Region-Based Representations of Image and Video: Segmentation Tools for Multimedia Services

P. Salembier, *Member, IEEE*, and F. Marqués, *Member, IEEE*

(Invited Paper)

***Abstract*—This paper discusses region-based representations of image and video that are useful for multimedia services such as those supported by the MPEG-4 and MPEG-7 standards. Classical tools related to the generation of the region-based representations are discussed. After a description of the main processing steps and the corresponding choices in terms of feature spaces, decision spaces, and decision algorithms, the state of the art in segmentation is reviewed. Mainly tools useful in the context of the MPEG-4 and MPEG-7 standard are discussed. The review is structured around the strategies used by the algorithms (transition based or homogeneity based) and the decision spaces (spatial, spatio-temporal, and temporal).**

The second part of this paper proposes a partition tree representation of images and introduces a processing strategy that involves a *similarity estimation* step followed by a *partition creation* step. This strategy tries to find a compromise between what can be done in a systematic and universal way and what has to be application dependent. It is shown in particular how a single partition tree created with an extremely simple similarity feature can support a large number of segmentation applications: spatial segmentation, motion estimation, region-based coding, semantic object extraction, and region-based retrieval.

***Index Terms*— Compression, indexing, motion estimation, MPEG-4, MPEG-7, object tracking, partition tree, regions, shot detection, spatial and temporal segmentation, video objects.**

I. Introduction

DATA and signal modeling for images and video sequences is experiencing important developments. Part of this evolution is due to the need to support a large number of new multimedia services. Traditionally, digital images were represented as rectangular arrays of pixels and digital video was seen as a flow of frames. New multimedia applications and services imply a representation that is closer to the real word or, at least, that takes into account part of the process that has created the digital information. Let us mention two examples.

- Applications supported by the MPEG-4 standard [38], [39] constitute a typical case: the representation based on an array of pixels is not appropriate if one wants to be able to interact with objects in the image, to encode differently the areas of interest, or to assign different behaviors to the entities represented in the image. In these applications, the notion of object is essential. As a consequence, the data modeling has to be modified and, for example, has to include regions of arbitrary shapes to represent objects. In the following, we make a distinction between an object, which is the visual two-dimensional (2-D) representation of an entity that has a semantic meaning and a region, which is a connected component of the space that is generally defined by a homogeneity criterion. An object may be represented by the union of several regions that may be connected or not.
- The MPEG-7 standard [40], [41] is also facing the same kind of challenges. For instance, the video representation based on a flow of frames is inadequate for a large number of video indexing applications. Among the large set of functionalities involved in a retrieval application, let us consider browsing. The browsing functionality should go far beyond the "fast forward" and "fast reverse" allowed by VCR's. One would like to have access to a table of contents of the video and to be able to jump from one item to another. This kind of functionality implies at least a structuring of the video in terms of individual shots and scenes. Of course, indexing and retrieval involve also a structuring of the data in terms of objects, regions, semantic notions, etc.

In both examples, the new data modeling has to take into account part of the creation process: an image is created by projection of a visual scene composed of three-dimensional (3-D) objects into a 2-D plane. Modeling the image in terms of regions is an attempt to know the projection of the 3-D object boundaries in the 2-D plane. Detecting shots in a video aims as well at finding what has been done during the video editing process. Note that, in both cases, an important goal of the data modeling is to define spatial or temporal transitions and discontinuities in a signal which was traditionally seen as a whole. In the sequel, we will refer to this problem as *segmentation*.

Segmentation can be an extremely easy task if one has access to the production process that has created the discontinuities. For example, the generation of a synthetic image or of a synthetic video implies the modeling of the 3-D world and of its temporal evolution. During the creation itself, it is very easy to recover and store the 2-D boundaries of the various objects. Another example is video editing, which creates a

Manuscript received October 1998; revised July 1999. This work was supported by the European Commission under the ACTS MoMuSys and DICEMAN projects. This paper was recommended by Guest Editor F. Pereira.

The authors are with the Department of Signal Theory and Communication, Universitat Politècnica de Catalunya, Barcelona, Spain (e-mail: philippe@gps.tsc.upc.es; ferran@gps.tsc.upc.es).

Publisher Item Identifier S 1051-8215(99)09581-6.

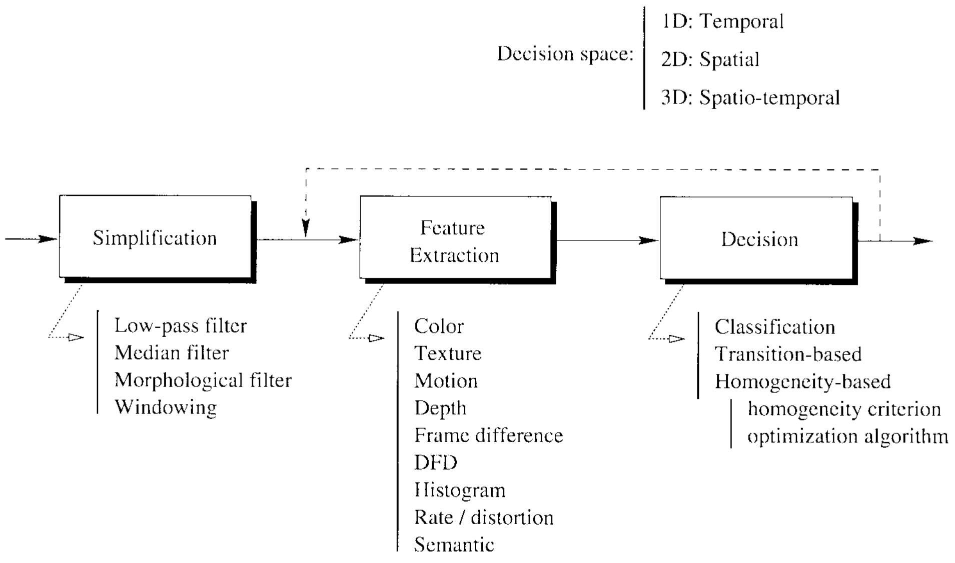

Fig. 1. Major segmentation steps and related choices.

large number of discontinuities either in space or in time. Spatial discontinuities are created by combining foreground objects that have been filmed over a blue screen with a background sequence that has been taken independently. Temporal transitions are produced by cutting and concatenating rushes. In both cases, the discontinuities detection is trivial if one has access to the information at this level of the production.

However, if the segmentation intents to estimate what has been done during the production process, its task is extremely difficult and one has to recognize that the state of the art has still to be improved to lead to robust segmentation algorithms able to deal with generic images and video sequences. The goal of this paper is to review the state of the art in tools creating region-based representations of images and video (segmentation algorithms), to show how region-based representations may be useful in the context of the MPEG-4 and MPEG-7 standards[1] and to discuss a region-based representation, called a partition tree, that tries to find a compromise between what can be done in a systematic and universal way and what has to be application dependent.

For multimedia services, segmentation may address very different goals. It includes, in particular, object detection for coding (MPEG-4) or description (MPEG-7), estimation of partitions allowing an efficient encoding, temporal tracking of regions, or shot-cut detection. The specific characteristics that have to be considered are:

1) most of the time the algorithm should be able to deal with generic images and sequences;
2) in the case of video, the amount of data to process is very large;
3) the algorithm should be efficient in terms of computational complexity and memory requirements.

[1] Note that segmentation is vital for a large number of applications supported by the MPEG-4 and the MPEG-7 standards. However, since it has no influence on the interoperability between systems, it is not a normative part of the standards.

The organization of this paper is as follows. Section II defines the main steps involved in a segmentation process and introduces the terminology that will be used. Section III reviews the state of the art on segmentation tools related to MPEG-4 and MPEG-7 applications. Section IV proposes a partition tree representation of images and a segmentation strategy that differentiates what can be done in a universal way from what has really to be matched to the application and the segmentation goal. Conclusions are given in Section V.

II. Segmentation Strategies

The word "segmentation" has a meaning that depends to a large extent on the application and the context in which it is used. The basic goal of any segmentation algorithm is to define a partition of the space. In the context of image and video, the space can be temporal [one-dimensional (1-D)], spatial (2-D), or spatio-temporal (3-D). In the following, this space is called the *decision space*. This section reviews the main steps involved in a segmentation algorithm and the main choices that have to be done. Moreover, the terminology is introduced. A general scheme for segmentation can be seen as the concatenation of three major steps represented in Fig. 1: simplification, feature extraction, and decision.

- *Simplification:* Most of the time, the original data in an image or in a video sequence contain information that is irrelevant for a given application. In such cases, data should be simplified by removing (e.g., filtering) irrelevant information. The simplification controls the amount and nature of the information that is preserved. Furthermore, the simplified data should contain areas that are easier to segment. For instance, simplification can reduce the complexity of textured areas or remove details smaller than a given size. Typical filtering tools are listed below the "Simplification" block in Fig. 1. The simplification should not modify the boundary information that is relevant for the application.

- *Feature extraction:* Segmentation is performed relying on specific features of the data. The selection of the *feature space* drives the type of homogeneity that is expected in the final partition. In some applications, the original data directly provide the feature space necessary for segmentation. For example, for color segmentation, the pixel values can directly correspond to the feature of interest. However, in a large number of cases, the features of interest have to be estimated from the original data. Typical features are listed below the "Feature Extraction" block of Fig. 1. The list includes texture, motion, depth, frame difference (FD), displaced frame difference (DFD), histograms, or even spaces characterizing some semantic notions. Note that, in some cases, the feature estimation has to be done on a region of support which should be homogeneous in terms of the same feature. As a result, a loop may be introduced in the segmentation process so that the estimation depends on the segmentation results. The final result is obtained through an iterative process.
- *Decision:* To finally obtain a partition of the data, the feature space has to be analyzed. The decision step decides on the position of the boundaries that form the partition in the decision space. Boundaries separate data areas that contain elements sharing the same characteristics in the selected feature space. For instance, in spatial segmentation, the decision may yield the precise shape of a region or, in temporal segmentation, the exact set of frames that form a shot.

In practice, three strategies may be used to define the partition: classification, transition-based, and homogeneity-based algorithms. Classification techniques start by creating a partition of the feature space and then translate this partition into a partition of the decision space. This approach has been seldom used for general purpose image or video segmentation because the topology of the regions in the decision space is not taken into account. Transition-based and homogeneity-based techniques are extensively used and will be discussed in Section III.

A *region* created by a segmentation algorithm is defined as a set of elements (pixels or images) homogeneous in the feature space and connected in the decision space. A region may not have any semantical meaning. On the contrary, an *object* is the visual 2-D representation of an entity that has a semantical meaning. An object may be formed by the union of several regions.

In order to name a specific segmentation algorithm, we will use the following terminology: first, we will define the *feature space* and then the *decision space*. For example, *motion spatial* segmentation corresponds to an algorithm that defines spatial regions (partition of a 2-D space) that are homogeneous in motion. In the case of spatial and spatio-temporal segmentation for video sequences, some approaches rely on the result obtained for the frame at time $T-1$ to segment the frame at time T [50]. They are usually based on the estimation of the motion between both frames and on the motion compensation of the previous partition information. This approach, in addition to increasing the robustness of the global segmentation method, allows creating a temporal link between representations of the same objects in consecutive frames [31]. These techniques are specified by the term *Inter* whereas the remaining approaches are *Intra* techniques. The terminology we will use is summarized by the following construction:

$$\left\{\begin{matrix}\textit{Inter}\\ /\\ \textit{Intra}\end{matrix}\right\}\left\{\begin{matrix}\textit{Feature}\\ \textit{space}\end{matrix}\right\}\left\{\begin{matrix}\textit{Decision}\\ \textit{space}\end{matrix}\right\}\textit{segmentation}.$$

Segmentation techniques often use more than one feature. This can be done either through the definition of a complex criterion combining several features or through the use of several segmentation steps that use different criteria. For example, the application of various degrees of simplification allows the analysis to be done at several levels of resolution and, at each resolution level, a specific feature space may be used. Feature space can also be very complex to define if the segmentation process allows user interaction. In these cases, the user can implicitly introduce semantic notions which might not be easily obtained by any automatic analysis of the data. As a result, it is often not possible to classify the segmentation algorithms as a function of the feature space they use.

In this paper, we analyze the main segmentation approaches for multimedia services from the viewpoint of the type of decision they use. Two classes will be discussed. The first one consists in estimating the position of the transitions that mark the separation between neighboring regions (*transition-based segmentation techniques*). This approach, although being applied to both spatial and temporal segmentation problems, has been mainly successful for the temporal case. The second approach consists in estimating the region of support of homogeneous elements (*homogeneity-based segmentation techniques*). This approach has been mostly applied to spatial and spatio-temporal segmentation.

III. State of the Art in Segmentation Tools for Generic Contents

A. Transition-Based Segmentation Techniques

1) Overview of the Decision Algorithms: In a segmentation process, the *decision* step should provide the final partition of the data. Transition-based techniques intend to estimate the position of discontinuities in the decision space. These discontinuities are evaluated in the feature space. The discontinuities are highlighted by a preprocessing that can be seen as a filtering. The filter output should contain high values in data positions close to the transition and low values in homogeneous data areas. Linear high-pass as well as nonlinear filters have been adopted.

However, the estimation of the discontinuities positions does not directly create a partition. Elements in the transition areas have values corresponding to the likelihood of being the actual position of the transition. This is due to the fact that even abrupt transitions in the scene may be represented by local smooth transitions in the image or video data and thus boundaries may not be easy to localize. For instance, estimated region transitions in spatial segmentation may have a width of various pixels, and estimated shot transitions in temporal segmentation may contain a few frames. This information has to be binarized to reduce the uncertainty in the positions of the transitions and, if necessary, thinned to define the estimated boundaries. That is, binary transitions are thinned to obtain boundary segments of width one pixel in spatial segmentation and of one frame in temporal segmentation. Thinning techniques should preserve the connectivity of the binary transitions. Several techniques for thresholding and thinning have been reported in the literature, and some of them are commented in Sections III-A2 and III-A3.

Last, the resulting boundaries may not fulfill the connectivity constraints of a real partition. This is a common problem in spatial and spatio-temporal segmentation where the estimated transitions can be unconnected and partially represent the spatial contours of regions. This problem is overcome using gap filling approaches that connect boundary segments to create final partitions. However, the gap filling process often relies on geometrical rules to connect the boundary segments. Such rules may not be generic or may not lead to the real contours of the video data. Note that this problem does not exist in temporal segmentation since the transitions are defined by a single (time) element and no connectivity is required.

The main drawback of transition-based approaches is their lack of robustness. In particular for spatial or spatio-temporal segmentation, if one element of a region contour is not detected, the entire region is merged with another region. In other words, a local error may have very significant consequences. Another weak point concerns the gap filling and thinning steps. Most of the time, they only rely on the binary geometric information obtained after thresholding. As a result, the localization of the transition after thinning or gap filling may not be accurate.

2) Transition-Based Approach to Spatial Segmentation: In spite of the previously commented drawbacks, some transition-based spatial segmentation methods have been proposed in the literature. The strong points of this type of techniques are basically two. First, in intra spatial segmentation, relevant segment boundaries can be easily detected avoiding over-segmented results. Second, in inter spatial segmentation, the tracking of object boundaries avoids the problems of assigning uncovered regions. In this section, three of these techniques are commented as representatives of different transition-based approaches and applications.

A pure transition-based gray-level spatio-temporal segmentation technique has been recently proposed in [33]. In this work, the goal is to extract and track moving objects. Images are first simplified using global motion estimation. Simplification is only applied to align the video frames in the case of a moving camera or background. Spatial boundaries are estimated directly on the grey level values by the Canny operator [6]. The final object is obtained as a blocky area containing a subset of boundaries that forms the model of the object to be tracked. The object model is created by comparing the spatial boundaries to the information obtained from the gray-level difference between consecutive images after binarization and thinning. Once the model of the object is obtained, it is tracked by comparing it to the following edge images in the sequence. The comparison is performed using the generalized Hausdorff distance. After comparison, the model is updated to compensate for rotation and changes in the shape of the object. A typical result of this technique is shown in Fig. 2. The blocky representation of the object shape [see Fig. 2(c)] does not yield its exact contours and, thus, it may not be useful for MPEG-4 applications such as video editing and manipulation. Nevertheless, this object representation may be good enough for several object-based video indexing applications [see Fig. 2(d)].

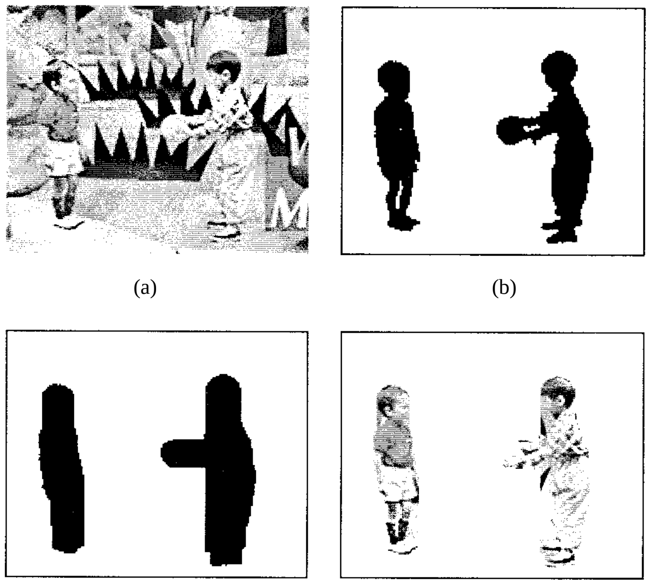

Fig. 2. Example of object representation. (a) Original image. (b) Accurate object shape representation. (c) Blocky object shape representation. (d) Blocky object representation.

A complete video analysis and retrieval system based on color and texture spatio-temporal segmentation is presented in [15]. Although the core of this technique is transition based, it cannot be said to be a pure transition-based segmentation since it uses a homogeneity-based step as well. In this approach, images are grouped and the middle frame of the group is segmented. The segmentation looks for transitions in a feature space that combines color and texture information. After estimating the transitions, disjoint boundaries are connected to form a partition. The other frames in the group are segmented using a (either forward or backward) motion compensated partition as initial estimate. Final partitions are found using a strategy that combines a homogeneity-based criterion for solving occlusions and a transition-based criterion for solving uncovered areas. The main problems with this technique are to decide the temporal extension of the group of frames as well as to ensure that the regions that are defined in the middle frame can correctly represent all the objects present in the complete group of frames.

The technique in [5] proposes a transition-based temporal link, although the initial partition is achieved by a homogeneity-based decision that uses the gray-level information as feature space. Motion between consecutive images is estimated [45], and the open boundaries and nodes (contact points of boundary segments) of the previous partition are motion compensated. Compensated boundaries are correctly placed in the current image by minimization of an energy function along the boundary segment. Final boundary segments are connected relying on pure geometrical criteria. As previously commented, the use of such criteria usually leads to final contours that locally do not represent the real transitions in the image.

3) Transition-Based Approach to Temporal Segmentation: Transition-based techniques are commonly applied to temporal segmentation and have been quite successful. The 1-D nature of the segments alleviates the lack of robustness and the connectivity issues of the approach. In the sequel, we analyze some of the techniques proposed in the literature. Techniques are grouped based on the feature space they use. Other reviews analyzing temporal segmentation approaches from other viewpoints can be found in [1], [69], and [70].

The first reported techniques for temporal segmentation rely on very simple feature spaces such as the difference of pixel values in consecutive frames [42]. A pixel is said to have changed if the difference in consecutive frames is larger than a given threshold. When the amount of changed pixels in a frame is large enough, a temporal transition is declared. The lack of robustness of this technique in front of noise or even slow camera or object motion is reduced by the use of a simplification step. This way, in [70] images are low-pass filtered before computing the frame difference. In addition, to decrease the sensitivity to camera and object motion, it is proposed to divide every image into a set of blocks and to estimate the mean and variance values of the pixels in each block. The decision is finally taken based on the amount of blocks with similar mean and variance features in successive frames.

The selection of a global image feature such as the image histogram helps improving the robustness against camera and object motion. Techniques based on the difference of consecutive histograms are presented in [42] and [70] for the gray-level and color cases. A simplification is proposed for the color histogram where the code for each color is created using only the two most significant bits of every color component. The final decision is taken by thresholding the sum of a function of the histogram value differences from consecutive frames. Among the different functions that are tested for histogram temporal segmentation, χ^2 comparison of color histograms is reported to be the most robust approach [42].

To further improve the robustness against camera and object motion, motion information can be estimated and applied in temporal segmentation of video sequences. Global motion is estimated and images are motion compensated to simplify the video data previous to feature extraction. This technique aims at removing the effect of camera motion in the shot transition detection. A global motion temporal segmentation algorithm is presented in [70]. Here, the direction of motion vectors is used as the feature to segment the video data. Therefore, temporal segments are associated with the different camera movements. A similar goal is pursued in [61] where the computed optical flow is classified as singular and nonsingular,

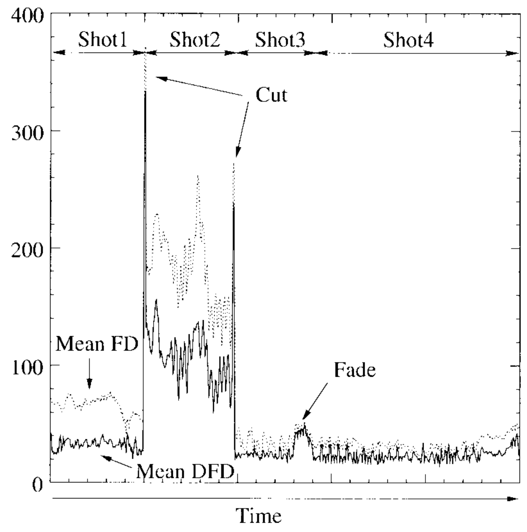

Fig. 3. Comparison between the use of FD and DFD for temporal segmentation in the cases of transitions due to scene cuts and fades.

depending on whether it vanishes or not in the camera center. The classification allows a more robust extraction of dominant motion. In [58], the feature space is a nonlinear combination of the matching values obtained in the block matching process between successive images. The decision is based on the assumption that, if the information in the current image can correctly predict the following image, both images belong to the same shot.

An example of the improvement achieved when using the motion information in temporal segmentation is presented in Fig. 3. The displaced frame difference (DFD) and frame difference (FD) are computed and used as feature space to segment a video sequence that presents four different shots. The transitions between the first three shots are due to the scene cuts, and they are better determined using the DFD as feature space. Note that FD values in the interior of the second region (shot with a high motion activity) are similar to that in the transition between the second and third regions. Therefore, the real position of the transition cannot be obtained by a simple threshold.

Editing effects represent a very important and specific problem in temporal segmentation. They produce gradual transitions that cannot be handled as abrupt ones. In the example of Fig. 3, the transition between the third and fourth shot is due to a fade. The so-called twin-comparison method [70] proposes a double threshold on the estimated transitions to cope with gradual as well as abrupt transitions. This approach can be used for a large number of feature spaces. A lower threshold detects those frames that are candidates to be either an abrupt transition or part of a gradual one. A higher threshold is used to detect the actual position of the transitions. Isolated frames with values greater than the higher threshold are associated to abrupt transitions. Gradual transitions are represented by a set of consecutive frames with values higher than the lower threshold. To assess whether such a set of frames should be considered as a gradual transition and to establish its position, the difference between consecutive

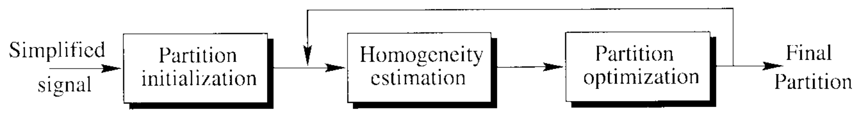

Fig. 4. Block diagram of homogeneity-based segmentation techniques. The process may iteratively estimate the homogeneity criterion and optimize the partition. At each step in the loop, a different homogeneity criterion may be selected.

frame values is added up. The transition is placed at the frame where the added value reaches the level of the higher threshold. If the higher threshold value is not reached, the set of frames is not associated to any transition.

An additional problem in temporal segmentation is the large amount of images that must usually be processed and the resulting computational load. The current existence of video data in compressed form has lead to the development of techniques that segment video sequences temporally without totally decoding the bitstream. Some techniques take advantage of the data structure in the compressed domain and simplify the image contents working in the so-called dc images [69]. DC images are spatially reduced versions of original images where each pixel corresponds to the dc value of its associated block in the compressed image. Various feature spaces are shown to work properly when applied to dc images and techniques for detecting abrupt and gradual transitions are proposed.

In addition to the previous one, several approaches have been proposed to further exploit the encoding strategies in the case of MPEG-1 and MPEG-2. Using the different characteristics of the three types of frames (I-, P-, and B-frames), temporal segmentation can be done at several levels of resolution and combining various features. The method presented in [3] performs first a simplification of the sequence only retaining the I-frames. The feature space is some of the discrete cosine transform (DCT) coefficients of a selected set of blocks in the frame. For each frame, a vector is formed chaining the selected DCT coefficients. The transition is estimated by computing the normalized inner product of the vectors of two consecutive frames. If the normalized inner product is close to zero, the two frames are said to be different and the shot boundary is located between both frames. Shot boundaries can be more precisely located if the simplification step preserves P-frames as well as I-frames [34]. When using P-frames, the feature space is related to the motion prediction. The ratio between motion compensated and intraframe coded macroblocks is used to estimate the temporal boundaries. Last, the use of B-frames has been proposed as well [34]. The feature space is associated to the amount of backward and forward motion vectors used in the prediction of a frame. If a B-frame is coded using mainly backward prediction, a temporal boundary is detected.

B. Homogeneity-Based Segmentation Techniques

1) Overview of the Feature Spaces and the Decision Algorithms: The decision step in homogeneity-based segmentation looks for the actual region of support where elements are homogeneous with respect to the feature space. Most relevant techniques implement the decision as an iterative process. A scheme of the global process is presented in Fig. 4.

a) Partition initialization: The algorithm is initialized by selecting a first estimation of the partition. This first estimation may be as rough as having all elements gathered in a single region or every element associated to a different region. Furthermore, the first estimation may not even be a partition itself. It may contain a set of components marking the core regions (markers), and a large number of elements may remain unassigned. The iterative algorithm builds up the final partition based on the selected homogeneity feature space(s) and on an optimization algorithm. The performance of the global process is less sensitive to the quality of the initial estimation if a random optimization is used instead of a deterministic one. Nevertheless, due to computational load, deterministic optimization strategies are commonly used.

b) Homogeneity estimation: As previously discussed, multimedia applications require the use of complex feature spaces. In temporal segmentation, the use of frame differences is not sufficient to partition the sequence into shots in the presence of moving camera or editing effects. Complex features are even more necessary in the case of spatial segmentation. To extract the spatial contours of objects evolving in a generic scene, various basic features should be combined.

An object is an entity with semantic meaning, which is commonly associated to a set of connected regions whose contours are defined in the original image. As a result, it should be possible to define an object by selecting a set of regions from a correctly segmented image. This concept leads to the use of color and texture features in the spatial segmentation process. Furthermore, contour features may be introduced to avoid noisy contours and to obtain regions conforming to the shape of natural objects. The automatic selection of the regions forming the objects generally relies on other homogeneity features. Motion information is largely applied since regions with homogeneous motion can be considered as objects in several applications. In addition, scenarios with foreground moving objects and static or quasi-static backgrounds are commonly assumed. As in transition-based approaches, global motion estimation and compensation can be used to simplify the image and to obtain a (quasi-) static background video sequence.

Although some techniques are only based on motion features, segmentation performance improves when combining motion with color and texture information. Direct segmentation of the motion information does not yield accurate transitions. Object contours are more accurately defined relying on color information. The combination of these feature spaces can be done following a parallel or a hierarchical strategy. The parallel approach computes a motion spatial segmentation on one side and a color spatial segmentation on the other side. Then both partitions are combined. The hierarchical strategy relies on merging steps. It starts, in a first step, by a color spatial segmentation and changes its criterion to deal with motion spatial segmentation in a second step.

Some objects do not fulfill the constrain of homogeneous motion but can be seen as a union of sub-objects with their own motion. This is for example the case for regions corresponding to human bodies. The depth information (distance to the camera) is a useful feature for segmenting such complex objects. As before, depth can be estimated and combined with previous features in a parallel or in a hierarchical manner [48]. For some specific objects and applications, it is possible to introduce in the segmentation process semantic features. Semantic features are selected after analyzing the behavior and characteristics of the object. A typical example includes segmentation of human face regions.

When the object to be tracked in a spatio-temporal segmentation is too complex, or its attributes may vary in time, user interaction can be applied to define or update the object. Note that, for the various features previously commented, it is always possible to help the automatic process by means of user interaction and, this way, to improve the segmentation performance.

Analysis of the feature space is done by assessing a homogeneity criterion. In the literature, one can find two main types of homogeneity criteria: deterministic and probabilistic. For deterministic criteria, a model $M_R(i)$ per region R is fit to the data $f(i)$, and a distance is computed between the model fitting the data and the data

$$\mathcal{D} = \sum_R \sum_{i \in R} \|f(i) - M_R(i)\|. \tag{1}$$

Typically, low-order polynomial models are used and the norm $\|\cdot\|$ is the Euclidean distance.

For probabilistic criteria, the region data are assumed to be a realization of a given random field $F_R(i)$ and, usually, the *a posteriori* likelihood of the data to be a sample of this random field is computed. Assuming independence between regions and neighboring pixels, the likelihood of a given partition has the following expression:

$$\mathcal{L} = \prod_R \prod_{i \in R} \text{Prob}\{F_R(i) = f(i)\}. \tag{2}$$

Gaussian models are commonly used for characterizing random fields. Based on the estimated likelihood or log-likelihood, the region is said to be homogeneous or not.

In a large number of cases, practical implementations of these two types of criteria lead to very similar algorithms. If the observed data are assumed to have a Gaussian distribution with zero mean around a deterministic region model, the log-likelihood is the Euclidean distance weighted by the covariance matrix of the distribution.

c) Partition optimization: Partitions (or marker information) are modified and updated to reach an optimum in the homogeneity criterion sense. Modifications rely on two basic processes: split and merge. Given a region, its goodness in terms of the selected homogeneity criterion is judged, and the region is split if its division leads to a better configuration. All possible divisions of the original region are not tested but only some predefined splits are analyzed. Typically, the region is divided into parts with equal area of support [50] or divided into two parts by a line segment [24].

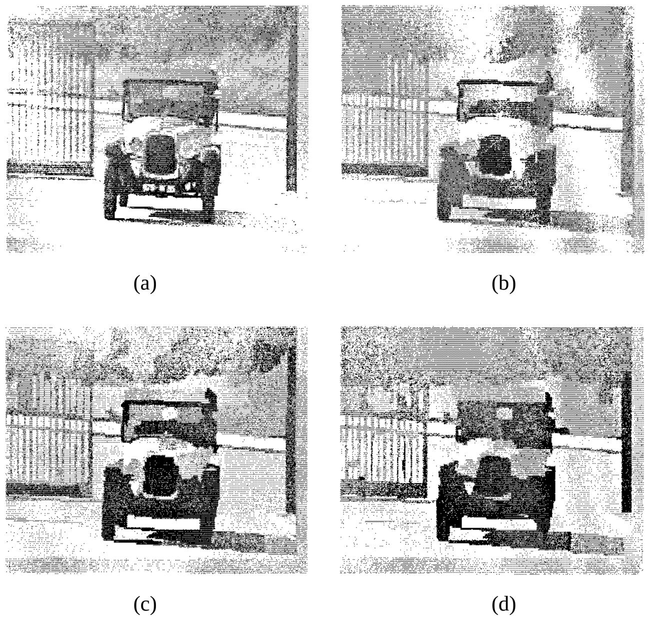
Fig. 5. Example of homogeneity criterion combining texture and contour features. The results are presented filling each region with its mean gray level and have been obtained changing the weight assigned to the contour smoothness with respect to the texture smoothness. (a) Original image. (b) Low weight. (c) Medium weight (d) High weight.

Merging is applied to two regions when their union into a single region leads to the best partition in the homogeneity criterion sense. Note that a merging can even result into an improvement in the homogeneity criterion. An example is presented in Fig. 5 where the same image has been segmented using as homogeneity criterion a combination of texture homogeneity and contour smoothness. In all cases, the number of regions is the same. Note that when the weight assigned to the contour smoothness decreases (increases), textured areas appear segmented into regions with complex (simple) shapes; e.g., the forest in the background of the image. On the other hand, the segmentation of highly contrasted regions produces similar shapes, regardless of the weight of the contour smoothness in the homogeneity criterion, e.g., the gate.

Region growing is the basic merging approach. Regions are merged based on a homogeneity criterion up to reaching a termination decision. The process can be initialized by defining a set of labeled markers partially covering the space or a complete partition where regions can be formed even by single pixels. Given a partition or set of markers, the analysis of all possible mergings may be cumbersome due to the complexity of the homogeneity criterion. Consequently, strategies to reduce the number of analyzed unions are frequently used. A possible solution to represent the merging steps of a region-growing algorithm is to define a spanning tree [37]. Once computed, the spanning tree can be used to derive several partitions resulting from the region growing process. This approach will be further extended and discussed in Section IV.

However, in spite of computing all possible mergings at each step, the final result may not be the optimum partition. Many homogeneity criteria are multimodal, that is, they present several local maxima or minima. Since the study of the possible region unions is usually carried out locally, the optimization algorithm can get trapped in a local optimum.

Pure split-based and merge-based algorithms have been largely applied, (e.g., the *quad-tree split* [50] and the *watershed* [35], respectively). Nevertheless, other algorithms for partition optimization have been proposed combining these two basic processes. This way, the so-called *split&merge* [50] applies a merging process on the partition resulting from a split step. The iterated conditional modes (ICM) approach [4] analyzes elements on the region boundaries. It compares a given partition with the result of splitting a region by removing an element lying on its boundary and merging it with any of its neighboring regions.

In the sequel, only homogeneity-based spatial segmentation approaches are discussed since, up to the authors' knowledge, no homogeneity-based temporal segmentation has been reported in the literature. The analysis of these approaches will be carried out from the point of view of application and feature space complexity.

2) Homogeneity-Based Approach to Spatial or Spatio-Temporal Segmentation:

a) Region-based coding algorithms: One of the first methods for generic spatio-temporal segmentation was presented in the framework of video coding [68]. Region-based image and video coding techniques [27], [63] aim at improving the coding efficiency by applying segmentation techniques before encoding the data. In the work presented in [68], a set of consecutive frames (a 3-D block) is taken as a single unit, and gray-level information is used to segment it. A 3-D *Split&Merge* approach is proposed to segment the sequence while the homogeneity criterion compares the data in each 3-D region with a polynomial model. The same type of model is used in the merging step. This approach presents several problems due to the 3-D block structure, such as blocky shapes of regions and lack of temporal relation between 3-D blocks. From the coding point of view, the major problems are associated to the large delay and low compression efficiency. The reason for this low compression efficiency is the large number of regions that is necessary to get a good visual quality. This problem is due to the use of only spatial information.

In [52], a quad-tree split spatial segmentation technique for coding purposes is presented. The technique works on a frame-by-frame basis and creates, for each frame, the complete quad-tree. For each node in the tree, several coding techniques are applied, and the quality of the coded result is stored. The final partition is defined by the combination of regions and coding techniques that optimizes the rate (distortion) under a constraint of distortion (rate). The feature space is therefore the rate-distortion space. This technique introduces the concept of creating a set of possible solutions that are jointly analyzed to achieved the final partition. The potential of this approach will be further discussed in Section IV. However, the simplicity of the quad-tree reduces the quality of the final partitions. In addition, there is no temporal link between partitions.

Several techniques for spatial segmentation have been developed having as a basic decision tool the watershed algorithm [35], [65], which is the morphological approach for image segmentation. The watershed technique is a region growing algorithm that analyzes the image as a topographic surface. It detects the minima of the gradient of the gray-level image and grows these minima according to the gradient values. It can be viewed as a flooding process. Points of contact between the propagation originating from different minima are defined as the boundaries of the regions and create the final partition. This basic algorithm has been extended to deal with markers and to process directly the original data instead of the gradient image. In this case, the watershed becomes an efficient way of implementing region growing. In addition, the watershed has been adapted and combined with other techniques to yield partitions with suitable characteristics for specific applications, including coding and progressive transmission [57], [22].

In the framework of region-based video coding, the work presented in [55] uses features such as contrast, size, and motion of regions to obtain a hierarchy of partitions. At each level of the hierarchy, the homogeneity criterion that is assessed in the watershed algorithm takes into account the complexity of the final contours. Contour complexity is introduced since contour coding cost is a very relevant issue in video coding. This hierarchy of partitions is input to a final decision step that combines regions from all levels in the partition hierarchy and creates a final partition of the initial image. The hierarchy of partitions is built up using motion as well as size and contrast criteria. Therefore, this segmentation approach combines several feature spaces to create a universe of possible partitions. As in [52], the final partition is chosen to be optimal in the rate-distortion sense. A similar strategy is adopted for the following frames in the sequence. The temporal link is created by an extension of the partition projection proposed in [49]. Furthermore, this technique partially decouples the problem of finding homogeneous regions and that of defining the final partition, which will be further discussed in Section IV.

b) Object tracking algorithms: Previously commented techniques were developed in the framework of region-based video coding, and therefore they did not specifically aim at object tracking. Object tracking enables video object (in the sense of MPEG-4) creation and, as a consequence, opens the door to content-based functionalities.

Morphological techniques allow accurate boundary localization and are broadly used for object tracking purposes in spatio-temporal segmentation. The usual approach is to obtain a partition of the initial image containing the object contours and to project this partition in the following frame. Projected regions are used as markers in the following frame and, if necessary, they are grown to conform to the actual boundaries [49]. The work presented in [67] follows this structure, with a modification in the projection step that allows

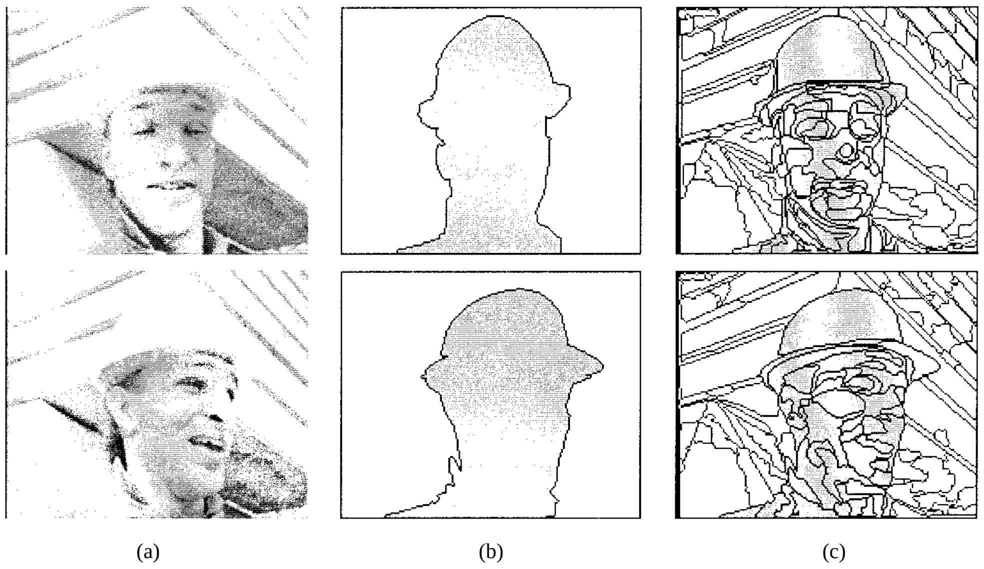
Fig. 6. Example of object tracking using a hierarchy of partitions. (a) Original images. (b) Object partitions. (c) Color-homogeneous partitions. The tracking is carried out at the color partition level.

the extraction of new regions. In [23], a predefined object is tracked using an extension of the projection step presented in [49]. The so-called global perspective motion estimation and compensation is used to improve the location of the markers in successive frames. Final boundaries are obtained using a color watershed algorithm. However, since projected markers are related to complex objects, they are not homogeneous in color. Therefore, only local homogeneity criteria (pixel-wise) can be applied.

This problem is solved in [30], where a hierarchy of partitions is used to track the object. The object partition is resegmented using a color homogeneity criterion. This color-based partition is projected, and projected markers are accommodated to the color information in a simplified version of the current image. Simplification is carried out by connected operators that yield a very fine partition of the current image in the color space. This technique allows tracking generic objects since they are divided into simple, color-homogeneous regions, which are easier to track (see Fig. 6). Nevertheless, this approach as well as all pure object tracking techniques require the initial definition of the object to be tracked.

c) Moving object segmentation and tracking: A common way to handle the object definition issue is to restrict the problem to the case of detecting and tracking moving objects. Such an approach is followed in the work presented in [11], where original images are simplified by global motion compensation and further filtered using morphological open–close by reconstruction. The watershed technique on a color space is applied to obtain the initial segmentation. From this partition, those regions containing more than 50% of their pixels with a value different from the previous image are marked as belonging to the moving foreground. The performance of this algorithm is constrained by the simplicity of the motion feature that is used, and new methods have been proposed combining it with techniques that better exploit the motion information.

The motion spatio-temporal segmentation suggested in [32] has been combined with the previous one [11]. The resulting algorithm has been involved in *core experiments* in the framework of the MPEG-4 standardization process. The technique will be described as an informative annex in the standard. The segmentation technique in [32] estimates the global motion and simplifies the video data by global motion compensation. In addition, a scene cut detector is used to establish whether the initial frame of a shot is under consideration. Motion information is used to create a change detection mask indicating the presence of moving objects. The shape of the moving object(s) is further refined by using motion features that allow the correct assignment of uncovered background. Final boundaries are obtained by adapting the partition to the luminance information. Different techniques for adapting the partition have been proposed. For example, this algorithm can be combined with a technique that uses color and motion features in a hierarchical way in a rule-based region processing [2]. Segmentation is carried out by means of a recursive shortest spanning tree first in the color space and then, relying on the color homogeneous regions, in the motion space. The combination of both algorithms is the analysis model currently adopted by the COST 211 group [2]. This type of algorithm performs quite well although it may detect false objects in scenarios containing shadows or reflections. Moreover, the algorithm loses track of objects that stop.

A typical result of this type of techniques is presented in Fig. 7. Three frames of the *Children* sequence are presented. Parts of the moving objects are included in the segmentation result as soon as they start moving. Previous moving parts are included thanks to the use of a memory in the segmentation process. As can be seen, the estimation of moving objects is improved by the tracking process.

Different approaches have been proposed in the literature to further exploit the motion feature space. For instance, in [44], higher order statistics are used to classify the image elements

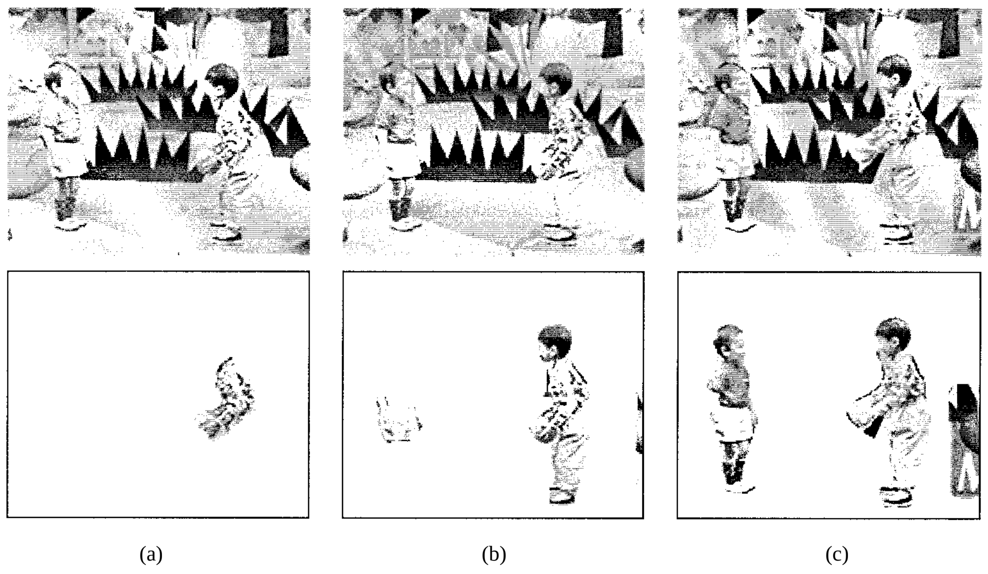

Fig. 7. Example of segmentation and tracking of moving objects. (a) Frame 001. (b) Frame 003. (c) Frame 005.

into foreground and background zones. Three-dimensional models have also been applied for the detection, segmentation, and tracking of moving objects in different scenarios [25], [59]. Kalman filtering has been proposed as well for allowing object tracking [16].

Last, a very popular approach to the problem of motion spatial segmentation relies on the use of compound Markov random field (MRF) models [21], [9]. A compound random field consists of two levels, called upper and lower. The upper level describes the observed image and is formed by several submodels related to the different homogeneous regions in the image. The lower level governs the transition between the observed submodels, whereby it is associated to the hidden partition of the image. This type of model is very appropriate to estimate and segment motion data. The introduction of a level in the model specifically handling the presence of boundaries allows a robust estimation of the motion information as well as an accurate definition of regions that are homogeneous in motion.

The estimation/segmentation is done by maximizing the *a posteriori* probability of a partition $(Q = q)$ for an observation $(X = x)$. Thanks to the Bayesian approach, the problem can be translated into maximizing $P(X = x/Q = q)P(Q = q)$. In the case of motion spatial segmentation, the observation is the motion field that has to be estimated from the original data. A usual approach to this problem is to iterate the estimation of the motion information and the maximization of the probability (see Fig. 4).

The motion can be estimated using parametric or nonparametric models. The work presented in [60] computes a nonparametric motion field. Images are simplified to perform the optimization using the ICM algorithm in a multiresolution approach. The features used to define the MRF model are related to spatial and temporal bindings of motion vectors. Temporal bindings are established accounting for motion trajectories. As a result, the concept of object tracking is directly introduced in the sequence model. Detected boundaries conform very well to natural contours, although the concept of object is not exploited.

Pure object tracking applications have also been addressed by MRF approaches. The work proposed in [46] breaks the typical loop of motion estimation and partition optimization, reducing it to a single step. This is done thanks to a robust multiresolution parametric affine motion estimation. The selected features take into account motion smoothness, spatial homogeneity properties, and temporal coherence of the partitions. The performance of the technique is further improved by the use of the so-called highest confidence first technique [12] for modifying the partition. This segmentation technique achieves very good tracking results, although contours do not always match with real boundaries because the algorithm does not use any spatial segmentation step.

d) Semantic segmentation through user interaction: All previous approaches may very likely fail to segment objects with semantic meaning, due to the complex definition of such objects. Nevertheless, some specific cases (such as human faces) have been successfully addressed, as will be discussed in Section IV. A common way to introduce complex homogeneity features is by allowing user interaction. This possibility is clearly necessary in object tracking applications, when the object to be tracked is too complex to be described in terms of low-level features [13]. Nevertheless, interaction should be as restricted and easy as possible. Three main types of interaction have been proposed in the literature.

- *Feature based:* In [8], a vector of features (including color, texture, and motion information) is estimated for each pixel. Each one of these features is assumed to have a probability density function (pdf) that can be approximated by a sum of Gaussian pdf's. To approximate the pdf's, the user selects a set of pixels belonging to the different regions that represent the object. The remaining pixels are classified into one of the classes specified by the user. However, connectivity aspects have to be taken into account after classification, and further user interaction may be necessary.

- *Contour based:* Another possibility for user interaction is to enable roughly marking the position of the object boundaries. An automatic algorithm accommodates the user's information to the real boundaries of the object. Actual boundaries can be found by a homogeneity-based technique as in [23], or by any of the transition-based approaches discussed in the previous section, as in [71]. In this case, object boundaries may not correspond to the type of boundaries needed for a proper tracking. A complete semiautomatic object tracking technique based on the concept of snakes has been presented in [26], leading to good spatio-temporal segmentation results.
- *Region based:* The third possibility is related to the definition of objects as the union of regions, which are homogeneous in some sense. The user is then allowed to interact with an initial partition by, usually, merging regions to create the final object [7]. In Section IV, a similar type of user interaction will be proposed. Since the initial partition is defined by the algorithm, it can use motion features to help the tracking process. Moreover, this technique allows an easy integration of the user interaction during the tracking process by allowing merging or splitting of erroneous regions.

In [29], the region-based user interaction approach has been extended, integrating features of the other two approaches. The work in [29] allows the introduction of rough markers proposed by the user, which is a type of interaction very close to that proposed in feature-based approaches while directly using connectivity concepts. Furthermore, it allows the definition of rough contours to obtain the object shape, enabling a type of interaction very close to the contour-based approach. However, in this case internal and external rough contours of the object are necessary. Last, the use of regions in the tracking process has been exploited to propose possible improvements of the final result to the user.

IV. Partition Tree as a Region-Based Image Representation: Universal Segmentation and Applications

The previous section reviewed the state of the art in segmentation. Segmentation aims at creating a partition (that is, a specific region-based representation) of the original data. One of the difficulties to be faced is related to the initial low-level representation of the original data. Most algorithms try to progressively convert the pixel-based representation into an initial region-based representation and then act on this region-based representation to obtain the final partition by splitting or merging steps. In this section, we propose a partition tree as a region-based representation that can be used for a large number of segmentation algorithms. The partition tree creation can be viewed as a preprocessing step creating a representation that is appropriate for many segmentation algorithms and that does not depend on the application.

Indeed, the previous section has highlighted the relation between the feature space, the decision space, and the applications. One of the major difficulties in segmentation is to adapt the analysis algorithm to the application. It is in general impossible to judge the quality of a partition produced by an algorithm without taking into account the subsequent use of this partition. One possible solution is to define a specific segmentation algorithm for each possible application. The drawback of this approach is that most of the time, it is not easy to reuse the work done for a given application to design a segmentation algorithm useful for a different application.

An alternative solution consists in finding a compromise between what could be done in a systematic and universal way and what has to be really application dependent. Note that this has to be a compromise since the "universal segmentation" does not exist. A possible solution is discussed in this section. It is summarized by Fig. 8: the three steps of the segmentation described in Fig. 1 are remapped into two steps: 1) the *similarity estimation* and 2) the *partition creation*. The similarity estimation deals with the simplification step as well as part of the feature extraction (generic features), whereas the partition creation addresses specific (nongeneric) feature extraction and the decision step. The output of the similarity estimation is already a region-based representation (but more than a partition in the sense that it is a hierarchial set of partitions).

The *similarity estimation* is, to a certain extent, independent of the application. Its goals are, first, to build a region-based representation by defining a set of regions that may be used by the *partition creation*, and second, to estimate the similarity between these regions. At this stage, the feature(s) used to measure the similarity should be rather generic. A *partition tree* structure can be used to represent the set of regions together with the similarity. An example of partition tree is shown in Fig. 9. The original image is shown in Fig. 9(a). An initial partition corresponding to this image is given in Fig. 9(b). Note that this partition is composed of 100 regions and would be considered as oversegmented for a large number of applications. In fact, in the present approach, it only defines the lowest resolution scale in term of regions. Ideally, it should represent all image elements that are perceptually relevant. Fig. 9(c) shows the image obtained by filling the regions of the initial partition with their mean gray value. As can be seen, almost all details of the original image are visible. Information about the similarity is encoded in the tree shown in the upper part of the figure. The tree leaves represent the regions of the initial partition. The remaining tree nodes are parent nodes. They represent regions that can be obtained by merging the regions represented by the child nodes. The information about the similarity is encoded by the tree structure itself: the set of regions that are the most similar to a given region R_0 is given by the set of siblings nodes of R_0. As a result, the similarity between regions represented in the lower (upper) part of the tree is very high (rather low). In practice, the tree can be arbitrary in the sense that a node can have an arbitrary number of child nodes. In the example of Fig. 9, the tree is binary (each node has either no child or two children). This is not a constraint on the approach itself, but a binary partition tree [54]

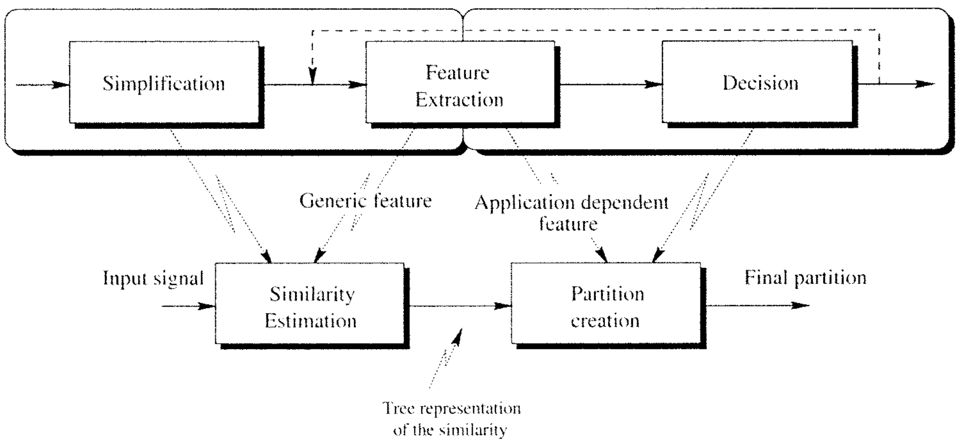

Fig. 8. Segmentation viewed as a two steps process. The *similarity estimation* is application independent and deals with the simplification steps as well as generic features. The *partition creation* is application dependent and covers the use of specific features and the decision step.

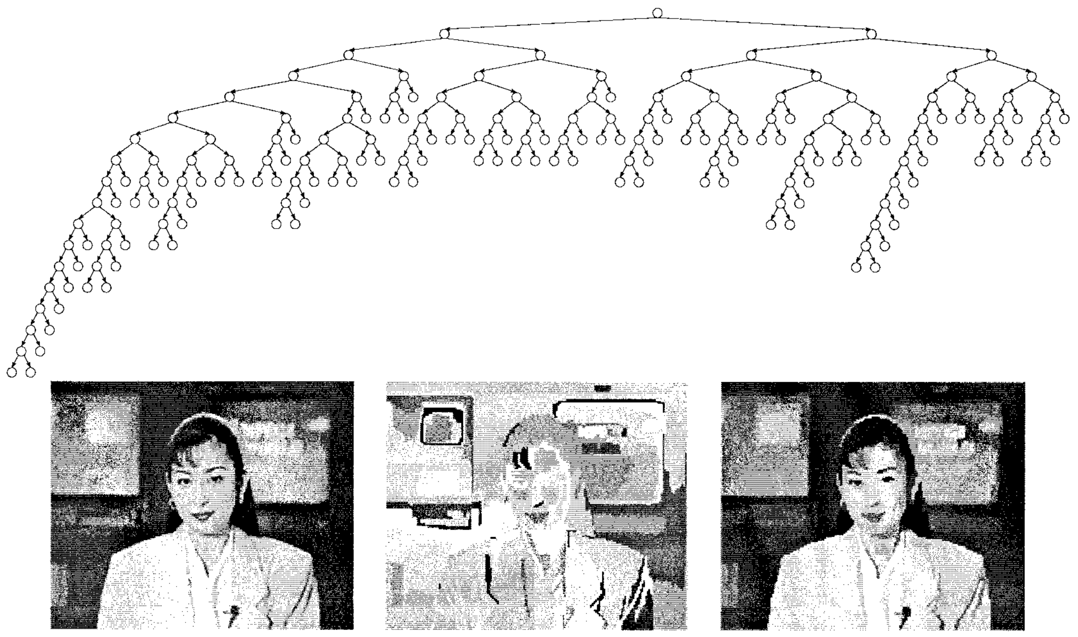

Fig. 9. Example of partition tree (top). (a) Original image. (b) Initial partition with 100 regions. (c) Regions of the initial partition represented by their mean value.

is simple to construct and sufficient to illustrate the interest of the approach. Last, let us mention that the *similarity estimation* can be viewed as a hierarchical segmentation problem. The approach reported here is related to the work presented in [37]. The main difference of our approach is that the tree is proposed here as an image representation that is the basis of a large number of different tools (that is various partition creation techniques). Moreover, the tree construction is more flexible than in [37], because it allows to deal with various criteria and with precomputed analysis results.

The *partition creation* block of Fig. 8 takes as input the partition tree and creates the final partition. This last step is closely related to the application. Its goal is to search, within the space of regions and of similarity encoded in the tree, the best set of regions for the application. The feature(s) used for this final decision can be very specific by contrast to the generic feature(s) used in the similarity estimation.

The main drawback of the proposed approach is that we do not take into account the application to define the similarity between regions. In order to show that this drawback does not prevent dealing with very different applications, we are going to discuss various examples: unsupervised and supervised spatial segmentation, motion spatial segmentation, region-based coding, semantic object extraction, and region-based retrieval. We assume that the *similarity estimation* has been computed once and that in all cases (except in Section IV-E), the *partition creation* starts with the partition tree shown in Fig. 9 (top). Furthermore, to show the flexibility of the *partition creation,* an almost trivial similarity criterion is used to create the partition tree. More precisely, the tree is created

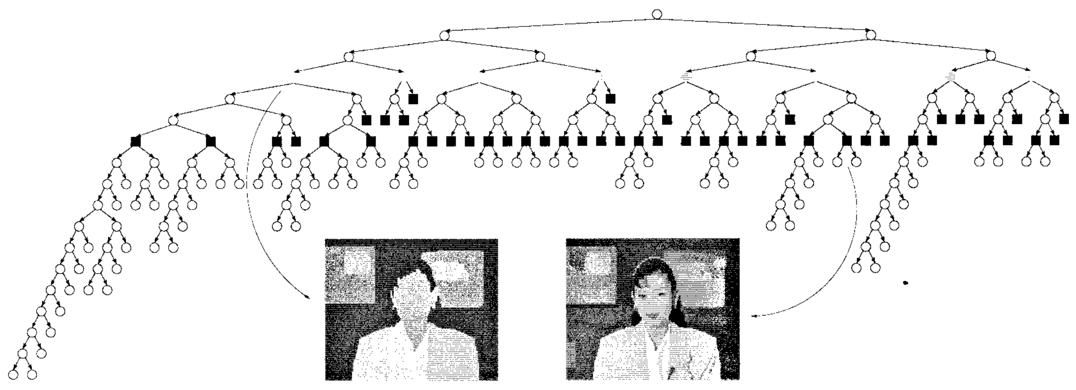

Fig. 10. Examples of unsupervised spatial segmentation: creation of partition with (a) 8 and (b) 48 regions.

by merging pairs of neighboring regions that belong to the initial partition of Fig. 9(b). The merging order is defined by the Euclidean distance in the YUV space between the mean of each region. More details about the actual merging algorithm implementation can be found in [20].

A. Unsupervised and Supervised Spatial Segmentation

An example of unsupervised spatial segmentation is illustrated in Fig. 10. The segmentation is unsupervised in the sense that the *partition creation* simply defines the number of regions that are useful for the application. As can be seen in Fig. 10, the extraction of a partition with a given number of regions is a trivial operation. For example, a partition with eight regions can be obtained by selecting all nodes that are at the third level below the root node (gray rhombus in Fig. 10). Note that in order to judge the segmentation results of Fig. 10, regions of the final partitions have been filled with the mean values of the original image. To obtain a partition with a higher number of regions, one has to descent in the hierarchy. If the homogeneity between siblings is stored with the tree, it is possible to define a partition with N regions, by performing the first $N-1$ splits from the root node following the homogeneity values [37], [66]. Last, any global criterion that is a function of the partition can be used to define the segmentation. For example, the *partition creation* can use the global peak signal-to-noise ratio (PSNR) to define the optimum partition [66], [54].

A less trivial example is illustrated by the supervised segmentation shown in Fig. 11. Assume that a user wants to extract from the image two objects of interest. With the help of a graphical user interface (GUI), two markers roughly indicating the center of the objects of interest are defined. In the example of Fig. 11, the two markers correspond to the face and the jacket regions. Note that the objects are of interest because of their semantic value, and their definition is not directly related to the color homogeneity criterion used to compute the partition tree.

The *partition creation* can rely on a propagation strategy similar to the one used for morphological segmentation [35]. The traditional strategy consists in propagating the markers by similarity until the entire image space is filled. Most of the time, the propagation involves complex algorithms and may be time consuming. However, if the partition tree has been previously computed, the propagation process based on similarity between neighboring regions becomes an easy task. Let us describe this propagation on a simple example.

Fig. 12(a) shows a simple image made of four regions of constant gray level value. Assume that the *similarity estimation* has been computed. The resulting partition tree indicates that regions 1 and 2 are the most similar. Once merged, their closest region is region 3. Region 4 is the most dissimilar. Consider now two markers A and B that have to be propagated by similarity. The two corresponding leaf nodes are indicated in Fig. 12(b). By construction of the partition tree, the most similar neighboring region with respect to a given node is represented by its sibling node and the result of the merging is represented by the parent node. Therefore, a marker associated to a node is propagated to its sibling and parent. Of course, this propagation can only be done if the sibling is not in conflict with the marker, that is, if none of the sibling's descendant has been assigned to a different marker. In the example of Fig. 12(c), the first marker to be propagated is the marker A corresponding to region 2. It is propagated to its sibling, that is, region 1, and their union is represented by their parent. At this level, the propagation has to stop because there is a conflict between the marker of the union of regions 1 and 2 (marker A) and the marker of region 3 (marker B).

This propagation process may not assign a marker to all nodes of the tree. In our example, region 4 remains without label. This situation means that the similarity between regions defined by markers A and B is higher than any combination with region 4 (region 4 is indeed the most dissimilar). The propagation process does not blindly assign all regions to markers. In most cases, this control is an attractive feature of the partition tree representation with respect to classical propagation strategies [35], [53]. If necessary, the algorithm may be modified so that it fills the entire space.

A complete example is shown in Fig. 11. In this example, we assume that a user has defined two markers (dark and gray). The first step is to assign the markers to the leaves of the tree [Fig. 11(a)]. Then, the propagation process creates three connected components [Fig. 11(b)]. The first two correspond

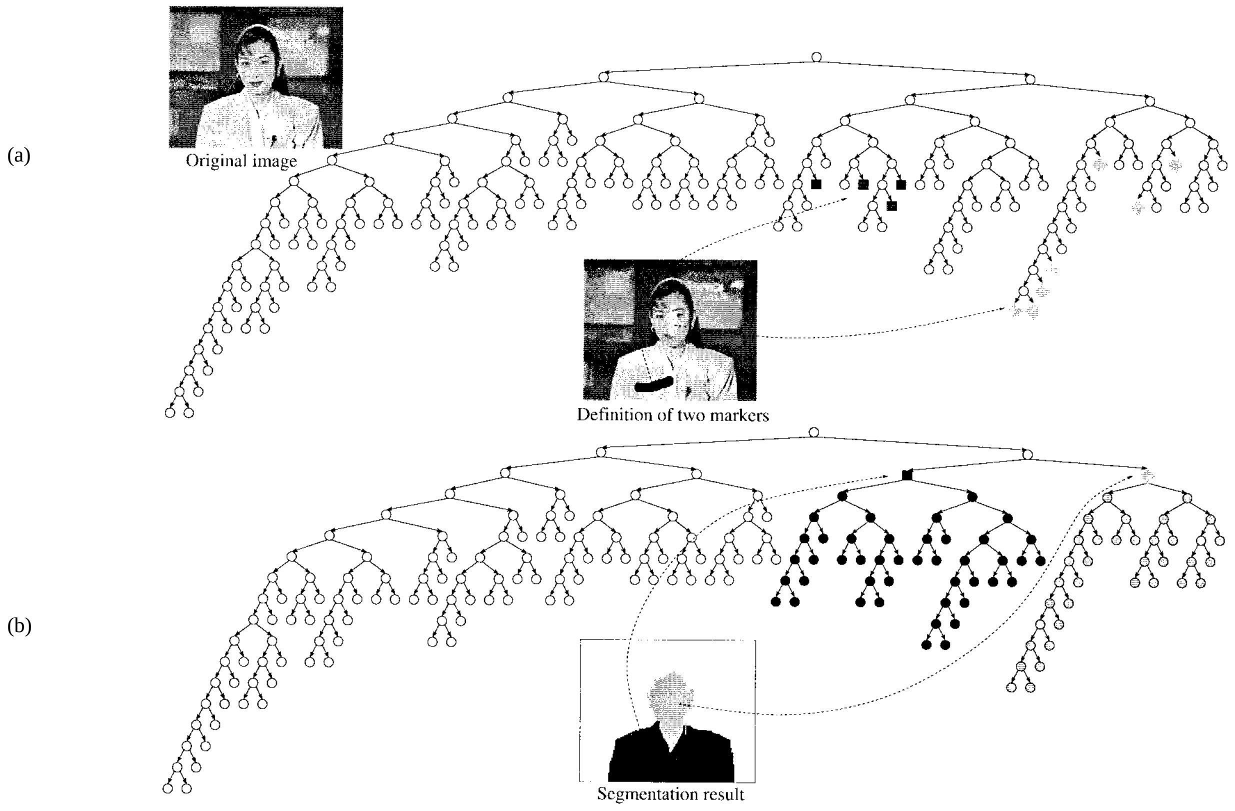

Fig. 11. Example of supervised spatial segmentation: the user marks two regions of interest with markers; then a tree propagation strategy creates the final partition. (a) Binary partition tree where the leaves intercepted by markers have been indicated. (b) Result of the propagation process.

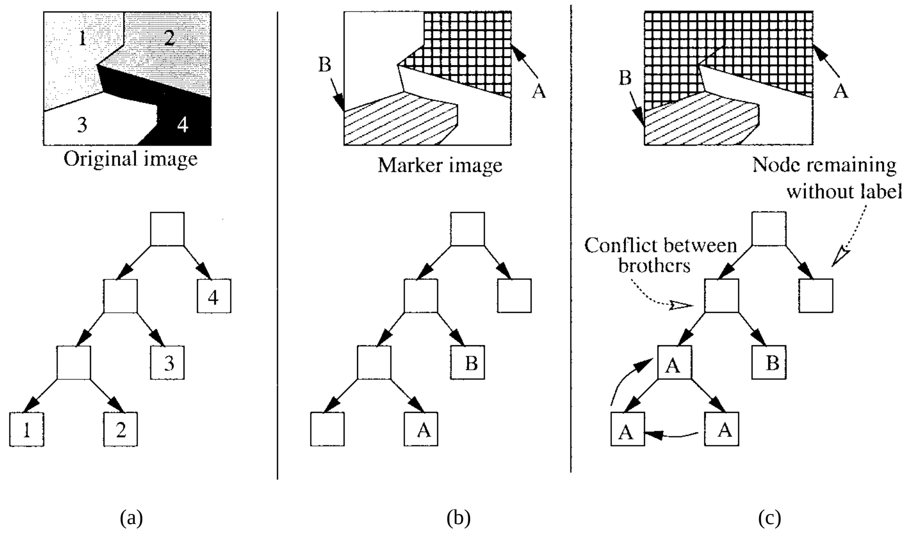

Fig. 12. Propagation process on the partition tree. (a) Binary partition tree. (b) Markers. (c) Propagation process.

to the zones of influence of the markers, whereas the last one remains without label because it is judged as being "too different." As can be seen, the face and jacket regions defined by the markers have been properly segmented and the background has been merged with none of these regions. Note that in this example, the background was judged as being "too different" with respect to the regions defined by the markers. In practice, the user may have to define markers in an iterative fashion to obtain the partition of interest.

In this context, one of the attractive features of the approach of similarity estimation and partition creation is its efficiency. Once the partition tree has been computed, the propagation process is very easy and almost instantaneous. This is particularly important for applications involving user interaction.

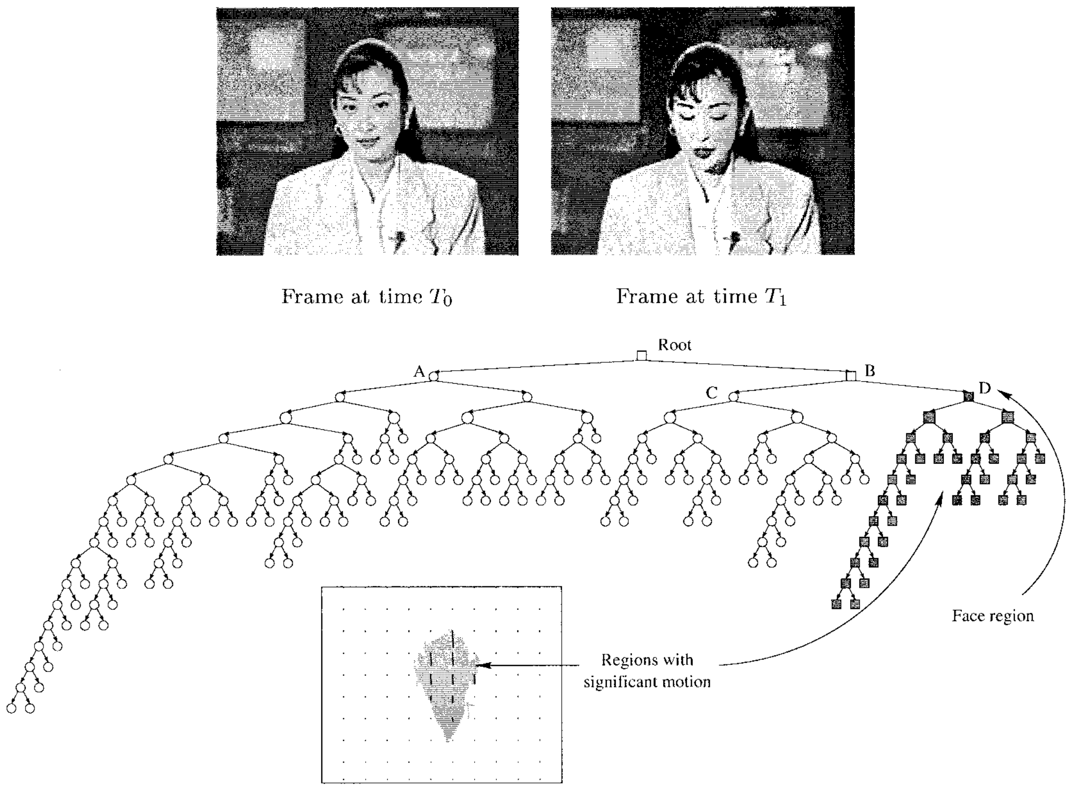

Fig. 13. Motion estimation/segmentation and face detection with the partition tree.

B. Motion Spatial Segmentation

A partition tree created according to a given feature is a useful representation to estimate various signal characteristics. For example, the partition tree of Fig. 9 has been defined by a color homogeneity criterion and can be used to deal with motion.

A typical case where a region-based representation of images provides robustness is motion detection. The strategy was proposed in [56] for Max_Tree and Min_Tree representation and is used here for partition trees. Let us describe the motion detection strategy on the example of Fig. 13. Two images belonging to the *Akiyo* sequence are shown in the upper part of the figure. The frame at time T_0 is the original image of Fig. 9. The motion between both frames is due to the face movements of the speaker.

A simple solution to detect the motion between frames T_0 and T_1 relies on the FD. If $f_{T_0}(i,j)$ and $f_{T_1}(i,j)$ denote the two images, the FD is given by $\text{FD} = \sum_{i,j} |f_{T_0}(i,j) - f_{T_1}(i,j)|$. Classical approaches analyze the FD at the pixel level. This strategy is not robust because the decision is very local and therefore sensitive to noise. Moreover, the FD does not detect the moving regions but only part of their contours (contours that are not parallel to the motion trajectory). The partition tree allows this analysis to be done at the region level. In fact, for each region of frame T_0 represented in the tree, a decision between "static" or "moving" area has to be taken. In the tree of Fig. 13, regions with significant FD are indicated by a square node. The mask of moving regions is obtained by the union of all leaves with a significant FD. It is shown in gray in the image of Fig. 13. Note that this example is rather simple. In particular, if the FD is small for a given region, it remains small for all its children. In practice, this may not always be the case and the classification between "static" and "moving" areas has to involve a robust decision strategy. The description of robust decision techniques goes beyond the scope of this paper, and the reader is referred to [56] for a detailed explanation of such a technique involving a Viterbi algorithm. Last, let us mention that the application of this change detection algorithm to cases where the camera is moving is straightforward. One has simply to estimate the dominant motion $(\Delta_x(i,j), \Delta_y(i,j))$ induced by the camera motion and compute the DFD to compensate for the dominant motion. The DFD is given by

$$\text{DFD} = \sum_{i,j} \left| f_{T_0}(i,j) - f_{T_1}(i-\Delta_x(i,j), j-\Delta_y(i,j)) \right|. \quad (3)$$

Motion estimation can also gain from the partition tree representation. The strategy consists in propagating a region-based motion estimation process in a top-down fashion. Assume, for example, that the motion is modeled by an affine transformation (six parameters)

$$\begin{pmatrix} \Delta_x(i,j) \\ \Delta_y(i,j) \end{pmatrix} = \begin{pmatrix} \alpha_x i + \beta_x j + \gamma_x \\ \alpha_y i + \beta_y j + \gamma_y \end{pmatrix}. \quad (4)$$

The dominant motion can be estimated by finding the optimum set of parameters, $\alpha_x, \ldots, \gamma_y$, so that the DFD is minimized over the region of support of the entire image (represented by the root of the tree). Gradient-based optimization techniques are appropriate for this purpose [17]. A significant DFD indicates that the motion is not homogeneous in the image. Therefore, the optimization is iterated individually on each child region denoted by A and B in the tree of Fig. 13. In this example, the motion parameters estimated for region A correspond to a static motion. and the corresponding DFD is insignificant. As a result, the propagation may stop. However, for region B, the DFD computed with the optimum parameters is still significant. Therefore, the estimation has to be performed individually on regions C and D. The motion of region C is static. whereas the motion of region D involves a vertical displacement. In both cases. the DFD is not significant and the estimation process may stop. The final motion estimation is presented in the image of Fig. 13. The motion is defined by dark lines indicating its magnitude and orientation. As can be seen, the approach combines the advantages of having a high precision on the contour definition (because the tree has been constructed using a color homogeneity criterion) and of dealing initially with large regions, which increases the robustness of the motion estimation. The drawback of the current approach is that it involves a pure split strategy. In the example of Fig. 13, the partition is made of three regions. However, regions A (background) and C (jacket) are both static. The algorithm can be improved by adding a postprocessing step that merges neighboring regions with similar motions (note that the regions that might be merged are regions that were not siblings in the partition tree).

C. Segmentation for Region-Based Coding

Region-based coding relies on segmentation algorithms producing partitions that could be efficiently encoded. Obviously, the criterion used to create the partitions is quite different from the one used in the previous sections. We will assume however that the similarity estimation is not modified. Starting from the same partition tree defined by color homogeneity, we are going to show how an appropriate partition creation strategy can produce partitions suitable for coding.

In the framework of region-based coding, bits are distributed between shape and color information [27], [63]. Color information can be encoded by modification of the well-known DCT, by region-based wavelet transform, or even low-order polynomials. The shape information may be encoded by techniques such as chain code, spline approximation, or the MPEG-4 shape coding tool. In this context, the segmentation should find a partition such that the global distortion is minimized and the coding cost (in terms of bits) is lower than a given budget. It is basically a "rate/distortion" problem. Note that this problem can be seen as a particular case of global optimization under constraint. As discussed in [55], a partition tree is a very attractive representation to perform an optimization in the rate/distortion sense.

The first step consists in analyzing the rate and distortion associated to each region. Assuming that several coding techniques are available, the rate and distortion analysis has to be done for each region and each coding technique. The computation of the distortion is rather straightforward. The squared error between the original and coded frames (i.e., the sum of the squared differences between the values of the original and coded frames for all pixels belonging to the region support) can be used. If each region is encoded separately, the rate is computed by adding the number of bits devoted to color and shape information.

The situation is more complex when regions are jointly encoded. A typical example is when a contour coding technique is used to encode the boundary between two regions in the partition. In such cases, the optimization involves complex dependencies between regions and, in general, the problem is not tractable. A practical solution consists in using an approximation so that individual rates can be assigned independently to regions. For example, if a contour is shared by two regions, the shape rate assigned to each region can be half of the total rate.

The rate/distortion optimization itself relies on the technique discussed in [47], [51]. The problem can be formulated as the minimization of the distortion $\mathcal{D}$ of the image with the restriction that the total cost $\mathcal{R}$ is below a given budget. It has been shown that this problem can be reformulated as the minimization of the Lagrangian $\mathcal{D} + \lambda\mathcal{R}$, where λ is the so-called Lagrange parameter. Both problems have the same solution if we find λ^* such that $\mathcal{R}$ is equal (or very close) to the budget. Therefore, the problem consists in using the partition tree to find a set of regions creating a partition and minimizing $\mathcal{D} + \lambda^*\mathcal{R}$ [55]. Assume in a first step that the optimum λ^* is known. How can the best set of regions be selected?

For each region, the encoding technique corresponding to the best Lagrangian is selected. Then, the definition of the best partition can be done by a bottom-up analysis of the partition tree.[2] One checks if it is better to code the area represented by a node as a single region R or as various independent regions $\{R_i\}$ corresponding to its two children. The selection of the best choice is done by comparing the Lagrangian of R with the sum of the Lagrangians of R_i. If the former is lower than the latter, the node corresponding to R is selected and the selection of children nodes is removed. This procedure is iterated up to the root node. Note that to use this approach, the distortion should be additive over the regions. In our experiments, the squared error has been used. However, any other additive measure can be used. At the end of the procedure, the best partition is defined by the regions corresponding to the selected nodes.

In practice, of course, the optimum λ^* parameter is not known, and the previous bottom-up analysis of the partition tree is embedded in a loop that searches for the best λ parameter. The computation of the optimum λ parameter can be done with a gradient search algorithm. The bottom-up analysis itself is not expensive in terms of computation since the algorithm has simply to perform a comparison of Lagrangians for all nodes of the tree. The part of the algorithm that might be expensive is the computation of the rate and distortion associated to the regions. This computation has to

[2] A top-down analysis process can also be used.

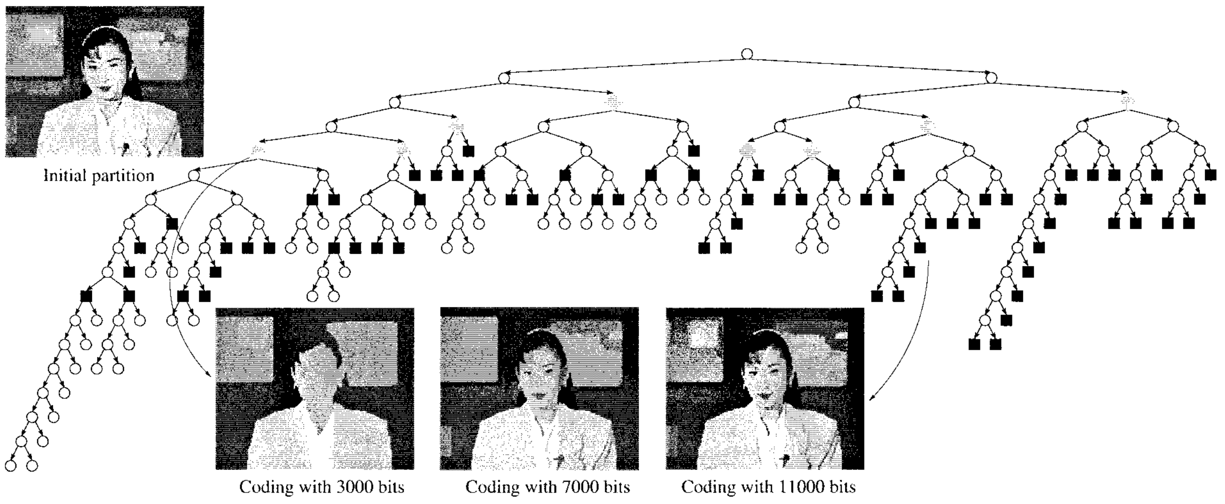

Fig. 14. Examples of rate/distortion optimization: black squares (gray rhombus) in the tree define the optimum solution for 11 000 bits (3000 bits).

be done once, but if the encoding techniques are complex, this tree population may be complex. In our example, it is not the case, since the encoding techniques are very simple.

Fig. 14 shows a binary partition tree corresponding to the initial partition involving 100 regions. If this original partition would have to be transmitted, and assuming that the coding of contours is done by chain code and that a constant color value is transmitted for each region,[3] the cost in term of bits would be equal to 14 000. However, for visualization purposes, this strategy is not optimum. We show in Fig. 14 three examples of coded images at 3000, 7000, and 11 000 bits. As can be seen, the image coded at 11 000 bits is visually equal to the original image. In the case where the transmission rate is very low, higher compression factors may be used while allowing the user to have an idea about the image content and may be useful for progressive transmission. Two coding strategies are shown in the tree representation of Fig. 14. The first one corresponds to the optimum solution for 11 000 bits and is shown with dark squares. The second one shown in the gray rhombus gives the optimum solution at 3000 bits. As can be seen, for low bit rates, the algorithm selects regions close to the root of the tree. For higher bit rates, a larger number of small regions providing details about the image content can be transmitted. Last, Fig. 15 gives the complete rate/distortion curve. One can see the evolution of the visual quality as a function of the number of regions introduced in the partition.

D. Segmentation of Regions with Semantic Meaning

Automatic segmentation of regions with a semantic meaning is a very active area of image analysis. Among this class, face detection is one of the most useful tools given its large number of applications [10]. Several techniques have been proposed to detect and track faces [10], [62]. Nevertheless, new multimedia applications rise a new need: the analysis algorithm should not only detect the position of the face but also really segment it and obtain its actual shape.

[3]A single and extremely simple coding technique is used here in order to highlight the effect of region selection. In practice, more efficient techniques should be used in particular to encode the color.

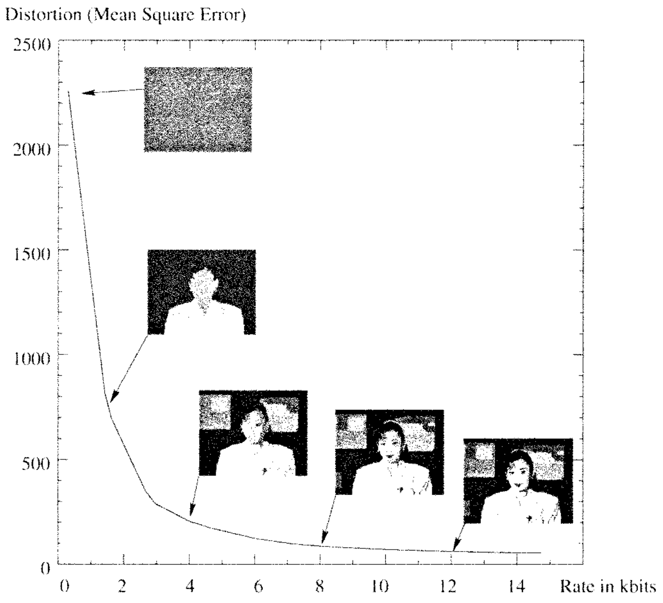

Fig. 15. Rate/distortion curve for the partition tree of Fig. 14.

One of the basic approaches in face detection relies on the notion of eigenspaces [36]. It assumes that the set of all possible face patterns is a low-dimensional linear subspace Ω within the high-dimensional space of all possible image patterns. An image pattern is classified as a face if its distance from the face subspace is below a certain threshold. With this technique, the position of a face in an image can be detected. Information about the face subspace is derived from the analysis of a face data base. The database is made of face images $\{f_n(i,j)\}$ with the same uniform background. For each image, a vector $\vec{x}_n$ of length N is constructed by scanning the lines and columns. The mean vector $\vec{\bar{x}}$ and the covariance matrix Σ are computed from the vector population $\{\vec{x}_n\}$.

Within this context, the partition tree is a very attractive representation not only to detect but also to segment face regions. The main modification with respect to the technique proposed in [36] is that one has to analyze each region R represented by the partition tree and measure its distance to the face subspace Ω [46]. If the distance is small, the region is classified as a face region.

To calculate the distance between a given region R of the partition tree and the face subspace, the first step consists in creating the vector $\vec{x}_R$ representing the region. To this end, the smallest rectangle including the region is considered as an auxiliary image. In the areas outside the region, the uniform data-base background is introduced. The auxiliary image is then scaled so that it matches the size of the data-base images, and the vector $\vec{x}_R$ is extracted by scanning the lines and columns of the scaled image.

The distance is defined as the image likelihood of being a face $P(\vec{x}_R/\Omega)$. The class membership of a scaled image represented by its vector $\vec{x}_R$ is generally modeled as a unimodal Gaussian density

$$P(\vec{x}_R/\Omega) = \frac{\exp\left[-\frac{1}{2}(\vec{x}_R - \bar{\vec{x}})^T \Sigma^{-1} (\vec{x}_R - \bar{\vec{x}})\right]}{(2\pi)^{\frac{N}{2}} |\Sigma|^{\frac{1}{2}}}. \tag{5}$$

The Mahalanobis distance is used as a sufficient statistic for characterizing this likelihood

$$d(\vec{x}_R, \Omega) = (\vec{x}_R - \bar{\vec{x}})^T \Sigma^{-1} (\vec{x}_R - \bar{\vec{x}}). \tag{6}$$

In the case of the *Akiyo* image, the algorithm successfully detects the face region (region D of Fig. 13). Last, note that this section has concentrated on face segmentation. The approach is however fairly general and can be extended to all cases where the semantic of interest can been considered as a visual pattern corresponding to a low-dimensional linear subspace within the high-dimensional space of all possible image patterns. Typical examples include eyes, nose, or even rigid objects [43].

E. Segmentation for Region-Based Retrieval

Image and video sequence indexing aims at describing the original content in such a way that it can be easily searched for and retrieved. Efficient searching is primarily supported by high-level descriptors that address the semantic content. Indeed, in a large number of cases, the user would like to find content where a specific event or a particular object is present. Event and object notions are easily handled by natural languages or keywords with a semantic meaning.

For high-level descriptors, the indexing process consists of defining the semantic entities (objects, events, properties) that are present in the image or in the video and that are useful or of interest. These entities are described by keywords or text and the description is structured so that efficient retrieval is supported. Note that the definition of "useful" or "interesting" entities is a subjective issue. Ideally, one has to foresee the most likely queries and anticipate the set of possible answers.

The drawback of a pure semantic description is its lack of flexibility. Indeed, it is extremely difficult to deal with an unforeseen query. Unforeseen queries often involve an object, an event, or a property that was not judged as being meaningful during the indexing process and therefore was not indexed. Flexibility can be obtained by introducing a low-level description of the signal (sometimes called syntactical description as opposed to the semantical description). This low-level description mainly deals with signal properties such as spatial or temporal regions, color, texture, shape, motion, etc.

Segmentation plays an important role for low-level description. The goal is to define regions either in space or in time to support information retrieval. During the indexing process, it is not possible to define the appropriate analysis resolution. Future queries may address large regions as well as very small details of the image. As a result, the segmentation has to produce more than a single partition. The issue of "universal segmentation" and "application" discussed at the beginning of Section IV is particularly important in the case of indexing since it is not possible to define a single retrieval application.

A possible solution consists in segmenting the image or the video sequence at various levels of resolution and in structuring the set of regions within a partition tree. Of course, the low-level description based on regions should be linked to the semantic description. An example of image description structure is illustrated in Fig. 16. In this example, the low-level description appears as a region tree [Fig. 16(a)]. As can be seen, the region tree defines a set of regions (tree nodes) at various scales and structures them depending on their inclusion relationships. Large regions appear on upper levels of the tree whereas fine details can be found on lower levels. The high-level description is addressed by the object tree [Fig. 16(b)] where nodes represent semantic notions. A link between an object and a region relates a semantic notion to its occurrence in the image. For example, object O_1 refers to region R_3. Object O_2 refers to regions R_5 and R_9 since two TV screens are visible in the image. Note that the region tree can be considered as the "table of contents" of the image. As in the case of a written document, the table of contents is a hierarchical representation that splits the document into elementary pieces. To follow the analogy with written document, the object tree can be viewed as the "index," since it defines a set of potentially interesting items and provides references to where these items are present.

As can be seen, the problem of segmentation for indexing is therefore equivalent to the "similarity estimation" step of Fig. 8. The example of Fig. 16 does not use the tree of Fig. 9. This is because the tree of Fig. 9 is constructed with a very simple homogeneity criterion (color), which is not suitable to produce regions with a high probability of corresponding to semantic notions. For region-based retrieval, the partition tree should be based on much more powerful criteria such as motion trajectory, relative depth, etc. Also, the result of specific preprocessing techniques such as foreground/background detection, face detection, etc., should be taken into account (see Section IV-F). Last, user interaction can also be used to improve the partition tree. Note that in this context, the goal of the user interaction is not to correct a partition or a set of regions [13] but to modify the tree itself (regions and structure). To the authors' knowledge, almost no work has been reported in this direction.

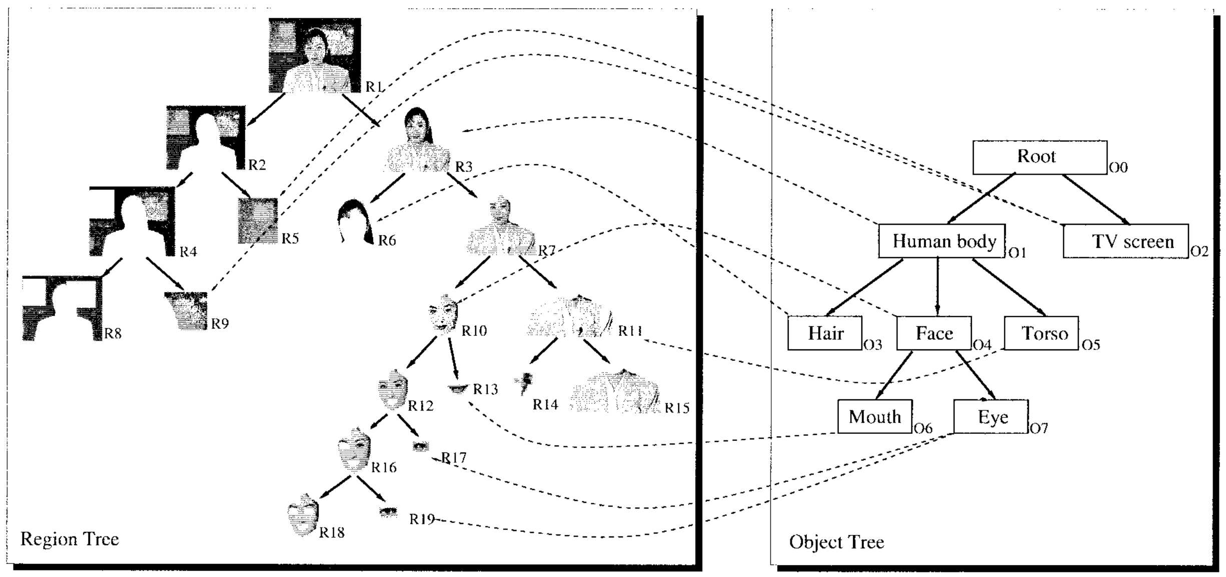

Fig. 16. Example of region and object trees. The region tree defines the spatial organization of regions as well as their inclusion relationship. The object tree is a hierarchical list of items referring to occurrences in the region tree.

Beside segmentation, the indexing process has also to assign region descriptors to each node of the tree. These descriptors typically deal with color, texture, shape, position, and orientation features as well as motion if the image is extracted from a sequence. Note that the tree structure introduces a notion of scalability in the description itself. Indeed, one region is described by its own descriptors but also by the set of descriptors of its children. Assume, for example, that the luminance information of a region is described by a constant value. This is a very crude approximation. However, for a given region R, this approximation can be improved if the set of luminance descriptors of all the regions included in R are considered. The tree structure is also an attractive solution to encode the descriptors. Indeed, the tree defines the correlation and inheritance relationships between descriptors.

F. Similarity Estimation

In the previous sections, we have illustrated the interest of the similarity estimation step of Fig. 8. A single partition tree (except in Section IV-E) has been used to support a large number of segmentation tasks. The goal of this section is to describe possible implementations of the similarity estimation. We will focus exclusively on the creation of a binary partition tree [54].

Several approaches can be followed to create this tree. An attractive solution relies on a region-based merging algorithm that follows a bottom-up approach. Starting from an initial partition, the algorithm recursively merges neighboring regions based on a homogeneity criterion until one region is obtained. As described by Fig. 9, the tree leaves represent the regions of the initial partition. The remaining nodes represent the regions that are obtained by merging the regions associated to the two children regions. In this representation, the root node represents the entire image support. Note that the resulting tree is binary since, at each step, only two neighboring regions are merged.

Using region-based merging algorithms such as [37], [66], [14], [20], and [28], the binary partition tree is created by keeping track of the regions that are merged at each iteration. The homogeneity criterion used in the example of Fig. 9 is based on color similarity. It should be noticed, however, that the homogeneity criterion has not to be restricted to color. For example, if the image for which we create the binary partition tree belongs to a sequence of images, motion information should also be used to generate the tree: in a first stage, regions are merged using a color homogeneity criterion, whereas a motion homogeneity criterion is used to merge regions in the second stage [18], [19]. Fig. 17(a) shows an example of binary partition tree that has been constructed exclusively with a color homogeneity criterion: the region merging order is defined by the Euclidean distance between the mean YUV values of the regions. In this case, it is not possible to concentrate the information about the foreground object within the same subtree. For example, the face mainly appears in the subtree hanging from region A, whereas the helmet regions are located below region D. In practice, the nodes that are close to the root have no clear meaning because they are not homogeneous in color. Fig. 17(b) presents an example of partition tree created with color and motion criteria. The nodes appearing in the lower part of the tree as white circles correspond to the color criterion, whereas the dark squares correspond to the motion criterion. As can be seen, the process starts with the color criterion as in Fig. 17(a), and then, when a given PSNR is reached, it changes to the motion criterion. Using motion information, the face and the helmet now appear as a single region E.

Additional information on previous processing or detection algorithms can also be used to generate the tree in a more robust way. For instance, a mask of an object included in the

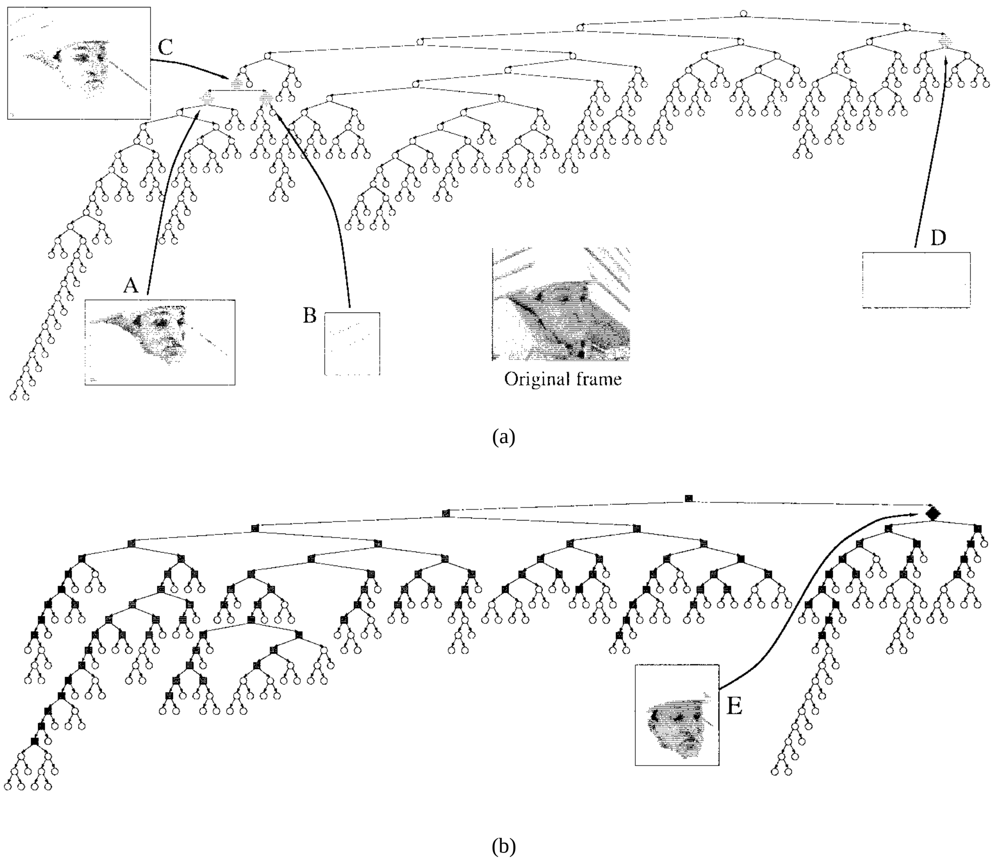

Fig. 17. Examples of creation of binary partition tree. (a) Color homogeneity criterion. (b) Color and motion homogeneity criteria.

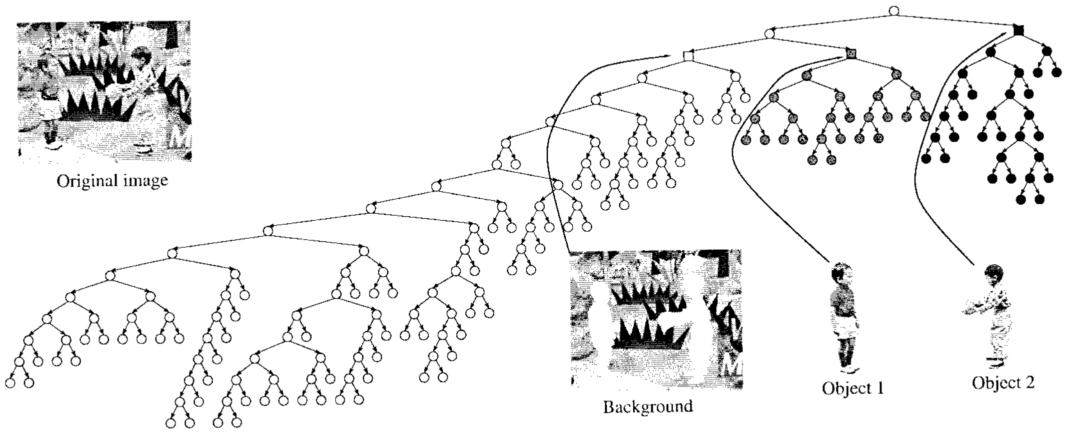

Fig. 18. Example of partition tree creation with restriction imposed with object masks.

image can be used to force the merging algorithm in such a way that the object itself is represented with a node in the tree. Typical examples of such algorithms are face, skin, character, and foreground object detection. An example is illustrated in Fig. 18. Assume for example that the original *Children* image sequence has been analyzed so that masks of the two foreground objects are available. If the merging algorithm is constrained to merge regions within each mask before dealing with remaining regions, the region of support of each mask will be represented as a single node in the resulting partition tree. In Fig. 18, the nodes corresponding to the background and the two foreground objects are represented by squares. The three subtrees further decompose each object into elementary regions.

Note that in most practical cases, the similarity estimation is much more complex to implement than the partition creation, and it may involve a high computational load. Since a single partition tree can support a large number of segmentation tasks, it makes sense to perform the "similarity estimation" once. The resulting partition tree can be stored for subsequent use by different "partition creation" algorithms. A future coding standard may encode the original image together with its partition tree, and, in the case of sequences, the encoded frames may be multiplexed with the corresponding partition trees.

V. Conclusion

This paper has discussed the main issues related to segmentation algorithms that may be used for multimedia applications. After a description of the main processing steps and the corresponding choices, the state of the art in segmentation has been reviewed. Segmentation tools are progressing rapidly, and, for specific applications where the input signal comes from a controlled environment or where the feature to analyze is simple, robust algorithms are already available. In other applications dealing with arbitrary images or video sequences or when the feature to analyze is complex, much work is still needed. One of the difficulties involved in most algorithms is related to the initial low-level representation of the original data. Most algorithms try to progressively convert the pixel-based representation into an initial region-based representation and then act on this region-based representation to obtain the final partition by splitting or merging steps.

The second part of the paper (Section IV) proposes a region-based representation, termed a partition tree, and its corresponding processing strategy. For segmentation, the partition tree representation has many advantages, including the definition of a limited set of regions, the hierarchical structuring of the region set, and the encoding of the similarity between regions by the tree structure. The processing approach involves a *similarity estimation* step followed by a *partition creation* step. This strategy tries to find a compromise between what can be done in a systematic and universal way and what has to be application dependent. It is shown in particular how a single partition tree (except in Section IV-E) created with an extremely simple and generic similarity feature can support a large number of segmentation tasks.

As conclusions, we would like to highlight three areas of work that seem of interest for the future.

Algorithms for the analysis and estimation of overlapping spatial, spatio-temporal, or temporal segments: It is difficult to consider these algorithms as segmentation tools since they do not produce a partition. However, visual data modeling allowing overlap goes one step further toward the goal of taking into account what has happened during the production process (see the introduction). Some tools are already available today, but this area deserves much attention.

New similarity features: Fig. 1 lists the set of feature spaces that are commonly used today. Most of the time either they are very low level or they involve strong restrictions. For video analysis, for instance, the notion of motion is generally constrained to be the motion between two successive frames. Motion trajectory, that is, the motion within a large number of frames, seems to offer a more reliable source of information. Another example is a feature space addressing semantic notions. Currently, a very low number of semantic notions, such as faces or specific rigid objects, can be analyzed.

Shift the complexity from the data to the algorithm: Segmentation is generally considered as a rather complex and time-consuming process. However, in a large number of cases, the complexity does not result from the algorithm but from the amount of pixels and frames to process. In the future, a possible trend may be to use more complex algorithms on a reduced number of elements. The approach described in Section IV can be viewed as a first step toward this goal. The similarity estimation can be considered as preprocessing that reduces the cardinality of the elements to process. Instead of processing $M * N$ pixels, one may work with a few tens or a few hundreds of regions stored in a partition tree.

Acknowledgment

The authors are very grateful to L. Garrido for his contribution to the work reported in Section IV.

References

[1] G. Ahanger and T. Little, "A survey of technologies for parsing and indexing digital video," *J. Visual Commun. Image Represent.*, vol. 7, no. 1, pp. 28–43, Mar. 1996.

[2] A. A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tuncel, and T. Sikora, "Image sequence analysis for emerging interactive multimedia services—The European cost 211 framework," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 802–813, Nov. 1998.

[3] F. Arman, A. Hsu, and M. Y. Chiu, "Image processing on compressed data for large video databases," in *Proc. 1st ACM Int. Conf. Multimedia Conf.*, Aug. 1993, pp. 267–272.

[4] J. Besag, "Spatial interaction and the statistical analysis of lattice systems (with discussion)," *J. Roy. Statist. Soc.*, series B, vol. 34, pp. 75–83, 1972.

[5] L. Bonnaud and C. Labit, "Multiple occluding objects tracking using a nonredundant boundary-based representation for image sequence interpolation after decoding," in *Proc. Int. Conf. Image Processing*, Oct. 1997, vol. II, pp. 426–429.

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

[7] R. Castagno, T. Ebrahimi, and M. Kunt, "Video segmentation based on multiple features for interactive multimedia applications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, no. 5, pp. 562–571, Sept. 1998.

[8] E. Chalom and V. Bove, "Segmentation of an image sequence using multi-dimensional image attributes," in *Proc. Int. Conf. Image Processing*, Lausanne, Switzerland, 1996, pp. 525–528.

[9] R. Chellappa and A. Jain, *Markov Random Fields. Theory and Application.* New York: Academic, 1993.

[10] R. Chellappa, C. Wilson, and S. Sirohey, "Human and machine recognition of faces: A survey," *Proc. IEEE*, vol. 83, pp. 705–740, May 1995.

[11] J. G. Choi, M. Kim, M. H. Lee, and C. Ahn, "Automatic segmentation based on spatio-temporal information," Tech. Rep. ISO/IEC JTC 1/SC 29/WG 11 MPEG97/2091, Bristol, U.K., Apr. 1997.

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

[13] P. Correia and F. Pereira, "The role of analysis in content-based video coding and indexing," *Signal Process.*, vol. 66, no. 2, pp. 125–142, Apr. 1998.

[14] J. Crespo, R. W. Shafer, J. Serra, C. Gratin, and F. Meyer, "A flat zone approach: A general low-level region merging segmentation method," *Signal Process.*, vol. 62, no. 1, pp. 37–60, Oct. 1997.

[15] Y. Deng and B. S. Manjunath, "Netra-v: Toward an object-based video representation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 616–627, Sept. 1998.

[16] F. Dufaux and F. Moscheni, "Segmentation-based estimation for second generation video coding techniques," in *Video Coding: The Second*

Generation Approach, L. Torres and M. Kunt, Eds. Norwell, MA: Kluwer Academic, 1996, pp. 219–264.

[17] J. L. Dugelay and H. Sanson, "Differential methods for the identification of 2D and 3D motion models in image sequences," *Signal Process. Image Commun.*, vol. 7, pp. 105–127, 1995.

[18] P. E. Eren, Y. Altunbasak, and A. M. Tekalp, "Region-based affine motion segmentation using color information," in *Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP)*, Munich, Germany, Apr. 1997, vol. 4, pp. 3005–3008.

[19] L. Garrido and P. Salembier, "Region based analysis of video sequences with a general merging algorithm," in *Proc. IX Eur. Signal Processing Conf. (EUSIPCO'98)*, Rhodes, Greece, Sept. 8–11, 1998, vol. III, pp. 1693–1696.

[20] L. Garrido, P. Salembier, and D. Garcia, "Extensive operators in partition lattices for image sequence analysis," *Signal Processing*, vol. 66, no. 2, pp. 157–180, Apr. 1998.

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

[22] C. Gu, "Multivalued morphology and segmentation-based coding," Ph.D. dissertation, Swiss Federal Institute of Technology, 1995, no. 1452.

[23] C. Gu and M. C. Lee, "Semiautomatic segmentation and tracking of semantic video objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 572–584, Sept. 1998.

[24] M. Vetterli, H. Radha, and R. Leonardi, "Image compression using binary space partitioning trees," *IEEE Trans. Image Process.*, vol. 5, pp. 1610–1624, Dec. 1996.

[25] I. Kompatsiaris, D. Tzovaras, and M. G. Strintzis, "3-D model-based segmentation of videoconference image sequences," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 547–562, Sept. 1998.

[26] S. Kruse, A. Graffunder, and S. Askar, "A new tracking scheme for semi-automatic video object segmentation," in *Proc. Workshop Image Analysis for Multimedia Application Services (WIAMIS'99)*, Berlin, Germany, June 1999, pp. 93–96.

[27] M. Kunt, A. Ikonomopoulos, and M. Kocher, "Second generation image coding techniques,"*Proc. IEEE*, vol. 73, pp. 549–575, Apr. 1985.

[28] B. Marcotegui, "Segmentation algorithm by multicriteria region merging," in *Proc. 3rd Workshop Mathematical Morphology and Its Applications to Image Processing*, P. Maragos, R. W. Schafer, and M. A. Butt, Eds. Atlanta, GA: Kluwer Academic, 1996, pp. 313–320.

[29] B. Marcotegui, P. Correia, F. Marques, R. Mech, R. Rosa, M. Wollborn, and F. Zanoguera, "A video object generation tool allowing friendly user interaction," in *Proc. IEEE Int. Conf. Image Processing (ICIP'99)*, Kobe, Japan, Oct. 1999, p. 26.

[30] F. Marqués and J. Llach, "Tracking of generic objects for video object generation," in *Proc. Int. Conf. Image Processing (ICIP'98)*, Chicago, IL, Oct. 4–7, 1998, p. WE.

[31] F. Marqués, M. Pardàs, and P. Salembier, "Coding-oriented segmentation of video sequences," in *Video Coding: The Second Generation Approach*, L. Torres and M. Kunt, Eds. Norwell, MA: Kluwer Academic, pp. 79–124, 1996.

[32] R. Mech and M. Wollborn, "A noise robust method for 2-D shape estimation of moving objects in video sequences considering a moving camera," *Signal Process.*, vol. 66, no. 2, pp. 203–218, Apr. 1998.

[33] T. Meier and K. N. Ngan, "Automatic segmentation of moving objects for video object plane generation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 525–538, Sept. 1998.

[34] J. Meng, Y. Juan, and S. F. Chang, "Scene change detection in a MPEG compressed video sequence," in *Proc. IST/SPIE*, Feb. 1995, vol. 2419, pp. 105–109.

[35] F. Meyer and S. Beucher, "Morphological segmentation," *J. Visual Commun. Image Represent.*, vol. 1, no. 1, pp. 21–46, Sept. 1990.

[36] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 696–710, July 1997.

[37] O. Morris, M. Lee, and A. Constantinidies, "Graph theory for image analysis: An approach based on the shortest spanning tree," *Proc. Inst. Elect. Eng.*, vol. 133, no. 2, pt. F, pp. 146–152, Apr. 1986.

[38] MPEG, MPEG-4: Applications document, Tech. Rep. ISO/IEC JTC1/SC29/WG11/w2724, MPEG, Seoul, Korea, Mar. 1999.

[39] MPEG, MPEG-4: Requirements document, Tech. Rep. ISO/IEC JTC1/SC29/WG11/w2723, MPEG, Seoul, Korea, Mar. 1999.

[40] MPEG, MPEG-7: Applications document, Tech. Rep. ISO/IEC JTC1/SC29/WG11/w2860, MPEG, Vancouver, Canada, July 1999.

[41] MPEG, MPEG-7: Requirements document, Tech. Rep. ISO/IEC JTC1/SC29/WG11/w2859, MPEG, Vancouver, Canada, July 1999.

[42] A. Nagasaka and Y. Tanaka, "Automatic video indexing and full-video search for object appearances," in *Visual Database Systems II*, E. Knuth and L. M. Wegner, Eds. Amsterdam, the Netherlands: Elsevier, 1992, pp. 113–127.

[43] S. K. Nayar, S. A. Nene, and H. Murase, "Subspace methods for robot vision," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 750–758, Oct. 1996.

[44] 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.

[45] J. Odobez and P. Bouthemy, "Robust multiresolution estimation of parametric motion models," *J. Visual Commun. Image Represent.*, vol. 6, no. 4, pp. 348–365, Dec. 1995.

[46] ——, "Direct incremental model-based image motion segmentation for video analysis," *Signal Processing*, vol. 66, no. 2, pp. 143–156, Apr. 1998.

[47] A. Ortega, K. Ramchandran, and M. Vetterli, "Optimal buffer-constrained source quantization and fast approximations," in *Proc. IEEE Int. Symp. Circuits and Systems*, May 1992, vol. 1, pp. 223–229.

[48] M. Pardàs, "Video object segmentation introducing depth and motion information," in *Proc. Int. Conf. Image Processing (ICIP'98)*, Chicago, IL, Oct. 4–7, 1998, p. WE.

[49] M. Pardàs and P. Salembier, "Time-recursive segmentation of image sequences," in *Proc. EUSIPCO'94, VII Eur. Signal Processing Conf.*, M. Holt, C. Cowan, P. Grant, and W. Sandham, Eds. Edinburgh, U.K., 1994, pp. 18–21.

[50] T. Pavlidis, *Structural Pattern Recognition*. New York: Springer-Verlag, 1977, ch. 4, pp. 68–70.

[51] K. Ramchandran and M. Vetterli, "Best wavelet packet bases in a rate-distortion sense," *IEEE Trans. Image Process.*, vol. 2, pp. 160–175, Apr. 1993.

[52] E. Reusens, "Joint optimization of representation model and frame segmentation for generic video compression," *Signal Process.*, vol. 46, pp. 105–117, Sept. 1995.

[53] P. Salembier, P. Brigger, J. R. Casas, and M. Pardàs, "Morphological operators for image and video compression," *IEEE Trans. Image Process.*, vol. 5, pp. 881–898, June 1996.

[54] P. Salembier and L. Garrido, "Binary partition tree as an efficient representation for filtering, segmentation and information retrieval," in *Proc. IEEE Int. Conf. Image Processing (ICIP'98)*, Chicago, IL, Oct. 4–7, 1998, vol. TU.

[55] P. Salembier, F. Marqués, M. Pardàs, R. Morros, I. Corset, S. Jeannin, L. Bouchard, F. Meyer, and B. Marcotegui, "Segmentation-based video coding system allowing the manipulation of objects," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 60–74, Feb. 1997.

[56] P. Salembier, A. Oliveras, and L. Garrido, "Anti-extensive connected operators for image and sequence processing," *IEEE Trans. Image Process.*, vol. 7, pp. 555–570, Apr. 1998.

[57] P. Salembier, L. Torres, F. Meyer, and C. Gu, "Region-based video coding using mathematical morphology," *Proc. IEEE*, vol. 83, pp. 843–857, June 1995.

[58] B. Shahraray, "Scene change detection and content-based sampling of video sequences," in *Proc. IST/SPIE*, Feb. 1995, vol. 2419, pp. 10–22.

[59] E. Steinbach, P. Eisert, and B. Girod, "Motion-based analysis and segmentation of image sequences using 3-D scene models," *Signal Process.*, vol. 66, no. 2, pp. 233–248, Apr. 1998.

[60] C. Stiller, "Object-based estimation of dense motion fields," *IEEE Trans. Image Process.*, vol. 6, pp. 234–250, Apr. 1997.

[61] G. Sudhir and C. M. L. John, "Video annotation by motion interpretation using optical flow streams," *J. Visual Commun. Image Represent.*, vol. 7, no. 4, pp. 354–368, Dec. 1996.

[62] K. K. Sung and T. Poggio, "Example-based learning for view-based human face detection," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 39–51, Jan. 1998.

[63] L. Torres and M. Kunt, Eds., *Video Coding: The Second Generation Approach*. Norwell, MA: Kluwer Academic, 1996.

[64] V. Vilaplana, F. Marqués, P. Salembier, and L. Garrido, "Region-based segmentation and tracking of human faces," in *Proc. IX Eur. Signal Processing Conf. (EUSIPCO'98)*, Rhodes, Greece, Sept. 8–11, 1984, vol. I, pp. 311–314.

[65] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE, Trans. Pattern Anal. Machine Intell.*, vol. 39, pp. 1845–1855, Dec. 1991.

[66] T. Vlachos and A. Constantinidies, "Graph-theoretical approach to color picture segmentation and contour classification," *Proc. Inst. Elect. Eng.*, Feb. 1993, vol. 140, pt. I, no. 1, pp. 36–45.

[67] D. Wang, "Unsupervised video segmentation based on watersheds and temporal tracking," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 539–546, Sept. 1998.

[68] P. Willemin, T. Reed, and M. Kunt, "Image sequence coding by split and merge," *IEEE Trans. Commun.*, vol. 39, pp. 1845–1855, Dec. 1991.

[69] B. L. Yeo and B. Liu, "Rapid scene analysis on compressed video," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 5, pp. 533–544, Dec. 1995.

[70] H. Zhang, A. Kankanhalli, and S. Smoliar, "Automatic partitioning of full-motion video," *ACM/Springer Multimedia Syst.*, vol. 1, no. 1, pp. 10–28, 1993.

[71] D. Zhong and S. F. Chang, "Amos: An active system for MPEG-4 video object segmentation," in *Proc. Int. Conf. Image Processing*, Chicago, IL 1998.

P. Salembier (M'96) received degrees from the Ecole Polytechnique, Paris, France, in 1983 and the Ecole Nationale Supérieure des Télécommunications, Paris, France, in 1985. He received the Ph.D. from the Swiss Federal Institute of Technology (EPFL), Lausanne, in 1991.

He was a Postdoctoral Fellow at the Harvard Robotics Laboratory, Cambridge, MA, in 1991. From 1985 to 1989, he worked at Laboratoires d'Electronique Philips, Limeil-Brevannes, France, in the fields of digital communications and signal processing for HDTV. In 1989, he joined the Signal Processing Lab of EPFL to work on image processing. At the end of 1991, after a stay at the Harvard Robotics Laboratory, he joined the Polytechnic University of Catalonia, Barcelona, Spain, where he is currently Associate Professor. He is lecturing on the area of digital signal and image processing. His current research interests include image and sequence coding, compression and indexing, image modeling, segmentation problems, video sequence analysis, mathematical morphology, and nonlinear filtering. He is involved in the definition of the MPEG-7 standard ("Multimedia Content Description Interface") where he is chairing the "Multimedia Description Scheme" group. He was an Area Editor of the *Journal of Visual Communication and Image Representation* from 1995 until 1998 and an AdCom officer of the European Association for Signal Processing in charge of the edition of the newsletter from 1994 until 1999. He has been a Guest Editor of special issues of *Signal Processing* on "Mathematical Morphology" (1994) and "Video Sequence Analysis" (1998). He is also co-editing a special issue of *Signal Processing: Image Communication* on the MPEG-7 proposals that were recently submitted for evaluation. He is Deputy Editor of *Signal Processing.*

F. Marqués (S'91–M'93) received the Ph.D. degree from the Polytechnic University of Catalunya (UPC), Barcelona, Spain, in 1992.

From 1989 to 1990, he worked in the Swiss Federal Institute of Technology in Lausanne (EPFL) in the group of "Digital Image Sequence Processing and Coding." In 1990, he joined the Department of Signal Theory and Communications of UPC. From June 1991 to September 1991, he was with the Signal and Image Processing Institute at the University of California, Los Angeles. Since 1995, he has been an Associate Professor at UPC and, since 1997, Associate Dean for International Relations. His current research interests include still-image and sequence analysis and segmentation, image sequence coding, motion estimation and compensation, mathematical morphology, and biomedical applications. In the area of image coding and representation, he has been a very active partner in the MPEG-4 standard process, mainly through the European project MoMuSys. In MoMuSys, he was a Work Package Leader of Video Algorithms, which, among other tasks, implemented the MPEG4 VM. Since 1994, he has been a member of the EURASIP AdCom, and since 1996, he has been Associate Editor of the *Journal of Electronic Imaging* in the area of image communications. He is author or co-author of more than 50 publications that have appeared as journal papers, proceeding articles, and book chapters. He has received three international patents.

Dr. Marqués received the Spanish Best Ph.D. Dissertation on Electrical Engineering Award in 1992.