Automatic Segmentation of Moving Objects in Video Sequences: A Region Labeling Approach

Yaakov Tsaig and Amir Averbuch

***Abstract*—The emerging video coding standard MPEG-4 enables various content-based functionalities for multimedia applications. To support such functionalities, as well as to improve coding efficiency, MPEG-4 relies on a decomposition of each frame of an image sequence into video object planes (VOPs). Each VOP corresponds to a single moving object in the scene. This paper presents a new method for automatic segmentation of moving objects in image sequences for VOP extraction. We formulate the problem as graph labeling over a region adjacency graph (RAG), based on motion information. The label field is modeled as a Markov random field (MRF). An initial spatial partition of each frame is obtained by a fast, floating-point based implementation of the watershed algorithm. The motion of each region is estimated by hierarchical region matching. To avoid inaccuracies in occlusion areas, a novel motion validation scheme is presented. A dynamic memory, based on object tracking, is incorporated into the segmentation process to maintain temporal coherence of the segmentation. Finally, a labeling is obtained by maximization of the *a posteriori* probability of the MRF using motion information, spatial information and the memory. The optimization is carried out by highest confidence first (HCF). Experimental results for several video sequences demonstrate the effectiveness of the proposed approach.**

***Index Terms*—Markov random fields, MPEG-4, video segmentation, VOP extraction, watershed segmentation.**

I. Introduction

With the phenomenal growth of the Internet and the World Wide Web, the interest in advanced interactivity with audio–visual contents is increasing rapidly. To address these growing needs, a new video coding standard, MPEG-4 [1], was introduced. Unlike its predecessors, MPEG-1 [2] and MPEG-2 [3], MPEG-4 targets more than just large coding gains. To provide new functionalities for multimedia applications, such as content-based interactivity and content-based scalability, it introduces a content-based representation. Scenes are treated as compositions of audio–visual objects, which are separately encoded and decoded.

To support the more advanced functionalities offered by MPEG-4, a prior decomposition of a scene into physical objects is required. A physical object in a frame is represented by a *video object plane* (VOP), which is a snapshot of a moving object at a given time and from a given view. Each frame of a video sequence is composed of VOPs corresponding to objects in the scene. From this definition, it is clear that VOPs carry a semantical meaning. As a result, VOPs cannot be uniquely characterized by a low-level feature such as motion, intensity, color, etc. Therefore, VOP segmentation is generally far more difficult than low-level segmentation. Furthermore, VOP extraction for content-based interactivity functionalities requires that the obtained object mask would be flawless, since even small errors in the object contour can render a VOP useless for such applications.

Segmentation of a video sequence into VOPs is not a normative part of the MPEG-4 video coding scheme. Yet, VOP segmentation constitutes the basis for content-based representation of natural video sequences. Therefore, accurate VOP segmentation is a crucial factor in the future success of MPEG-4 as a content-based video coding standard.

Most of the segmentation techniques suggested for extraction of VOPs from image sequences rely on change detection as the source of temporal information [4]–[8]. This approach is motivated by the assumption that moving objects usually entail intensity changes between successive frames. Hence, a change detection mask (CDM) can be computed by applying a decision rule on the intensity differences between successive frames in a sequence. This approach is also attractive from a computational point of view, since CDMs are much easier to compute than motion fields. However, this approach suffers from two major drawbacks. First, unless moving objects are sufficiently textured, only occlusion areas (covered background) will be marked as changed, while the interior of objects will remain unchanged. Second, uncovered background will be marked as changed in the process as well, thus the boundaries of the moving objects are likely to be inaccurate. To overcome this drawback, a post-processing step that distinguishes between moving objects and uncovered background has to be applied [4], [8].

This paper presents a new algorithm for automatic segmentation of moving objects in image sequences. The approach underlying our algorithm is to classify regions obtained in an initial partition as foreground or background, based on motion information. A block diagram of the proposed algorithm is depicted in Fig. 1.

We formulate the segmentation problem as detection of moving objects over a static background. Thus, the first step in the algorithm is to compensate the motion of the camera. The global motion is modeled by an eight-parameter perspective motion model and estimated using a robust gradient-based technique [9].

An initial spatial partition of the current frame is obtained by applying the watershed segmentation algorithm. For this purpose, the spatial gradient is first estimated in the color space

Manuscript received May 13, 2001; revised April 16, 2002. This paper was recommended by Associate Editor K. Aizawa.

The authors are with the Department of Computer Science, School of Mathematical Sciences, Tel-Aviv University, Tel-Aviv 69978, Israel (e-mail: tsaig@post.tau.ac.il; amir@post.tau.ac.il).

Publisher Item Identifier 10.1109/TCSVT.2002.800513.

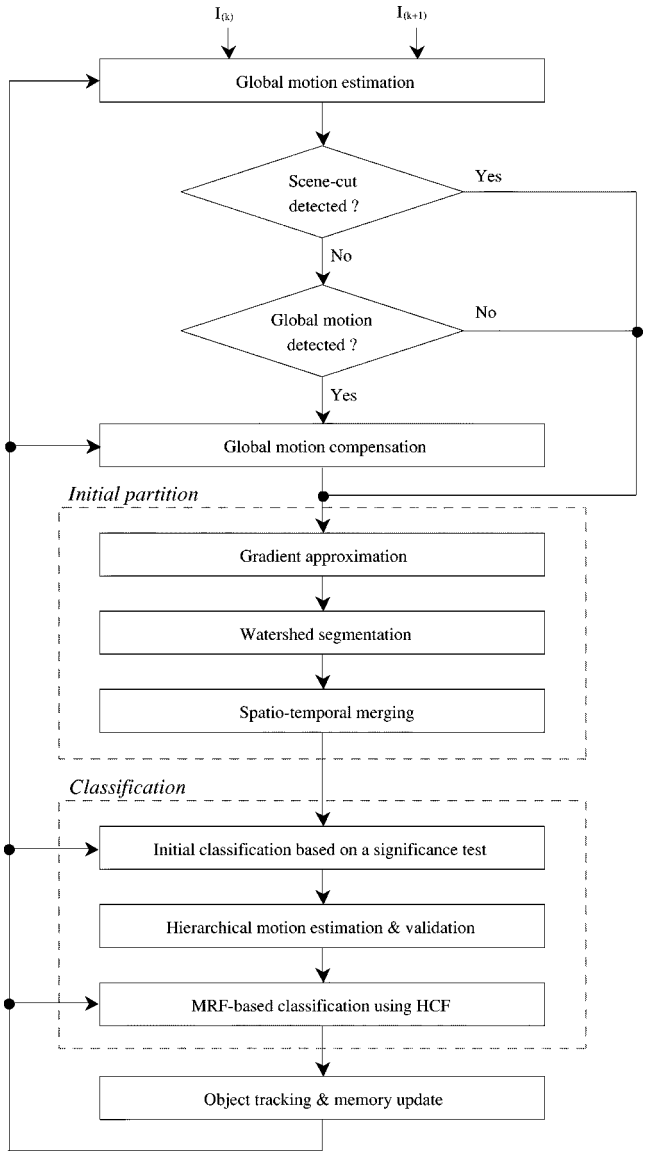


Fig. 1. Block diagram of the proposed algorithm.

through the use of Canny's gradient [10]. Then, the optimized rainfalling watershed algorithm is applied [11]. To reduce oversegmentation, small regions are merged together in a post-processing step based on spatio-temporal information.

Based on the initial partition, the classification stage labels regions as foreground or background. It begins with an initial classification based on a statistical significance test, which marks regions as foreground candidates. This initial stage is useful in increasing the efficiency and robustness of the classification. The motion of each foreground candidate is estimated by region matching in a hierarchical framework. To avoid false movements caused by occlusion, an iterative validation scheme examines the estimated motion vectors and corrects them, if necessary. The estimated motion vectors are then incorporated into a Markov random field model, along with spatial information and information gathered from previous frames. The optimization of the Markov random field (MRF) model is performed using highest confidence first (HCF) [12], leading to a classification of the regions in the initial partition.

Finally, a dynamic memory is introduced into the algorithm to ensure temporal coherency of the segmentation process. The memory is updated using the estimated motion vectors and the MRF-based classification. Each region is tracked to the next frame in the sequence and the memory is updated accordingly.

The paper is organized as follows. Section II addresses the preliminary stages of the algorithm, namely, global motion estimation and compensation and scene-cut detection. Section III discusses the initial partition and Section IV explains the classification process. Experimental results are presented in Section V, and concluding remarks are given in Section VI.

II. Preliminary Steps

A. Global Motion Estimation

For the estimation of the global motion, we utilize a gradient-based parametric motion estimation technique suggested in [9]. The camera motion is modeled by the eight-parameter perspective motion model

$$\begin{pmatrix} x_i' \\ y_i' \end{pmatrix} = \begin{pmatrix} \dfrac{a_1 + a_2 x_i + a_3 y_i}{a_7 x_i + a_8 y_i + 1} \\ \dfrac{a_4 + a_5 x_i + a_6 y_i}{a_7 x_i + a_8 y_i + 1} \end{pmatrix} \tag{1}$$

and the global motion is estimated using the Levenberg–Marquardt (LM) nonlinear minimization algorithm. To increase the robustness of the estimation process, as well as to reduce computation time, the LM algorithm is applied within a hierarchical framework, using a three-level multiresolution pyramid. To assure convergence of the algorithm, an initial stage is performed that computes a coarse estimate of the translation component of the displacement, by applying a matching technique at the coarsest level of the pyramid.

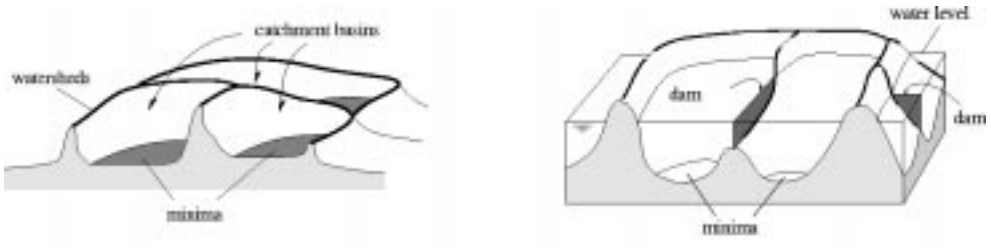
To remove the influence of local motions on the estimation process, the estimation is carried out considering only pixels within background regions of the current frame. Background regions are determined according to the tracked mask of the previous frame (the tracking mechanism is described in Section IV-D). This also reduces the computation time of the estimation process. In case the background regions are not yet known, e.g., in the first frame or after a scene-cut, the global motion estimation is carried out over the entire frame.

B. Scene-Cut Detection

To allow the segmentation of sequences composed of several shots, the transitions (cuts) between adjacent shots need to be detected. For this purpose, a scene-cut detector evaluates whether or not the difference between the current original frame I_k and the motion-compensated reference frame I_{k+1} exceeds a given threshold. In detail, we examine the average sum of absolute differences (ASAD) between the two frames

$$\frac{1}{M} \sum_i \left| I_{k+1}(x_i', y_i') - I_k(x_i, y_i) \right| \tag{2}$$

where the summation is carried over all M pixels that participate in the GME. A scene-cut is detected by comparing the ASAD with a predetermined threshold T_{sc}. If the ASAD is



(a) (b)

Fig. 2. (a) Minima, catchment basins, and watersheds on the topographic representation of a gray-scale image. (b) Building dams at the places where the water coming from two different minima would merge.

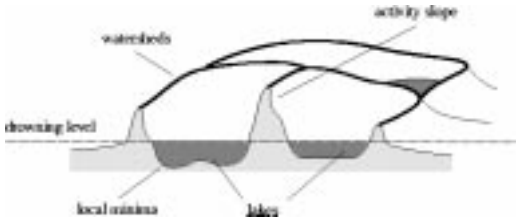


Fig. 3. Topographic watershed cross section, after the drowning process. Lakes are formed by merging neighboring pixels below the drowning threshold.

larger than $T_{\rm sc}$, then a scene-cut is detected between frames k and $k+1$, thus frame $k+1$ cannot serve as a reference frame for frame k. Hence, the classification for frame k is based only on the memory. Furthermore, the segmentation algorithm is reset in frame $k+1$, i.e., all parameters are set to their initial values.

C. Global Motion Compensation

Following the global motion estimation, the camera motion is compensated. Before applying the compensation, the estimated parameters of the motion model defined in (1) are evaluated to determine if a camera motion actually occurred. This step is performed to increase the robustness of the algorithm, since the global motion estimation may falsely detect small global motion due to local motions in the scene, or due to noise. In order to prevent this, the sum of the absolute values of the estimated motion parameters $\{a_1, \ldots, a_8\}$ is compared to a threshold T_{GMC}. Global motion is compensated only when

$$\left(\sum_{i=1}^{8} |a_i|\right) - 2 > T_{\rm GMC}.$$

If this is the case, then the global motion between frames I_k and I_{k+1} is compensated by warping frame I_{k+1} to frame I_k according to the perspective projection defined in (1).

III. Initial Partition

After the global motion has been estimated and compensated, an initial partition of the current frame is formed by watershed segmentation. The initial partition consists of three steps. First, the spatial gradient is approximated using Canny's method [10]. Then, the optimized rainfalling watershed algorithm is applied [11] using the gradient image as input. Finally, a spatio-temporal merge is performed on small regions to reduce the oversegmentation caused by the watershed algorithm. The obtained partition serves as input to the classification phase, which labels each region in the partition as foreground or background.

A. Gradient Approximation

The spatial gradient of the current frame is estimated using Canny's gradient approximation [10]. Specifically, the image is convolved with the first derivative of a Gaussian

$$\dot{G}(x) = \frac{-x}{\sqrt{2\pi}\sigma_G^3} \cdot e^{-x^2/2\sigma_G^2} \tag{3}$$

where the standard deviation of the Gaussian is chosen to be $\sigma_G = 1$.

Many researchers base the gradient approximation solely on the luminance component of the color, which contains most of the information. However, it is clear that some information is lost by discarding the chrominance components. In cases where the objects are not clearly distinct from the background, this extra information may have a profound impact on the segmentation results. Therefore, we incorporate color information into the segmentation process. For this purpose, Canny's gradient is first computed on each of the three color components, resulting in the gradient magnitudes G_Y, G_{C_r} and G_{C_b}. Then, a common gradient image is generated, that combines the information of the three color components. (The use of principal component analysis on the color components was investigated in [13], but the improvement gained did not justify the excess computations involved). Since the luminance component (Y) contains most of the information, it is likely that its associated gradient image differs considerably in magnitude from the gradient images of the chrominance components (C_r and C_b). Thus, if we attempt to generate a common gradient image by a simple summation, the luminance component will prevail over the other components and some very useful complementary details are likely to be lost. To overcome this problem, some sort of scaling must be incorporated in order to normalize the three gradient images. Therefore, the common gradient image $G_{\rm col}$ is generated using

$$G_{\rm col} = \max\left\{ w_Y \cdot \frac{G_Y}{\sqrt{\sigma_Y}}, w_{C_r} \cdot \frac{G_{C_r}}{\sqrt{\sigma_{C_r}}}, w_{C_b} \cdot \frac{G_{C_b}}{\sqrt{\sigma_{C_b}}} \right\} \tag{4}$$

where σ_Y, σ_{C_r}, and σ_{C_b} are the variances of the gradient magnitudes of the three color components and w_Y, w_{C_r}, w_{C_b} are the associated weight coefficients. Notice that this scaling also has the obvious drawback of amplifying the noise in the chrominance components. To partly eliminate this side-effect, we used a weighted maximization over the three color components rather than a weighted sum (as was used, e.g., in [14]). Experimentally,

we found that setting the weights to $w_Y = 0.5$, $w_{C_r} = w_{C_b} = 0.25$ leads to satisfactory results. This set of weights gives the luminance component more importance than the other two components, in order to suppress false gradient maxima due to noise in the gradients of the chrominance components. Fig. 4(b) shows the gradient image obtained for the first frame of the sequence *Foreman* using this set of weights.

B. Watershed Segmentation

To obtain an initial partition of the current frame, the watershed algorithm is applied on the gradient image G_{col}. Watershed segmentation draws its origins from mathematical morphology and is, in fact, a region-growing algorithm that treats the input image as a topographic surface, and through the intuitive process of water-filling, creates a partition of the image. Commonly, two approaches exist for the implementation of the watershed algorithm. The first approach relies on rainfalling simulations. Assume that a drop of water falls on a topographic surface. According to the law of gravitation, it will flow down along the steepest slope path until it reaches a minimum. The whole set of points of the surface whose steepest slope paths reach a given minimum constitutes the catchment basin associated with this minimum. The watersheds are the zones dividing adjacent catchment basins. This is illustrated in Fig. 2(a). The second approach makes use of *immersion* or *flooding* simulations. Consider again the topographic surface and assume that holes have been punctured in each regional minimum of the surface. The surface is then slowly immersed into a lake. Starting from the minima at the lowest altitude, the water will progressively flood the catchment basins of the image. In addition, dams are erected at places where the water coming from two different minima would merge [see Fig. 2(b)]. At the end of this flooding procedure, each minimum is completely surrounded by dams, which delineate its associated catchment basin. The resulting dams correspond to the watersheds. They provide us with a partition of the input image into its different catchment basins.

The latter definition was used in [15] to derive an efficient implementation using FIFO queues. However, the implementation required that the input image be discrete. Hence, the gradient image must be quantized, which may lead to inaccuracies in the resulting segmentation. Recently, a new algorithm for computation of watersheds was presented [11]. This algorithm relies on the definition of watershed in terms of rainfalling simulations. As a result, it does not require the input image to be discrete. Moreover, the optimized implementation is 3–4 times faster than the one presented in [15]. In addition, the algorithm introduces a new *drowning* step that removes some of the weakest edges and helps reduce the influence of noise. This drowning step can be thought of as flooding the surface with water at a certain level. This process will create a number of *lakes* grouping all the pixels that lie below the drowning level (see Fig. 3).

While the watershed algorithm in [11] produces very accurate results, it has one inherent drawback: it is extremely sensitive to gradient noise and usually results in oversegmentation. This is evident from Fig. 4(c), which shows the result of the watershed algorithm on the first frame of the sequence *Foreman*. In order to eliminate the oversegmentation caused by the watershed algorithm, it is common to employ a post-processing step that merges small regions. In the following section, we suggest a region-merging scheme, which relies on temporal information as well as spatial information, to maintain an accurate segmentation.

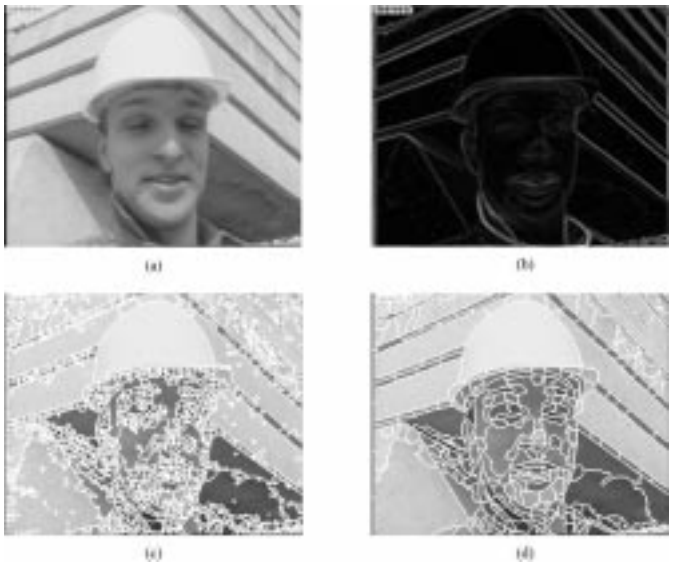


Fig. 4. Initial partition for the first frame of *Foreman*. (a) Original image. (b) gradient magnitude. (c) Partition after watershed segmentation (2020 regions). (d) Partition after spatio-temporal merging (312 regions).

C. Spatio-Temporal Merging

The goal of the merging step is to reduce the number of regions in the partition by eliminating small regions, while maintaining the accuracy of the partition. If we base the region merging solely on spatial information, we may jeopardize the accuracy of the segmentation, since moving regions might mistakenly be merged with the background. In order to avoid this, it is obvious that we must consider temporal information, as well as spatial information. Therefore, we suggest the following scheme for spatio-temporal merging.

Suppose that the output of the watershed segmentation consists of N regions $\{R_1, \ldots, R_N\}$, where the number of pixels within the region R_i is N_i. We assume that the spatial relations between regions in the partition are known. Let $\mathbf{I}(x,y) = (I^1(x,y), I^2(x,y), I^3(x,y))$ denote the intensity functions of the three color components (Y, C_r and C_b) of the current frame (the frame index k is omitted to simplify notation) and $\mathbf{A}_i = (A_i^1, A_i^2, A_i^3)$ denote the vector of mean intensity values of the three color components, with components

$$A_i^l = \frac{1}{N_i} \sum_{(x,y)\in R_i} I^l(x,y), \qquad l = 1, \ldots, 3. \tag{5}$$

Define two distance measures between neighboring regions R_i and R_j as

$$A_{ij} = \sum_{l=1}^{3} \left| A_i^l - A_j^l \right| \tag{6}$$

$$G_{ij} = \sum_{l=1}^{3} \frac{1}{N_{ij}} \sum_{(x_i,y_i),(x_j,y_j)} \left| I^l(x_i,y_i) - I^l(x_j,y_j) \right| \tag{7}$$

where the inner summation in (7) is carried out over all pairs of 4-connected pixels $(x_i, y_i) \in R_i$, and $(x_j, y_j) \in R_j$ and N_{ij} is their cardinality. A_{ij} represents the difference between the average intensities of the two regions, whereas G_{ij} measures the weakness of the common boundary between the two regions.

Based on these two distance measures, the spatial distance D_{ij} between two neighboring regions R_i and R_j is defined as

$$D_{ij} = \frac{1}{2}\left(A_{ij} + G_{ij}\right). \tag{8}$$

In addition, we define a temporal distance measure, based on the differences between the current frame and the reference frame. Specifically, let $d_k^{\text{abs}}(x,y) = |I_{k+1}^1(x,y) - I_k^1(x,y)|$ denote the image of absolute luminance differences between frames k and $k+1$. Then, the temporal distance B_{ij} between R_i and R_j is defined as

$$B_{ij} = \frac{1}{N_{ij}} \sum_{(x_i,y_i)\in R_i,(x_j,y_j)\in R_j} \left| d_k^{\text{abs}}(x_i,y_i) - d_k^{\text{abs}}(x_j,y_j) \right|. \tag{9}$$

B_{ij} is the difference of luminance differences between successive frames on the common boundary of the two regions R_i and R_j. A high value of B_{ij} indicates that one of the regions is moving relative to the other, whereas a low value of B_{ij} indicates that the regions either belong to the background or to a single moving object and can be merged without concern. Thus, by adding a constraint on B_{ij}, we can avoid merging moving regions with the background.

Using this set of distance measures, the spatio-temporal merging process can now be described as follows.

1) Set $T_{\text{size}} = 20$.
2) For each region R_i, $i = 1, \ldots, N$ whose size (number of pixels) is smaller than T_{size}, starting with the smallest region:
   a) Among the neighbors of region R_i, find the region R_j which satisfies
   $$R_j = \arg \min_{R_j : B_{ij} \le T_B} D_{ij} \tag{10}$$
   that is, the region R_j whose distance D_{ij} to R_i is minimal, among all neighbors R_i that their temporal distance measure is lower than a predefined threshold T_B.
   b) If $D_{ij} \le T_D$, where T_D is a predefined similarity threshold, or R_j is the only neighbor of R_i, merge R_i and R_j.
3) $T_{\text{size}} = T_{\text{size}} + 20$.
4) If $T_{\text{size}} > 100$, stop. Otherwise, go back to step 2.

Selection of the thresholds T_D and T_B allows us to control the sensitivity of the merging process. The higher we set T_D and T_B, the more regions will be merged, resulting in a smaller number of regions, yet it is more likely that true boundaries will be violated in the process as well. Experimentally, we have found that using $T_D = 20$ and $T_B = 25$ leads to satisfactory results.

The effect of the spatio-temporal merge is illustrated in Fig. 4(d) for the test sequences *Foreman*. We can see that the spatio-temporal merge led to a dramatic decrease of about 85% in the number of segments, while maintaining the integrity of the object.

IV. Classification

The core of the proposed algorithm is the classification phase, which determines whether each region in the initial partition is part of a moving object or part of the background. The classification phase consists of three steps. First, an initial classification is performed that marks regions as potential foreground candidates based on a significance test. Then, the motion of each foreground candidate is estimated and validated. The motion information, along with spatial information, is used to define a MRF model. Finally, the classification problem is formulated as an optimization problem in the Bayesian framework and a solution is obtained using HCF.

A. Initial Classification

The classification phase begins with an initial classification that is based on a statistical significance test [16]. This initial phase serves two main purposes. First, it reduces the computational load of the following motion estimation phase by eliminating most of the background regions from the estimation process. This is especially important if the moving objects are small compared to the overall frame size. Second, it increases the robustness of the motion estimation by eliminating noisy background regions that may be falsely detected as moving.

Let $d_k(x,y) = I_{k+1}(x,y) - I_k(x,y)$ denote the image of gray-level differences between frames I_k and I_{k+1}. Under the hypothesis than no change occurred at position (x,y) (the null hypothesis H_0), the corresponding difference $d_k(x,y)$ follows a zero-mean Gaussian distribution

$$p\left(d_k(x,y)|H_0\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-d_k^2(x,y)/2\sigma^2} \tag{11}$$

where the noise variance σ^2 is equal to twice the variance of the camera noise, assuming that the camera noise is white. Rather than performing the significance test directly on the values $d_k(x,y)$, it is better to evaluate a local sum of normalized differences

$$\Delta_k(x,y) = \sum_{(x',y')\in W(x,y)} \frac{d_k^2(x',y')}{\sigma^2} \tag{12}$$

where $W(x,y)$ is a window of observation centered at (x,y). Under the assumption that no change occurs within the window, the normalized differences d_k/σ obey a Gaussian distribution $N(0,1)$ and are spatially uncorrelated, thus the local sum $\Delta_k(x,y)$ follows a χ^2 distribution with N degrees of freedom, N being the number of pixels within the window $W(x,y)$.

With the distribution $p(\Delta_k(x,y))$ known, we can obtain a decision rule for each pixel by performing a significance test on $\Delta_k(x,y)$. Specifically, for a given significance level α, we can compute a corresponding threshold T_α using

$$\alpha = \Pr\left\{\Delta_k(x,y) > T_\alpha | H_0\right\}. \tag{13}$$

The significance level α is in fact the false alarm rate associated with the statistical test. The higher the value of α is, the more likely we are to classify unchanged pixels as changed.

From the description above, it is obvious that the significance test depends on the noise variance σ^2. Thus, an accurate estimate of the noise variance is crucial for the performance of the test. To ensure this is obtained, the variance is estimated only within background regions of the current frame, to remove the influence of changed regions. The background regions are determined according to the tracked mask of the previous frame (the tracking mechanism is described in Section IV-D). In case the background regions are not yet known, e.g., in the first frame or after a scene-cut, a robust estimation scheme is used, in which the highest 5% differences are removed from the estimate.

It is important to note that the suggested significance test has one major drawback. Like all change-detection based methods, moving pixels will be marked as changed only within areas that are sufficiently textured. In other words, within smooth, uniform areas, only a small percentage of the overall pixels will be marked as changed, mostly due to covered/uncovered background. Thus, it is obvious that we cannot make a decision for each region based on a majority rule, since many truly moving regions will be eliminated in the process. Therefore, a region is classified as a foreground candidate if more than 10% of its pixels are marked as changed, otherwise it is marked as background.

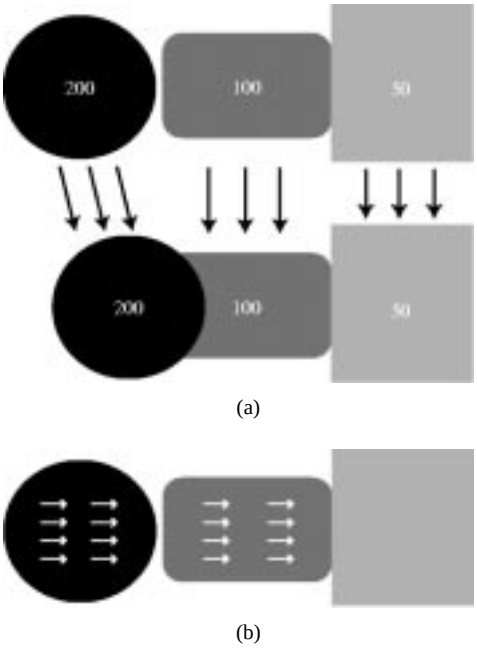
Finally, in order to avoid elimination of slowly moving regions, we must also consider the information gathered in previous frames of the sequence. For this purpose, regions that a majority of their pixels appear in the tracked mask of the previous frame, are marked as foreground candidates as well.

Experimentally, we have found that using a significance level $\alpha = 10^{-2}$ with an observation window of 5×5, leads to good results.

B. Hierarchical Motion Estimation and Validation

Following the initial classification stage, the motion of potential foreground candidates is estimated. The estimation takes place within a segmentation-based framework. That is, the motion of each region is determined by estimating a parametric motion model for the region. As the motion model, we selected the simple 2-parameter translational motion model. Our choice of this motion model is supported by the following facts. First, the regions resulting from the initial segmentation are usually small enough to justify the assumption of piecewise constant motion. Second, the goal of the motion estimation phase is to identify moving regions, rather than minimize the motion-compensation errors, thus the use of a complex model which accurately describes the motion of each region is not required. Finally, the use of a simple 2-parameter motion model allows for a very efficient implementation.

The motion estimation is carried out by intensity matching in a hierarchical framework [17]. The 3-level multi-resolution pyramid constructed in the global motion estimation stage (Section II-A) is used. The hierarchical search begins at the coarsest level of the pyramid and propagates to the higher levels of the hierarchy, while refining the displacement estimate in each level.



(a)

(b)

Fig. 5. Illustration of the occlusion problem. (a) Synthetic scene where the left object is moving over the middle object. The numbers denote the intensity values. (b) Estimated motion. The middle object was falsely detected as moving.

The hierarchical estimation technique suggested above, like all motion estimation techniques, suffers from the occlusion problem. Let us illustrate this problem with a simple example. Fig. 5(a) shows a synthetic scene composed of three objects, where the leftmost object (the black circle) is moving to the right, while partially occluding the middle object. Now, suppose that the motion estimation is carried out for each object separately. The motion of the rightmost and leftmost objects is determined correctly. However, when estimating the motion of the middle object, the occlusion created by the black circle causes a high difference in the occluded area. Since the ultimate goal of the motion estimation is to minimize the sum of differences, the differences caused by the occlusion induce motion toward the right object in order to compensate for these differences. Since the intensity differences between the left circle and the middle rectangle are significantly higher than the differences between the middle and right rectangles, the middle rectangle will be identified as moving, as shown in Fig. 5(b).

For the segmentation task at hand, this poses a serious problem, since it causes background regions to be detected as moving, which will inevitably lead to their misclassification as foreground. In order to overcome this problem, we propose a motion-validation scheme, based on a statistical hypothesis test.

The idea underlying the hypothesis test is to validate the motion vector of each region, by examining the motion-compensated differences (MCDs) within the occlusion area formed by its movement. If the estimated motion vector is valid, then the MCDs inside the region should be smaller than the frame differences inside it. If, on the other hand, the estimated motion was induced by occlusion, then, within the occlusion area, the MCDs are likely to be much bigger than the frame differences. Let us illustrate this point with an example. In Fig. 5(b), the estimated motion of the middle object formed an occlusion area over the

right square. Inside the occlusion area the MCD is 50, while the frame differences in this area equal zero (assuming that the sequence is noiseless). Thus, the estimated displacement is wrong.

Let us formalize this notion, in order to obtain a hypothesis test. If we assume that the estimated motion vector $(\Delta x, \Delta y)$ for the region R_i is valid, then, according to the brightness change constraint, we have the following:

$$I_k(x,y) = I_{k+1}(x+\Delta x, y+\Delta y)\ \forall (x,y) \in R_i. \tag{14}$$

Assuming that the sequence is subject to camera noise, then for pixels (x,y) within the occlusion area, the frame differences d_k satisfy

$$\begin{aligned} d_k(x+\Delta x, y+\Delta y) =& I_{k+1}(x+\Delta x, y+\Delta y) \\ & - I_k(x+\Delta x, y+\Delta y) + n \\ =& I_k(x,y) - I_{k+1}(x+\Delta x, y+\Delta y) + n \end{aligned} \tag{15}$$

where the noise n obeys a zero-mean Gaussian distribution

$$p(n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-n^2/2\sigma^2} \tag{16}$$

and its variance σ^2 is equal to twice the variance of the camera noise. Conversely, if the estimated motion vector for the region R_i is invalid, then the differences within the occlusion area are attributed only to camera noise

$$d_k(x+\Delta x, y+\Delta y) = n. \tag{17}$$

Therefore, we can define the two hypotheses as

$$\begin{aligned} H_0 &: d_k(x+\Delta x, y+\Delta y) = n \\ H_1 &: d_k(x+\Delta x, y+\Delta y) = D + n \end{aligned} \tag{18}$$

where we define $D = I_k(x,y) - I_{k+1}(x+\Delta x, y+\Delta y)$ to simplify notation. Using (16), we can write

$$\begin{aligned} p\left(d_k|H_0\right) =& \frac{1}{\sqrt{2\pi\sigma^2}} e^{-d_k^2/2\sigma^2} \\ p\left(d_k|H_1\right) =& \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(d_k-D)^2/2\sigma^2}. \end{aligned} \tag{19}$$

Now, with the *a priori* probabilities unknown, the optimal decision rule is the *maximum likelihood* (ML) criterion

$$\text{Decide } H_1 \text{ if } p\left(d_k|H_1\right) > p\left(d_k|H_0\right). \tag{20}$$

Substituting (19) into (20), we obtain

$$\text{Decide } H_1 \text{ if } |d_k(x,y)| > \frac{|D|}{2}. \tag{21}$$

Thus, a decision rule is obtained for the validation of the motion vector for a pixel in the occlusion area of a given region R_i. Assuming that the noise is spatially uncorrelated, the decision for the entire region is obtained by applying the above decision rule for each pixel within the occlusion area and deciding based on a majority rule.

If, based on this decision, we find that the estimated motion vector is invalid, it does not imply that the region is not moving, only that its current displacement estimate is wrong. Hence, we must estimate its motion again, while denying movement to the same location. This is done by applying the hierarchical matching algorithm again, while considering only displacements that do not coincide with the occlusion area. After a new motion vector has been estimated, it is validated in the same manner and if it is not valid, then the new occlusion area is appended to the previous one and the entire process is repeated. This leads to an iterative motion estimation and validation scheme.

To increase the efficiency of the validation process, only regions that lie on the border of a moving object (have a common boundary with a background region) are validated, since these regions are the most likely to be affected by the occlusion problem.

Following is a summary of the hierarchical motion estimation and validation scheme.

1) Estimate the motion for all the regions that were classified as foreground candidates in the initial classification using hierarchical region matching.
2) For each region R_i that lies on the boundary of a moving object and its estimated motion vector $(\Delta x_i, \Delta y_i) \neq (0,0)$.
   a) Perform a hypothesis test on each pixel in the occlusion area using (21).
   b) If the majority of the pixels in the occlusion area are valid, then the estimated motion vector for the region is valid.
   c) Otherwise, mark the occlusion area and apply the hierarchical region matching algorithm again, while avoiding displacements that coincide with the marked area.
   d) Go back to (a).

The performance of the suggested motion estimation and validation scheme is demonstrated in Fig. 6 for the test sequence *Foreman*. Notice that this sequence exhibits global motion due to a moving camera, thus global motion compensation was applied before estimating the local motion. In the example shown, the person's head is moving to the left, leading to occlusion in the areas left to the head. The effect of the occlusion can be seen in Fig. 6(a) and (b), which displays the estimated motion before the validation. We can see that many regions on the left side of the head were mistakenly detected as moving (moving regions are marked by white). The estimated motion after the validation scheme is displayed in Fig. 6(c) and (d). The motion vectors of regions that were mistakenly estimated as moving were corrected, resulting in a more accurate estimate of the motion in the scene.

C. MRF-Based Classification

1) MRFs on Graphs: Without doubt, the most prominent stochastic models in image processing and computer vision are based on Markov processes. Due to their ability to capture the spatial continuity that is inherent in natural images, MRFs have been used extensively in the past years for the solution of many problems in these fields. Applications of MRFs range from low-level vision tasks like image restoration [18] and image segmentation [19]–[21], through mid-level vision problems such

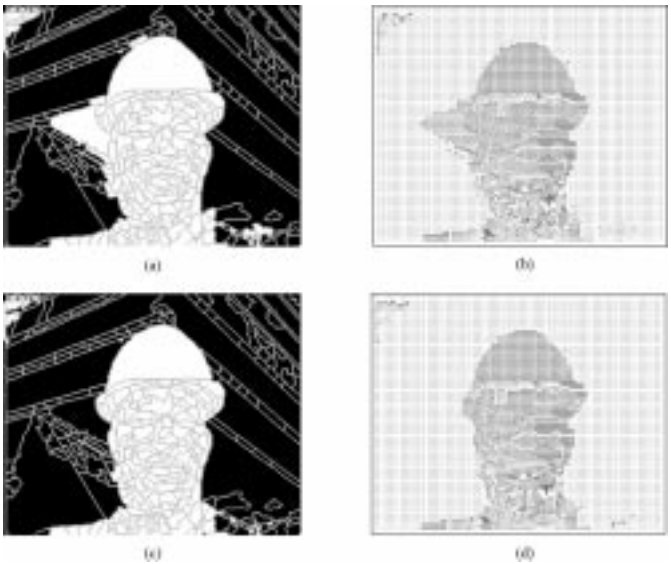


Fig. 6. Estimated motion between frames 22 and 25 of the test sequence *Foreman*: moving regions and corresponding motion field. (a), (b) Before motion validation. (c), (d) After motion validation. Moving regions are marked by white.

as texture classification [22] and change detection [8], [16], to high-level tasks such as image interpretation and understanding [23]. The main drawback of MRF models is their high computational load, since they are usually defined directly over the pixels in the image. However, we define the MRF model over the set of regions obtained in the initial partition, rather than on the rectangular lattice that composes the image. Since the number of regions is relatively small (a few hundreds), the optimization of the MRF is remarkably efficient.

Let $R = \{R_1, \ldots, R_N\}$ denote the set of regions obtained in the initial partition. Then R can be represented by a set of nodes in a connected graph, called the *region adjacency graph* (RAG), as depicted in Fig. 7. Formally stated, a RAG $G = (S, E)$ is an undirected graph such that

- $S = \{S_1, \ldots, S_N\}$ is the set of nodes in the graph, where node S_i corresponds to region R_i;
- $(S_i, S_j) \in E$ iff the corresponding regions R_i and R_j are spatially adjacent (connected).

Representation using RAGs is very common in the formulation of a segmentation problem using MRF models [20], [21], [23].

We can also define a neighborhood system on a RAG G, denoted by $\mathbf{n} = \{n(S_1), n(S_2), \ldots, n(S_N)\}$, where $n(S_i)$, $i = 1, 2, \ldots, N$, is the set of all the nodes in S that are neighbors of S_i, such that

- $S_i \notin n(S_i)$;
- if $S_j \in n(S_i)$, then $S_i \in n(S_j)$.

Let $X = \{X_1, X_2, \ldots, X_N\}$ be a set of discrete-valued random variables, where X_i is the random variable representing the label of the node S_i. The value ω_i of a random variable X_i may be any member l_j of the label set L. An assignment of values to all the variables in the random field is called a configuration and is denoted ω. The set of all possible configurations is denoted Ω.

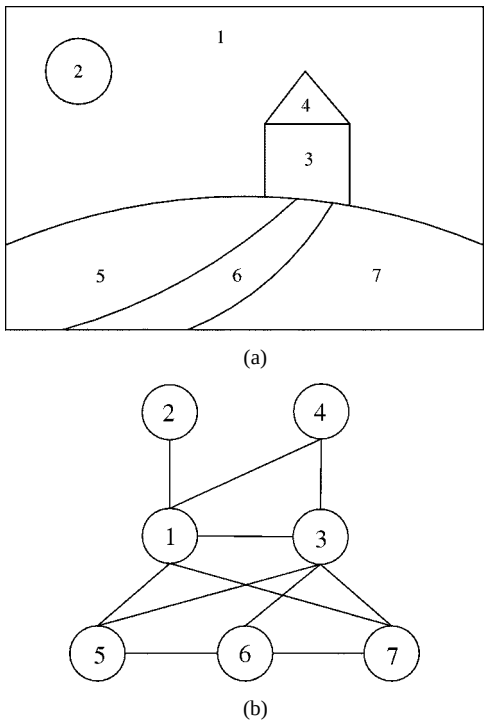


Fig. 7. Example of a RAG. (a) Segmented image and (b) its RAG.

X is *Markov Random Field* with respect to the neighborhood system $\mathbf{n}$ if and only if

$$P(X = \omega) > 0, \forall \omega \in \Omega \tag{22}$$

$$P\Big(X_i = \omega_i | X_j = \omega_j, \forall j \neq i\Big) = P\Big(X_i = \omega_i | X_j = \omega_j, \forall S_j \in n(S_i)\Big). \tag{23}$$

According to (23) above, a MRF is characterized by the conditional distributions, called the *local characteristics* of the random field. An alternative characterization is given by Hammersley–Clifford theorem [24]: X is a MRF with respect to a neighborhood system $\mathbf{n}$ if and only if its joint distribution is a Gibbs distribution

$$P(X = \omega) = \frac{1}{Z} e^{-U(\omega)} \tag{24}$$

where $U(\omega)$, the *energy function*, is given by

$$U(\omega) = \sum_{c \in C} V_c(\omega) \tag{25}$$

and the term Z, often called the *partition function*, is given by

$$Z = \sum_{\omega} e^{-U(\omega)}. \tag{26}$$

C is the set of all cliques in the graph G. The partition function Z is simply a normalizing constant, so that the sum of the probabilities of all realizations ω, add to unity. The functions $V_c(\omega)$ are called the *clique potentials*. The only condition on $V_c(\omega)$ is that it depends only on the nodes within the clique c. Clique functions provide a mechanism to express soft constraints between labels at neighboring variables.

In order to obtain a classification of the regions, we wish to take into account not only the prior probabilities of particular configurations, but also external evidence. Suppose that we have an observation $O = \{O_1, O_2, \ldots, O_N\}$ where O_i is a set of features for the site (region) S_i. We are interested in the configuration $\hat{\omega}$ for the MRF X that induced this observed set of features O. The optimal estimator (minimum probability of error estimate) is the *maximum a posteriori* (MAP) estimator

$$\hat{\omega} = \arg\max_{\omega} P(\omega|O). \tag{27}$$

From Bayes' rule, we know that

$$P(\omega|O) = \frac{P(O|\omega)P(\omega)}{P(O)}. \tag{28}$$

Assuming the likelihoods $P(O_i|X_i)$ are local and spatially distinct, it is reasonable to assume that they are conditionally independent. That is

$$P(O|\omega) = \prod_{i=1}^{N} P\left(O_i|\omega_i\right). \tag{29}$$

Now, under this assumption and using the Hammersley-Clifford theorem, it can be shown [25] that the *a posteriori* probability in (28) follows a Gibbs distribution

$$P(\omega|O) = \frac{e^{-U_p(\omega|O)}}{Z_p}. \tag{30}$$

So the maximum *a posteriori* estimate $\hat{\omega}$ is obtained by minimizing the posterior energy function $U_p(\omega|O)$.

2) Region Classification Based on MRF Model: To define a MRF model, we use the set of observations $O = \{O_1, \ldots, O_N\}$, where the observation O_i for the region R_i is defined as $O_i = \{MV_i, A_i, M_i\}$, where

- $MV_i = (\Delta x_i, \Delta y_i)$ is the estimated motion vector of the region R_i;
- A_i is the sum of the average intensity values of the three color components Y, C_r and C_b within the region R_i, as defined in (5);
- M_i is the average value of the memory within the region R_i (the memory will be explained in detail in Section IV-D).

In addition, the label set is defined as $L = \{F, B\}$, where F denotes foreground and B denotes background.We define the energy function $U_p(\omega|O)$ as a composition of three terms

$$U_p(\omega|O) = \sum_{i=1}^{N} V_i^M(\omega, O) + V_i^T(\omega, O) + \sum_{(i,j)\in E} V_{ij}^S(\omega, O) \tag{31}$$

where, for practical reasons, only singleton and pairwise cliques are considered.

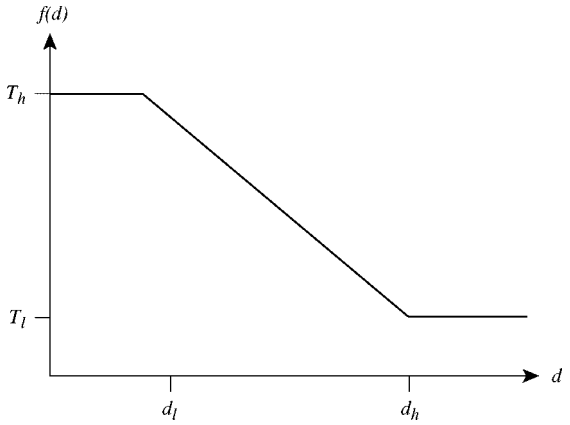


Fig. 8. Similarity function $f(d)$.

- The first term $V_i^M(\omega, O)$ is the motion term, which represents the likelihood of the region R_i to be classified as foreground/background, based on its estimated motion

$$V_i^M(\omega, O) = \begin{cases} -\alpha N_i, & (\omega_i = F \text{ and } MV_i \neq 0) \text{ or} \\ & (\omega_i = B \text{ and } MV_i = 0) \\ \alpha N_i, & (\omega_i = F \text{ and } MV_i = 0) \text{ or} \\ & (\omega_i = B \text{ and } MV_i \neq 0) \end{cases} \tag{32}$$

  where N_i is the size (number of pixels) of the region R_i. The motion term simply states that moving regions should be classified as foreground, whereas static regions should be classified as background. Note that the magnitude of the motion vector is not taken into consideration, only whether it is different from zero. This ensures that the classification process is not biased by incorrect motion vectors with large magnitudes.
- The second term $V_i^T(\omega, O)$ is a temporal continuity term

$$V_i^T(\omega, O) = \begin{cases} -\beta \cdot M_i \cdot N_i, & \omega_i = F \\ \beta \cdot M_i \cdot N_i, & \omega_i = B. \end{cases} \tag{33}$$

  The temporal continuity term allows us to consider the segmentation of prior frames, thus maintaining the coherency of the segmentation through time. If a region has been classified as foreground several times in the past, its memory value will be high and it is likely to be classified as foreground again (the memory mechanism is explained in Section IV-D). This will allow us to maintain an accurate segmentation, even when parts of the object stop moving for long periods of time.
- The last term $V_{ij}^S(\omega, O)$ is a spatial continuity term

$$V_{ij}^S(\omega, O) = \begin{cases} -\gamma_F \cdot f\left(A_i - A_j\right) N_{ij}, & \omega_i = \omega_j = F \\ -\gamma_B \cdot f\left(A_i - A_j\right) N_{ij}, & \omega_i = \omega_j = B \\ \gamma_{diff} \cdot f\left(A_i - A_j\right) N_{ij}, & \omega_i \neq \omega_j \end{cases} \tag{34}$$

  where N_{ij} is the length of the common boundary between R_i and R_j and the function $f(d)$ is given by

$$f(d) = T_l - \frac{T_h - T_l}{d_h - d_l}\left(d - d_l\right) \tag{35}$$

  as depicted in Fig. 8.

Fig. 9. Frames: (a) 10, (b) 30, and (c) 40 of the test sequence *Foreman*, respectively. (e), (d), and (f) Corresponding memory status. Bright regions indicate high memory values.

The spatial continuity term expresses the relationships between pairs of regions. A similarity measure is used in order to incorporate the spatial properties of the regions into the optimization process. Specifically, two regions with similar spatial properties are likely to belong to the same moving object, thus more weight is given to an equal labeling for the two regions. Specification of the values T_h, T_l, d_h, and d_l in the function $f(d)$ will determine the effect of neighboring regions on one another. In our experiments, we have used $T_h = 2$, $T_l = 0.5$, $d_h = 80$, $d_l = 20$. This implies that similar regions (whose average intensities differ by no more than 20) have a similarity coefficient 2 (their mutual effect is doubled). Dissimilar regions (whose average intensities differ by more than 80) have a similarity coefficient 0.5 (their mutual effect is halved). For regions whose intensity differences range from 20 to 80, the similarity coefficient varies linearly from 2 to 0.5, according to $f(d)$ in (35).

The constants α, β, γ_F, γ_B and γ_{diff} determine the relative contributions of the three terms to the energy functions. Experimentally, we have found that using $\alpha = 1.75$, $\beta = 0.75$ and $\gamma_F = \gamma_B = \gamma_{diff} = 5$ leads to satisfactory results.

The minimization of the energy function $U_p(\omega|O)$ is performed using an iterative deterministic relaxation scheme known as HCF, which was presented in [12]. Our implementation uses a slightly modified version of HCF. Rather than starting with an all-uncommitted configuration, we set the initial configuration based on the initial classification phase. Specifically, all regions that were detected as foreground candidates are set to the uncommitted state, while all regions that were classified as background are set to the background state. This modification increases the computational efficiency of the algorithm, since the number of uncommitted sites is reduced, especially when the moving objects are small compared to the background.

D. Object Tracking and Memory Update

To impose temporal coherency on the segmentation process, we incorporate a memory into the algorithm. Rather than using a static memory, as was suggested in [4], we use a dynamic memory, based on region tracking. This memory will contain, for each region, the number of times it was classified as foreground in past frames. However, unlike the static memory in [4], the update of the dynamic memory consists of tracking each region to its new location in the frame, using the displacement vector estimated in the hierarchical motion estimation phase. This enables us to track objects as they move throughout the sequence, without accumulating uncovered background in the process. Error propagation is avoided by slowly decreasing memory values of regions that stop moving. Thus, if a background region was mistakenly detected as moving once during the sequence, its effect is diminished through time.

Let MEM_k denote the memory in the kth frame. The tracking memory is updated based on the following scheme:

- Initialization: $\forall(x,y),\ \text{MEM}_0(x,y) = 0$;
- For each region R_i, $i = 1, \ldots, N$

— If $(\Delta x_i, \Delta y_i) \neq (0,0)$ and $\omega_i = F$ (moving region), then

$$\forall(x,y) \in R_i,\ \text{MEM}_{k+1}(x + \Delta x_i, y + \Delta y_i) = \text{MEM}_k(x,y) + 1.$$

— Otherwise

$$\forall(x,y) \in R_i,\ \text{MEM}_{k+1}(x,y) = \text{MEM}_k(x,y) - 0.25$$

where k and $k+1$ are the indices of the current frame and the next frame, respectively and ω is the labeling obtained by the MRF.

We update the memory each frame by setting the memory value of all the pixels each region to the average value of the pixels inside that region.

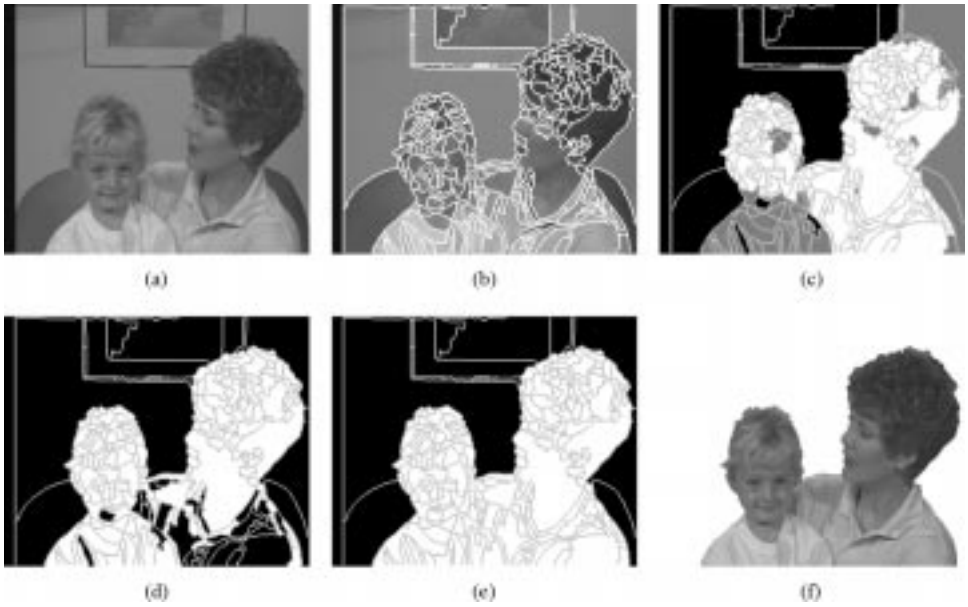


Fig. 10. Segmentation of the 43rd frame of *Mother & Daughter*. (a) Original frame. (b) Initial partition. (c) Initial classification. (d) Moving regions. (e) MRF classification. (f) Segmentation mask.

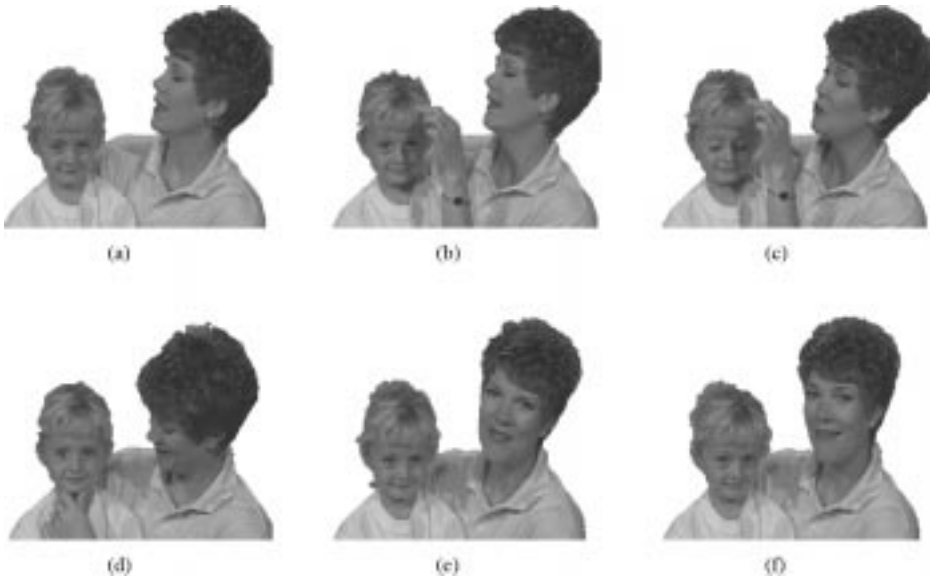


Fig. 11. Segmentation masks for the sequence *Mother & Daughter*. Frames: (a) 55; (b) 91; (c) 127; (d) 169; (e) 211; (f) 292.

Of course, the spatial segmentation may be inconsistent throught time, i.e., regions which correspond to the same object may be segmented differently in consecutive frames, resulting in incoherent memory values. In order to maintain a constant memory value for each region and to avoid leaving residues due to inaccuracies in the tracking mechanism, a post-processing step is employed following the segmentation of the next frame. In this step, a constant memory value for each region is determined by averaging the memory values over the entire region. If the majority of values is zero, then all memory values in this region are set to zero, otherwise, all memory values are set to the average value of the memory in this region.

Finally, note that in case of camera motion, the memory must be adapted to the global motion as well, in order to maintain an accurate tracking. Therefore, the memory is warped to the next frame in the global motion compensation phase, based on the estimated global motion parameters.

Fig. 9 illustrates the use of the memory for the test sequence *Foreman*. As we can see, the memory successfully tracked the moving object (the person's head), without marking uncovered background as moving.

V. Experimental Results

The proposed algorithm for VOP segmentation was experimentally investigated by means of computer simulations. The sequences *Mother & Daughter* (10 Hz), *Silent* (10 Hz), *Foreman* (30 Hz), *Table Tennis* (30 Hz), and *Stefan* (30 Hz) were used in CIF format (352×288 pixels).

The sequence *Mother & Daughter* is a typical video-conference scene that exhibits slow and smooth motion over a sta-

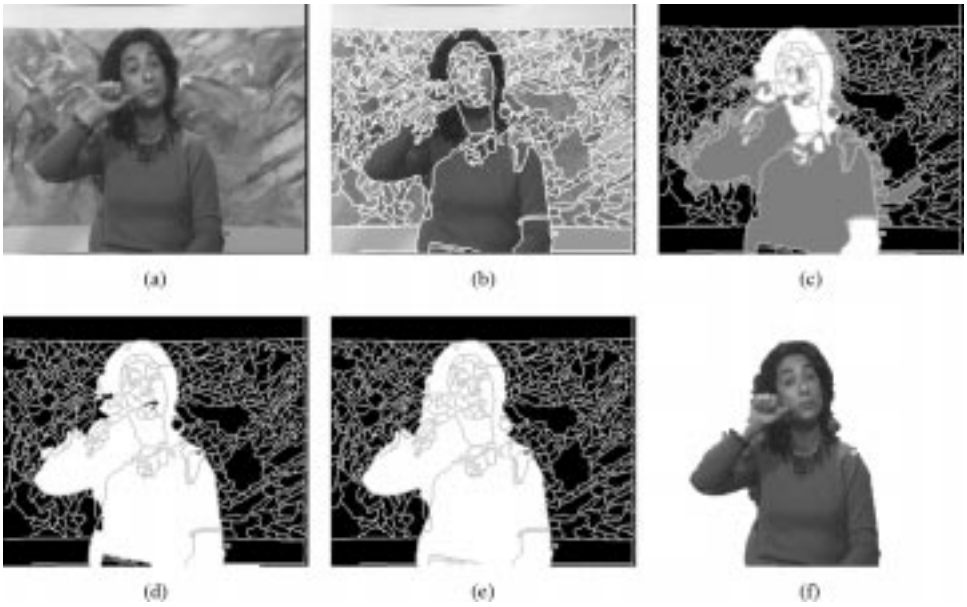


Fig. 12. Segmentation of the 49th frame of *Silent*. (a) Original frame. (b) Initial partition. (c) Initial classification. (d) Moving regions. (e) MRF classification. (f) segmentation mask.

Fig. 13. Segmentation masks for the sequence *Silent*. Frames: (a) 55; (b) 91; (c) 118; (d) 136; (e) 175; and (f) 181.

tionary background. The segmentation process of frame 43 of the sequence is illustrated in Fig. 10. The initial partition in Fig. 10(b), which is slightly oversegmented, is accurate enough to obtain a reliable segmentation. Fig. 10(c) shows the result of the initial classification, where white areas correspond to regions detected by the memory and gray areas correspond to regions detected based on the significance test. Moving regions, as detected by the hierarchical motion estimation and validation scheme, are depicted in Fig. 10(d). As we can see in Fig. 10(e), the MRF-based classification provided a perfect labeling, resulting in an accurate segmentation mask [Fig. 10(f)]. Segmentation masks for several other frames in the sequence are shown in Fig. 11.

Silent is another typical video-conference scene. Yet, unlike *Mother & Daughter*, this sequence exhibits a combination of rapid, nonrigid motion (the woman's hands), along with slow motion (the woman's body), over a textured background. The segmentation process of frame 49 is shown in Fig. 12. As we can see in Fig. 12(b), the initial partition is oversegmented, due to the textured background of the scene. Yet, most of the background regions were eliminated already in the initial classification [Fig. 12(c)]. The obtained segmentation mask shown in Fig. 12(f) is quite satisfactory, except for the woman's right hand, which was not detected accurately. VOPs extracted from several other frames in this sequence are depicted in Fig. 13. Despite the fast movement of the woman's hands, the moving object was tracked successfully throughout the sequence.

The sequence *Foreman* is another "head and shoulders" scene. Yet, unlike the previous sequences, *Foreman* exhibits a moving camera. Moreover, the background in this sequence is not planar, which may lead to difficulties in the global motion compensation. In addition, we note that the boundaries of the

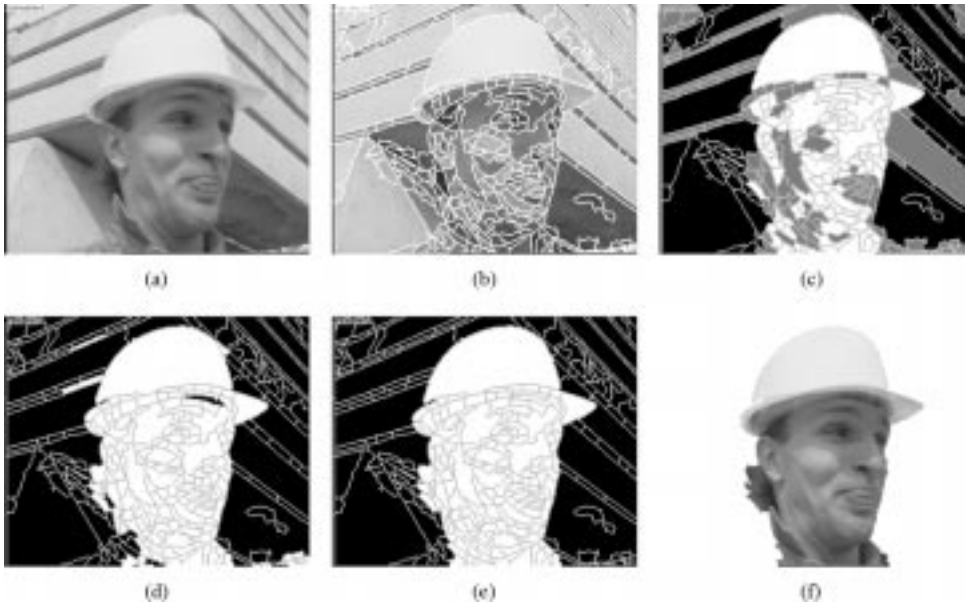


Fig. 14. Segmentation of the 7th frame of *Foreman*. (a) Original frame. (b) Initial partition. (c) Initial classification. (d) Moving regions. (e) MRF classification. (f) Segmentation mask.

Fig. 15. Segmentation masks for the sequence *Foreman*. Frames: (a) 11; (b) 24; (c) 39; (d) 71; (e) 90; and (f) frame 120.

moving object, in particular the person's hat, are not clearly distinct. The segmentation process of the seventh frame of *Foreman* is shown in Fig. 14. The initial partition obtained in Fig. 14(b) is very accurate. This is attributed to the use of color information in the watershed segmentation. Fig. 14(f) displays the resulting segmentation mask, where only the person's right shoulder was not detected, since it lacks sufficient motion. Segmentation masks obtained for subsequent frames in the sequence are shown in Fig. 15.

Table Tennis is a very dynamic sequence that exhibits fast global and local motion, as well as a cluttered and textured background. Moreover, the background is not planar in most of the sequence, which may lead to difficulties in the global motion compensation. Due to the strong motion of the moving objects in the scene, they are easily separated from the background, leading to a very accurate segmentation, as seen in Fig. 16 (the segmentation masks in this sequence are depicted over a gray background, to emphasize the moving objects). Segmentation masks for subsequent frames in the sequence are depicted in Fig. 17. As we can see in Fig. 17(b) and (c), the tennis table was classified as foreground, even though it is not moving. This is due to the fact that the table violates the assumption of a planar background and cannot be incorporated into the global motion model. Thus, it is detected as moving in the motion estimation phase and is subsequently classified as foreground. Finally, note that a scene-cut occurred between frames 131 and 132, thus the segmentation algorithm was reset in frame 132.

The last sequence that we will consider, and perhaps the most challenging, is the sequence *Stefan*. This sequence displays a scene from a tennis match, with a mobile camera. The back-

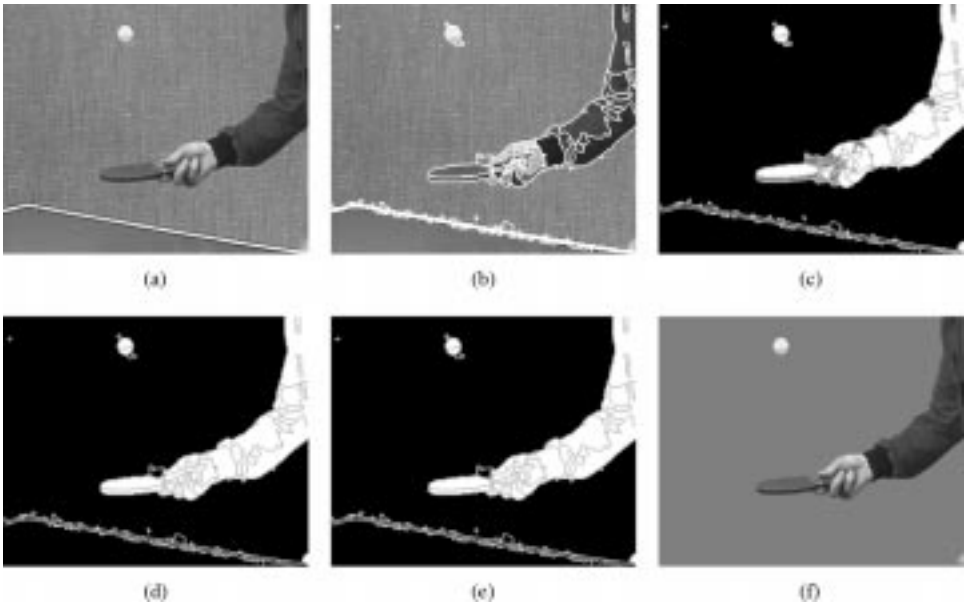


Fig. 16. Segmentation of the 4th frame of *Table Tennis*. (a) Original frame. (b) Initial partition. (c) Initial classification. (d) Moving regions. (e) MRF classification. (f) Segmentation mask.

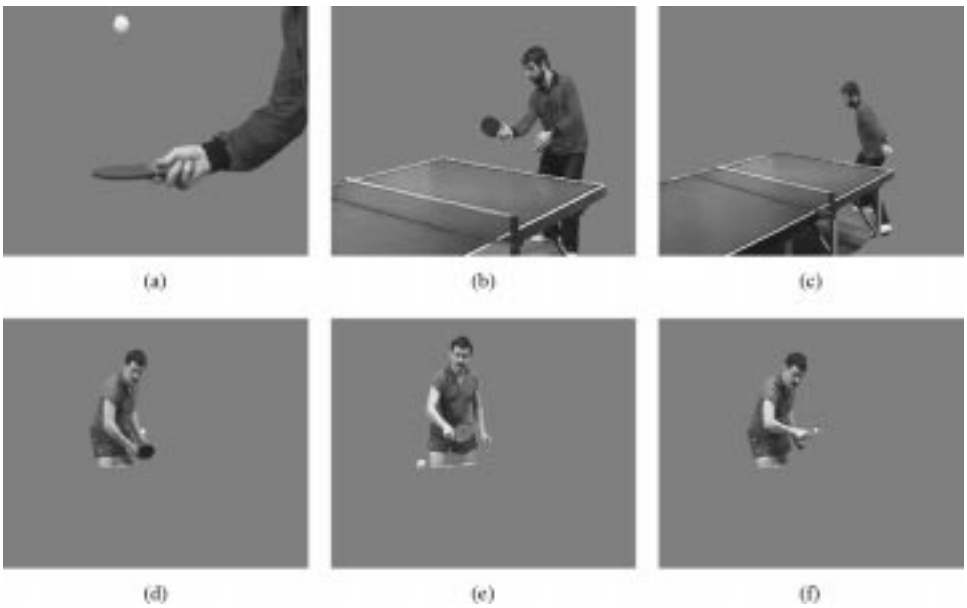


Fig. 17. Segmentation masks for the sequence *Table Tennis*. Frames: (a) 18; (b) 84; (c) 126; (d) 143; (e) 232; (f) 296.

ground in this sequence is very cluttered and the moving object exhibits fast, nonrigid motion. The segmentation process for the first frame in this sequence is shown in Fig. 18. Due to the cluttered background, the initial partition is severely oversegmented. Moreover, the initial classification marked many of the background regions as potential foreground candidates. However, the motion estimation effectively captured the local motion of the object, as seen in Fig. 18(d). The MRF-based classification eliminated some background regions, yet it also eliminated part of the moving object, namely, the player's shoes. This is contributed to the fact that the shoes are inherently different than the rest of the object and therefore considered to be part of the background. As a result, the obtained VOP is not accurate enough to allow perfect extraction of the moving object. This is also inherent in the other VOPs, shown in Fig. 19.

Finally, we note that the proposed algorithm, while still not suitable for realtime applications, is of modest computational requirements. Currently, most of the computational load lies in the global motion estimation phase. For sequences with a stationary background (without application of the GME phase), the execution time of the algorithm is about 2 s for each frame in CIF format. For sequences with a moving background, execution times are in the range of 5–10 s for each frame. The experiments were carried out on a Pentium III 500-MHz workstation.

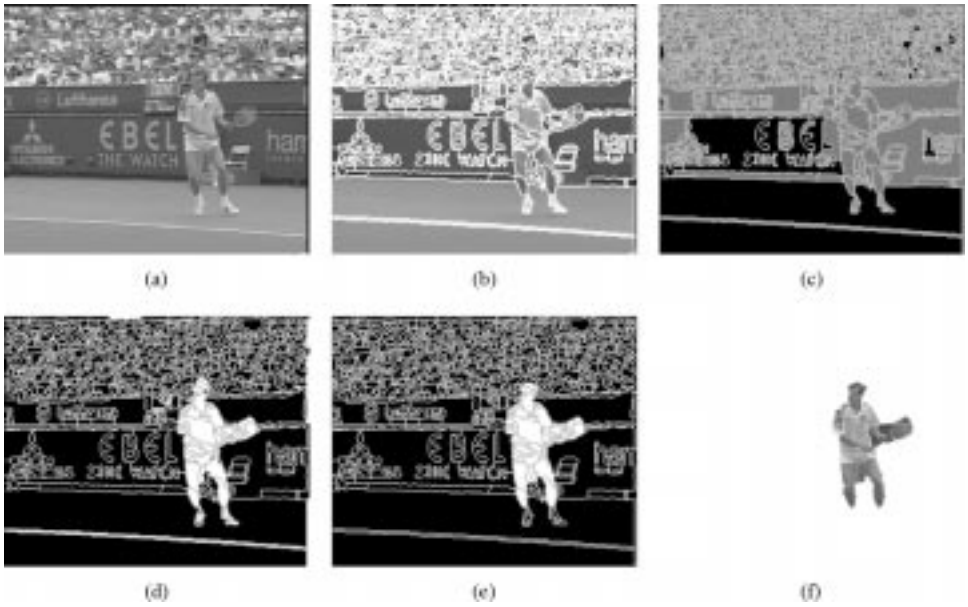


Fig. 18. Segmentation of the 1st frame of *Stefan*. (a) Original frame. (b) Initial partition. (c) Initial classification. (d) Moving regions. (e) MRF classification. (f) Segmentation mask.

Fig. 19. Segmentation masks for the sequence *Stefan*. Frames: (a) 25; (b) 53; (c) 82; (d) 92; (e) 125; (f) 142.

VI. Discussion and Conclusions

This paper introduces a new algorithm for automatic segmentation of moving objects in image sequences for VOP generation. The algorithm is based on a MRF model defined over a region adjacency graph, which uses motion information to classify regions as foreground or background. The location of object boundaries is guided by the initial partition, which consists of a color-based watershed segmentation. Therefore, the proposed technique succeeds in locating objects boundaries that are not clearly distinct, where other techniques fail. A hierarchical motion estimation and validation scheme detects moving regions in the scene, while avoiding misdetections caused by the occlusion problem. A tracking memory ensures that a reliable segmentation is maintained throughout the sequence, even when the objects stop moving. In addition, the tracking memory also accommodates sequences with rapidly moving objects, without marking uncovered background as foreground. The memory contents are incorporated into a MRF model, along with motion information and spatial information, to obtain a spatio-temporal segmentation of the scene.

Experimental results demonstrated that our proposed technique can successfully extract moving objects from various sequences, with stationary or mobile camera. Nevertheless, the boundaries of the extracted objects are not always accurate enough to place them in different scenes, which requires a nearly perfect boundary location. Furthermore, in the case of insufficient motion, the algorithm converges to the correct segmentation only after several frames. However, the VOPs obtained by our proposed technique could be used to provide other content-based functionalities, such as content-based scalability.

Currently, automatic segmentation of video sequences remains an unsolved problem, since none of the proposed tech-

niques can accomplish this task for generic video sequences. This is mainly due to the fact that VOPs cannot be characterized by homogeneous low-level features such as color, texture or motion. The key to developing segmentation techniques that achieve the performance of the human visual system is to incorporate higher level information into the segmentation process.

Future work should concentrate on incorporating temporal information in the form of change or motion into the initial partition, rather than relying only on spatial information. This way, the number of resulting regions can be reduced dramatically, while maintaining the structural integrity of moving objects in the scene. For instance, the entire background can be segmented as one region, while only moving objects are partitioned to smaller segments. This will surely reduce the computational load of the algorithm, while increasing its robustness at the same time.

In addition, further work should be put into the global motion estimation and compensation phase. As we saw in Section V, the current technique does not handle scenes in which the background cannot be considered planar in a satisfactory manner. Moreover, most of the computational burden of the algorithm currently lies within the global motion estimation phase. Therefore, other techniques should be investigated in order to overcome these drawbacks.

REFERENCES

[1] *MPEG-4 Video Verification Model Version 15.0*, ISO/IEC JTC1/SC29/WG11 N3093, 1999.

[2] *Information Technology—Coding of Moving Pictures and Associated Audio for Digital Storage Media at up to About 1.5 mbit/s*, ISO/IEC 11 172, 1993.

[3] *Information Technology—Generic Coding of Moving Pictures and Associated Audio Information*, ISO/IEC 13 818, 1994.

[4] R. Mech and M. Wollborn, "A noise robust method for 2D shape estimation of moving objects in video sequences considering a moving camera," *Signal Processing*, vol. 66, no. 2, pp. 203–217, Apr. 1998.

[5] A. Neri, S. Colonnese, G. Russo, and P. Talone, "Automatic moving object and background separation," *Signal Processing*, vol. 66, no. 2, pp. 219–232, Apr. 1998.

[6] J. G. Choi, M. Kim, M. H. Lee, and C. Ahn, "Automatic segmentation based on spatio-temporal information," ISO/IEC JTC1/SC29/WG11 MPEG97/m2091, Bristol, U.K., 1997.

[7] M. Kim, J. G. Choi, D. Kim, H. Lee, M. H. Lee, C. Ahn, and Y. Ho, "A VOP generation tool: Automatic segmentation of moving objects in image sequences based on spatio-temporal information," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 9, pp. 1216–1226, Dec. 1999.

[8] N. Paragios and G. Tziritas, "Adaptive detection and localization of moving objects in image sequences," *Signal Processing: Image Commun.*, vol. 14, pp. 277–296, Feb. 1999.

[9] F. Dufaux and J. Konrad, "Efficient, robust and fast global motion estimation for video coding," *IEEE Trans. Image Processing*, vol. 9, pp. 497–500, 2000.

[10] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-8, pp. 679–698, Nov. 1986.

[11] P. De Smet and D. De Vleeschauwer, "Performance and scalability of a highly optimized rainfalling watershed algorithm," in *Proc. Int. Conf. Imaging Science, Systems and Technology*, vol. CISST 98, Las Vegas, NV, July 1998, pp. 266–273.

[12] P. B. Chou and C. M. Brown, "The theory and practice of Bayesian image labeling," *Int. J. Comput. Vis.*, vol. 4, pp. 185–210, 1990.

[13] Y. Tsaig, "Segmentation of Moving Objects in Video Sequences," M.Sc. thesis, Tel-Aviv University, Tel-Aviv, Israel, 2001.

[14] P. De Smet, R. Pires, and D. De Vleeschauwer, "Activity driven nonlinear diffusion for color image segmentation," in *Noblesse Workshop on Non-linear Model Based Image Analysis, NMBIA'98*, Glasgow, U.K., July 1998, pp. 183–187.

[15] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 583–598, 1991.

[16] T. Aach, A. Kaup, and R. Mester, "Statistical model-based change detection in moving video," *Signal Processing*, vol. 31, no. 2, pp. 165–180, 1993.

[17] J. R. Bergen, P. Anandan, K. J. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Proc. Eur. Conf. Computer Vision, ECCV'92*, Santa Margherita Ligure, Italy, May 1992, pp. 237–252.

[18] S. Geman and D. Geman, "Stochastic relaxation, gibbs distributions and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. PAMI-6, pp. 721–741, Nov. 1984.

[19] T. Meier, K. N. Ngan, and G. Crebbin, "A robust Markovian segmentation based on highest confidence first ()," in *IEEE Int. Conf. Image Processing (ICIP'97)*, vol. 1, Santa Barbara, CA, Oct. 1997, pp. 216–219.

[20] I. Y. Kim and H. S. Yang, "A systematic way for region-based image segmentation based on Markov random field model," *Pattern Recognit. Lett.*, vol. 15, pp. 969–976, Oct. 1994.

[21] A. Sarkar, M. K. Biswas, and K. M. S. Sharma, "A simple unsupervised MRF model based image segmentation approach," *IEEE Trans. Image Processing*, vol. 9, pp. 801–812, May 2000.

[22] H. Derin and W. S. Cole, "Segmentation of textured images using Gibbs random fields," *Comput. Vis. Graph. Image Process.*, vol. 35, pp. 72–98, 1986.

[23] J. W. Modestino and J. Zhang, "A Markov random field model-based approach to image interpretation," in *Markov Random Fields: Theory and Applications*, R. Chellappa and A. Jain, Eds. New York: Academic, 1993, pp. 369–408.

[24] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *J. Roy. Statist. Soc. B*, vol. 36, no. 2, pp. 192–236, 1974.

[25] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *J. Appl. Statist.*, vol. 16, pp. 131–164, 1989.

Yaakov Tsaig received the B.Sc. degree in electrical engineering and computer science and the M.Sc. degree in computer science from Tel Aviv University, Israel, in 2000 and 2001, respectively.

His research interests include digital image processing, image segmentation, computer vision, object tracking, and image and video coding.

Amir Averbuch was born in Tel Aviv, Israel. He received the B.Sc and M.Sc degrees in mathematics from the Hebrew University, Jerusalem, Israel, in 1971 and 1975, respectively. He received the Ph.D degree in computer science from Columbia University, New York, NY, in 1983.

From 1966 to 1970 and 1973 to 1976, he served in the Israeli Defense Forces. From 1976 to 1986, he was a Research Staff Member with IBM T.J. Watson Research Center, Yorktown Heights, NY. In 1987, he joined the Deptartment of Computer Science, School of Mathematical Sciences, Tel Aviv University, where he is currently a Professor of computer science. His research interests include wavelet and multiscale applications for signal/image processing and numerical computation, multiresolution analysis, scientific computing, fast algorithms, parallel and supercomputing (software and algorithms).