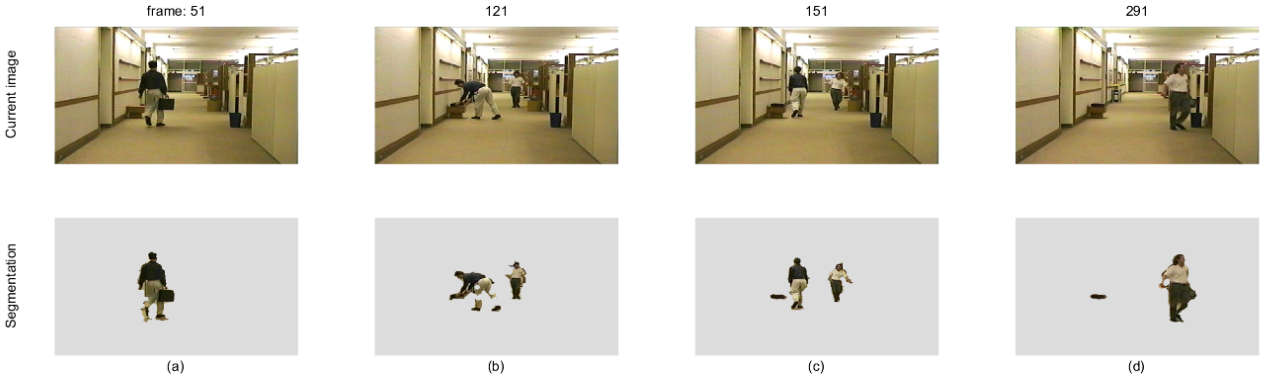

Figure 12. TCM segmentation results for some frames of the *Hall Monitor* sequence. For each column from top to bottom, the first box contains the current image and the second shows the final segmentation after applying post-processing..

To illustrate reactivity of the TCM stage to eliminate ghosts, based on the permanence memory concept, our strategy will be compared with that proposed by (Dedeoglu, 2004), based on the idea of dynamically updating the background model with eq. (16). To compare both strategies a sequence[1] piece will be used where the first frame, used as initial background model, already contains moving objects (see Figure 13a). The displacement of these objects in successive frames will cause ghosts to appear when they abandon the positions that they had in the initial frame. As mentioned in section 4.4, the saturation charge constant, C_{max}, see eq. (19), and β, see eq. (16), condition the reaction speed. Thus, a low value in both instances will provide a greater reactivity. Nevertheless, the segmentation deteriorates because the updating of the background model is so fast that the background takes the moving object's pixels, causing a characteristic wake-effect (see Figure 13b). Consequently, so that the goodness of the result of both methods are not dependent on the value chosen for these constants, their value is taken in such a way that the capacity to react is maximum but without causing any wake-effect on the sequence being studied. Specifically, C_{max}=48 and β=0.97. The qualitative result of the process for eliminating ghosts for both methods can be seen in Figure 13. Note that at the instant when the ghost (vehicle initial position) is eliminated in the background model (Figure 13c) and, consequently, in the segmentation output (Figure 13d) with the TCM, it just begins to fade into the background model (Figure 13e) obtained with the Dedeoglu's strategy, but it is still present in the segmentation output (Figure 13f).

[1] PETS'2001 Dataset (http://ftp.pets.rdg.ac.uk/)

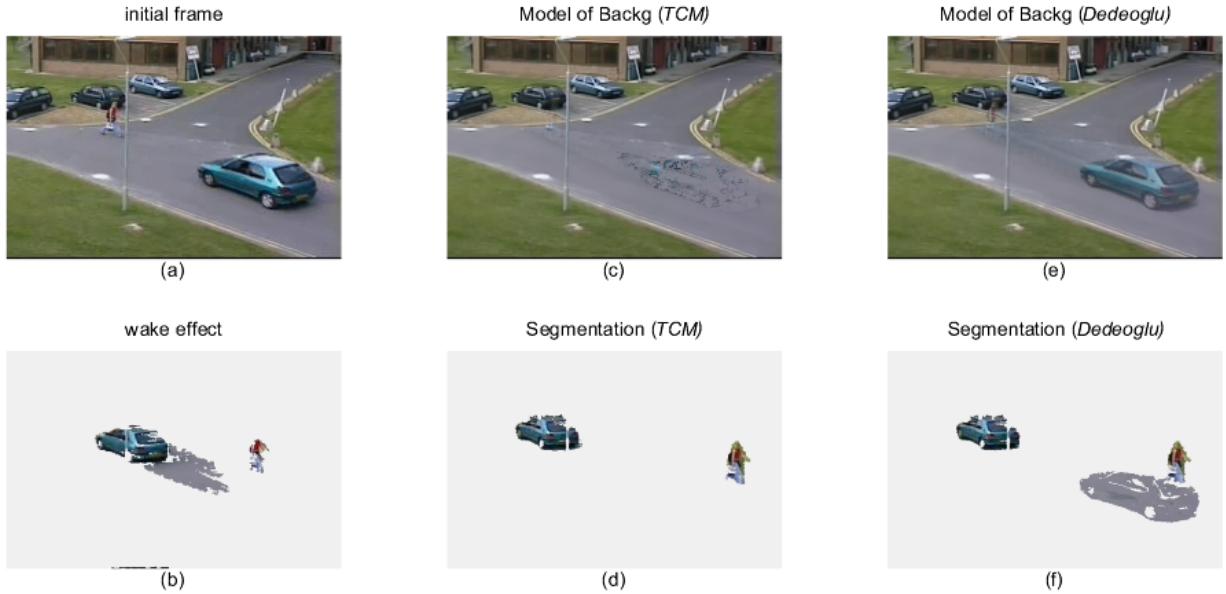

Figure 13. Qualitative comparison of the capacity to react to eliminate ghosts using the truncated cone method (TCM) and the strategy proposed by (Dedeoglu, 2004). (a) Frame no. 608, initial frame, (b) example of increased reactivity, which can be achieved with both methods, at the expense of bad segmentation, (c) frame no. 634, elimination of the ghost from the background model and (d) from the segmentation output using the TCM, (e) frame nº 634, ghost remains in the background model and, therefore, (f) it is still present in the segmentation output according to the strategy proposed by Dedeoglu.

Figure 14 shows a more detailed analysis, where the number of FP and FN for each sequence frame is counted. Obviously, when the ghosts are eliminated, the FP provide the most information on the process because, at very moment, every ghost pixel always increases the FP count. Thus in Figure 14b three very different zones can be established. The first zone, FP increase to a maximum, corresponds to the appearance of the ghost that increases its size as the vehicle follows its trajectory and abandons the position in the initial frame. The second zone, FP decrease to a low value, corresponds to the progressive updating of the background model because of the progressive charging of the permanence memory to the saturation value. The third zone, FP stay around a low value, corresponds to the stationary situation where the ghost has been completely eliminated from the background model. As it is shown in the Figure 14b, our strategy, as well as identifying the ghost pixels sooner, also updates the background model more quickly. Furthermore, the evolution of the FN (Figure 14a), as was to be expected, is practically the same in the two instances. The high value of the FN in the initial frames is because, first of all, the background model contains the very vehicle and until this abandons its initial position there will be colour coincidences between part of the vehicle and the background, causing background noise in this zone. In this example, bearing in mind that the value chosen for the saturation charge was C_{max}=48, and that the sequence was recorded at 25 fps, it means that the ghost would appear in the scene during a time equivalent to less than 2 seconds recording time. Note that if the situation had been that of a parked car that starts up and begins its movement or that of a moving car that parked, the ghost associated with both situations would also be eliminated after C_{max} frames. It should also be mentioned that with the ghost eliminating strategy the sequence on which the TCM is going to be applied no longer needs to be initialized with a set of frames where there are no moving objects. Finally, compared with other ghost eliminating strategies, ours has the advantage of being very fast computationally because it only requires checking one condition that is a function of the permanence memory values and, in turn, the updating of the PM only implies updating the values of a matrix by the addition of a constant. Thus, for example, in *Sakbot* (Cucchiara et al, 2003) several conditions have to be checked. One of them implies some computational cost because it has to calculate the average optical flow of the pixels of each blob that appears in the foreground. Nevertheless, *Sakbot* has the

advantage of not depending on the value of an initialisation constant, as indeed occurs in our strategy with the constant C_{max}.

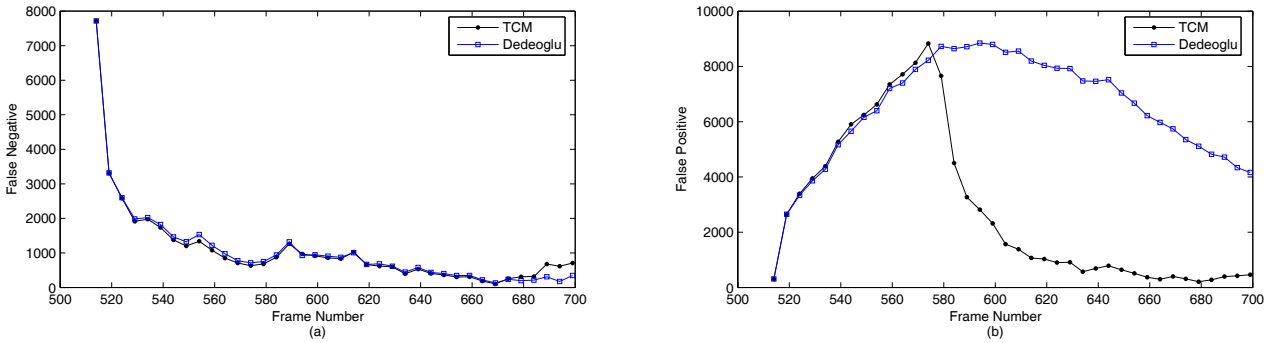

Figure 14. Comparison of reactivity to eliminating ghosts using the trunk cone method (TCM) and the strategy based on the proposal by (Dedeoglu, 2004). (a) False negatives, and (b) False positives.

7. Conclusions

The principal contribution of this work is twofold: first, it provides a new segmentation method based on the well known background subtraction method and, secondly, it establishes a set of rules that allows the characterization of the different objects detected in the segmentation process: MVOs, shadows, reflections, fluctuations, ghosts and background noise. With this blob-level knowledge, the method proposed here significantly improves MVO segmentation and background updating. The application of shadow, ghost and reflection filtering operators is based on blob-level knowledge and therefore leads to better understanding and prediction of the result of their action. This aspect is important as it facilitates the understanding of the result of each stage of the process. Furthermore, the truncated cone method is completely independent of later stages typical of a vision system like, for example, tracking and classification. In addition, it uses colour in an intuitive and natural way. Although the description of the method has been done in the framework of the RGB space, TCM could also be applied to any other colour space.

It is also important to highlight one of the most powerful characteristics of the TCM: its low computational cost. Unlike other background subtraction methods that do complex statistical background calculations for each frame, ours only requires the application of low computational cost rules and operators, and simple updatings of different model maps (background, thresholds and permanence memory). Thus, for example, for a Pentium M 1.8GHz, the video sequence used in Figure 10 (320x240 pixel image frames) and a implementation of TCM in MATLAB, the average frame rate is of 0.47 s. If we bear in mind that MATLAB is an interpreted language, it is not disproportionate to think that, for a compilable language, the previous ratio can be reduced by a factor of 10, i.e., reaching a ratio lower than 0.047 s. Consequently, ratios higher than 21 fps could be reached and thus, we would be talking of the possibility of doing tracking and classification in real time. The *Sakbot* system (Cucchiara et al, 2003) reports an average frame rate of 9.82 fps, for a sequence with the same dimension frames and with Pentium 4, 1.5 GHz. As well as low computational cost, the method also includes eliminating shadows, reflections and ghosts compared with other approaches based on background subtraction method that make segmentation but do not eliminate any kind of noise (Kim and Kim, 2003). Specifically, in (Kim and Kim, 2003) the authors report a processing time of 0.23s on average, with a image frame size of 320x340, in a Pentium III 850-MHz PC and using an MS Visual C++ implementation. Finally, the TCM was successfully tested in different types of scenes, indoor and outdoor environments and with artificial/natural light. Figure 15 shows the result of segmentation in different frames from some of these video sequences.

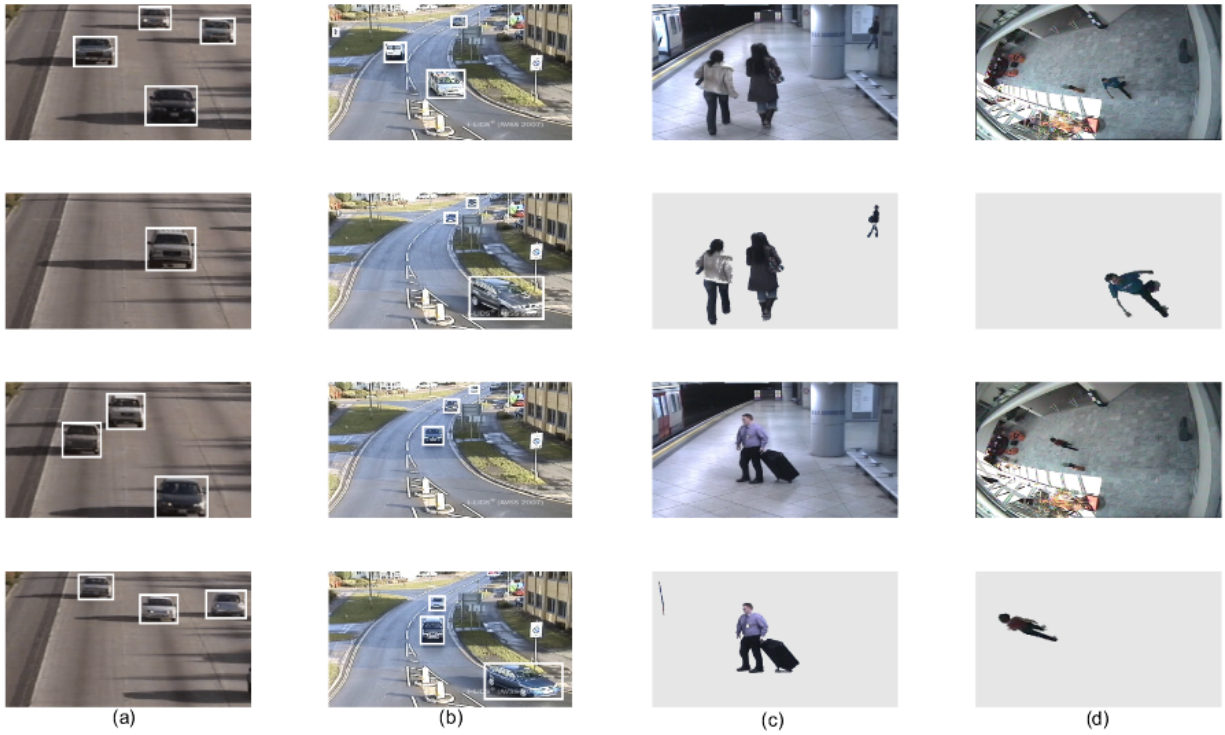

Figure 15. Examples of segmentation with the TCM applied to different types of sequences: indoor and outdoor. (a) Four frames of the *Highway*[1] sequence, (b) Four frames of the *AVSS PV Easy*[2] sequence, (c) Two frames of the *AVSS AB Easy*[2] sequence, (d) Two frames of the *Walk1*[3] sequence (in this last sequence, in order to increase the detail, a zoom has been done in the segmented frame).

Acknowledgement

This research was supported by *CICYT* via project *TIN2004-07661*.

References

Carmona E.J., Martínez-Cantos J., Mira J. 2006. Posprocesamiento morfológico adaptativo basado en algoritmos genéticos y orientado a la detección robusta de humanos. Campus Multidisciplinary in Perception and Intelligence, CMPI-2006, 10-14 July 2006, Albacete (Spain), 249-261.

Collins, R., Lipton, A., Kanade, T., Fujiyoshi, H., Duggins, D., Tsin, Y., Tolliver, D., Enomoto, N., Hasegawa, O., 2000. A system for video surveillance and monitoring: VSAM final report. Technical report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University.

Cucchiara, R., Piccardi, M., Prati, A., 2003. Detecting Moving Objects, Ghost and Shadows in Video Streams". IEEE Trans. Pattern Analysis and Machine Intelligence, 25 (10), 1337-1342.

Dedeoglu, Y., 2004. Moving Object Detection, Tracking and Classification for Smart Video Surveillance. Ph.D. Thesis, Department of Computer Engineering and The Institute of Engineering and Science of Bilkent University.

Fernández, M.A., Fernández-Caballero, A., López, M.T., Mira, J., 2003. Length–speed ratio (LSR) as a characteristic for moving elements real-time classification. Real-Time Imaging, 9 (1), 49-59.

[1] http://cvrr.ucsd.edu/aton/shadow (ATON project)
[2] http://www.elec.qmul.ac.uk/staffinfo/andrea/avss2007_d.html (AVSS 2007)
[3] http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/ (CAVIAR project)

Haritaoglu, I., Harwood, D., Davis, L.S., 2000. W4: Real-Time Surveillance of People and Their Activities. IEEE Trans. Pattern Analysis and Machine Intelligence, 22 (8), 809-830.

Horprasert, T., Harwood, D., Davis, L.S., 1999. A statistical approach for realtime robust background subtraction and shadow detection. In Proc. of IEEE Frame Rate Workshop, Kerkyra, Greece, 1-19.

Jadon, R.S., Chaudhury, S. and Biswas, K.K., 2001. A fuzzy theoretic approach for video segmentation using syntactic features, Pattern Recognition Letters, 22 (13), 1359–1369.

Kim, J.B., Kim, H.J., 2003. Efficient region-based motion segmentation for a video monitoring system. Pattern Recognition Letters, 24 (1-3), 113–128.

Kim, E.Y., Park, S.H., 2006. Automatic video segmentation using genetic algorithms. Pattern Recognition Letters 27 (11), 1252-1265.

Lee, D-S., 2005. Effective Gaussian Mixture Learning for Video Background Subtraction. IEEE Trans. Pattern Anal. Mach. Intell.,27 (5), 827-832.

Leone, A., Distante, C., 2007. Shadow detection for moving objects based on texture analysis. Pattern Recognition 40, 1222-1233.

Leone, A., Distante, C., Buccolieri, F., 2006. A shadow elimination approach in video-surveillance context. Pattern Recognition Letters, 27 (5),345-355.

Lipton, A.J., Fujiyoshi, H., Patil, R.S., 1998. Moving target classification and tracking from real-time video. In Proc. of Workshop Applications of Computer Vision, 129-136.

Martínez-Cantos, J., Carmona, E.J., Fernández-Caballero, A., López, M.T., 2006. Mejora paramétrica de la interacción lateral en computación acumulativa. Campus Multidisciplinary in Perception and Intelligence, CMPI-2006, 10-14 July 2006, Albacete (Spain), 262-273.

McKenna, S.J., Jabri, S., Duric, Z., Rosenfeld, A., Wechsler, H., 2000. Tracking Groups of People. Computer Vision and Image Understanding, vol. 80, no. 1, 42-56.

Salvador, E., Cavallaro, A., Ebrahimi, T., 2004. Cast shadow segmentation using invariant color features. Computer Vision and Image Understanding 95(2), 238–259.

Stauffer, C., Grimson, W., 1999. Adaptive background mixture models for real-time tracking. In Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 246-252.

Stauffer, C., Grimson, W., 2000. Learning Patterns of Activity Using Real-Time Tracking. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 22, no. 8, 747-757.

Wang, L., Hu, W., Tan, T., 2003. Recent developments in human motion analysis. Pattern Recognition, 36 (3), 585-601.

Wren, C., Azarbayejani, A., Darrell, T., Pentland, A.P., 1997. Pfinder: Real-Time Tracking of the Human Body. IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 19, no. 7, 780-785.

Xu, D., Li, X., Liu, Z., Yuan, Y., 2005. Cast shadow detection in video segmentation. Pattern Recognition Letters, 26 (1), 91-99.