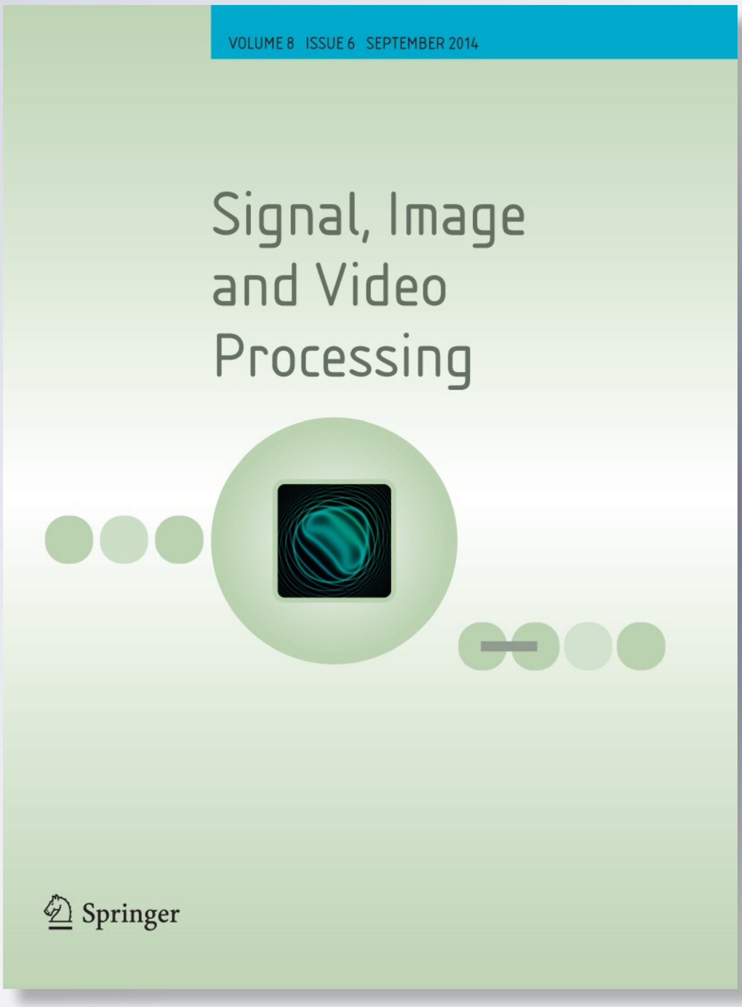
Color video segmentation by lateral inhibition in accumulative computation

Antonio Fernández-Caballero, María T. López, Juan Serrano-Cuerda & José Carlos Castillo

Signal, Image and Video Processing

ISSN 1863-1703
Volume 8
Number 6

SIViP (2014) 8:1179-1188
DOI 10.1007/s11760-014-0656-1


VOLUME 8 ISSUE 6 SEPTEMBER 2014
Signal, Image and Video Processing
Springer

SIViP (2014) 8:1179–1188
DOI 10.1007/s11760-014-0656-1

ORIGINAL PAPER

Color video segmentation by lateral inhibition in accumulative computation

Antonio Fernández-Caballero · María T. López · Juan Serrano-Cuerda · José Carlos Castillo

Received: 2 April 2013 / Revised: 21 February 2014 / Accepted: 4 June 2014 / Published online: 28 June 2014

Abstract The lateral inhibition in accumulative computation (LIAC) algorithm has proved to be an efficient method for moving object segmentation in gray-level video sequences. This paper reviews the main steps and features of the LIAC algorithm, and assesses the suitability of applying the LIAC algorithm to the segmentation of color videos. Two widely used color spaces, namely RGB and HLS, are used for validating the LIAC algorithm, and a comparison is provided after performance evaluation of the algorithm in both color spaces.

Keywords Lateral inhibition in accumulative computation · Moving objects · Color spaces · Video segmentation

1 Introduction

Color by itself is a powerful feature in the distinction and recognition of objects. There are numerous papers dedicated to color-based segmentation (e.g. [2,17]). HSI and RGB color spaces are used for traffic sign detection [14]. Recently, color and texture are modeled with the RGB space and LBP method, respectively, to classify banknotes of different countries [13]. Also, some approaches use several image features combined to color for the segmentation problem (e.g. [23,24]). Other approaches use mixtures of deformable part models to represent highly variable object classes [10]. This work relies on the classification of partially labeled models using *latent SVM* formalisms. In a recent paper, a new color space, called the RGB color ratio space, is proposed and defined according to a reference color such that an image can be transformed from a conventional color space to the RGB color ratio space [6]. Another recent technique [3] copes with contour detection and image segmentation integrated in the so called algorithm *gPb-owt-ucm*. Combining motion and color is also a well-known option [4]. An approach is proposed where color segmented regions are used to constrain the motion fitting [1]. Another approach also combines color and motion information in an attempt to fuse color segmentations with motion estimates obtained using block correlations [25]. Another paper exploits color information for both background subtraction and shadow detection to improve moving object segmentation and background update [7]. Another framework takes as input two label fields [16], a quickly estimated and to-be-refined segmentation map and a spatial region map that exhibits the shape of the main objects of the scene. Another recent approach combines morphological, color and textural feature [5].

Now, lateral inhibition in accumulative computation (LIAC) has proved to be an efficient method for moving object segmentation in gray-level video sequences [22] and has been implemented in real time [9]. Due to its versatility, the LIAC method has been applied successfully to dynamic visual attention [19] in surveillance applications [18] in order to monitor human activities [11]. Also some works have provided enhancements through the inclusion of genetic algorithms [21] and stereoscopy [20]. The current paper reviews two widely used color spaces—RGB and HLS—and assesses the suitability of using the LIAC method in both color spaces. The choice of a color model is of great importance for many computer vision algorithms. The RGB model specifies colors using three primary intensities—red (R), green (G) and blue (B)—which can be plotted along the axes

A. Fernández-Caballero (✉) · M. T. López
Departamento de Sistemas Informáticos, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
e-mail: antonio.fdez@uclm.es

J. Serrano-Cuerda · J. C. Castillo
Instituto de Investigación en Informática de Albacete (I3A), 02071 Albacete, Spain

of a unit cube. The HLS model is based on the perceptual variables hue (H), luminance (L) and saturation (S); it is one of several related color spaces based on polar coordinates.

The rest of the paper is structured as described next. Section 2 introduces the LIAC method in the motion detection task in color videos. Section 3 introduces a set of examples which enable comparing the performance of the LIAC method in two well-known color spaces, namely, RGB and HLS. Lastly, Sect. 4 offers the most important discussion and conclusions.

2 Lateral inhibition in accumulative computation (LIAC)

The problem we are stating by means of LIAC is the discrimination of moving objects capable of holding our attention in a scene. Motion allows to obtain gradually all moving objects' shapes through a mechanism called accumulative computation. Then, the algorithm fuses spots obtained by means of neurally inspired lateral inhibition (LI) and thresholding. Figure 2a, b shows two consecutive input images (frames 133 and 136) from the BEHAVE Interactions Test Case Scenarios (downloaded from http://groups.inf.ed.ac.uk/vision/BEHAVEDATA/INTERACTIONS/). Figure 2i shows the result of applying LIAC to the sequence after frame number 136. The input images will be used as a running example for a better understanding of the proposed method.

The complete LIAC architecture is shown in Fig. 1, where the reader may have a first contact with the modules of the method. From [12], we cite and reformulate the most important concepts and equations of the LIAC method for the motion detection task on videos. The adaptation of the LIAC algorithm to color video requires to expand from one unique gray-level component to three color components of the color space used, be it RGB or HLS. In the equations adapted for the color spaces $(r/g/b)$ and $(h/l/s)$ are used for the three components of the color spaces RGB and HLS, respectively. Also (κ) stands generically for r, g, or b, and h, l, or s, when the same equation applies to all color components. Next, each one of the modules is described in detail. Also, the influence of the most important parameters of the LIAC algorithm are briefly explained. A more detailed explanation of the parameters is available in a previous work [12].

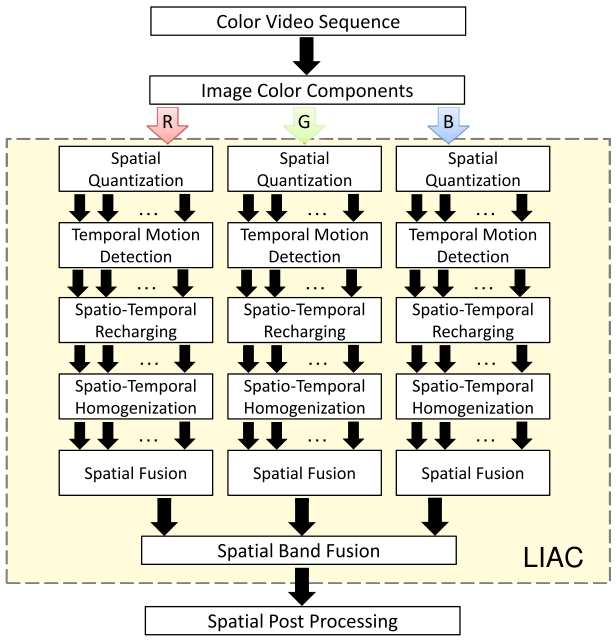


Fig. 1 *RGB*-based LIAC architecture for color video sequences

2.1 Spatial quantization

The module performs an uniform quantization of the color input image $C(x, y; t)$ segmenting each component κ of the color, that is to say $C_i^\kappa(x, y; t)$, into a preset group of bands (N). A high value of N usually enables to better discriminate the whole shapes of the moving non-rigid objects. Nevertheless, a too high value of this parameter may include some image background into the shapes. This may even lead to fuse more than one different shape into one single silhouette.

Now, there is a clear difference in RGB and HLS color spaces. In the RGB color space, we have Eq. (1) for each one of the color components:

$$\lambda_i^{r/g/b}(x, y; t) = \begin{cases} 1, & \text{if } C^{r/g/b}(x, y; t) \in [Q^{r/g/b} \cdot i, Q^{r/g/b} \cdot (i+1) - 1] \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $i \in [0..N-1]$ is the band and $C^{r/g/b}(x, y; t)$ is the r, g or b component of the color input image at time instant t. Notice that the value of quantization step Q differs depending on the color space. For instance, r, g and b components range from values 0 to 255. Therefore, the value 256 is used in the formula. However, in the HLS color space, remember that the h component ranges from 0° to 360°, whereas the s and l components range from 0 to 100 %. Thus, we have Eqs. (3) and (4), with Q calculated as shown in Eq. (2).

$$Q^\kappa = \begin{cases} \frac{256}{N}, & \text{if } \kappa \in r/g/b \\ \frac{360}{N}, & \text{if } \kappa = h \\ \frac{100}{N}, & \text{otherwise} \end{cases} \tag{2}$$

$$\lambda_i^{h}(x, y; t) = \begin{cases} 1, & \text{if } C^{h}(x, y; t) \in [Q^{h} \cdot i, Q^{h} \cdot (i+1) - 1] \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$\lambda_i^{l/s}(x, y; t) = \begin{cases} 1, & \text{if } C^{l/s}(x, y; t) \in [Q^{l/s} \cdot i, Q^{l/s} \cdot (i+1) - 1] \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

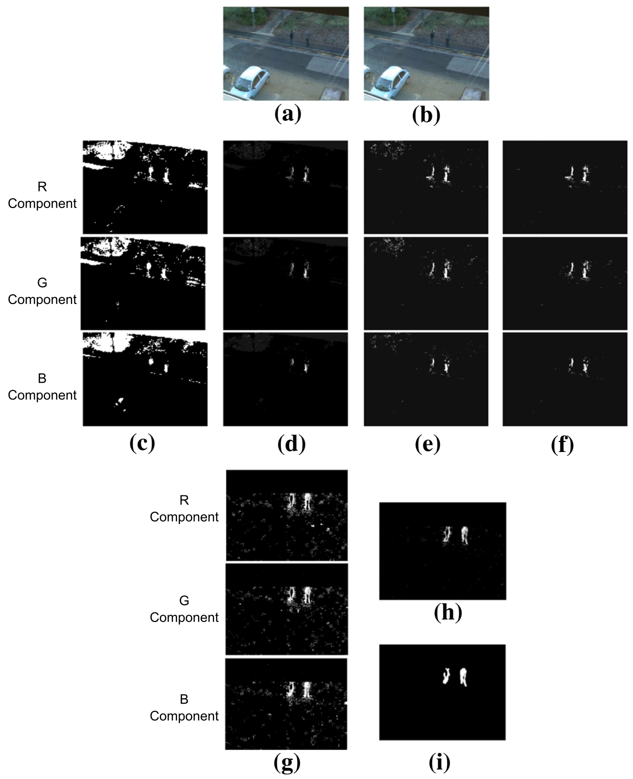


Fig. 2 Use of LIAC in color videos. **a** Running example input image number 133 of the sequence. **b** Running example input image number 136 of the sequence. **c** R, G and B components for spatial quantization on band $N = 0$. **d** R, G and B components for temporal motion detection on band $N = 0$. **e** R, G and B components for spatio-temporal recharging on band $N = 0$. **f** R, G and B components for spatio-temporal homogenization on band $N = 0$. **g** Spatial fusion results for R, G and B components of the input image number 136. **h** Result after fusing the charges of all bands. **i** Result after thresholding and human size fitting

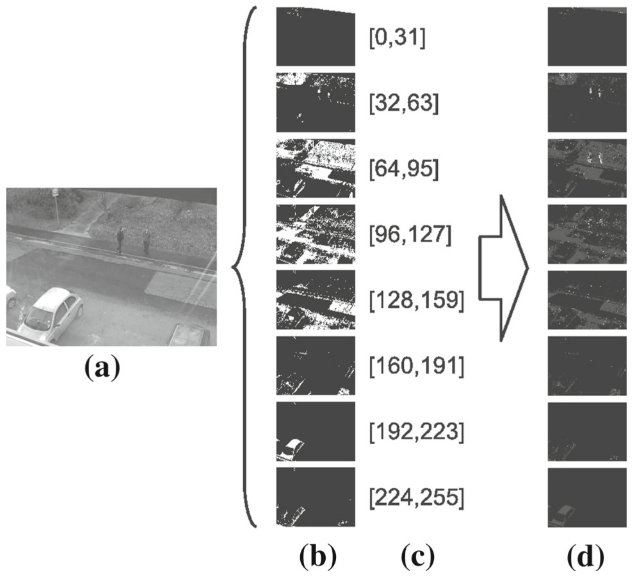


Fig. 3 Results of the spatial and temporal quantization for image number 136 of the sequence. **a** Input R-channel of the frame. **b** Spatial quantization results. **c** Ranges within each spatial quantization result. **d** Temporal quantization results

Here $C^{h/l/s}(x, y; t)$ is the h, l or s component of the color input image in HLS format at time instant t. For instance, see the result for RGB on band $N = 0$ from the quantization into $N = 8$ bands of the running example at Fig. 2c as well as the quantization into 8 bands for the *R-channel* of an image at Fig. 3b. The ranges of the original image within each band are shown at Fig. 3c. From this point on, the formulas are the same for both color spaces RGB and HLS.

2.2 Temporal motion detection

Now, a charge or discharge due to motion detection is performed. This module has been designed to obtain the accumulated charge $q_i^{\kappa}(x, y; t)$ on a quantization basis in 3 layers (color components), and each one of them will memorize the value of the accumulative computation present at time scale t for each pixel (x, y). Indeed, the accumulated charge value at each band i ($i = 0, \ldots, N-1$), q_i^{κ}, related to motion detection at each input image pixel is obtained, as shown in the following formula:

$$q_i^{\kappa}(x, y; t) = \begin{cases} v_{\mathrm{dis}}, & \text{if } ((\lambda_i^{\kappa}(x, y; t) = 0) \wedge \\ \quad (\lambda_i^{\kappa}(x, y; t - \Delta t) = 0)) & \\ v_{\mathrm{sat}}, & \text{if } ((\lambda_i^{\kappa}(x, y; t) = 0) \wedge \\ \quad (\lambda_i^{\kappa}(x, y; t - \Delta t) = 1) \wedge & \\ \quad (\lambda_i^{\kappa^{+th}}(x, y; t) = 0) \wedge & \\ \quad (\lambda_i^{\kappa^{-th}}(x, y; t) = 0)) & \\ v_{\mathrm{sat}}, & \text{if } ((\lambda_i^{\kappa}(x, y; t) = 1) \wedge \\ \quad (\lambda_i^{\kappa}(x, y; t - \Delta t) = 0) \wedge & \\ \quad (\lambda_i^{\kappa^{+th}}(x, y; t - \Delta t) = 0) \wedge & \\ \quad (\lambda_i^{\kappa^{-th}}(x, y; t - \Delta t) = 0)) & \\ \max[q_i^{\kappa}(x, y; t - \Delta t) - v_{\mathrm{dm}}, v_{\mathrm{dis}}], & \text{otherwise} \end{cases}$$

where $\lambda_i^{\kappa^{+th}}$ and $\lambda_i^{\kappa^{-th}}$ are defined to relax the division into bands for RGB (and in a similar way for HLS) as:

$$\lambda_i^{\kappa^{+th}}(x, y; t) = \begin{cases} 1, & \text{if } C^{\kappa^{+th}}(x, y; t) \in \\ \quad \left[\left(\frac{range}{N} + th\right) \cdot i, \frac{range}{N} \cdot (i + 1) - 1\right] & \\ 0, & \text{otherwise} \end{cases}$$

$$\lambda_i^{\kappa^{-th}}(x, y; t) = \begin{cases} 1, & \text{if } C^{\kappa^{-th}}(x, y; t) \in \\ \quad \left[\left(\frac{range}{N} - th\right) \cdot i, \frac{range}{N} \cdot (i + 1) - 1\right] & \\ 0, & \text{otherwise} \end{cases}$$

where $range = 256$ for r, g and b components; $range = 360$ for h component; $range = 100$ for l and s components.

At each pixel (x, y), we are in front of three possibilities:

1. The charge value at pixel (x, y) is discharged down to v_{dis} (the minimum allowed charge value) when no motion

information may be detected at band i. No motion information is available as pixel (x, y) does not correspond to band i.

2. The charge value at pixel (x, y) is saturated to v_{sat} (the maximum charge value) when motion is detected at t. Motion is detected as image pixel now belongs to this band at time instant t, and it did not correspond to the band at the previous instant $t - \Delta t$, or vice versa taking into account $\lambda_i^{\kappa^{+th}}$ and $\lambda_i^{\kappa^{-th}}$.
3. The charge value at pixel (x, y) is decremented by a value v_{dm} when motion goes on being detected in consecutive intervals t and $t - \Delta t$. Of course, the permanence value cannot get off a minimum value v_{dis}. Notice that the discharge of a pixel by a quantity of v_{dm} is the way to stop maintaining attention to a pixel of the image that did capture our interest in the past. As it will be seen later on, if a pixel is not directly or indirectly bound by means of lateral inhibition mechanisms to a maximally charged pixel, v_{sat}, it goes down to the total discharge with time.

The results for the running example may be found in Fig. 2d and in more detail in Fig. 3d. The influence of v_{dm} is as follows. Different values of the discharge value due to motion detection offer different trails of the movement in the consecutive output images. When lowering the value of v_{dm}, more information of the history of the movement is obtained through the offered trail.

2.3 Spatio-temporal recharging

Lateral inhibition is thought here to reactivate the accumulated charge with an extra charge v_{rv} of those pixels which are partially loaded (charge different from v_{dis} and v_{sat}) and directly or indirectly connected to maximally charged pixels (whose charge is equal to v_{sat}). Thus, v_{rv} is the recharge value. Initially, the new charge values Q_i^{κ} are initialized to the charge values provided at the previous step, q_i^{κ}.

Spatio-temporal recharging occurs in steps after t and before the next frame. The value of $\Delta\tau$ will determine the number of times the value at each pixel is calculated in accordance with the distance of the connectivity.

In order to explain the notion of this step, we will say that the activation toward the lateral modular structures (up, down, right and left) is based on the following basic ideas: (1) All modular structures with maximum accumulated charge value v_{sat} (saturated) output the charge toward the neighbors. (2) All modular structures with a non-saturated charge value allow passing this information through them if activated by some neighbor (they behave as transparent structures to the charge passing). (3) The modular structures with minimum permanence value v_{dis} (discharged) stop the passing of the charge information toward the neighbors (they behave as opaque structures). Therefore, we are in front of an explosion of lateral activation which begins at the structures with accumulated charge set to v_{sat}, and spreads lineally toward all directions, until a structure appears in the pathway with a complete discharge. One important issue is that the recharge at each pixel takes place at most once. The variable r is used for controlling this fact. Passing the charge to the neighbors is performed by using the variable o.

Initially, the values for variables o and r are setup depending on the charge value of each pixel (x, y), as follows:

- For pixels with maximum accumulated charge value v_{sat}: $o_i^{\kappa}(x, y) = 1$ - to let passing the charge $r_i^{\kappa}(x, y) = 0$ - to not accept a recharge
- For pixels with minimum accumulated charge value v_{dis}: $o_i^{\kappa}(x, y) = 0$ - to not let passing the charge $r_i^{\kappa}(x, y) = 0$ - to not accept a recharge
- For pixels with intermediate accumulated charge value $v_{\mathrm{dis}} < v < v_{\mathrm{sat}}$: $o_i^{\kappa}(x, y) = 0$ - to not let passing the charge $r_i^{\kappa}(x, y) = 1$ - to accept a recharge

The spread of charge toward the neighbors, starting from the pixels with maximum accumulated charge value v_{sat}, may be formulated as:

$Q_i^{\kappa}(x, y; t + l \cdot \Delta\tau) = v_{\mathrm{sat}}$,

where l controls the number of iterations before the next frame, as saturated pixels do not change their charge value. Also, the control variables now change to:

$o_i^{\kappa}(x, y) = 0$ - to not let passing the charge

$r_i^{\kappa}(x, y) = 0$ - to not accept a recharge

The opaque discharged (v_{dis}) pixels neither change their charge value. No change is provided in variables i or o.

$Q_i^{\kappa}(x, y; t + l \cdot \Delta\tau) = v_{\mathrm{dis}}$

$o_i^{\kappa}(x, y) = 0$ - to not let passing the charge

$r_i^{\kappa}(x, y) = 0$ - to not accept a recharge

Lastly, let us consider the situation of those pixels with intermediate charge value. A recharge is only possible when $r_i^{\kappa}(x, y) = 1$ and when one direct neighbor is offering the possibility of a recharge, that is $o_i^{\kappa}(x \pm 1, y \pm 1) = 1$. So:

$Q_i^{\kappa}(x, y; t + l \cdot \Delta\tau) = \min[Q_i^{\kappa}(x, y; t + (l-1) \cdot \Delta\tau) + v_{\mathrm{rv}}, v_{\mathrm{sat}}]$,

if $((r_i^{\kappa}(x, y) = 1) \wedge ((o_i^{\kappa}(x-1, y) = 1) \vee (o_i^{\kappa}(x+1, y) = 1) \vee (o_i^{\kappa}(x, y-1) = 1) \vee (o_i^{\kappa}(x, y+1) = 1)))$

After the recharge, we have:

$o_i^{\kappa}(x, y) = 1$ - to let passing the charge

$r_i^{\kappa}(x, y) = 0$ - to not accept a recharge

From this point on, as the pixel will not be recharged again:

$Q_i^{\kappa}(x, y; t + l \cdot \Delta\tau) = Q_i^{\kappa}(x, y; t + (l-1) \cdot \Delta\tau)$

$o_i^{\kappa}(x, y) = 0$ - to not let passing the charge

$r_i^{\kappa}(x, y) = 0$ - to not accept a recharge

This scheme is repeated until it is not possible to recharge any more pixels or the next frame arrives. See the results for the running example in Fig. 2e. Notice that the recharge has as secondary effect, recovering part of the history of motion. The accumulated charge of each pixel will be offered to the following module as output.

2.4 Spatio-temporal homogenization

In this module, the charge is distributed among all the connected neighbors holding a minimum charge (greater than υ_{dis}), once again by means of lateral inhibition mechanisms. Θ_i^κ, the homogenized charge value, is initialized to Q_i^κ. This occurs according to Eq. (6).

$$\begin{aligned}&\Theta_i^\kappa(x,y;t+m\cdot\Delta\tau)\\&=\frac{1}{1+\delta_{x-1,y}+\delta_{x+1,y}+\delta_{x,y-1}+\delta_{x,y+1}}\\&\quad\times[\Theta_i^\kappa(x,y;t+(m-1)\cdot\Delta\tau)+\\&\quad+\delta_{x-1,y}\cdot\Theta_i^\kappa(x-1,y;t+(m-1)\cdot\Delta\tau)\\&\quad+\delta_{x+1,y}\cdot\Theta_i^\kappa(x+1,y;t+(m-1)\cdot\Delta\tau)\\&\quad+\delta_{x,y-1}\cdot\Theta_i^\kappa(x,y-1;t+(m-1)\cdot\Delta\tau)\\&\quad+\delta_{x,y+1}\cdot\Theta_i^\kappa(x,y+1;t+(m-1)\cdot\Delta\tau)]\end{aligned}\tag{5}$$

where

$$\begin{aligned}&\forall(\alpha,\beta)\in[x\pm1,y\pm1],\delta_{\alpha,\beta}\\&=\begin{cases}1, & \text{if } \Theta_i^\kappa(\alpha,\beta;t+(m-1)\cdot\Delta\tau)>\upsilon_{\text{dis}}\\0, & \text{otherwise}\end{cases}\end{aligned}\tag{6}$$

The explanation of this data clustering-based method is as follows. Starting from the values of the accumulated charge values in each pixel on a band basis, we will see how it is possible to obtain all the parts of a moving object. A part of an object is just the union of pixels that are together and in a same band. The charge is homogenized among all the pixels that pertain to the same band and that are directly or indirectly united to each other. This way, a double objective will be obtained:

1. Diluting the charge due to the false image background motion along the other pixels of the background. So, there should be no presence of the motion characteristic of the background, but we will rather keep motion of the objects present in the scene.
2. Obtaining a parameter common to all the pixels of the part of the object in a surrounding window with a same band.

See the result of the homogenized accumulated charges on the running example at Fig. 2f.

2.5 Spatial fusion

During this step, we take the maximum value of all outputs of the i bands, as described in Eq. (7), to show the detected blobs associated to a moving object as obtained for each color component (see Fig. 2g):

$$\Theta^\kappa(x,y;t)=\arg\max_i \Theta_i^\kappa(x,y;t)\tag{7}$$

2.6 Spatial band fusion

The final output segmentation result is obtained as a logical AND of the three Θ partial outputs, that is:

$$\Theta(x,y;t)=\Theta^{\kappa1}(x,y;t)\wedge\Theta^{\kappa2}(x,y;t)\wedge\Theta^{\kappa3}(x,y;t)\tag{8}$$

Also, you may take a look at Fig. 2h.

2.7 Spatial post-processing

This module performs a binarization with threshold Θ_{obj} [see Eq. (9)]. Values over threshold are set to max (255) and below threshold are set to min (0). Once the image is binarized, some morphologic operations leading to eliminate image noise are performed. Firstly, an erosion is performed in order to eliminate isolated and small spots [see Eq. (10)]. This is done a number of times n_o. And, secondly, a dilation operation is computed (a number of times n_c) to enhance the remaining spots [see Eq. (11)].

$$\Theta_b(x,y;t)=\begin{cases}min, & \text{if } \Theta(x,y;t)\le\Theta_{obj}\\max, & \text{otherwise}\end{cases}\tag{9}$$

$$\Theta_o(x,y;t)=\Theta_b(x,y;t)\ominus\begin{vmatrix}0&1&0\\1&1&1\\0&1&0\end{vmatrix}\tag{10}$$

$$\Theta_c(x,y;t)=\Theta_o(x,y;t)\oplus\begin{vmatrix}0&1&0\\1&1&1\\0&1&0\end{vmatrix}\tag{11}$$

Finally, spots are filtered based on their features, such as height, width and compactness. For this purpose, minimum and maximum values are established: $h_{\min}$ and $h_{\max}$ for the height, $w_{\min}$ and $w_{\max}$ for the width, and $c_{\min}$ and $c_{\max}$ for the compactness. Visually, we have the result shown on Fig. 2i.

3 Data and results

A qualitative and quantitative comparison of the performance of the LIAC algorithm in color video for RGB and HLS color spaces is provided in this section. Also, a quantitative comparison between color-based and gray-level-based LIAC is introduced. Firstly, Sect. 3.1 introduces the metrics used for the quantitative assessment of the LIAC performance. Then, in Sect. 3.2 the two input data video sequences used as case studies, namely BEHAVE and CAVIAR, are described.

Lastly, some qualitative and quantitative results are offered in Sects. 3.3 and 3.4, respectively.

3.1 Metrics definition and setup

Before explaining the metrics to evaluate the performance of the algorithm in the color spaces, let us define the basic concepts used for their calculation. First of all, we have to highlight that all the metrics are related to the bounding boxes of the moving objects detected by the LIAC algorithm. In agreement with the computer vision community, we accept the following standard definitions:

- True Positive (TP): the system has detected a real situation (bounding box exists in reference data and algorithm results).
- False Positive (FP): the system has detected a situation that is not real (bounding box exists only in algorithm results).
- False Negative (FN): a real situation has been missed by the system (bounding box exists only in reference data).

The previous data provide input to some of the most accepted metrics. These are:

- Precision: indicates the true positives ratio over the whole set of detections (true positives + false positives).

$$Precision = \frac{TP}{TP + FP} \tag{12}$$

- Recall: approximates the probability of the positive label being true; in other words, it assesses the effectiveness of the algorithm on a single class. A recall of 100 % means that the test recognizes all positives.

$$Recall = \frac{TP}{TP + FN} \tag{13}$$

- *F-score*: is a measure of a test's accuracy. It considers both the precision and the recall of the test to compute the score. The *F-score* can be interpreted as a weighted average of the precision and recall (see Eq. (14)), where an *F-score* reaches its best value at 1 and worst score at 0.

$$F - score = 2 \times \frac{precision \times recall}{precision + recall} \tag{14}$$

We have experimentally established that a spot detected by the LIAC algorithm (and represented as its bounding box) matches a bounding box existing in the reference data if *F-score* > 0.15. This minimum *F-score* value ensures that bounding boxes in algorithm result and reference data with low overlapping areas are discarded, and considered as a false positive (FP) plus a false negative (FN).

A threshold *F-score* of 0.15 was experimentally selected. This value is motivated through the two reasons explained next. Firstly, let us point out the motion direction trails generated by the LIAC algorithm (see Fig. 4a, where the green box corresponds to the detected blob including the trail due to accumulative computation and the blue box corresponds to the ground truth). Due to the trails, the bounding boxes of the detected objects use to be greater than the real objects contained. Indeed, the bounding boxes not only contain the moving object at its current location, but also a part of the moving object in some previous locations depending on the LIAC parameters established and the speed of the moving object. This imposes a low value of the *F-score* threshold to avoid a number of missed detections.

Secondly, there is the situation of little overlapping between the ground truth and the detected objects. This is motivated due to two possible reasons. In first place, there is the typical problem of moving objects only partially detected (e.g. due to occlusions or to bad segmentation performance in some specific lightning conditions; see Fig. 4b, c). In this case, a too low value of the *F-score* threshold does not work properly. Indeed, if the *F-score* threshold is too low, a small overlapping can be considered as a hit (see Fig. 4c, d). This kind of small overlapping may be caused by the intersection between the trails of two objects or the overlapping of a false positive with the ground truth.

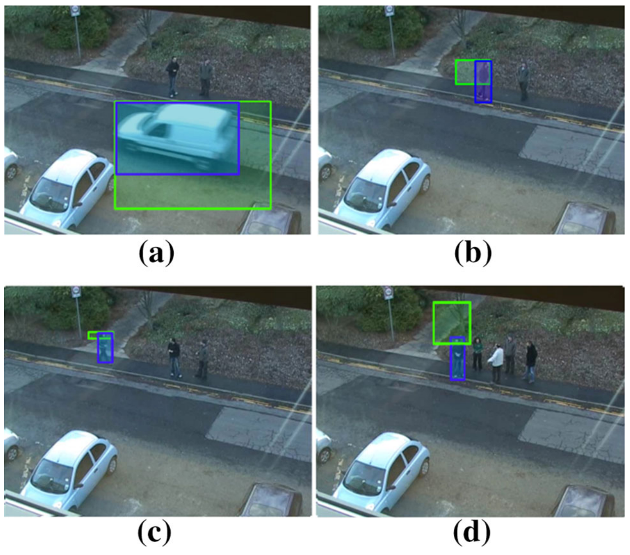


Fig. 4 Toward an optimal *F-score* threshold. **a** The blob detected by the LIAC algorithm is greater than the ground truth due to the characteristic accumulative computation trail. **b** Part of the human has not been detected by the LIAC algorithm, and there is part of the trail in the detected blob. **c** Most of the human has not been detected, and the ground truth contains the detected blob. **d** Overlapping of a false positive with the ground truth

3.2 Input videos description

3.2.1 BEHAVE and CAVIAR scenarios

In first place, we have tested the LIAC algorithm in the BEHAVE Interactions Test Case Scenario 0 (see http://groups.inf.ed.ac.uk/vision/BEHAVEDATA/), composed of 11, 200 image frames. The data set comprises of various scenarios of people acting out various interactions. The data are captured at 25 frames per second. The resolution is 640 $\times$ 480. A lot (but not all) of the video sequences have ground truth bounding boxes of the humans in the scene. In the BEHAVE Interactions Test Case Scenario 0, there are five subsequences: grouped humans with little motion, groups with median motion, a vehicle crossing the scenario, a cyclist crossing the scenario and humans crossing the scenario.

Then, we have tested the algorithm in the CAVIAR Test Case Scenario "Walk1" (see http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/). For the CAVIAR project, a number of video clips were recorded acting out the different scenarios of interest. These include people walking alone, meeting with others, window shopping, entering and exiting shops, fighting and passing out and leaving a package in a public place. The first section of video clips (including the "Walk1" scenario) was filmed with a wide angle camera lens at the entrance lobby of the INRIA Labs at Grenoble, France. The resolution is half-resolution PAL standard (384 $\times$ 288 pixels, 25 frames per second). In CAVIAR, we may find only one type of subsequence, where some persons are crossing the scenario. Table 1 shows the values used for the most important parameters of the LIAC algorithm.

Table 1 Values for the most relevant parameters of the LIAC algorithm

Test case	BEHAVE	CAVIAR
Number of bands	8	8
Maximum charge value (v_{sat})	255	255
Minimum charge value (v_{dis})	0	0
Discharge value (v_{dm})	63	63
Recharge value (v_{rv})	31	31
Number of erosions (n_o)	1	2
Number of dilations (n_c)	3	5
Minimum height ($h_{\min}$)	30	20
Maximum height ($h_{\max}$)	500	70
Minimum width ($w_{\min}$)	30	20
Maximum width ($w_{\max}$)	500	70
Minimum compactness ($c_{\min}$)	1	1
Maximum compactness ($c_{\max}$)	100	100

3.2.2 Ground truth files description

In order to evaluate the performance of the algorithm, we have used the reference data provided by the BEHAVE and CAVIAR projects as ground truth files. It has been necessary to implement a tool for transforming the data models of both projects into a common model, capable of providing the data needed to perform the performance assessment. The selected model is the one used in the OTCBVS Benchmark Dataset Collection [8] (see http://www.cse.ohio-state.edu/otcbvs-bench/). Here, each frame indicates the number of objects and the rectangle containing each object, $(x_{\min}, y_{\min}, height, width)$. The overall structure of the original ground truth files is the following one:

– Ground truth for BEHAVE: organizes the sequence in an XML file on the basis of the objects that are present. For each object, the frame ranges where the object appears are provided, as well as the position of the object in each frame range.
– Ground truth for CAVIAR: provides much more information on the objects. Firstly, for each frame, CAVIAR provides information at object and at group of objects level. And, for each object or group, it indicates its position, height and width, and some other features related to the activities performed. It also gives the orientation in the vertical axis.

Let us also highlight that a direct comparison of the algorithm results and the reference data has not been possible. This is due to two reasons. In first place, the ground truth files (CAVIAR and BEHAVE) only contain information about the persons, and not about other kinds of moving objects that might be detected from their motion. Nevertheless, the LIAC algorithm will detect the moving vehicle, which would be classified as false positive when matched toward the ground truth. Therefore, this case, for instance, has not been considered when obtaining the statistics, although it is shown as a result of the algorithm segmentation.

In second place, the ground truth data offer the positions of the people even if they are motionless during some time. As the LIAC algorithm is motion-based, it is not able to detect these people unless they move a sufficient distance. This is why for every non-detected human, we consider the possibilities that the human has not changed its position since the previous frame (he/she is stationary) or that the position has only varied slightly (he/she is quite still). For this last case, a tolerance of ± 2 pixels at each corner of the bounding box is assumed.

3.3 Qualitative comparison between RGB- and HLS-based LIAC

Some qualitative results for the BEHAVE Test Case Scenario 0 are offered in Fig. 5. Figure 5a–f shows some frames with

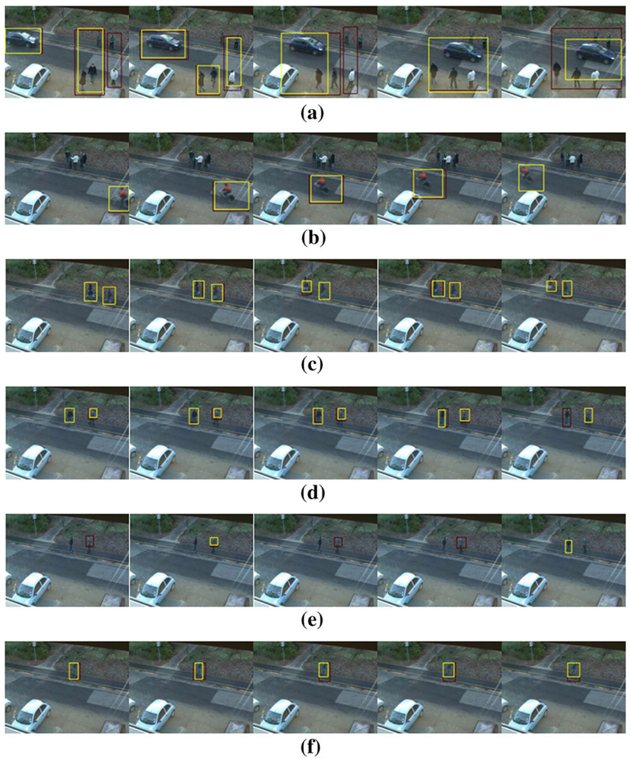


Fig. 5 Qualitative results of applying the color-based LIAC algorithm to the BEHAVE test case; *yellow and red bounding boxes* correspond to the output of the RGB-based and HLS-based LIAC algorithm, respectively. **a** Moving car and people moving slightly. **b** Moving cyclist and non-moving people. **c** and **d** Moving people. **e** People moving slowly. **f** A moving human (color figure online)

significant outputs. In the input video frames, red and yellow bounding boxes stand for moving objects as segmented in the RGB and HLS color spaces, respectively. Figure 5a shows frames 9,303, 9,320, 9,333, 9,359 and 9,363 containing a moving car and people moving slowly. In Fig. 5b, we have frames 3,684, 3,692, 3,704, 3,713 and 3,728, where you may observe a moving cyclist and some stationary people. Figure 5c, d are examples of the presence of moving people (frames 6,327, 6,357, 6,369, 6,376 and 6,398, as well as frames 102–104, 107 and 109). Figure 5e, composed of frames 122–125 and 133, contains people with little motion. Lastly, we offer the results of the segmentation of a single moving human in Fig. 5f (frames 4,379–4,383).

As you may easily observe in the figure, generally LIAC in RGB color space has a better performance than LIAC in HLS in all subsequences. Indeed, in RGB, a greater number of moving objects is detected. This is the case even in sequences with little motion (see Fig. 5e), where in RGB produces a greater number of true positives. In Fig. 5b, we may observe that the LIAC algorithm is capable of detecting the cyclist in both color spaces with a great precision, while the humans are ignored as they have too little motion. In the last frame shown, both bounding boxes coincide. In Fig. 5a, the algorithm tends to unite the grouped persons, independently of the color space used. Also in this case the performance has better hits (true positives) in the RGB color space.

3.4 Quantitative comparison between RGB- and HLS-based LIAC

After calculating the metrics introduced previously, we get the results shown in Table 2. Again, it is easy to observe that the LIAC algorithm in the RGB color space outperforms the algorithm in HLS color space. The number of true positives in RGB is greater for both test cases BEHAVE and CAVIAR. Also the number of false negatives is much lower in RGB for both test cases. Therefore, although the LIAC algorithm possesses a great recall in both color spaces, it is closer to the ideal case (a value of 1) for RGB. The same is confirmed through the comparison of the *F-score* values in the couple of test cases. The values offered by the LIAC algorithm used in the RGB color space are quite better.

3.5 Quantitative comparison between gray-level- and color-based LIAC

Also, a comparison with processed gray-level videos (obtained through the L component of the HLS color space) is offered in this new version. As shown in Table 2, the results largely suffer from illumination changes in the tested environment. CAVIAR images belong to an indoor data set (with quite constant illumination), while BEHAVE videos are composed of outdoor images (with important illumination changes). The results of the LIAC algorithm using gray-level images show how high illumination changes lower the algorithm performance. This is translated into the results with an *F-score* of 0.79 and values of precision and recall of 0.79 and 0.66, respectively, produced by high false positives and negatives rates.

At this point, it is worth explaining that blobs in the L component are often discarded since their shadows are sometimes included within the proper area of the blob, resulting on a height higher than $h_{\max}$ or a width above $w_{\max}$. For this reason, some blobs are discarded, provoking false negatives. The AND operation between the three color components results in a reduction of the object area, since the H and S components might contain more accurate representations of the object's area and location. The new width and height of the object after this operation enable the blob to be within the acceptable ranges of width and height, leading to a higher number of true positives. Notice that the running parameters are the same for all input videos (color or gray-level) as offered in Table 1.

Table 2 Quantitative results of applying the color-based LIAC algorithm to BEHAVE and CAVIAR test cases

	BEHAVE			CAVIAR		
	L	RGB	HLS	L	RGB	HLS
TP	5,623 (55.87 %)	9,996 (97.06 %)	7,812 (75.88 %)	232 (88.89 %)	224 (91.05 %)	199 (77.13 %)
FP	1,495 (14.85 %)	125 (1.21 %)	127 (1.23 %)	28 (10.62 %)	19 (7.72 %)	31 (12.02 %)
FN	2,946 (29.27 %)	177 (1.71 %)	2,356 (22.88 %)	1 (0.38 %)	3 (1.22 %)	28 (10.85 %)
Precision	0.790	0.988	0.984	0,892	0.922	0.865
Recall	0.656	0.983	0.768	0,996	0.987	0.877
F-score	0.717	0.985	0.863	0,941	0.953	0.871

Table 3 Quantitative results of LIAC and MoG in BEHAVE and CAVIAR test cases

	BEHAVE			CAVIAR		
	LIAC L	LIAC RGB	MoG	LIAC L	LIAC RGB	MoG
TP	5,623 (55.87 %)	9,996 (**97.06 %**)	8,093 (76.11 %)	232 (88.89 %)	224 (91.05 %)	229 (**94.63 %**)
FP	1,495 (14.85 %)	125 (1.21 %)	2,538 (23.86 %)	28 (10.62 %)	19 (7.72 %)	12 (4.96 %)
FN	2,946 (29.27 %)	177 (1.71 %)	2 (0.00 %)	1 (0.38 %)	3 (1.22 %)	1 (0.41 %)
Precision	0.790	**0.988**	0.761	0,892	0.922	**0.950**
Recall	0.656	0.983	**0.999**	**0,996**	0.987	0.995
F-score	0.717	**0.985**	0.864	0,941	0.953	**0.972**

3.6 Comparison with other approaches

In first place, based on the quantitative results shown in [26] and [15], we establish the following qualitative comparison. As you may observe, our approach in RGB color space offers a ratio of true positives around 97 %. In most cases, this result is better than the results provided in the before mentioned papers processing outdoor sequences. Moreover, taking into account the numerical results provided in [26], it seems clear that our approach throws excellent results.

Now, the validity of this proposal is assessed through a quantitative comparison with a state-of-the-art technique that has proved to offer good results, namely background/foreground segmentation based on a mixture of Gaussians (MoG) [27]. One important feature of this algorithm is that it selects the appropriate number of Gaussian distribution for each pixel. It provides good adaptability to varying scenes due illumination changes etc. As shown in Table 3, the LIAC RGB version outperforms the MoG one with a wide margin. In the case of the CAVIAR data set, both RGB and L versions of LIAC offer competitive results in comparison with the MoG-based approach. Moreover, when comparing the average F-score for our best performing LIAC alternative, 0.969, to the one from MoG, 0.918, some advantage of our approach is emphasized. Notice that values in bold show the best results obtained.

A qualitative comparison with other well-known approaches has also been included in order to highlight the performance of our proposal. There are two main reasons that do not enable presenting a strong quantitative comparison with other approaches. Firstly, the results provided within the different experiments do not show which criteria have been considered when selecting a hit from the overlapping area between the ground truth data and the detected region. And, secondly, as the processed sequences are also different from one paper to another, it is impossible to establish a fully objective comparison.

4 Conclusions

This paper is motivated by the results obtained during the last few years by LIAC for moving object segmentation in gray-level video sequences. Nowadays, color video is commonly used in many applications, such as visual surveillance. Therefore, we have studied the suitability of using the LIAC method in two widely used color spaces (RGB and HLS). Firstly, the LIAC algorithm for motion detection in color video has been introduced. Then, the performance of LIAC algorithm in both color spaces has been compared. For this purpose, some videos from the BEHAVE and CAVIAR test cases, as well as their respective reference data, have been tested for color spaces RGB and HLS. Also, the tests have been performed on the L component of the video images. It has been demonstrated (qualitatively and quantitatively) that the LIAC algorithm has an overall better behavior in RGB color space over HLS color space and gray level.

Acknowledgments This work is partially supported by Spanish Ministerio de Economía y Competitividad / FEDER under projects TIN2010-20845-C03-01 and TIN2013-47074-C2-1-R. The authors would like to acknowledge the publicly available data sets of the projects CAVIAR and BEHAVE.

References

1. Altunbasak, Y., Eren, P.E., Tekalp, A.M.: Region-based parametric motion segmentation using color information. Graph. Models Image Process. **60**(1), 13–23 (1998)
2. An, N.Y., Pun, C.M.: “Color image segmentation using adaptive color quantization and multiresolution texture characterization”. Signal Image Video Process. 1–12 (2012). doi:10.1007/s11760-012-0340-2
3. Arbelaez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **33**(5), 898–916 (2011)
4. Brox, T., Rousson, M., Deriche, R., Weickert, J.: Colour, texture, and motion in level set based segmentation and tracking. Image Vis. Comput. **28**(3), 376–390 (2010)
5. Chatterjee, S., Bhattacherjee, A.: Genetic algorithms for feature selection of image analysis-based quality monitoring model: an application to an iron mine. Eng. Appl. Artif. Intell. **24**(5), 786–795 (2011)
6. Chen, C.L., Tai, C.L.: Adaptive fuzzy color segmentation with neural network for road detections. Eng. Appl. Artif. Intell. **23**(3), 400–410 (2010)
7. Cucchiara, R., Grana, C., Piccardi, M., Prati, A.: Detecting moving objects, ghosts, and shadows in video streams. IEEE Trans. Pattern Anal. Mach. Intell. **25**(10), 1337–1342 (2003)
8. Davis, J.W., Keck, M.A.: A two-stage approach to person detection in thermal imagery. In: Proceedings of the IEEE Workshop on Applications of Computer Vision, vol. 1, pp. 364–369 (2005)
9. Delgado, A.E., López, M.T., Fernández-Caballero, A.: Real-time motion detection by lateral inhibition in accumulative computation. Eng. Appl. Artif. Intell. **23**(1), 129–139 (2010)
10. Felzenszwalb, P.F., Girshick, R.B., McAllester, D., Ramanan, D.: Object detection with discriminatively trained part-based models. IEEE Trans. Pattern Anal. Mach. Intell. **32**(9), 1627–1645 (2010)
11. Fernández-Caballero, A., Castillo, J.C., Rodríguez-Sánchez, J.M.: Human activity monitoring by local and global finite state machines. Expert Syst. Appl. **39**(8), 6982–6993 (2012)
12. Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E.: Spatio-temporal shape building from image sequences using lateral interaction in accumulative computation. Pattern Recognit. **36**(5), 1131–1142 (2003)
13. García-Lamont, F., Cervantes, J., López, A.: Recognition of Mexican banknotes via their color and texture features. Expert Syst. Appl. **39**(10), 9651–9660 (2012)
14. Gómez-Moreno, H., Maldonado-Bascón, S., Gil-Jiménez, P., Lafuente-Arroyo, S.: Goal evaluation of segmentation algorithms for traffic sign recognition. IEEE Trans. Intell. Transp. Syst. **99**, 1–14 (2010)
15. Haritaoglu, I., Harwood, D., Davis, L.S.: W4: real-time surveillance of people and their activities. IEEE Trans. Pattern Anal. Mach. Intell. **22**, 809–830 (2000)
16. Jodoin, P.M., Mignotte, M., Rosenberger, C.: Segmentation framework based on label field fusion. IEEE Trans. Image Process. **16**(10), 2535–2550 (2007)
17. Khan, A., Ullah, J., Jaffar, M.A., Choi, T.S.: Color image segmentation: a novel spatial fuzzy genetic algorithm. Signal Image Video Process. 1–11. (2012). doi:10.1007/s11760-012-0347-8
18. López, M.T., Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E.: Visual surveillance by dynamic visual attention method. Pattern Recognit. **39**(11), 2194–2211 (2006)
19. López, M.T., Fernández-Caballero, A., Fernández, M.A., Mira, J., Delgado, A.E.: Motion features to enhance scene segmentation in active visual attention. Pattern Recognit. Lett. **27**(5), 469–478 (2006)
20. López-Valles, J.M., Fernández, M.A., Fernández-Caballero, A.: Stereovision depth analysis by two-dimensional motion charge memories. Pattern Recognit. Lett. **28**(1), 20–30 (2007)
21. Martínez-Cantos, J., Carmona, E., Fernández-Caballero, A., López, M.T.: Parametric improvement of lateral interaction in accumulative computation in motion-based segmentation. Neurocomputing **71**(4–6), 776–786 (2008)
22. Mira, J., Delgado, A.E., Fernández-Caballero, A., Fernández, M.A.: Knowledge modelling for the motion detection task: the algorithmic lateral inhibition method. Expert Syst. Appl. **27**(2), 169–185 (2004)
23. Moreno-Noguer, F., Sanfeliu, A., Samaras, D.: Dependent multiple cue integration for robust tracking. IEEE Trans. Pattern Anal. Mach. Intell. **30**(4), 670–685 (2008)
24. Ross, M.G., Kaelbling, L.P.: Segmentation according to natural examples: learning static segmentation from motion segmentation. IEEE Trans. Pattern Anal. Mach. Intell. **31**(4), 661–676 (2009)
25. Tweed, D.S., Calway, A.D.: Integrated segmentation and depth ordering of motion layers in image sequences. Image Vis. Comput. **20**(9–10), 709–723 (2002)
26. Zhao, T., Nevatia, R.: Bayesian human segmentation in crowded situations. Comput. Vis. Pattern Recognit. **2**, 459–466 (2003)
27. Zivkovic, Z., Verbeek, J.J.: Efficient adaptive density estimation per image pixel for the task of background subtraction. Pattern Recognit. Lett. **27**(7), 773–780 (2006)