Available online at www.sciencedirect.com

SCIENCE DIRECT®

ELSEVIER

Real-Time Imaging 10 (2004) 127–143

www.elsevier.com/locate/rti

Real-time motion segmentation from moving cameras

Rita Cucchiara*, Andrea Prati, Roberto Vezzani

Dipartimento di Ingegneria dell'Informazione, University of Modena and Reggio Emilia, Via Vignolese, 905, 41100 Modena, Italy

Received 1 September 2003; received in revised form 20 February 2004; accepted 23 March 2004

Abstract

This paper describes our approach to real-time detection of camera motion and moving object segmentation in videos acquired from moving cameras. As far as we know, none of the proposals reported in the literature are able to meet real-time requirements. In this work, we present an approach based on a color segmentation followed by a region-merging on motion through Markov Random Fields (MRFs). The technique we propose is inspired to a work of Gelgon and Bouthemy (Pattern Recognition 33 (2000) 725–40), that has been modified to reduce computational cost in order to achieve a fast segmentation (about 10 frame per second). To this aim a modified region matching algorithm (namely Partitioned Region Matching) and an innovative arc-based MRF optimization algorithm with a suitable definition of the motion reliability are proposed. Results on both synthetic and real sequences are reported to confirm validity of our solution.
© 2004 Elsevier Ltd. All rights reserved.

1. Introduction

Moving object segmentation in videos is a key process in a plethora of multimedia and computer vision applications such as content-based retrieval, 3D scene reconstruction, video surveillance, and so on. Most of the difficulties associated to this task are due to the presence of a relative motion of the observer with respect to the motion of objects and background. When the background is stationary and videos are acquired by fixed cameras, moving object detection is a relatively "easy" problem: well-known methods such as background suppression and frame differencing were implemented with success in many applications. Even when background is moving with a known path, for instance when the camera's motion is constrained (for example in surveillance Pan-Tilt-Zoom cameras with a programmed camera path) modified methods of background suppression and frame differencing have been proposed [2]. Instead, the segmentation of moving objects becomes more critical when the video is acquired by a moving camera with an unconstrained and a priori unknown motion. More in general, we consider the framework of videos with moving foreground objects on a moving background. In this case, segmentation cannot be accomplished computing visual motion only, but other features must be exploited such as color, shape, texture, and so on. In addition, our aim is to develop a system able to compute the dominant motion in real time, since the final goal is to provide a fast segmentation that could be adopted in an on-line video segmentation process, suitable for applications such as indoor/outdoor surveillance, video-conferencing, multimedia on the fly transcoding, and augmented reality, where (hard) time constraints are mandatory.

Many works propose the integration of multiple features for detecting and tracking moving objects on video with a moving background. Among them, a previous work of Gelgon and Bouthemy [1] proposes an approach based on a color segmentation followed by a motion-based region merging. The authors adopt the affine motion model and the merging of regions is performed with Markov Random Fields (MRF, hereafter). An implicit tracking of the objects is also included, and it is obtained with the initialization of the MRF according with a label prediction. This approach is based on two basic assumptions on the objects: they must have different colors (the segmentation of the scene in objects is based only on color

*Corresponding author.
E-mail addresses: cucchiara.rita@unimore.it, rita.cucchiara@unimo.it (R. Cucchiara), prati.andrea@unimore.it (A. Prati), vezzani.roberto@unimore.it (R. Vezzani).

1077-2014/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.rti.2004.03.002

information) and rigid motion (expressible with affine equations). This solution produces good results, but the computational complexity is too high to allow real-time implementation: authors in [1] state to be able to process a frame every 80 s.[1]

In this paper we present an approach inspired by the one of [1] substantially modified in order to abate the computational time. To this aim our algorithm assumes some simplifications with respect to [1] but also some important new features. The basic simplification is the hypothesis of a translational motion model instead of an affine one: this hypothesis limits the applicability to the (very frequent) cases of pure translational movements or that can be approximated as translational between two consecutive frames.

However we introduce some important novelties:

- the "Partitioned Region Matching": we propose a new motion estimation algorithm based on region matching, but emphasizing advantages of both region matching and block matching;
- a measure of "motion reliability": we use in the MRF model a factor which takes into account the reliability of the motion estimation phase;
- a new minimization algorithm of the MRF function that approximates the classical approach with a search based on arcs instead that on nodes. This method abates the computational costs, by preserving most of the efficacy of the algorithm. In this way we have defined a technique for very fast segmentation reaching up to 10 frames per second (fps)[2] in frames of 100×100 pixels.

In the paper we present the approach and we discuss the results and the computational time performance. The next section contains a taxonomy of related works available in the literature, with a schematic representation of the differences. In Section 3 our algorithm is presented and discussed. Section 4 reports experimental results on several videos, with performance analysis with respect to ground-truths. In Section 5 the new MRF optimization algorithm is described and in Section 6 the related performance analysis is shown. Eventually, Section 7 provides concluding remarks.

2. Related work

Several researchers have approached the problem of motion detection on video acquired by moving camera. Table 1 tries to summarize the most relevant contributions to this field reported in the literature. We tried to classify them by means of several features such as whether they explicitly compute camera motion or not, which motion model (translational, affine, quadratic, etc.) is assumed, which features are used to segment the video (visual features as color, or motion, or both), and whether they achieve real-time performance or not.

In practice, we classify the approaches by means of eight factors (see Table 1):

(1) *Type of camera motion* (II[3]): The first assumption copes with the camera's motion model, namely:

- *fixed* camera (camera not in motion);
- *constrained moving* camera (e.g., pan-tilt camera, camera with a fixed path);
- *unconstrained moving* camera (when the motion of the camera is a priori unknown).

In the paper we analyze only approaches with a relative motion between the camera (i.e. the observer) and the background, and in particular videos acquired with unconstrained moving camera or with an unconstrained moving background.

(2) *Acquisition system* (III): It refers to the number and the type of cameras involved in the acquisition process and the model of the scene.

- 2*D* system (single camera);
- 3*D* system with *stereoscopic* vision (two normal cameras);
- 3*D* system (e.g., with a normal camera and a range camera).

Our proposal assumes a 2D system with a single moving camera.

(3) *Computation of camera motion* (IV and V): The camera motion can be estimated through the evaluation of the dominant motion with different techniques and models. Some approaches (labeled with "yes" in column IV) exploit camera motion computation to produce compensated videos from original ones in order to apply algorithms developed for fixed camera. Other approaches do not distinguish camera motion from the motions of foreground objects.

(4) *Motion of objects* (VI and VII): A very relevant difference among proposals is the adoption of either a *motion detection* or a *motion estimation* approach. In the former, each pixel/region is simply classified as "fixed" or "in motion", while with motion estimation a quantitative measure of motion is also provided.

(5) *Motion model* (V and VII): To describe the motion of a pixel in an image, the two components (along axis *x* and *y*) of the shift vector are enough. Instead, the motion of a region is more complex and could require more parameters to describe it. In this case, it is possible to adopt different models to represent

[1]On a ULTRASPARC Machine.

[2]On a dual Pentium III 1000 MHz with 512 MB of RAM.

[3]Roman numbers between brackets are referred to the columns of the Table 1.

Table 1
A review of related works

Ref.	Camera motion[a]	Capture system	Egomotion		Objects motion		Segmentation[b]	Tracking	Real time[c]	Notes
			Comput.	Model[d]	D/E[e]	Model[d]				
[21]	U	3D stereo	No		E: n/a	3D	Mix: Disparity and motion	No	n/a	
[23]	U	2D estimation	No		E: iterative	A	VF: Color/gray level	No	No	
[3]	U	2D	Yes	A	D: FD		M	Yes	SPHS	
[13]	U	2D	No		E: dominant components retrieval	A	M	No	No	Work with transp.
[4]	U	2D	Yes	A	D: FD		Mix	No	No	
[9]	C (only pan-tilt)	2D	No		D: BGS		M	Yes	Yes	
[14]	U	2D	No		E: OF	P or A	M: simultaneously with E	No	No	
[29]	U	2D	Yes	Quadratic	E: RMR	A	M	No	No	
[5]	U	2D	Yes	n/a	D: FD		M	Yes	Yes	For periodic motion
[30]	U	2D	Yes	n/a	E: OF	A	VF: luminance	No	No	
[24]	U	2D	Yes	A	D: FD		VF: Color	No	No	
[1]	U	2D	No		E: n/a	A	VF: Color	Yes	No	
[15]	U	2D	No		E: region matching	T	M	Yes	No	User-guided
[16]	U	2D	No		E: OF	A	M	No	No	
[6]	C (only pan-tilt)	2D	Yes	T	E: temporal derivative	n/a	M	With only 1 moving object	Yes	Egom. with spiral scanning
[22]	U	2D	No		E: feature tracking	A	Mix: some image cues	No	No	
[31]	F	2D	No		D: FD		M	No	SPHS	
[27]	U	2D	No		E: n/a	A	Presegmentation required	No	No	
[17]	U	3D stereo	No		E: OF	n/a	Mix: Edge, depth and optical flow	Yes	SPHS	
[7]	U	2D	Yes	A	D: FD		M: through multiscale MRF	No	No	
[8]	U	2D	Yes	A	D: FD		M: through multiscale MRF	No	No	
[10]	U	2D	Yes	A & 3D	D: FD and BGS		M	No	No	Background mosaic
[26]	U	2D	No		E: edge tracking		VF: Edges	Yes	No	
[18]	U	2D	No		E: corner based	A	Mix: clustering of edges	Yes	Yes	
[28]	U	2D	No		E: edge tracking	A	User guided initialization	Yes	Yes	
[25]	U	2D	No		E: BC	T	VF: gray level	No	No	
[19]	U	3D with stereo	No		E: OF	A	M	Yes	SPHS	
[20]	U	2D	No		E: OF	A	M	No	No	
*Our prop.	U	2D	No		E: partitioned region matching	T	VF: color	No	Yes	

[a] U: unconstrained, C: constrained, F: fixed.
[b] Feature of the first and basic segmentation; VF: visual features, M: motion, Mix: mixed.
[c] SPHS: real time on a special purpose hardware system.
[d] E: estimation, D: detection; OF: optical flow, FD: frame difference, BGS: background suppression, BC: block correlation.
[e] Only if motion estimation is adopted; A: affine, T: translational, 3D: six 3D-parameters, P: perspective.

the motion of the regions: *translational*, *affine*, *quadratic*, etc. The more the parameters, the higher the quality of the estimation is, but, at the same time, the more complex and computationally expensive the algorithm becomes.

(6) *Segmentation features* (VIII): The extraction of moving objects from the background is performed through image segmentation exploiting features. Thus, approaches are characterized by the feature used as:

- segmentation based on *visual features* (e.g. color, edge, texture),
- segmentation based on *motion*,
- *mixed* segmentation (motion and visual features together).

In the first case, the objects are separated independently from their motion; this typically brings to an over-segmentation. For example, a segmentation based only on color information can separate a person with shirt and pants of different colors into several objects. In this case the use of another feature (i.e., motion) helps in merging different regions of the same object. Another possible approach is to compute the segmentation by considering only the motion information, but in this case the aperture problem can be critical, as in the case of non-textured moving objects. The most reliable solution to this problem is, typically, to implement a segmentation that takes into account visual features and motion at the same time in a mixed segmentation.

(7) *Tracking* (IX): Not all the approaches include a *tracking phase* that allows the system to trace objects' positions over time. In some cases an implicit tracking is integrated in the motion segmentation process (e.g., in the system proposed in [1] the tracking is performed through the initialization of the labels based on prediction: objects that confirm the prediction, maintain the same label over time).

(8) *Real time implementation* (X): As stated before, our aim is to meet *real-time* requirements.[4] For this reason, we are interested in evaluating whether an algorithm works in real time or not. In Table 1 we reported only if the time requirements are considered and if the real-time constraints are satisfied or not (as stated by the respective authors).

In conclusion, we can group the proposals into three classes:

(a) *based on camera motion computation*: these methods compute camera motion and, after its compensation, they apply an algorithm defined for fixed-camera;
(b) *based on motion segmentation*: the objects are mainly segmented by using the motion vector computed at pixel level;
(c) *based on region merging with motion*: the objects are obtained with a segmentation based on visual features, and next merged on motion parameters computed on a region-level.

Among approaches based on the computation of the camera motion followed by the application of a fixed-camera technique, some of them use frame differencing [3–8] and some other uses background suppression with background obtained through mosaicing techniques [9,10]. Refer also to [11] for background mosaicing construction and to [12] for a comparison of approaches to camera motion computation.

System described in [13–20] belong to class (b). The algorithm used for segmentation is the characterizing factor for these works. Also the proposals reported in [21] and [22] could be included into this class; unlike the others, they use motion and some visual features simultaneously in the segmentation process.

The approaches described in [23–25] belongs to the third class, that use color as visual feature for initial segmentation, [1] that assumes a possible exploitation of gray level, texture or color, [26] that makes use of edges. In [27] and [28], the authors present only the merge phase and suppose the existence of a previous region segmentation step. Our proposal belongs to this class.

Finally, [29] and [30] are two mixed approaches: [29] performs an initial camera motion computation to remove camera motion similarly to the approaches of class (a) and obtains the region in motion by frame differencing. Then, to separate single objects, it exploits a motion-based segmentation only with a "split and merge" algorithm on the regions extracted by frame differencing. [30] computes camera motion for compensation and then uses an approach similar to [1], composed by a color segmentation process followed by motion based region merging.

Our proposal (the one labeled with "$*$" in Table 1) follows the same guidelines of [1] aiming to a real time implementation with general-purpose platforms. By the analysis of the Table 1, we could note that few proposals take into account the time requirements; among these, many use a special purpose hardware [3,31,17,19], or a manual initialization [28] or are oriented to constrained motion model [9,32] or a specific application (as [5] for periodic model). [6] and [18] are general purpose systems, but they do not assure the extraction of the entire moving objects, because they make use only of the motion information in the segmentation process.

3. Description of the system

Our approach can be summarized as follows: each frame of the sequence is segmented into regions by using *color*, with the assumption that each region contains only one (or part of one) object. For this task a *region growing* algorithm is applied. According with color segmentation, a *Spatial Region Graph* (*SRG*), also called adjacency graph, is created. In SRGs, nodes represent regions, whereas arcs represent topological adjacencies. We consider an attributed graph: area size, the coordinate of the bounding-box and the centroid's position are associated with each node. Then, *motion estimation* is computed with the adoption of the translational motion model. To accomplish this task there are basically two ways: the first requires *pixel-level motion estimation* (e.g., with optical flow or block matching) followed by a statistical function (e.g., mean or mode) over regions; the second (adopted in our system) directly exploits computation of the *region-level*

[4]We use the term *frame-rate* to say 25/30 fps (European/American acquisition standards) and the term *real time* as a more "soft" constraint that could be considered acceptable for many applications (for instance in surveillance systems 10 fps are often assumed as a real time).

motion vectors through region matching (and in particular with *Partitioned Region Matching*, described in the following). In addition to motion vector a measure of *motion reliability* is defined and computed for each region, based on the presence of "strong" gradients (that allows correct estimation of the motion) and a good match. Computed information are then stored in the spatial graph. Then SRG is the starting point of a Markov Random Field (MRF) framework, where regions are merged according with an energy function influenced by the motion parameters (velocity and reliability). The largest region is assumed as background and its motion as the motion of the camera, while other regions are classified as moving objects. Temporal continuity of motion field is supposed to perform a prediction of the next frame. This prediction is used to initialize the region-level MRF framework during the analysis of the next frame, with two advantages: the reduction of merging phase duration and the implicit tracking of the objects.

In Fig. 1 the complete algorithm is schematically reported; on the left are presented the functional blocks and on the right the partial outputs obtained with a synthetic sequence are shown. Fig. 1b and c are the color segmentation output and the spatial region graph, respectively. Fig. 1d and e describe the segmented regions and the graph after motion computation and MRF estimation, respectively. Finally Fig. 1f shows detected moving objects.

As above-mentioned, our proposal has been initially inspired by the work of [1], and initial description reported in [33]. According with that, we mix color feature and motion and exploit MRF to optimize the spatial region graph in order to extract objects as regions with similar motion parameters. Nevertheless, many substantial differences have been introduced in all sub-tasks and in particular in:

- *Color segmentation*: In accordance with [1] we believe that color is the most salient feature to be exploited. In [1] color segmentation is performed at the first frame only and then refined frame by frame by means of the MRF. Instead, focusing on speed, MRF optimization has been substituted by a color (re-)segmentation at each frame. In particular in our system we apply a region growing algorithm to

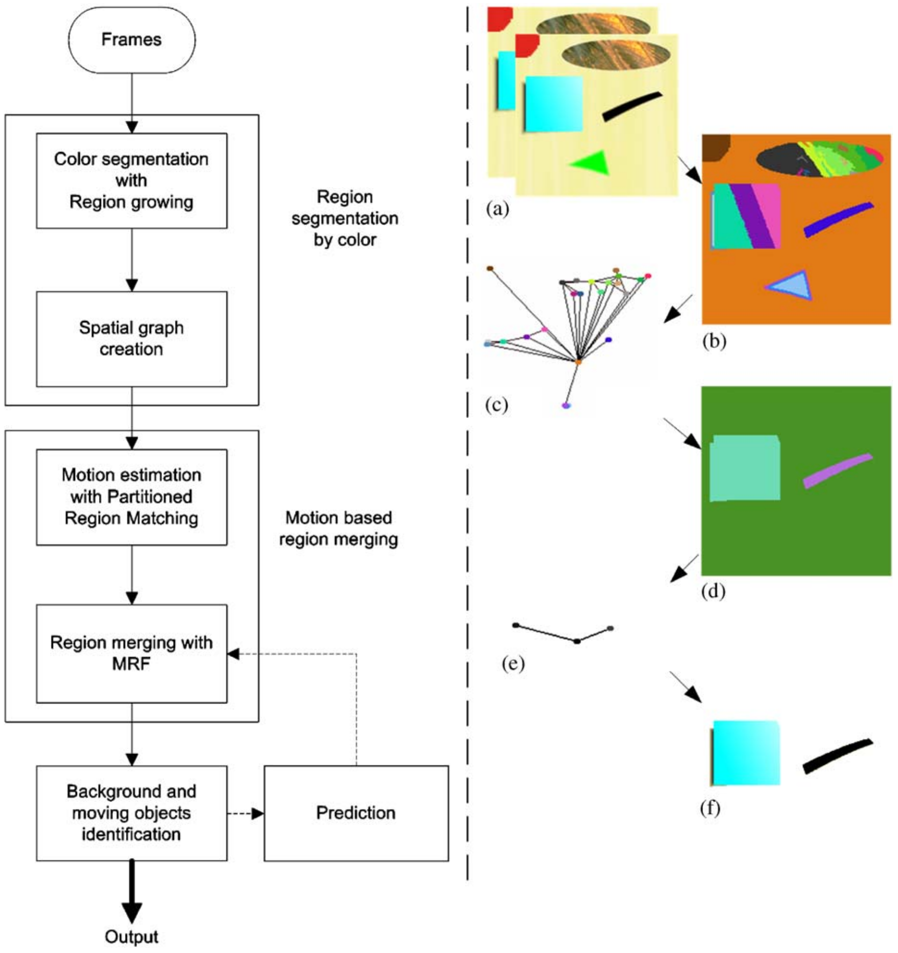

Frames
Color segmentation with Region growing
Spatial graph creation
Region segmentation by color
Motion estimation with Partitioned Region Matching
Region merging with MRF
Motion based region merging
Background and moving objects identification
Prediction
Output
(a)
(b)
(c)
(d)
(e)
(f)

Fig. 1. The block diagram of the proposal.

every frame. This last solution has proved to be less accurate but much faster. Indeed, the construction of the pixel level MRF prediction has been tested, as defined in [1], and has resulted useless if color segmentation on every frames is adopted; thus, the pixel level prediction and the optimization of the segmentation with MRF are not implemented in our version.

- *Motion model and estimation*: We have adopted the translational motion model, less realistic than the affine one, but characterized by a much lower complexity. However, when camera and objects are moving slowly the simplification results acceptable. Correlation-based algorithms (such as block matching) could be sufficient for motion estimation; in particular, taking advantage from the presence of a region segmentation, we propose an original region matching method, the Partitioned Region Matching (PRM) (see Section 3.2). Lastly, we define a motion reliability term exploited in MRF.
- *MRF optimization*: The energy function used is changed due to the different motion model adopted. In addition, Section 5 describes a new optimized algorithm that allows a real time processing.

3.1. Color segmentation

Color segmentation is implemented with a region growing algorithm. Since our goal is to detect moving objects, we work principally on an object-level and a perfect color segmentation at pixel level is not required. Thus, we defined a simple iterative approach inspired to [34], with suitable modifications. At each iteration an unlabeled point (or seed) is extracted from the image: a new label is assigned in order to create a region with a single pixel. Then, the 4-connected neighbor points are evaluated. A point x is merged to the current region S if it is unlabeled and if it has a color close enough to the mean color of the region. To check if the sum of the absolute difference between xcolor components of the point x and the correspondent mean values evaluated on the current region is lower than a fixed threshold α, we use the following equation:

$$dist(x,S) = |x_R - \overline{R}_S| + |x_G - \overline{G}_S| + |x_B - \overline{B}_S| \leqslant \alpha, \qquad (1)$$

where x_R, x_G, x_B are the three RGB color components of the evaluated point x, $\overline{R}_S$, $\overline{G}_S$, $\overline{B}_S$ the mean values of RGB color components computed over the region S and α a fixed threshold.

The dependency on initial seeds and scanning order is a possible limitation of this implementation. Only for computational reasons, seeds are not extracted with a particular algorithm that could extrapolate significant points (e.g., points of local minimum/maximum of the intensity) but the first unlabeled pixel starting from the top-left corner of the image is selected.

To reduce segmentation dependency on the visiting order of the neighbor pixels we have adopted a propagative search with the following strategy. First, the pixels adjacent to the seed are pushed into a stack A. Then, we pop one-by-one pixels from stack A and insert their neighbors into a stack B. We evaluate pixels from stack B and push new pixels in stack A. In this way the regions grow on all directions with homogeneity.

The presence of strong edges or noise implies the creation of little regions, some of those with only one pixel. To overcome this problem we have introduced a second threshold β: regions with a dimension lower than β are merged with an adjacent one. Thresholds α and β are less critical than expected, but depend on the tradeoff between precision and speed. If α or β are "low" more regions are created. Nonetheless, the following MRF phase will solve a possible initial over-segmentation.

3.2. Motion estimation with partitioned region matching (PRM)

The aim of the motion estimation phase is the computation of a motion vector for each region produced with color segmentation. Several variants of block matching are available in the literature (e.g. [35,36]) to evaluate the motion of a block with a translational model; moreover, a statistical function is needed to integrate the blocks motion measures over the whole region. When a previous color segmentation is available, as in this case, region-matching techniques produce often better results than block matching in a relative short time. To this aim we present here a variation of the classic region matching.

Differently from the block matching algorithms, that confront fixed sized blocks, region matching exploits the whole regions extracted with segmentation as comparing patterns and a function defined over the entire regions as matching value (i.e. distortion rate). For example, we could adopt the measure expressed in Eq. (2), that is an adapted version of the classical sum of absolute differences (SAD) over a region:

$$SAD_R(\mathbf{v}) = \sum_{(x,y)\in R} \left(\sum_{i\in\{R,G,B\}} |I_t(x,y)_i - I_{t+1}(x+v_x, y+v_y)_i| \right), \qquad (2)$$

where $I_t(x,y)_i$ is the ith color component of the point of the image I at the (x,y) coordinates at the frame t and $\mathbf{v} = (v_x, v_y)$ is the displacement to evaluate.

The most important advantage of region matching with respect to block matching is the possibility to estimate motion of uniform regions without texture, by using the shape. In fact, the presence of at least two not-parallel gradients is required to correctly compute a motion vector (also known as the "aperture problem", [37]). The adoption of regions in substitution of blocks makes more probable that the previous requirement will be satisfied, both for the bigger size and for the presence of the object border inside the region. At the same time, problems with region matching based on Eq. (2) can arise when the motion is not strictly translational[5] or when the shape varies over time (for example in presence of occlusions). In this case, the variation of the occluded object's shape can introduce errors. To reduce this problem, we define a new matching criterion that we call *Partitioned Region Matching* (PRM in the following), that ensembles features of both block and region matching.

PRM is defined as follows:

(1) a region R is partitioned in a number n of disjoint sub-regions SR^k, so that

$$R = \bigcup_{k=1}^{n} SR^k \quad \text{and} \quad SR^i \cap SR^j = \emptyset | i, \ j \in 1 \dots n. \quad (3)$$

(2) The integer, finite set $\mathbf{V}$ of shifts in all directions is defined as follows:

$$\mathbf{V} = \{\mathbf{V}_j = (v_{xj}, v_{yj}) \,|\, v_{xj}, v_{yj} \in [-s, +s]\}, \quad (4)$$

where s is the maximum shift that the system can evaluate; please note that this integer set does not allow sub-pixel accuracy in motion estimation; once again, we prefer to limit the complexity of the system to be able to meet real-time constraints.

(3) For each sub-region SR^k the function $SAD_{SR^k}(\mathbf{v}_j)$ is evaluated, according with Eq. (2); the motion vector $\mathbf{v}^k \triangleq (v^k_x, v^k_y)$ of the sub-region SR^k is assumed equal to:

$$\mathbf{v}^k = \arg\min_{\mathbf{v}_j \in \mathbf{V}} \{SAD_{SR^k}(\mathbf{v}_j)\}. \quad (5)$$

(4) A probability function $P(\mathbf{v})$ is associated to the region and it is computed as the product of the a posteriori probability (i.e., number of occurrences of each motion vector) with the a priori probability.

$$P(\mathbf{v}) = \frac{\left(\sum_{k=1}^{n} \delta(\mathbf{v}, \mathbf{v}^k)/n\right) \cdot P_{priori}(\mathbf{v})}{z}, \quad (6)$$

where

$$\delta(\mathbf{v}, \mathbf{v}^k) = \begin{cases} 1 & \text{if } \mathbf{v} = \mathbf{v}^k \\ 0 & \text{if } \mathbf{v} \neq \mathbf{v}^k \end{cases} \quad (7)$$

and z is the normalization factor. The a priori probability could be computed taking into account the motion in the previous frame, physical constraints, and knowledge on the scene or application. To reduce computational complexity, in the experiments reported in this paper the a priori probability has been set identically to $\frac{1}{|\mathbf{V}|}$. In this case, $P(v)$ is computed simply as

$$P(\mathbf{v}) = \frac{\sum_{k=1}^{n} \delta(\mathbf{v}, \mathbf{v}^k)}{n}. \quad (8)$$

(5) Finally, the motion vector $\mathbf{MV}_R = (MVx_R, MVy_R)$ of the whole region R is assumed to be the vector $\mathbf{v}_j$ that maximizes $P(\mathbf{v})$:

$$\mathbf{MV}_R = \arg\max_{\mathbf{v}_j \in \mathbf{V}} \{P(\mathbf{v}_j)\}. \quad (9)$$

Using Eq. (8) to compute the probability values, the motion vector $\mathbf{MV}_R$ results to be the mode of the shift vectors $\mathbf{v}^k$.

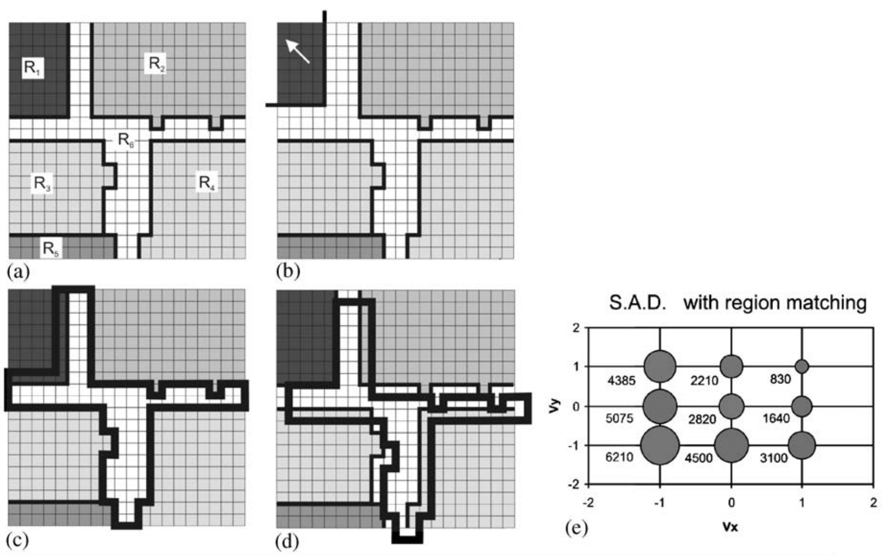
However, we must define how to compute the sub-regions as in (1). The region partitioning should present some characteristics: (a) the sub-regions SR^k should not be too small, (e.g. 8×8, 16×16,), otherwise PRM turns into block matching; (b) the sub-regions SR^k should have almost the same area; in this manner all the SAD_{SR} are comparable; otherwise, a suitable factor coping with different area should be included in the a posteriori probability; (c) the edges border should be fragmented into more than one part, to reduce the influence of occlusions.

The region partitioning adopted in this work, compatible with the previous criteria, is the following: let BB_R be the bounding-box (or extent) of the region R and define a partitioning window W_L of $L \times L$ pixels. For instance, L could be 8,16, so that $Area(W_L) \ll Area(BB_R)$. Thus, BB_R is decomposed in windowed bounding-blocks WBB_i, i.e. sub-squares LxL as large as the partitioning window is $\left(BB_R = \bigcup_i WBB_i\right)$. An example is in Fig. 2g. The sub-regions SR_k are the not null intersections between the windowed bounding blocks and the region (see Fig. 2h): $R = \bigcup_{k=1}^{n} SR_k$, $SR_k = WBB_k \cap R$, $SR_k \neq \emptyset$.

In this manner, the criterion (a) and (c) are always satisfied, while the (b) criterium is not, because some border sub-regions could be smaller than central ones. In spite of this, the central sub-regions give the same contribution in the statistical function of the smaller border sub-regions. This means to assume that the border motion is more significant than the central one.

The defined PRM reduces the problems of occlusion typical of region matching, since occlusions affect only a limited number of sub-regions that are not significant due to Eqs. (6), (8), and (9); moreover it resolves the drawback of motion vector locality of block matching, working on larger patterns.

[5]If the affine model is assumed, the Eq. (2) can be modified as follows: $SAD'_R(\mathbf{v}) = \sum_{(x,y)\in R}\left(\sum_{i\in\{R,G,B\}} |I(x,y)_i - I(a_1(x - x^R_G) + a_2(y - y^R_G) + a_3, a_4(x - x^R_G) + a_5(y - y^R_G) + a_6)_i|\right)$, where (x^R_G, y^R_G) are the centroid coordinates of the region R and $a_1 \dots a_6$ are the six affine motion parameters to evaluate. The introduction of six parameters instead of two makes more complex and heavy the motion estimation.



Vx	Vy	Region:	R_1	R_2	R_3	R_4	R_5	R_6	SAD_R
		Gray level:	20	200	230	240	50	255	
		$d(R_x,R_6)$:	235	55	25	15	205	0	
-1	-1		22	12	7	0	1	58	6210
-1	0		19	0	8	0	2	71	5075
-1	1		14	0	15	7	3	61	4385
0	-1		16	13	1	0	0	70	4500
0	**0**		**12**	**0**	**0**	**0**	**0**	**88**	**2820**
0	1		7	0	9	9	1	74	2210
1	-1		8	20	0	8	0	64	3100
1	0		4	10	0	10	0	76	1640
1	**1**		**0**	**7**	**7**	**18**	**0**	**68**	**830**

Composition of the match

(f)

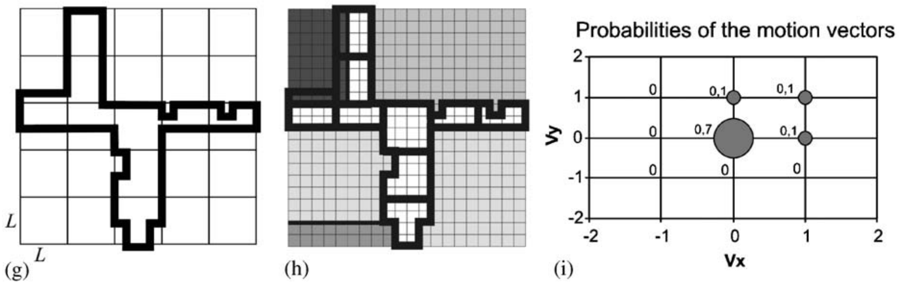


Fig. 2. Example of partitioned region matching (PRM). (a,b) two consecutive frames where R_1 is the only moving object; (c,d) two possible matches of R_6, obtained with $(v_x, v_y) = (0, 0)$ and $(v_x, v_y) = (1, 1)$ respectively; (e,f) SAD_R values with Eq. (2), the columns R_i in (f) are the number of pixel matching $R_6 - R_i$ that must weighted with the correspondent distance d in color (e.g., first row: $6210 = 22 \times 235 + 12 \times 55 + 7 \times 25 + 1 \times 205$); (g) parts creation for PRM; (h) best match with PRM, (i) probabilities of the motion vectors.

To better understand and appreciate the power of this solution we can consider the two consecutive frames in Fig. 2(a) and (b); we suppose that only R_1 is in motion, while the other regions are fixed. To compute motion of R_6 (i.e., the background), we search the best match between the second and the first frame. In Fig. 2(c) and (d) are represented two possible matches, obtained with $(v_x, v_y) = (0, 0)$ and $(v_x, v_y) = (1, 1)$, respectively. Obviously, the first is the correct match; Fig. 2(f) shows that is also the case with the maximum number of matching pixels (the pattern extracted from the second frame covers 88 pixel of R_6 and 12 of R_1 in the first frame), but the SAD_R, if computed over the whole region, is smaller in the second case (see Fig. 2(e)), where the pixel of the region R_6 covered are only 68. Therefore, a classical region matching based on SAD_R fails. This problem often occurs in real sequences, when the foreground moving objects have very different colors from the background. In this case the background's motion may be wrongly evaluated as equal to the motion of a foreground occluded object. With PRM, instead, we can estimate the correct motion vector. The partitioning of R_6 is made as Fig. 2(g): for graphical reasons the size of each window is 4×4 pixels, but in the implemented system partitioning windows are larger (e.g., $L = 8, 16, \ldots$). The entire region results segmented in ten sub-regions. In Fig. 2(h) we can see that only three of ten sub-regions of the R_6 are affected by the occlusion of R_1, while most of sub-regions can correctly estimate the motion vector. Fig. 2(i) shows the probabilities $P(v_x, v_y)$ associated to the motion vectors (v_x, v_y): the vector with the maximum value is the right: $(v_x, v_y) = (0, 0)$.

For each region R we also define a *motion reliability* as:

$$\rho_R = \frac{\sum_{(x,y)\in R}\left[\sum_{i\in\{R,G,B\}}\left(|\partial I(x,y)_i/\partial x| + |\partial I(x,y)_i/\partial y|\right)\right]}{SAD_R(\mathbf{MV}_R) + 1}. \tag{10}$$

ρ_R takes into account two factors: the gradient in the color image and the goodness of the best match found, evaluated as the SAD (obtained with a shift of $\mathbf{MV}$) increased by 1 to prevent division by 0. If a region presents a low texture, the choice of the best match could be ambiguous, therefore the reliability of the motion estimation is low too. Reliability is low also if the best match presents an high distortion and thus an high SAD (e.g., in presence of occlusions and deformations).

Moreover, the motion reliability value resolves another problem: region matching (as all the other motion estimation techniques) cannot evaluate the motion in the image border. If a region has a bounding-box entirely contained in the image border (large as s of Eq. (4)), we set ρ_R to 0.

3.3. Region level MRF

The Markov Random Fields result very powerful to resolve segmentations and classifications. In this kind of problem, the undirected graph is the typical representation of the MRFs, with nodes corresponding to a set of variables and the arcs reporting their interactions. The nodes contain both variables based on the observations of the system modeled (e.g., color, motion, etc...) and the output values (e.g., the labels). Moreover, a function of these variables (i.e., the energy function), is defined over the whole graph. The main property of the MRFs is the independency between the variables of nodes not connected with an arc; consequently, the energy function could be decomposed as the sum of local terms defined over the *two-site cliques*[6] (i.e., couple of nodes connected with an arc). Goal of a MRF framework is the minimization of the energy function and, to this aim, an optimization algorithm should be introduced.

We use a Markov Random Field framework to merge regions with similar motion. As proposed in [1], the energy function is composed by three terms: one based on motion (U_1), one used to perform a geometrical regularization (U_2), and one based on the number of labels assigned (U_3):

$$U(e, o) = U_1(e, o) + U_2(e) + U_3(e), \tag{11}$$

where e and o are the labels and the observation fields respectively.

Differently from [1], we exploit motion reliability too, including it in the energy term based on motion:

$$U_1(e, o) = \sum_{(s,t)\in\Gamma} V_1(e_s, o_s, e_t, o_t), \tag{12}$$

$$V_1(e_s, o_s, e_t, o_t) = \begin{cases} 0 & e_s \neq e_t, \\ c_1 \cdot \sqrt{(MVx_s - MVx_t)^2 + (MVy_s - MVy_t)^2} \cdot \sqrt{\rho_s \cdot \rho_t} & e_s = e_t, \end{cases} \tag{13}$$

where Γ is the set of two-dimensional cliques and s and t are two sites of a clique (corresponding to regions); MVx and MVy are the two components of the motion vector, ρ is the motion reliability and c_1 is a positive constant. The motion distance is computed with Euclidean formula, while the reliability term is included to make motion-based energy term U_1 more significant than the geometrical term in presence of high values of reliability.

In addition, we can decompose the geometrical regularization term into the sum of local terms defined over the cliques as in [1]:

$$U_2(e) = \sum_{(s,t)\in\Gamma} V_2(e_s, e_t), \tag{14}$$

$$V_2(e_s, e_t) = \begin{cases} 0 & e_s \neq e_t, \\ -c_2 \dfrac{\xi_{s,t}}{\xi_{s,t} + \sqrt{(G_x^s - G_x^t)^2 + (G_y^s - G_y^t)^2}} & e_s = e_t, \end{cases} \tag{15}$$

where c_2 is a positive constant, ξ is the length of the shared border and $\mathbf{G} = (G_x, G_y)$ is the region centroid. This term enhances the fusion of two adjacent regions with a long shared border and a small distance between centroids.

The last term takes into account the number of labels assigned, i.e. the cardinality of e ($\#e$):

$$U_3 = c_3 \cdot \#e, \tag{16}$$

where c_3 is a positive constant.

[6] Hereinafter, we will refer to a two-sites clique simply with the term "clique".

In conclusion, the motion based term tries to keep separate regions with different motion, geometrical term decrements the weight of motion distance if there is a strong adjacency and U_3 represents a sort of motion difference threshold under which two regions will be merged.

The optimization of the energy function is performed with a multi-scale iterative algorithm. A label and a binary stability flag (preset to "unstable") are associated to each node; initial label is assigned according with a prediction computed on the previous frame. Known the labels of regions in the previous frames we predict the label of each node by computing the maximum overlapped area and imposing motion continuity. The algorithm extracts one by one in random order the regions with "unstable" flag and evaluates which label (taken from a set composed by the old label, the labels of the adjacent nodes and an outlier label to allow separation of connected regions) minimizes the energy function. The new label is assigned to the node, the stability flag is set to "stable" and, if the label has been changed, the state of the adjacent nodes is set to "unstable". When all the nodes become stable, a compression of the graph is performed by grouping nodes with the same label into a single label; then the algorithm is iteratively exploited on the new graph.

3.4. *Background and moving object identification*

After MRF optimization, we should obtain a final graph with each node corresponding to an object (or a blob containing objects with the same motion). The largest one is assumed to be the background. Nevertheless, assuming that all the other regions are moving objects could be incorrect, since the background could be separated in more parts by interposed objects. To prevent this, every region with the same motion of the background is merged with it, even if not adjacent. To be less sensitive to noise, we can do the same with regions that exhibit similar motion (i.e., the Euclidean distance in the velocity space must be under a suitable threshold). Remaining regions are assumed to be moving objects. If small regions are not relevant for the application, a size threshold can be introduced.

4. Experimental results

The presented system has been implemented and tested on both synthetic and real sequences.[7] *Ground-truth tests* (i.e. tests with manually segmented frames as ground-truth) have been performed to evaluate the efficacy. Finally, a time analysis has been carried out to study dependencies on parameters and the capability to satisfy real-time requirements. In the following subsections we will analyze the results obtained on different sequences.

4.1. *"Object sequence"*

"Object" sequence (Fig. 3) is synthetic video composed of 20 frames of 200×200 pixel each. It has been made properly to test the system and contains five objects in strictly translational motion over a mobile background. In particular, there are:

- a red circle O_1, with homogeneous color but with a vanished border. The object is entering into the scene and its shape and size are varying on the time; this tests the reliability of the region matching in presence of objects that are entering/exiting from the scene;
- an ellipse O_2 with a very strong texture: it represents an object that color segmentation cannot entirely extract (see Fig. 3) and only the motion field can correctly reconstruct;
- a blue square O_3, with vanished color: darker in the bottom left with a shadow;
- a black bar O_4, with uniform color: is an ideal object for this application;
- a green triangle O_5, with a transparency;
- the background: is yellow, with a weak texture, just sufficient to compute its motion.

The output of the system confirms correctness of the approach: all the moving objects are correctly retrieved in the whole sequence.

The output was also subjected to a ground-truth analysis: see Fig. 4 with false positives and false negatives at pixel-level. As it can be seen from Table 2 the objects with shadows or vanishing borders have some pixels that are not correctly segmented, but are assigned to the background; this causes the false negatives of Fig. 4. The false positives at frame 10 correspond to a new region that the circle leaves in the top-left corner of the image while is still entering (the motion of a new region is unknown and it is assumed to be an indefinite value).

The analysis of the centroids' coordinates is more significant for our application, because it consents to evaluate better the efficacy of the motion estimation and the region merging phases (rather than the color segmentation); Table 2 shows that the estimation error is null for the bar and irrelevant for the other objects.

4.2. *"Indoor sequence"*

Fig. 5 shows an original frame of an indoor video, the color segmentation, and the final output. The camera

[7]The videos are available at the Imagelab Laboratory page http://imagelab.ing.unimo.it.

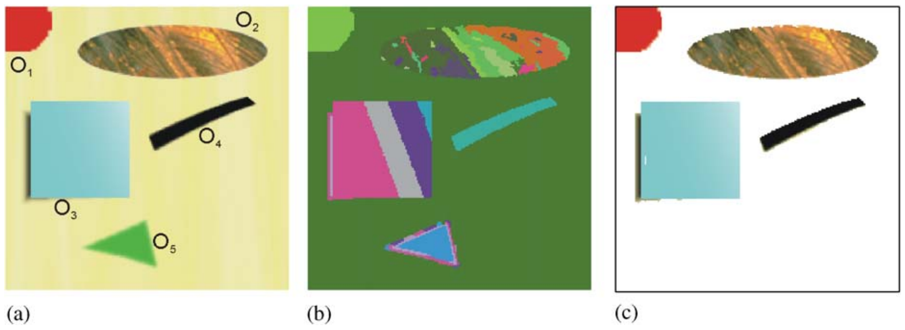


Fig. 3. Frame 6 of object sequence: (a) original frame, (b) after color segmentation with region growing and (c) moving objects reported on output (in this frame O_5 is not in motion).

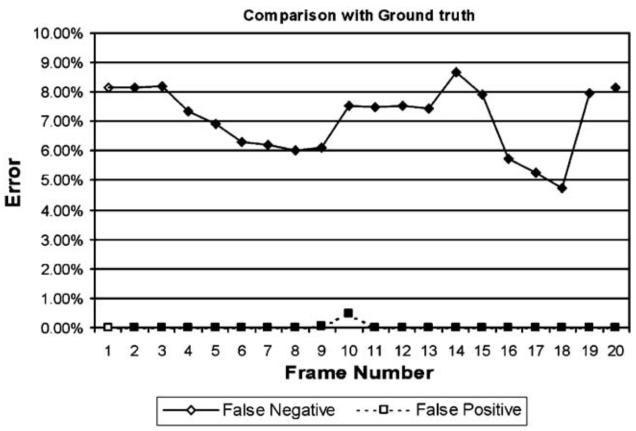


Fig. 4. Errors expressed in percentage of the total (in pixels) over object sequence. The values refer to objects in motion (different from the background). Thus, false negatives are moving objects' points that are not detected and false positives are points of still objects or of the background labeled as belonging to moving objects.

Table 2
Size and centroid coordinates estimation errors on object sequence

Object	Size mean error	Centroid's coordinates mean error (in pixel)
Circle	n/a (2.3%)	0.24
Square	−352 pixels (6.5%)	0.27
Ellipse	−185 pixels (4.5%)	0.12
Bar	−2 pixels (0.3%)	0
Triangle	−135 pixels (12.6%)	0.32

The size mean error of the circle is carried out only in percentage because the circle is entering on the scene and its size is not fixed.

and a person are moving in opposite directions (see superimposed arrows).

A ground-truth analysis is performed (Fig. 6). The presence of false positives between frames 41 and 61 is caused by a wrong motion estimation. The motion of the background region at the right of the man is equal to the man's motion and the two regions are merged (see Fig. 5d). In this case, neither standard region matching, nor PRM can correctly compute the motion value; in fact, they cannot exploit the texture, because the region is homogeneous, nor the shape because the area presents half border affected by the occlusion of the man and the other half corresponding to the image border, always stationary. In the frames from 75 to 100 the person is stopped: the system works correctly, also in presence of the unavoidable camera noise. From frame 100 to 108 the person is not completely detected due to its initial very slow motion, and this is shown in the peak of false negatives reported in the right graph of Fig. 6.

4.3. "Hand sequence"

In hand sequence (see Fig. 7) a hand is moving over a desk and the camera is in motion too. This sequence is very hard to analyze, because the background is composed by a lot of small regions (of the mouse pad) with some large and fine-textured ones (of the desk) in addition. Motion estimation is difficult on both: on the small regions when the hand occludes them and on the white desk for the absence of gradients. Fig. 8 reports ground-truth analysis: in this case, the false positives are some details of mouse pad connected to the hand.

4.4. Computational performance analysis

The system proposed on [1] produces satisfactory results in terms of accuracy of segmentation and motion detection (as stated by the authors), but it is computationally too expensive to satisfy strict time requirements (as indicated in [1] and in [38], they needed 80 s for frame on a ULTRASPARC). Our initial idea was not to develop a more reliable implementation, but to meet real-time requirements and, and the same time, maintain acceptable results.

In this section we present a detailed time analysis that shows the cost of each phase and their possible dependencies on application parameters. In particular:

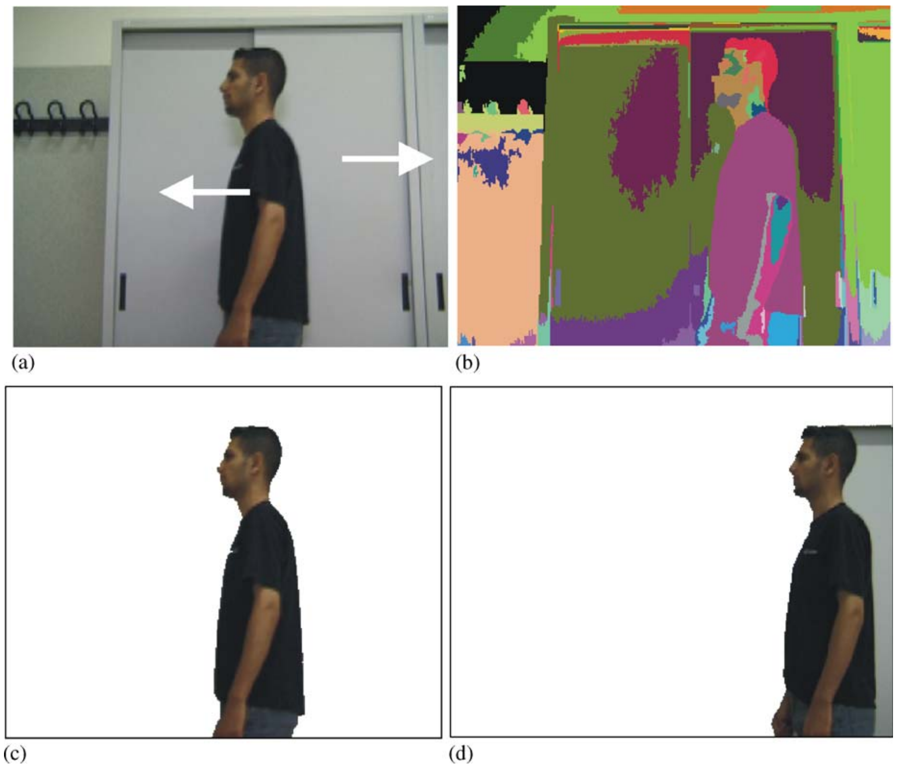


Fig. 5. Frame 140 from indoor sequence: (a) original frame, (b) after color segmentation with region growing, (c) moving objects reported on output. (d) Output of frame 50, with a background's region wrongly merged with the foreground moving object.

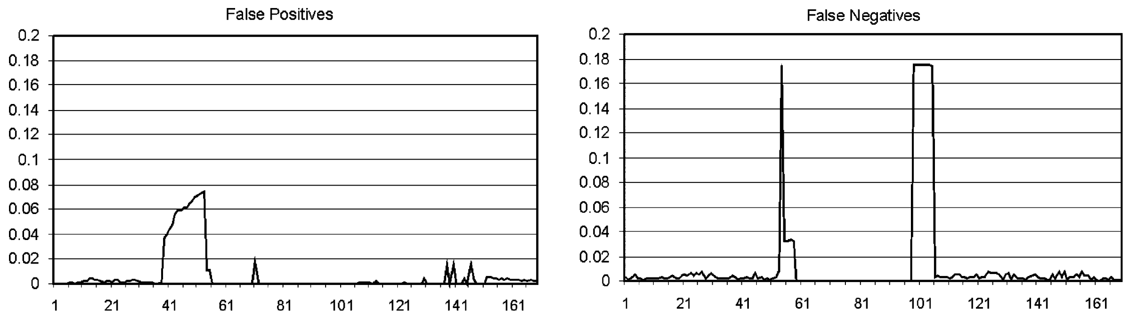


Fig. 6. Positives and negatives (as in Fig. 4) over the indoor sequence.

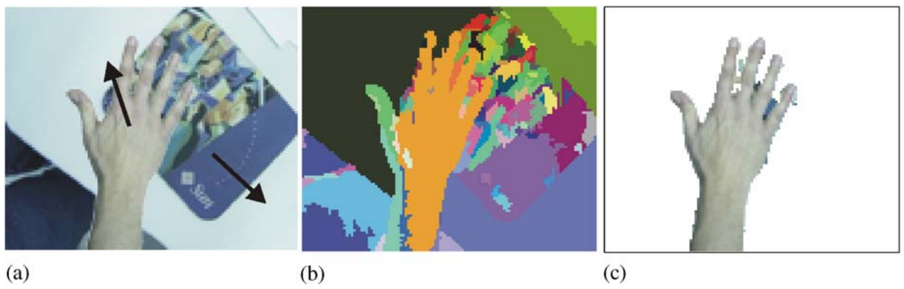


Fig. 7. Hand sequence: (a) an original frame, (b) after color segmentation, and (c) output.

- *Color segmentation*: the time necessary for this phase depends on the frame size (see Table 3); instead, number of created regions and color threshold do not affect it (see also Fig. 9).
- *Spatial graph creation*: is very fast and its time cost is negligible.
- *Motion estimation*: is the most critical phase, not only for its algorithmic complexity, but also for the several

factors that concur to make difficult the computational time estimation. In fact, it depends on the frame size s, and on the number of regions created by the color segmentation. Tables 3 and 4 show the execution time of motion estimation with PRM.

- *Markov Random Field optimization*: for this task, the frame size is obviously not relevant, because the application works only on the graph. On the other hand, the time occurred for energy minimization varies with the number of regions (see Table 3 and Fig. 9). As it is possible to see, time required for MRF increases not linearly with the number of regions. Table 4 shows that PRM is affected by the maximum shift s. If we suppose not to have fast objects (e.g., with $s = 2$) PRM asks for 43.8 ms for frame (it grows up to 1 s if we shift the region up to 20 pixels).

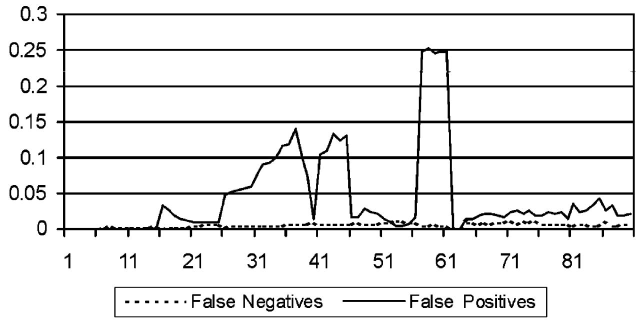


Fig. 8. False negatives and positives (as in Fig. 4) over the hand sequence.

Calling n the number of nodes of the graph, i.e. number of regions, the minimization algorithm proposed in Section 3.4 has a complexity of $O(n^4)$. In fact, each iteration analyzes all the n nodes; for each of them it computes the energy value on the n arcs linking it with the others. If the state of the analyzed node changes, all the other n nodes may be re-evaluated. Supposing that each node is evaluated n times, we obtain a complexity of $O(n^3)$. Then a multi-scale optimization is performed, that consists, in the worst case, of n iterations.

As indicated by the total time costs in Table 3 and Fig. 9, we are far from satisfying real time constraints. In fact, although downscaling the frame size can allow very fast computation of region growing and motion

Table 3
Processing time dependencies from parameters

Size	Param. of R.G. (α, β)	R.M. Max shift	N regions	Computational time (s)			
				Segm. + SRG	Motion estimation	Region MRF	Total
100 × 100	70-8	2	43	0.125	0.047	0.031	0.203
200 × 200	70-25	4	36	0.578	0.531	0.031	1.140
400 × 400	70-120	8	32	2.562	6.625	0.094	9.281
100 × 100	70-20	6	23	0.125	0.172	0.016	0.313
200 × 200	70-20	6	41	0.562	0.938	0.031	1.531
400 × 400	70-20	6	116	2.594	4.156	0.391	7.141

In the first three rows the parameters are scaled with the dimension, while in the other three are fixed.

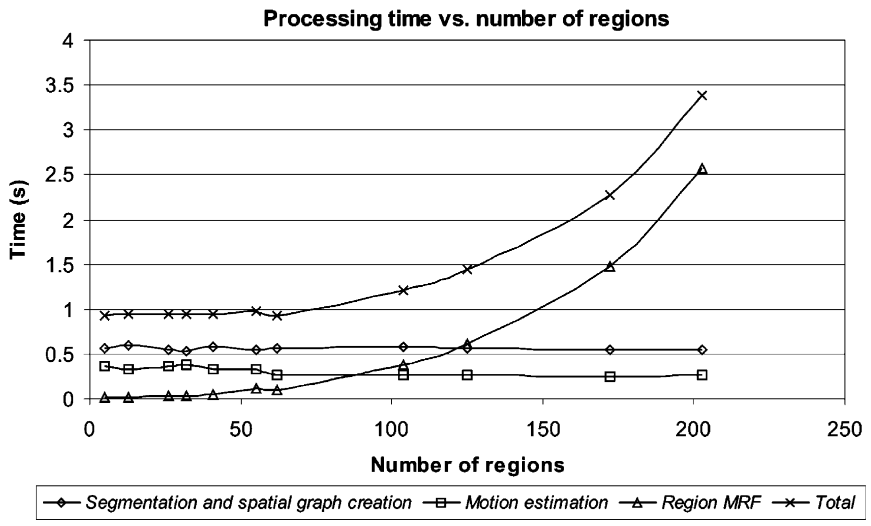


Fig. 9. Processing time vs. number of regions (on object sequence 200 × 200).

Table 4
Times (in seconds) for motion estimation vs. maximum shift

Max shift (s)	PRM times (seconds)
2	0.0438
3	0.0658
4	0.1022
5	0.1344
6	0.1804
8	0.2814
12	0.5124
15	0.7406
20	1.0606

Frame size: 100×100.

estimation, the MRF optimization requires too high computational times (consider that region growing usually produce almost 30 regions). For this reason, in the next section we propose an innovative algorithm of optimization based on arcs instead of nodes characterized by a quadratic complexity.

5. Arc-based optimization

5.1. Energy function

First, the energy function previously presented has been slightly modified; in particular, the term U_3 is now defined as the number of *broken arcs* (i.e. pairs of regions geometrically adjacent but with different labels). The third energy term is, thus, expressed in a form similar to the other two terms:

$$U_3(e) = c_3 \sum_{(s,t)\in\Gamma} V_3(e_s, e_t), \tag{17}$$

$$V_3(e_s, e_t) = \begin{cases} 1 & e_s \neq e_t, \\ 0 & e_s = e_t, \end{cases} \tag{18}$$

where c_3 is a positive constant.

The weak proportionality between number of labels and *broken arcs* justifies this change. The global energy function can be now entirely decomposed into the sum of local terms defined over cliques:

$$U(e,o) = \sum_{(s,t)\in\Gamma} U_{s.t}(e_S, e_t, o_s, o_t) = \sum_{(s,t)\in\Gamma} V_1(e_s, e_t, o_s, o_t) + V_2(e_s, e_t) + V_3(e_s, e_t). \tag{19}$$

Analyzing the definitions of the three energy terms we can define a unified local term as:

$$U_{s,t}(e_s, e_t, o_s, o_t) = \begin{cases} V_1(e_s, o_s, e_t, o_t) + V_2(e_s, e_t) & e_s = e_t, \\ V_3(e_s, e_t) & e_s \neq e_t. \end{cases} \tag{20}$$

We can also assign a binary state to each arc that identifies if the two connected regions have the same label or not.

5.2. Energy optimization

The main advantage of Eq. (20) consists on the possibility of a faster optimization. The algorithm here proposed is composed by two scanning phases: the first on the arcs and the second on the nodes. As above reported, the computational complexity becomes quadratic $O(n^2)$ (evaluation of n^2 arcs plus n nodes).

For each arc (s,t) we evaluate if it is more convenient to break it or not, by considering only the sub-graph composed by the arc and the two connected nodes (s and t). To this aim we define $W_{s,t}$ as in Eq. (21) and the test to perform on each arc is shown in Eq. (22):

$$W_{s,t} = W_{s,t}(e_s, e_t, o_s, o_t) = V_1(e_s, o_s, e_t, o_t) + V_2(e_s, e_t) - V_3(e_s, e_t), \tag{21}$$

$$W_{s,t} \geqslant 0 \Rightarrow (s,t) \textit{ broken}. \tag{22}$$

After that, the nodes of the graph are labeled using a modified region growing algorithm. Starting from the first node, all the nodes connected with non-broken arcs are merged with the same label. This optimization is sub-optimal because the graph may present a sort of inconsistence.

In fact, Fig. 10a shows a segmented frame of the indoor sequence and the correspondent SRG. A particular containing three adjacent regions (labeled with r, s, t) is zoomed in Fig. 10b. If we suppose that the signs of $W_{s,t}$, $W_{s,r}$, and $W_{r,t}$ are as in Fig. 10c (positive the first and negative the others two) the graph will become inconsistent. The arc (s,t) is *broken*, but the growing algorithm assigns the same label to nodes s and t walking through arcs (s,r) and (r,t) with the consequent increase of the energy associated to the arc (s,t) and the global energy is not minimized. Instead, an optimum algorithm should evaluate if preserving a unique label for the three nodes is "more convenient" than assign a different label to node s or node t. For this reason, the defined algorithm is not generalizable for any graph, while it is applicable with good results in our application. In fact, the energy function is principally based on motion difference. The situation presented in figure is very infrequent, because it means that r and t have a similar motion, the same for s and r, but not for s and t.

The main task for our purposes is to merge regions with same motion and separate regions with *very different* motion. In other cases, possible errors due to segmentation and motion estimation result to be more relevant than previous simplifications in MRF optimization. Unfortunately, with this new algorithm, we lose the implicit tracking of the objects, because the labeling

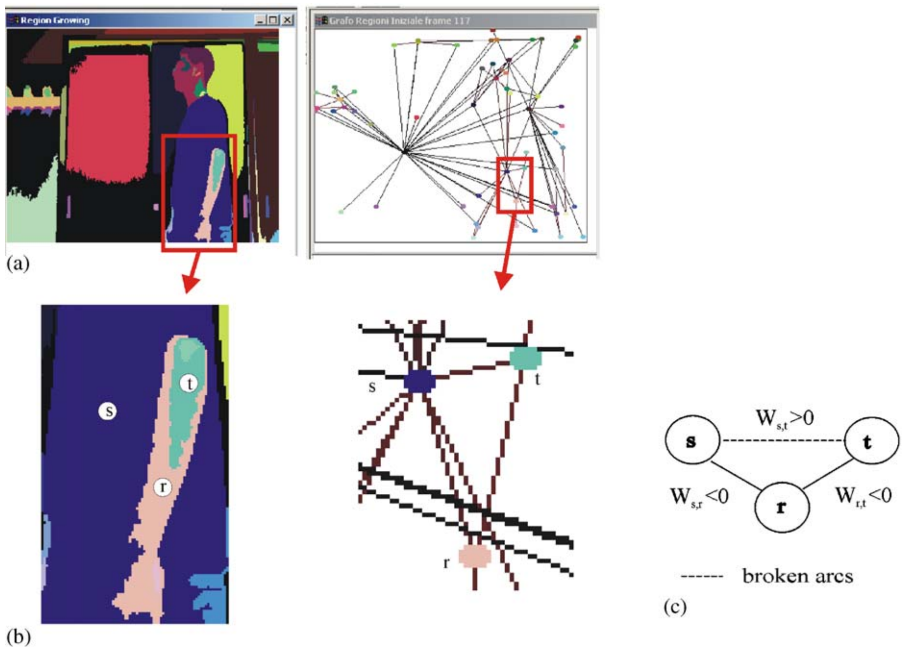


Fig. 10. Possible graph inconsistence during arc based optimization.

phase does not consist in an optimization of a prediction as in Section 3.3 and in [1]. The prediction is useless since labels are given by following the non-broken arks. Indeed, a final tracking phase after the end of motion segmentation could be added, if needed. The performance analysis reported in the next section shows the validity of the method and the advantages in terms of speed.

6. Performance analysis on the modified algorithm

6.1. *Processing times*

The main advantage of this new solution is the reduction of computational complexity. The graph in Fig. 11 shows time analysis for arc based MRF optimization with respect to the number of initial regions. Comparing it with the one of node-based MRF of Fig. 9, the improvement of speed is evident; this phase is not the bottleneck of the system anymore. For instance, for 50, 100, and 200 regions it requires only 23, 26 and 45 ms, respectively, and node-based MRF requires about 970, 1200, and 3300 ms, respectively. In particular, if we analyze ‘‘object sequence’’ and ‘‘hand sequence’’, we downscale them to the size of 100×100 pixel, and assuming s equal to 2, we are able to reach 10 fps on a standard dual Pentium III 1000 MHz with 512 MB of RAM. Obviously, using a more off-the-shelf, powerful PC would allow us to reach better performance.

6.2. *Qualitative metrics*

It remains to prove that the sub-optimal optimization introduced does not affect much the efficacy of our system. To do this, the new system was tested on the same sequences used previously. In the ‘‘objects sequence’’ we obtained identical results: the color segmentation is unchanged and the region motion values are always correctly computed; both the MRF algorithms of Sections 3.3 and 5.2 can extract the objects in motion correctly. Instead, in the case of the ‘‘hand sequence’’ the output is slightly worst with the new optimizer, because wrong motion values are assigned to the parts of small regions affected by occlusion (compare Figs. 12 and 13). In Fig. 13 the number of pixels in real motion and the ones computed as in motion with the new optimizer are reported. Anyway, as it is possible to see, the results are satisfactory since the difference is limited between the two profiles: the ‘‘output’’ curve is higher than the ‘‘real’’ curve because of the merging of small regions of background to the moving hand. Similar performance results in efficacy have been tested in the ‘‘indoor sequence’’ and in other videos.

7. Conclusions

We proposed a real-time approach based on color and motion segmentation with MRF for moving object detection in moving background or in general in video acquired by moving cameras. The method achieves

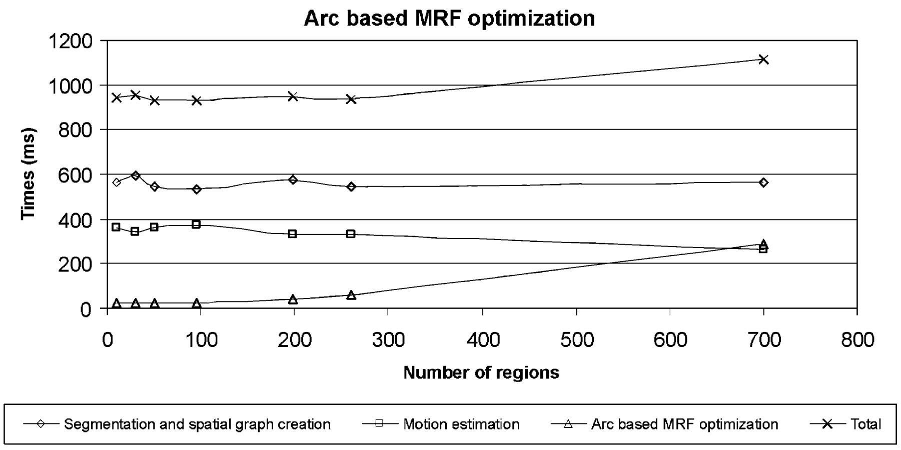


Fig. 11. Computational time for arc based MRF optimization (on object sequence 200 × 200).

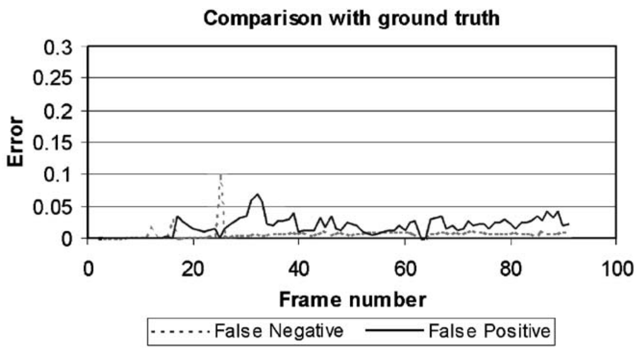


Fig. 12. False negatives and positives (as in Fig. 4) over the hand sequence with arc-based optimization.

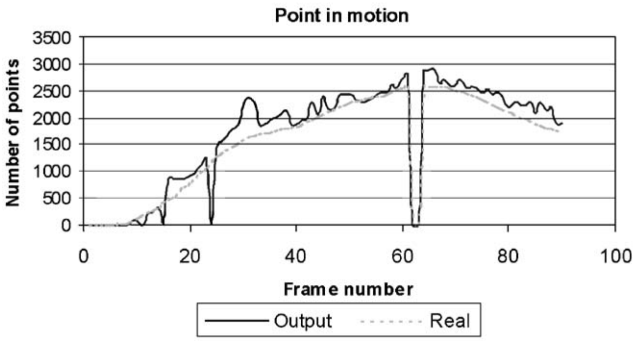


Fig. 13. Points in motion on hand sequence calculated with arc based optimization.

satisfactory results when the motion can be approximated with a translational model. A new Partitioned Region Matching has been proposed to perform a good motion estimation also in presence of occlusion or shape variation; moreover, a motion reliability value in the MRF allows us to weight differently the motion or the geometry of regions in the case of low texture or large shape changes. However, the method suffers from the typical limitation of methods based on color as the unique visual feature of segmentation: adjacent moving objects with the same color are merged by definition. The method could be improved by adding other features in the initial segmentation.

Then, a new sub-optimal energy minimization algorithm applicable on this particular case of Markov Random Fields has been defined. With its introduction, we can process a sequence composed by frame of 100 × 100 pixels with a frame rate of about 10 fps on a standard PC platform.

In conclusion, the system presented in this work is able to segment videos acquired from moving camera in an effective way and in a very fast manner: thus, it can be used not only in off-line processing of stored videos for information retrieval, but also on-line, as, for instance, in live camera for video surveillance.

Acknowledgements

This work is partially supported by the *FIRB 2002-04 - PERF* "*Valutazione delle prestazioni nei sistemi complessi tecniche metodologie e strumenti*" project. The authors are also thankful to Vahabi Hossein for providing the indoor sequence and his help during the test phase of the system.

References

[1] Gelgon M, Bouthemy P. A region-level motion-based graph representation and labeling for tracking a spatial image partition. Pattern Recognition 2000;33:725–40.
[2] Collins R, Tsin Y. Calibration of an outdoor active camera system. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, vol. 1, 1999. p. 528–34.
[3] Araki S, Matsuoka T, Takemura H, Yokoya N. Real-time tracking of multiple moving objects in moving camera image sequences using robust statistics. In: Proceedings of

International Conference on Pattern Recognition, vol. 2, 1998. p. 1433–5.
[4] Jehan-Besson S, Barlaud M, Aubert G. Region-based active contours for video object segmentation with camera compensation. In: Proceedings of IEEE International Conference on Image Processing, vol. 2, 2001. p. 61–4.
[5] Cutler R, Davis L. Robust real-time periodic motion detection. IEEE Transactions on Pattern Analysis and Machine Intelligence 2000;22(8):781–96.
[6] Lee K, Ryu S, Lee S, Park K. Motion based object tracking with mobile camera. Electronics Letters 1998;34(3):256–8.
[7] Odobez J, Bouthemy P. Detection of multiple moving objects using multiscale mrf with camera motion compensation. In: Proceedings of IEEE International Conference on Image Processing, vol. 2, 1994. p. 257–61.
[8] Paragios N, Perez P, Tziritas G, Labit C, Bouthemy P. Adaptive detection of moving objects using multiscale techniques. In: Proceedings of IEEE International Conference on Image Processing, vol. 1, 1996. p. 525–8.
[9] Bhat K, Saptharishi M, Khosla P. Motion detection and segmentation using image mosaics. In: Proceedings of International Conference on Multimedia and Expo, vol. 3, 2000. pp. 1577–80.
[10] Sawhney H, Ayer S. Compact representations of videos through dominant and multiple motion estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence 1996;18(8):814–30.
[11] Megret R, Saraceno C, Kropatsch W. Background mosaic from egomotion. In: Proceedings of International Conference on Pattern Recognition, vol. 1, 2000. p. 571–4.
[12] Tian T, Tomasi C, Heeger D. Comparison of approaches to egomotion computation. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 1996. p. 315–20.
[13] Bergen J, Burt P, Hingorani R, Peleg S. A three frame algorithm for estimating two-component image motion. IEEE Transactions on Pattern Analysis and Machine Intelligence 1992;14(12):886–96.
[14] Chang M, Teklap A, Sezan M. Simultaneous motion estimation and segmentation. IEEE Transactions on Image Processing 1997;6(9):1326–33.
[15] Grinias I, Tziritas G. Motion segmentation and tracking using a seeded region growing method. In: Proceedings of European Signal Processing Conference, 1998.
[16] Hennebert C, Rebuffel V, Bouthemy P. A hierarchical approach for scene segmentation based on 2d motion. In: Proceedings of International Conference on Pattern Recognition, vol. 1, 1996. p. 218–22.
[17] Ninomiya Y, Matsuda S, Ohta M, Harata Y. A real-time vision for intelligent vehicles. In: Proceedings of IEEE Intelligent Vehicle Symposium, 1996. p. 315–20.
[18] Smith S, Brady J. Asset-2: real-time motion segmentation and shape tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence 1995;17(8):814–20.
[19] Kam J. A real-time 3D motion tracking system. Tech. Rep. 93-16, Laboratory for Computational Intelligence, Department of Computer Science, University of British Columbia. April 1993.
[20] Wang J, Adelson E. Representing moving images with layers. IEEE Transactions on Image Processing 1994;3(5):625–38.
[21] Altunbasak Y, Tekalp A, Bozdagi G. Simultaneous motion-disparity estimation and segmentation from stereo. In: Proceedings of IEEE International Conference on Image Processing, 1994. p. 73–7.
[22] Lin Y, Chen Y, Kung S. Object-based scene segmentation combining motion and image cues. In: Proceedings of IEEE International Conference on Image Processing, vol. 1, 1996. p. 957–60.
[23] Altunbasak Y, Eren P, Tekalp A. Region-based affine motion segmentation using color information. In: Proceedings of International Conference on Acoustic, Speech, and Signal Processing, vol. 4, 1997. p. 3005–8.
[24] Fablet R, Bouthemy P, Gelgon M. Moving object detection in color image sequences using region-level graph labeling. In: Proceedings of IEEE International Conference on Image Processing, vol. 2, 1999. p. 939–43.
[25] Tweed D, Calway A. Motion segmentation based on integrated region layering and motion assignment. In: Proceedings of Asian Conference on Computer Vision, 2000. p. 1002–7.
[26] Smith P, Drummond T, Cipolla R. Segmentation of multiple motions by edge tracking between two frames. In: Proceeding of British Machine Vision Conference, 2000. p. 342–51.
[27] Moscheni F, Bhattacharjee S. Robust region merging for spatio-temporal segmentation. In: Proceedings of IEEE International Conference on Image Processing, vol. 1, 1996. p. 501–4.
[28] Techmer A, Contour-based motion estimation and object tracking for real-time applications. In: Proceedings of IEEE International Conference on Image Processing, vol. 3, 2001. p. 648–51.
[29] Csurka G, Bouthemy P. Direct identification of moving objects and background from 2d motion models. In: Proceedings of IEEE International Conference on Computer Vision, vol. 1, 1999. p. 566–71.
[30] Dufaux F, Moscheni F, Lippman A. Spatio-temporal segmentation based on motion and static segmentation. In: Proceedings of IEEE International Conference on Image Processing, vol. 1, 1995. p. 306–9.
[31] Lohier F, Lacassagne L, Garda P. Generic programming methods for the real time implementation of a mrf based motion detection algorithm on a multi-processor dsp with multi-dimensional dma. In: Proceedings of Symposium GRETSI on Signal and Image Processing, 1999.
[32] Van Leeuwen M, Groen F. Motion estimation with a mobile camera for traffic applications. In: Proceedings of IEEE Intelligent Vehicle Symposium, 2000. p. 58–63.
[33] Cucchiara R, Prati A, Vezzani R. Object segmentation in videos from moving camera with mrfs on color and motion features. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, vol. 1, 2003. p. 405–10.
[34] Adams R, Bischof L. Seeded region growing. IEEE Transactions on Pattern Analysis and Machine Intelligence 1994;16(6):641–7.
[35] Musmann H, Pirsch P, Grallert H. Advances in picture coding. Proceedings of the IEEE 1985;73(4):523–46.
[36] Zhu S, Ma K. A new diamond search algorithm for fast block matching. IEEE Transactions on Circuits and System for Video Technology 2000;9(2):287–90.
[37] Todorovic D. A gem from the past: Pleikart stumpf's (1911) anticipation of the aperture problem, reichardt detectors, and perceived motion loss at equiluminance. Perception 1996;25: 1235–42.
[38] Gelgon M, Bouthemy P. A region-level graph labeling approach to motion-based segmentation. In: Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition, 1997. p. 514–9.