PERGAMON

Pattern Recognition 35 (2002) 563–579

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Visual segment tree creation for MPEG-7 Description Schemes

Philippe Salembier[a,*], Joan Llach[b], Luis Garrido[a]

[a] *Universitat Politècnica de Catalunya, Jordi Girona, 1-3, 08034 Barcelona, Spain*
[b] *Laboratoires d'Electronique Philips (LEP), 22 avenue Descartes, F-94453 Limeil-Brevannes, France*

Received 16 November 2000; accepted 16 November 2000

Abstract

This paper deals with the creation of visual segment trees involved in MPEG-7 Description Schemes. After a brief overview of MPEG-7 Description Schemes in general and of the Segment Description Scheme in particular, tools allowing the creation of segment trees are discussed. The emphasis of the paper is on the creation of hierarchical relationship among segments. For this purpose, it is proposed to create a Binary Partition Tree in a first step and to restructure the tree in a second step. This approach is both efficient and flexible. Several examples involving spatial and temporal partition trees are presented. The proposed approach is appropriate to create both *tables of contents* and *indexes*. © 2001 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: MPEG-7; Multimedia description schemes; Segment trees; Hierarchical segmentation; Indexing; Retrieval; Browsing; Partition trees

1. Introduction

The goal of the MPEG-7 standard [1] is to allow interoperability among devices that deal with audio–visual content description. The standard will involve four types of normative elements: descriptors (Ds), description schemes (DSs), description definition language (DDL), and encoding tools for descriptions. A descriptor is a representation of a feature that characterizes the audio–visual content. A description scheme specifies the structure and semantics of the relationships between its components, which may be both descriptors and description schemes. The description definition language allows the specification of description schemes and descriptors. It also allows the extension and modification of existing Description Schemes. Finally, description encoding schemes are needed to satisfy requirements such as compression efficiency, error resilience, random access, etc.

Descriptors deal with low-level features such as color, texture, motion, audio energy, etc., as well as high-level features such as semantic description of objects and events, production process or information about the storage media, etc. It is expected that most descriptors corresponding to low-level features will be instantiated by automatic analysis tools whereas human interaction will be used for most high-level descriptors. DSs combine individual descriptors as well as DSs within common structures and define relationships among them. As for descriptors, the instantiation of the relationships can rely on automatic tools or on human interaction. The goal of this paper is to discuss hierarchical relationships that are involved in the main MPEG-7 tool used to represent the structure of the AV content, *the segment DS*, and to propose a set of simple analysis tools to instantiate them.

The organization of this paper is as follows. Section 2 briefly reviews the current MPEG-7 DS and focuses in particular on the Segment DS. Analysis tools are

* Corresponding author. Tel.: +34-93-401-7404; fax: +34-93-401-6447.
E-mail address: philippe@gps.tsc.upc.es (P. Salembier).

PII: S0031-3203(01)00060-7

discussed in Section 3 and conclusions are reported in Section 4.

2. Overview of MPEG-7 description schemes

2.1. General overview of the MPEG-7 description tools

The elements, Ds and DSs, described in this section are mainly structured on the basis of the functionality they provide. An overview of the organization of MPEG-7 description tools is shown in Fig. 1.

At the lower level, basic elements can be found. They deal with basic data-types, mathematical structures, linking and media localization tools as well as basic DSs, that are found as elementary components of more complex DSs. Based on this lower level, content description and management elements can be defined. These elements describe the content from several viewpoints. Currently, five viewpoints are defined: Creation & Production, Media, Usage, Structural aspects and Conceptual aspects. The three first elements address, primarily, information related to the management of the content whereas the two last ones are mainly devoted to the description of perceivable information (content description). The functionality of each set of elements is more precisely defined as follows:

- *Creation and production information*: It provides information about the creation and production of the content. Typical features include title, creator, classification, purpose of the creation, etc. This information is most of the time author-generated since it cannot be extracted from the content.
- *Usage meta information*: This set of elements is related to the usage of the content. Typical features involve rights holders, access right, usage history, and financial information. This information may very likely be subject to change during the lifetime of the AV content.
- *Media description*: It describes the storage media, in particular the storage format and the encoding of the AV content. It also provides elements for the identification of the media. Note that several instances for the same AV content can be described.
- *Structural aspects*: These elements address the description of the AV content from the viewpoint of its structure. The description is organized around segments that represent physical spatial, temporal or spatio-temporal components of the AV content. Segments may be organized in a hierarchical structure and can define Tables of Content or Indexes. Each segment may be described by signal-based features (color, texture, shape, motion, audio features) and some elementary semantic information using simple textual annotation.
- *Conceptual aspects*: It is the description of the AV content from the viewpoint of its conceptual notions. It involves entities such as objects, events, abstract concepts and their relationships. The structural and semantic descriptions may relate by a set of links. The links relate semantic notions with their instances described by segments.

The five sets of elements are presented here as separate entities. In practice however, they are interrelated and may be partially included in each other. For example, Media, Usage or Creation & Production elements can be attached to individual segments involved in the structural description of the content. Depending on the application, some areas of the content description will have to be emphasized and others may be minimized or discarded.

Besides the direct description of the content provided by the five sets of elements previously described, tools are also defined for *navigation and access*. Browsing is supported by the summary elements. Two navigation modes are defined: hierarchical and sequential. In the hierarchical mode, the information is organized into a succession of levels, each describing the audio–visual content at a specific level of detail. In general, levels closer to the root of the hierarchy provide coarse summaries and levels further away from the root provide more detailed summaries. The sequential summary is used to specify a single audio–visual summary, which contains a sequence of images or video frames, possibly synchronized with audio, composing a slide show or audio–visual skim. Furthermore, information about possible variations of the content is also given. Variations of the AV content can replace the original, if necessary, to adapt different multimedia presentations to the capabilities of the client terminals, network conditions or user preferences.

Content organization elements address the organization of a set of AV documents by classification, by definition of collections and by modeling. The Collection DS groups segments, events, and/or objects from multiple descriptions in collection clusters, specify properties common to all elements in a cluster and describes statistics of their attribute values.

Finally, the last set of tools is the *User Preference DS*. It is used to describe users' preferences pertaining to the consumption of multimedia material. The correspondence between user preferences and content descriptions facilitates accurate and efficient personalization of content access and content consumption.

2.2. Segment description scheme

Let us detail more precisely the segment DS. It addresses the description of the physical and logical aspects of AV content. Segment DSs may be used to form segment trees to define the structure of the AV content, i.e.

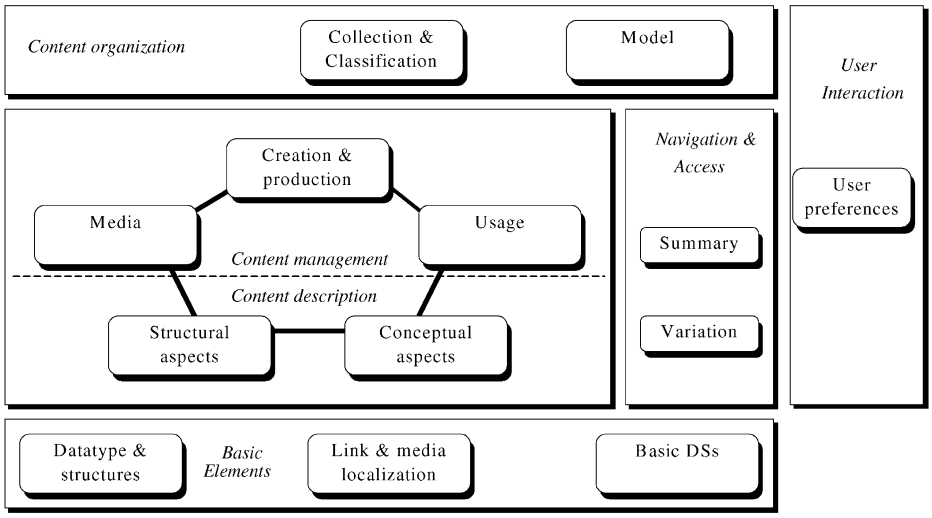

Fig. 1. MPEG-7 Multimedia description schemes.

a table of contents or an index. MPEG-7 also specifies a SegmentRelation DS. It is used to describe temporal, spatial, and spatio-temporal relationships, among others, between segments that are not described by the tree structures. However, in this paper we will focus on Segment tree structures.

A segment represents a section of an AV content item. The Segment DS is an abstract class (in the sense of object-oriented programming). It has five major subclasses: AudioVisualSegment DS, AudioSegment DS, StillRegion DS, MovingRegion DS and VideoSegment DS. Therefore, it may have both spatial and temporal properties. A temporal segment may be a set of samples in an audio sequence, represented by an AudioSegment DS, a set of frames in a video sequence, represented by a VideoSegment DS or a combination of both represented by an AudioVisualSegment DS. A spatial segment may be a region in an image or a frame in a video sequence, represented by a StillRegion DS. Finally, a spatio-temporal segment may correspond to a moving region in a video sequence, represented by a MovingRegion DS. The Segment DS is abstract and cannot be instantiated on its own: it is used to define the common properties of its subclasses. Any segment may be described by creation information, usage information, media information and textual annotation. Moreover, a segment can be decomposed into subsegments through the Segment-Decomposition DS.

A segment is not necessarily connected, but may be composed of several unconnected components. Connectivity refers here to both spatial and temporal domains. A temporal segment (VideoSegment and AudioSegment) is said to be temporally connected if it is a sequence of continuous video frames or audio samples. A spatial segment (StillRegion) is said to be spatially connected if it is a group of connected pixels. A spatio-temporal segment (MovingRegion) is said to be spatially and temporally connected if the temporal segment where it is instantiated is temporally connected and if each one of its temporal instantiations in a frame is spatially connected (note that this is not the classical connectivity in a 3D space). Fig. 2 illustrates several examples of temporal or spatial segments and their connectivity. Figs. 2(a) and (b) illustrate a temporal and a spatial segment composed of a single connected component. Figs. 2(c) and (d) illustrate a temporal and a spatial segment composed of three connected components. Note that, in the latter case, the descriptors and DSs attached to the segment are global to the union of the connected components building the segment. At this level, it is not possible to describe individually the connected components of the segment. If connected components have to be described individually, then the segment has to be decomposed into various subsegments corresponding to its individual connected components.

The Segment DS is recursive, i.e., it may be subdivided into sub-segments, and thus may form a hierarchy (tree). The resulting segment tree is used to define the media source, the temporal and/or spatial structure of the AV content. For example, a video program may be temporally segmented into scenes, shots, and micro-segments; a table of contents may thus be generated based on this structure. Similar strategies can be used for spatial and spatio-temporal segments. A segment may also be decomposed into various media sources such as various audio tracks or various viewpoints from several cameras. The hierarchical decomposition is useful to design efficient search strategies (global search to local search). It

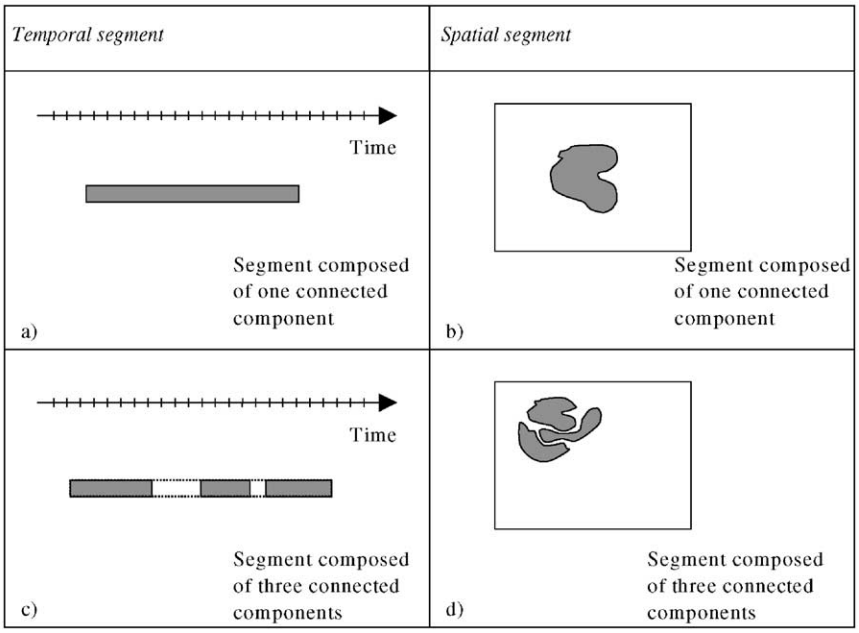

Fig. 2. Examples of segments: (a) and (b) segments composed of a single connected component; (c) and (d) segments composed of three connected components.

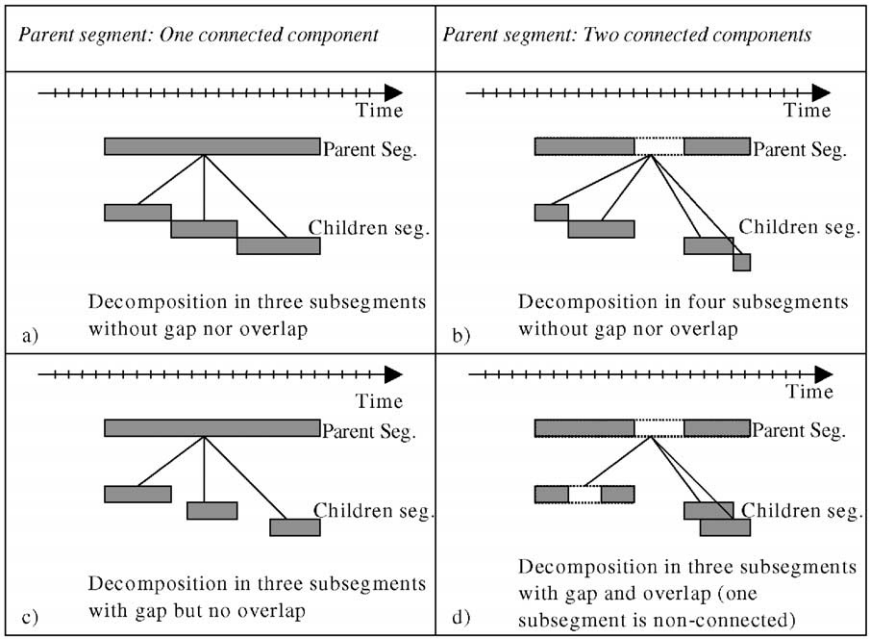

Fig. 3. Example of temporal segment decomposition: (a) and (b) segment decomposition without gap or overlap; (c) and (d) segment decomposition with gap or overlap.

also allows the description to be scalable: a segment may be described by its direct set of descriptors and DSs, but it may also be described by the union of the descriptors and DSs that are related to its sub-segments. Note that a segment may be subdivided into sub-segments of different types, e.g. a video segment may be decomposed in moving regions that are themselves decomposed in static regions.

Table 1
Specific features describing the various segments. All segments may also be described by creation, usage, media information and textual annotation

Features	Video segment	Still region	Moving region	Audio segment
Time	X		X	X
Color, Texture	X	X	X	
Spatial geometry		X	X	
Editing effect	X		X	X
Motion	X		X	
Camera motion	X		X	
Mosaic	X		X	
Audio features				X

As it is done in a spatio-temporal space, the decomposition has to be described by a set of attributes defining the type of subdivision: temporal, spatial, spatio-temporal or media sources. Moreover, the spatial and temporal subdivisions may leave gaps and overlaps between the subsegments. Several examples of decompositions are described for temporal segments in Fig. 3. Fig. 3(a) and (b) describe two examples of decompositions without gaps or overlaps (partition in the mathematical sense). In both cases, the union of the children corresponds exactly to the temporal extension of the parent, even if the parent is itself non-connected (see the example of Fig. 3(b)). Fig. 3(c) shows an example of decomposition with gaps but no overlaps. Finally, Fig. 3(d) illustrates a more complex case where the parent is composed of two connected components and its decomposition creates three children: the first one is itself composed of two connected components, the two remaining children are composed of a single connected component. The decomposition allows gap and overlap. Note that, in any case, the decomposition implies that the union of the spatio-temporal space defined by the children segments is included in the spatio-temporal space defined by their ancestor segment (children are contained in their ancestors).

Specific features describing specialized segment types are reported in Table 1. Most of the descriptors corresponding to these features can be extracted automatically from the original content. For this purpose, a large number of tools have been reported in the literature (see Ref. [2] and the reference therein). Less attention has been devoted to the design of tools allowing the instantiation of the decomposition involved in the Segment DS. This decomposition can be viewed as a hierarchical segmentation problem where elementary entities (region, video segment, etc.) have to be defined and structured by an inclusion relationship within a tree. In the following section, we propose and discuss a possible strategy to create VideoSegment and StillRegion trees.

3. Tools for segment tree creation

3.1. Visual Segment Tree creation

The extraction of individual spatial or temporal regions and their organization within a tree structure can be viewed as a hierarchical segmentation problem also known as a partition tree creation [2]. The strategy we propose here involves three steps illustrated in Fig. 4: The first step is a conventional segmentation; its goal is to produce an initial partition with a rather high level of details. Depending on the nature of the segment tree, this initial segmentation can be a shot detection algorithm (for VideoSegment) or a spatial segmentation following a color homogeneity criterion (for StillRegion). The second step is the creation of a Binary Partition Tree (BPT). Combining the segments (VideoSegment or StillRegion) created in the first step, the BPT defines the set of segments to be indexed and encodes their similarity with the help of a binary tree. Finally, the third step restructures the binary tree into an arbitrary tree. In this proposal, an intermediate representation relying on a BPT is used because, on the one hand, there exist efficient strategies to create the binary tree and, on the other hand, this representation provides enough flexibility to create the final tree.

The first step is a classical segmentation. Efficient spatial segmentation techniques can be found in Refs. [3–7] Temporal segmentation algorithms are described in Refs. [8–10]. Finally, an overview can be found, for example, in Ref. [2] and the references contained therein. The following sections focus on the two last steps: BPT creation and tree structuring.

3.2. Binary Partition Tree creation

The second step of the Segment Tree creation is the computation of a BPT [11]. Each node of the tree represents connected components either in space (StillRegion) or in time (VideoSegment). The leaves of the tree represent the segments defined by the initial segmentation step. The remaining nodes represent segments that are obtained by merging segments represented by the children. This representation should be considered as a compromise between representation accuracy and processing efficiency. Indeed, all possible mergings of the initial segments are not represented in the tree. Only the most "useful" merging steps are represented. However, the main advantage of the binary tree representation is that efficient techniques are available for its creation (see Refs. [12,13] for example) and it conveys enough information about segment similarity to construct the final Segment Tree.

3.2.1. Spatial segments

In this section we describe the creation of spatial binary partition trees (temporal trees will be discussed in

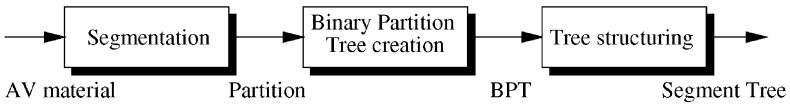

Fig. 4. Outline of the Segment Tree creation.

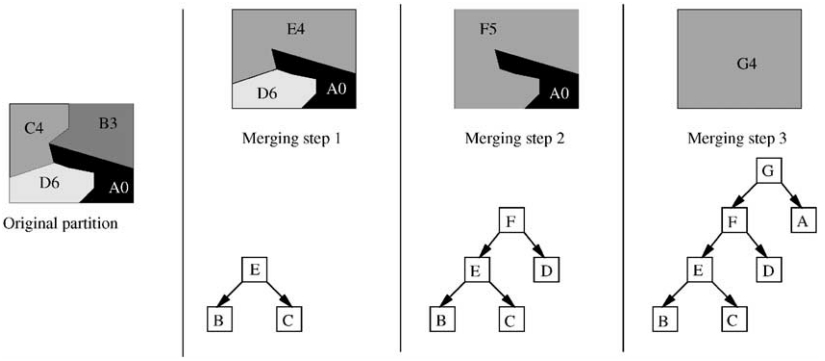

Fig. 5. Example of Binary Partition Tree creation with a region merging algorithm.

Section 3.2.2). The BPT should be created in such a way that the most "useful" segments are represented. This issue can be application dependent. However, a possible solution, suitable for a large number of cases, is to create the tree by keeping track of the merging steps performed by a segmentation algorithm based on segment merging (see Refs. [12,14,15]). In the following, this information is called the *merging sequence*. Let us consider a simple case with StillRegions: starting from the initial partition produced in the first step, the algorithm merges neighboring regions following a homogeneity criterion until a single region is obtained. An example is shown in Fig. 5. The original partition involves four regions. The regions are indicated by a letter and the number indicates the mean gray level value. The algorithm merges the four regions in three steps. In the first step, the pair of most similar regions, B and C, are merged to create region E. Then, region E is merged with region D to create region F. Finally, region F is merged with region A and this creates region G corresponding to the region of support of the whole image. In this example, the merging sequence is: $(B,C)|(E,D)|(F,A)$. This merging sequence progressively defines the Binary Partition Tree as shown in Fig. 5. As can be seen, the merging is done iteratively between pairs of regions. The resulting tree is therefore binary.

A simple way to create a BPT is to use a color homogeneity criterion. At each step of the merging algorithm, the most similar regions in terms of color are merged together. The distance between regions R_1 and R_2 is defined by the following expression:

$$d = N_1||M_{R_1} - M_{R_1\cup R_2}||_2 + N_2||M_{R_2} - M_{R_1\cup R_2}||_2, \qquad (1)$$

where N_1 and N_2 are the numbers of pixels of regions R_1 and R_2; $||.||_2$ denotes the $\mathscr{L}_2$ norm; and M_R represents the model for region R. The model is a mathematical function used to represent the region. In our example, it consists of three constant values describing the YUV components of the region. The distance between regions given by Eq. (1) defines the order of merging of regions. The interest of this merging order, compared to other classical criteria, is discussed in Ref. [12].

However, it should be noticed that the homogeneity criteria (or distance d) is not always restricted to color. For example, if the image for which we create the BPT belongs to a sequence of images, motion information can 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 in the second stage [16,17]. Fig. 6 presents an example of BPT created with color and motion criteria on the *Foreman* sequence. The nodes appearing in the lower part of the tree as white circles correspond to the color criterion, whereas the gray squares correspond to the motion criterion. The process starts with a color criterion and then, when a given peak signal-to-noise ratio (PSNR) is reached, it changes to the motion criterion. As can be seen, regions that are homogeneous in motion such as the face and helmet are represented by a single node (B) in the tree.

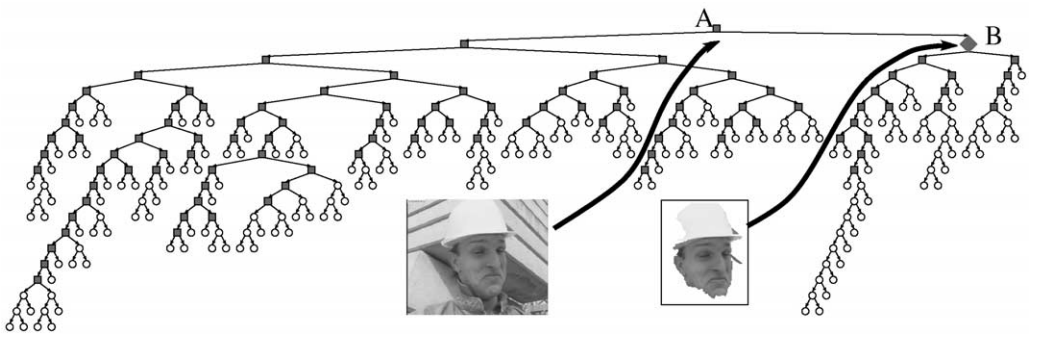

Fig. 6. Examples of creation of Binary Partition Tree with color and motion homogeneity criteria.

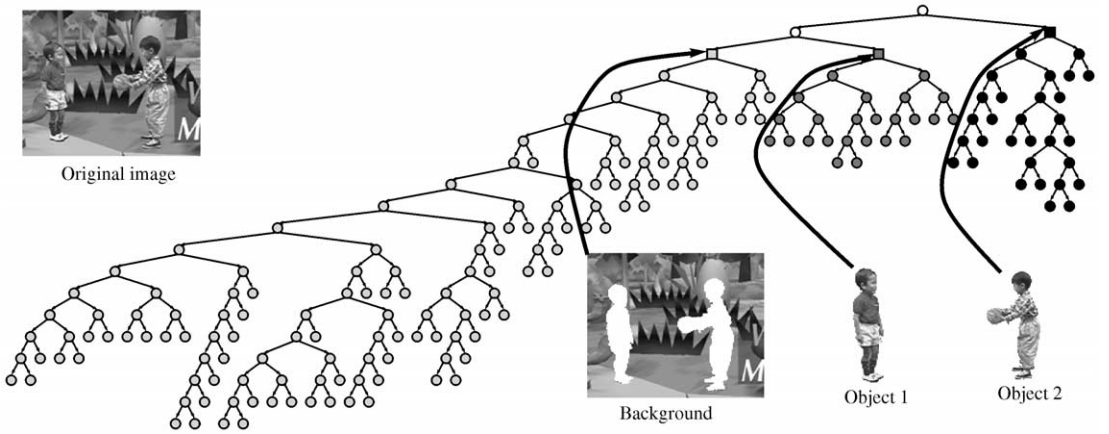

Fig. 7. Example of partition tree creation with restriction imposed with object masks.

Furthermore, additional information about previous processing or detection algorithms can also be used to generate the tree in a more robust way. For instance, an object mask can be used to impose constraints on the merging algorithm in such a way that the object itself is represented as a single node in the tree. Typical examples of such algorithms are face, skin, character or foreground object detection. An example is illustrated in Fig. 7. Let us 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 the remaining regions, the region of support of each mask will be represented as a single node in the resulting BPT. In Fig. 7, 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.

3.2.2. Temporal segments

The approach described in the previous section can also be used for VideoSegments [13]. In this case each node of the tree represents a connected component in time. Assume, for example, that the initial segmentation of the *drama* sequence has produced the set of elementary shots shown in Fig. 8. In this example, one can see: the initial shot (#1), a hall scene (#2–#8), the living room scene (#9–#13 and #15) and a dialogue scene (#16–#18). A BPT can be constructed by keeping track of the merging sequence of a shot merging algorithm.

The merging algorithm used in the temporal BPT creation is the same as the one for the spatial case. Using a homogeneity criterion, the most similar temporal segments are merged together in an iterative way.

Each shot is described by a model from which the homogeneity criterion is derived. The shot model may involve average images, histograms of luminance/chrominance information (*YUV* components) and histograms of motion information (x and y components of the motion vectors). In Fig. 9, an example of average images and color histograms for a given shot is shown. Notice that all the frames of the shot are used to compute the models. This approach requires a higher computation load than using, for example, a single key-frame,

Fig. 8. Initial temporal segmentation of the *drama* sequence. Each image represents the central frame of the shot. The numbers indicate the shot starting and ending frame number.

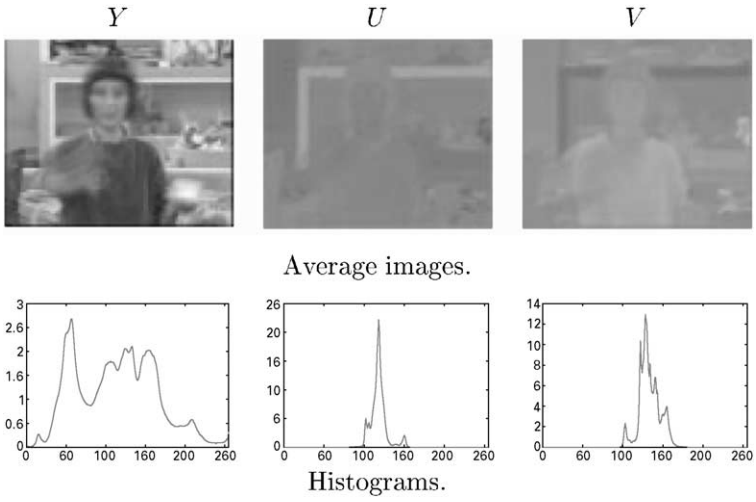

Fig. 9. Mean images and histograms for shot #9 of *drama* sequence (see Fig. 8).

but it provides better results. Motion information is represented in an analogous way by creating average images and histograms for the x and y motion vector components. Both luminance/chrominance [18,19] and motion [19,20] seem to provide useful information, but from our experimental results, luminance/chrominance information appears to be more important than the motion information, which implies that it is not mandatory to compute the motion vectors saving, therefore, some computation load.

At the beginning of the merging process, each node represents a single shot. In the following merging steps, a node may represent more than one shot. For example, in Fig. 10 nodes 25 and 26 represent shots $\{2,\ldots,8\}$ and $\{11,12,13\}$, respectively. Since the content associated to each shot can be quite different, the model associated to

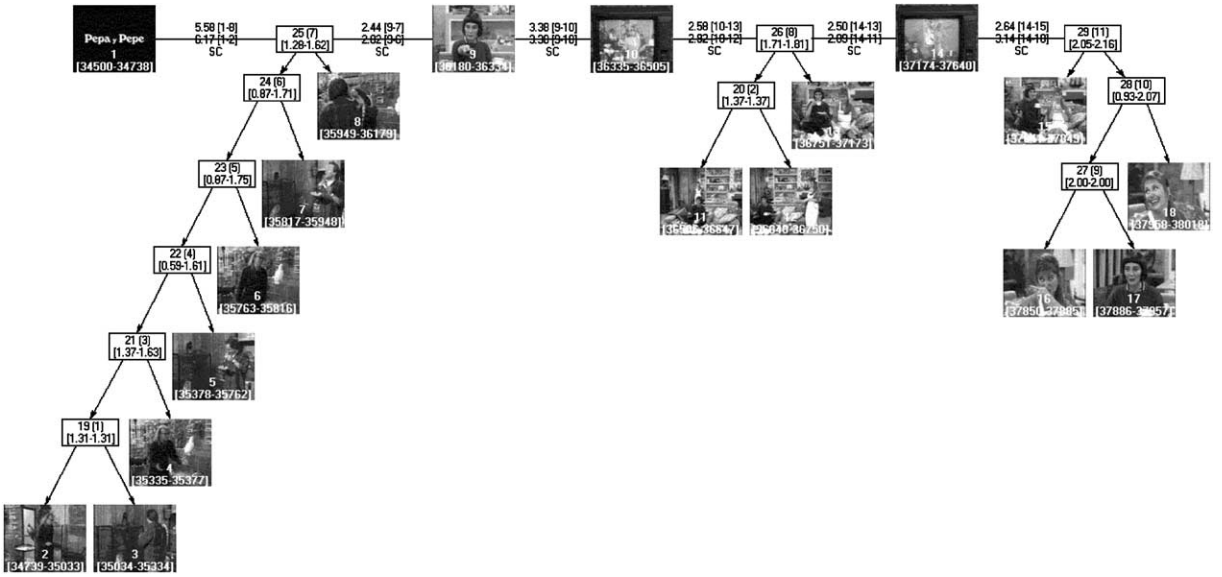

Fig. 10. Intermediate step in the merging process for the VideoSegment tree creation. The first row of nodes represents the *Region Adjacency Graph* after eleven iterations. The numbers above the links give the maximum and minimum distances as defined by Eq. (3) (in brackets the involved shots).

a node which represents more than a single shot is the *union* of all the models of the shots that it contains (a similar idea is used in Ref. [21]). The node is not modeled using, for example, the average of the models associated to each shot it contains. Experimental evidences have shown that the merging of different shots rapidly leads to random statistical estimation.

The merging order relies on a distance computed from the features represented in the segment model. At the beginning of the merging process, the VideoSegments represent individual shots and the shot similarity is computed as follows: if A and B denote two different shots, and f_k^A (resp. f_k^B) the feature k of the shot A (resp. B) (an average image or a histogram), an elementary Euclidean distance between the shot features is given by

$$d(f_k^A, f_k^B) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(f_k^A(i) - f_k^B(i))^2}, \qquad (2)$$

where N is the number of pixels if f_k is an image, or the number of bins if f_k is a histogram.

The distance between two shots is the weighted average of the feature distances:

$$d(A,B) = \frac{1}{\sum_{k=1}^{M} w_k}\sum_{k=1}^{M} w_k d(f_k^A, f_k^B), \qquad (3)$$

where M is the number of features to compare and w_k the weight of each individual feature distance.

In the following merging steps, the distance has to be computed between nodes that may represent a large number of shots. Since the merging process may combine shots with quite different luminance/chrominance or motion information, two distances are computed: the minimum and the maximum distances (d_{min}, d_{max}) between all pairs of shots included in each VideoSegment [21] (see Eq. (3)).

The merging order is defined by the minimum distance but the merging is actually performed only if the maximum distance is below a given threshold. An example of BPT is shown in Fig. 11. Inside each leaf node, the associated shot number is indicated together with its starting and ending frame number (between brackets). In each remaining node, a label, the step at which the node has been created (between parenthesis, next to the node label), and the minimum and maximum distances (between brackets) are indicated. Note that the BPT creation stops when all of the segments have been merged into a single node. In this example, it can be seen that all shots belonging to the initial hall scene appear in the same branch (node label: 19–24). Also, the living room shots are grouped in the same branch (node label: 20 and 26).

Furthermore, as in the case of the spatial BPT, it is possible to use additional information in order to construct the tree in a more robust way. Information concerning the type of transition between the initial shots may be used to estimate the distance between nodes. Such transitions can be classified in *scene cuts* or *soft transitions* (dissolves, fadings, wipes, etc.). It is known that the presence

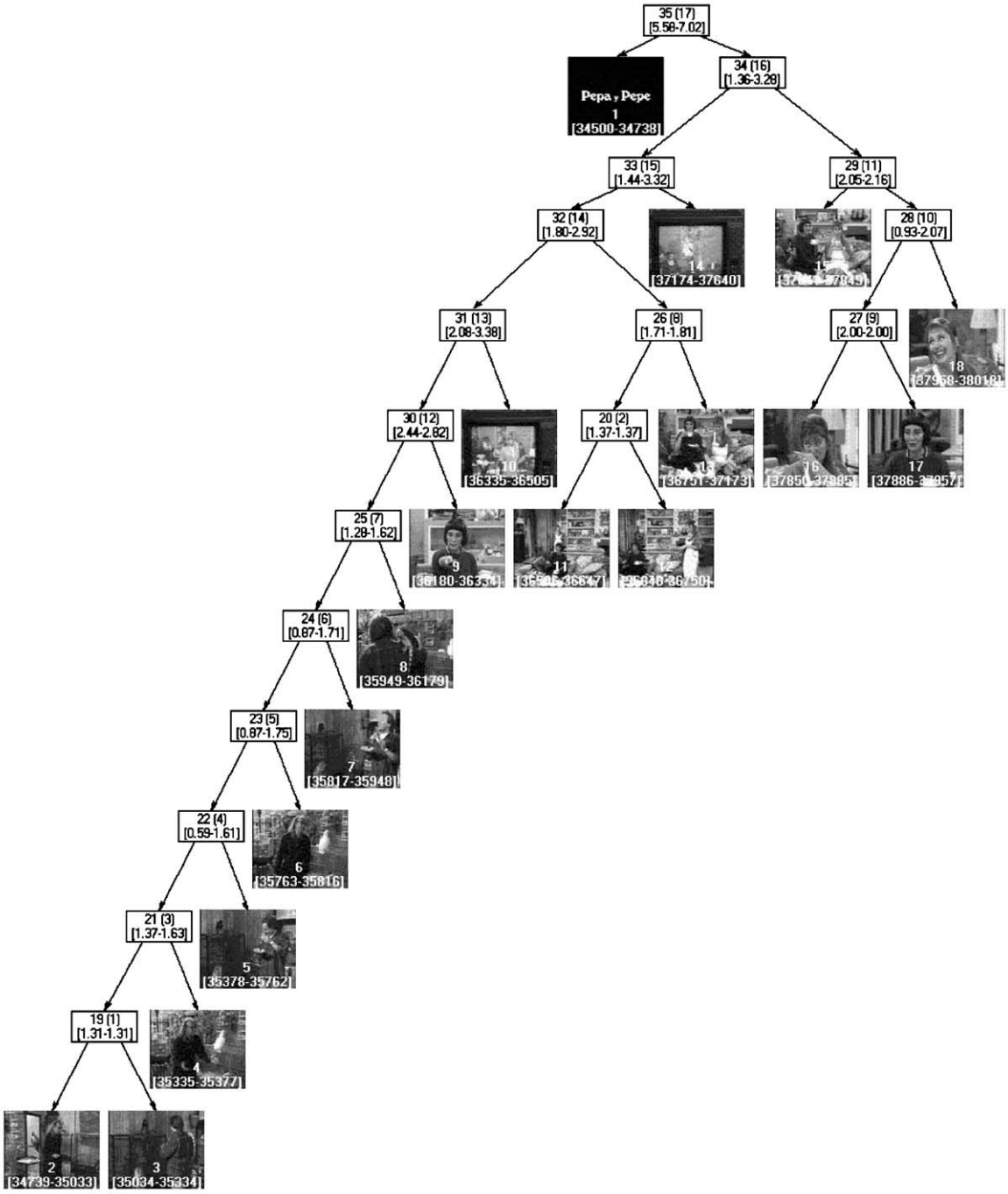

Fig. 11. Binary Partition Tree created by the shot merging algorithm. The initial shots are shown in Fig. 8. The top node represents the whole sequence, while the leaf nodes represent the initial shots.

of a soft transition is likely to be related to a change of semantic in the content. As a result, a high weight should be used in order to delay the merging of nodes until all the low-weight transitions get merged.

3.2.3. Table of contents and indexes

Until now, we have assumed that the segments that were allowed to be merged were neighbors (either in time or in space). The resulting BPT can be considered

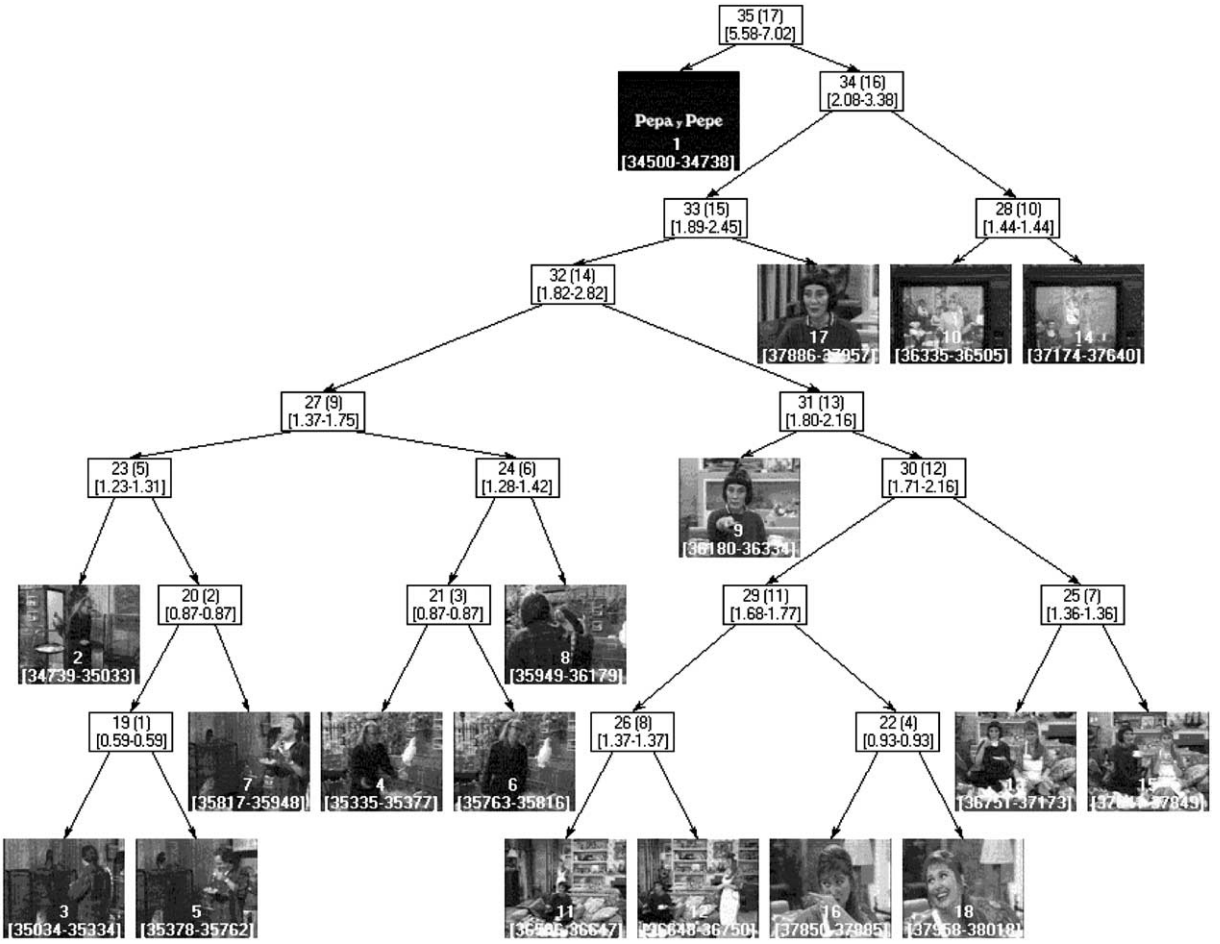

Fig. 12. Binary Partition Tree created by the shot merging algorithm without constraint on the neighborhood relationship between segments. The result is a hierarchical clustering algorithm.

as a kind of Table of Contents since it is constrained by the temporal or spatial organization of the content. However, if we remove the constraint of having neighboring segments, then the algorithm performs a classification or clustering of the segments based on their similarity. This classification approach leads to hierarchical indexes. An example based on VideoSegment is shown in Fig. 12. This example has to be compared with the one of Fig. 11. Note, in particular, how all the shots belonging to the hall scene appear below node 27 and shots belonging to the living room scene are below node 31. Note that both structures are considered in MPEG-7 Segment DS since segments may have several connected components.

3.3. Restructuring of the BPT

3.3.1. Restructuring strategy

The purpose of restructuring the Binary Partition Tree is to create an arbitrary tree that reflects more clearly the video or image structure. To this end, nodes that have been created by the binary merging process but do not convey any relevant information should be removed. These nodes appear due to the fact that all the merging steps necessary to progress from an initial partition to a partition with one region are represented within the tree.

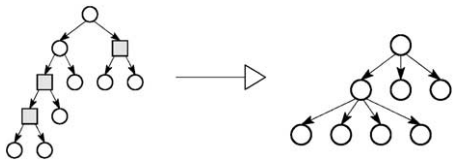

Fig. 13. Example of Binary Partition Tree restructuring. The original Binary Partition Tree is shown on the left. Square shaped nodes have to be removed, whereas circular shaped nodes have to be kept. The resulting restructured tree is shown on the right.

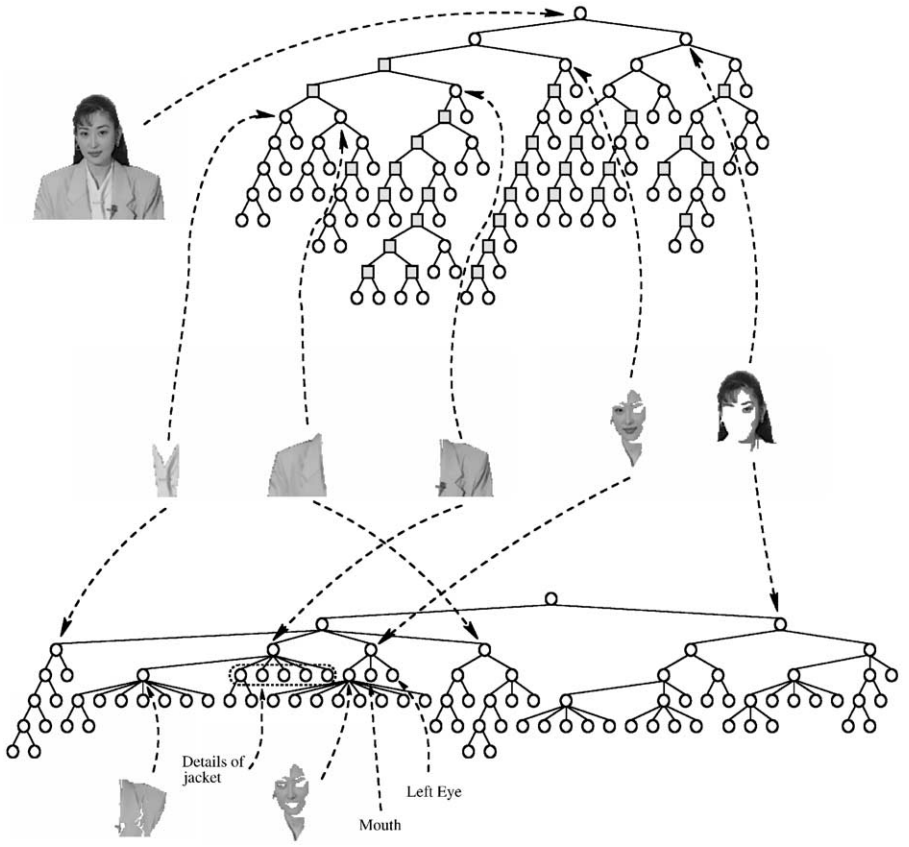

Fig. 14. Example of Region Tree restructuring. Top: original Binary Partition Tree. Square shaped node are nodes to be removed, whereas circular shaped nodes are nodes to be kept. Bottom: restructured tree. Several regions with their associated node in the Binary Partition Tree and the restructured tree are also shown.

Therefore, some merging steps are more reliable than others. For instance, very similar segments may merge in an arbitrary order depending on the similarity measure used to merge them and the level of noise included in each segment. The restructuring algorithm should be able to detect such nodes and remove them.

The restructuring algorithm can be divided in two stages. In the first stage, a criterion ψ is assessed for each node of the BPT. If the criterion ψ is below a certain threshold Θ, the corresponding node is marked as *to be removed*. In the second stage, the decisions taken in the previous stage are used to restructure the tree. Nodes *to be removed* are eliminated whereas the remaining nodes are preserved and linked to their first preserved ancestor. Note that the reconstruction creates new relationships between nodes and that the resulting tree is not binary anymore. Fig. 13 shows an example of tree restructuring. On the left-hand side, the original BPT is shown. Nodes *to be removed* are depicted with a squared shape, whereas nodes to be kept have a circular shape. The restructured tree is shown on the right. Note that the leaves and the root node should always be kept in the final tree.

In the following section, the algorithms used to restructure the tree for the spatial and the temporal segments are explained. The only difference between them is the criterion used to decide which nodes have to be removed and which have to be kept.

3.3.2. Spatial segments

As explained in the previous section, restructuring is based on assessing a criterion for each node of the BPT (excluding the leaves and the root node). The criterion used to decide if a node must appear in the final tree is based on the variation of the distance d (homogeneity criterion) used to create the BPT. If the node has been created following a color homogeneity criterion, it is removed from the tree if the following condition is satisfied:

$$\frac{|d(\text{analyzed node}) - d(\text{parent node})|}{d(\text{analyzed node})} < \Theta. \qquad (4)$$

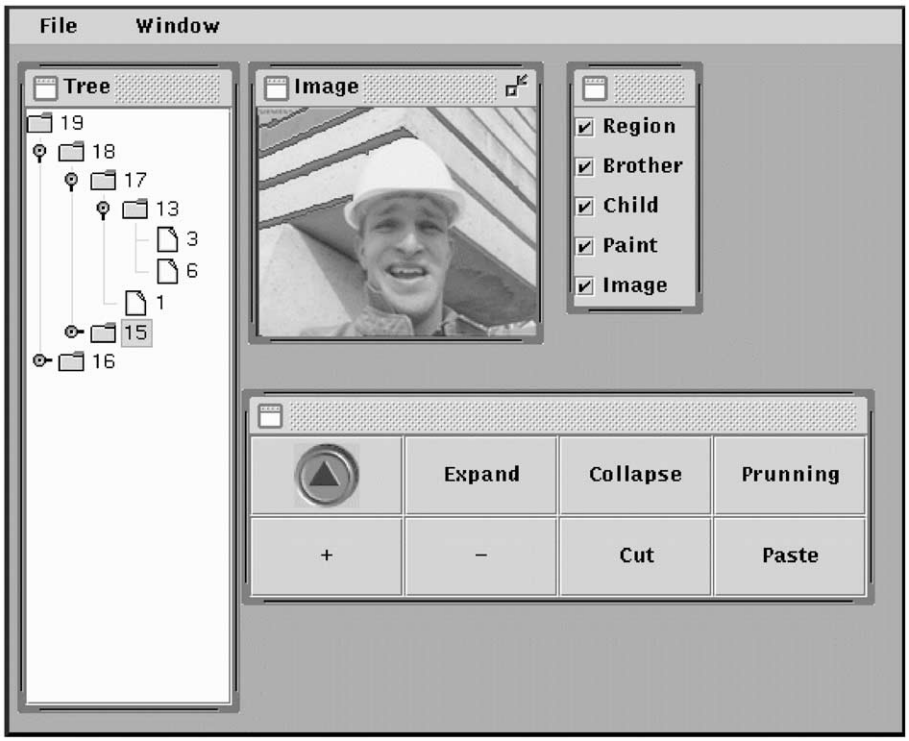

Fig. 15. Interactive tool allowing the manipulation of the resulting tree.

Eq. (4) measures the relative difference between the distance (Eq. (1)) associated to a node and its parent. Note that the simple difference between the distances cannot be considered since the color homogeneity criterion that has been used to create the BPT (see Eq. (1)) depends on the size of the regions to be merged. Therefore, the distance d increases proportionally to the area as regions are merged together.

Furthermore, Eq. (4) may be interpreted as follows: if a set of connected regions have similar color, their associated distances will be similar. As a result of noise or minor color differences, the regions merge in any order. Eq. (4) is a criterion that may be used to identify these random mergings.

Fig. 14 shows a complete example of tree restructuring. A BPT (on top) has been created, for the object shown on the left, following a color homogeneity criterion (Eq. (1)). That is, the region of support of the root node of the tree is the object itself rather than the whole image. Eq. (4) has been used to define the set of nodes to be removed (indicated with a squared shape in Fig. 14). The restructuring algorithm constructs the tree shown at the bottom of Fig. 14. This tree conveys only the most important information of the BPT.

Several regions indicating their associated node in the BPT and in the final tree are shown to illustrate the improvement of the restructuring algorithm. Note, for example, that the face is represented in the original BPT as a node that has been obtained by merging several similar (in color) neighboring regions. Therefore, the regions associated to the skin of the face may merge in any order creating a set of nodes which do not convey any information (observe the large branch of descendants of the node associated to the face in the BPT). The criterion (Eq. (4)) decides to remove most of these intermediate nodes. In the restructured tree, the node associated to the face has three children: one associated to the mouth, one to the left eye and one to the skin of the face. The latter may be further decomposed in several subregions composing the skin of the face.

The process described so far is purely automatic. However, in practical situations, it may be useful to check and possibly to modify the tree structure on the basis of human interaction. The interactive editor shown in Fig. 15 allows one to browse the tree structure, to modify the tree structure by cut and paste of tree branches and to prune the tree. Moreover, some semantic keywords may be assigned to the segments. The main difference between this kind of tool and interactive tools developed for classical segmentation [22,23] is that here the goal is to modify the tree structure rather than the partition itself.

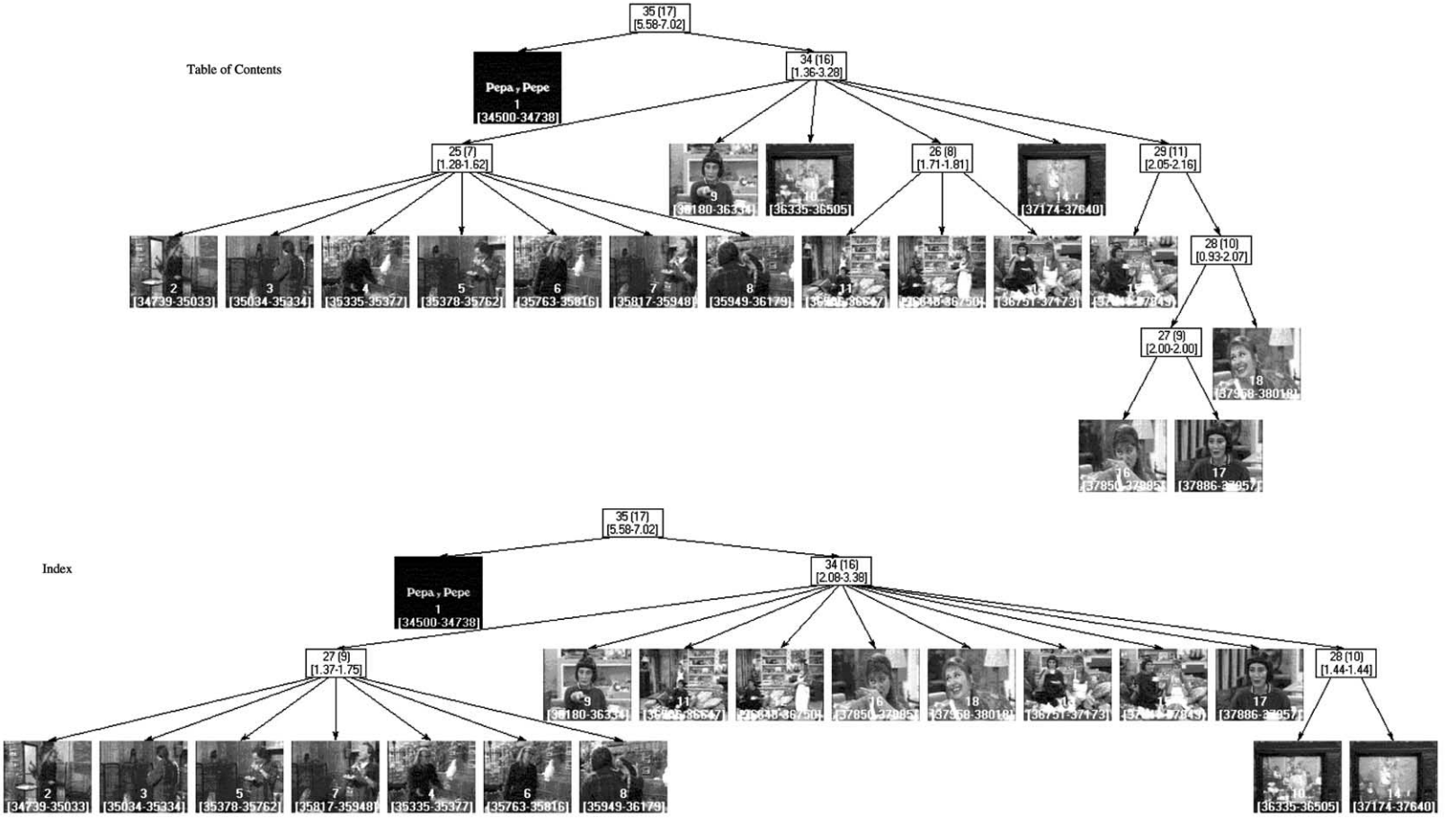

Fig. 16. Final VideoSegment trees resulting from the tree structuring algorithm. The tree on the top represents the *Table of Contents* of the sequence while the tree on the bottom represents the *Index*.

Fig. 17. Screen shots of the VideoSegment tree editor, where the table of contents of the *drama* sequence is presented. On the left, a view of the tree completely expanded, which shows, the key-frames around the selected node in the same level and some data of the selected node. On the right, a view of the timeline panel, which shows the succession of key-frames and the borders between them.

3.3.3. Temporal segments

The only difference between the spatial and temporal cases is the criterion used to decide which nodes must be preserved in the restructured tree. In the restructuring of the temporal BPT, where a more complex model is used for shot merging, the minimum and maximum distances are used and the condition to remove a node from the tree becomes

$$|d_{min}(\text{analyzed node}) - d_{min}(\text{parent node})| < \Theta,$$
$$|d_{max}(\text{analyzed node}) - d_{max}(\text{parent node})| < \Theta. \quad (5)$$

By using both the maximum and minimum distances, it is possible to check the similarity variation of the pair of closest shots as well as of the most different pair of shots contained in the analyzed node. In this way, the analyzed node will only be removed from the restructured tree if both the minimum and maximum similarities do not differ significantly from those in the parent node.

The restructuring of the BPTs presented in Fig. 11 and 12 is shown in Fig. 16. For the Table of Contents (left) the final tree represents more clearly the structure of the video sequence. In the first level of the hierarchy, there are two nodes (one for the header of the program and another for the program itself). In the second level, the decomposition creates several sections: the hall scene, a small transition with shots 9 and 10, the living room scene, shot 14 and a dialogue (below node 29).

For the Index (right), the number of levels in the tree has also been greatly reduced with respect to the binary one. This tree represents more clearly the clustering of shots according to their similarity. For example, shots 2–8 representing the hall scene have been grouped and TV screen shots 10, 14 appear jointly below node 28. Although they are connected in time, shots 9, 11, 12, 13, 15, 16, 17, 18 have been grouped together because they all occur in the same living room.

In order to be able to refine the automatically generated result, an interactive editor allowing the modification of the structure of the tree has been implemented (see Fig. 17). This tool displays the tree in two different ways: as a tree or as a set of key-frames, [1] allowing, in both cases, the adaptation of the tree visualization at different hierarchical levels.

In the tree visualization mode, a label is assigned to each node depending on its position in the tree and the position of its neighbors. In the example of Fig. 17, the labels assigned to each level follow the following hierarchy: Chapter, Section, Subsection, Paragraph and Shot (Chapter, Section, Subsection and Paragraph are commonly considered as scenes). This visualization mode allows *cut and paste* tree branches as well as insertion of new intermediate nodes. It is also possible to temporarily move nodes to the auxiliary tree (below the main tree) in order to create more complex structures.

[1] For a node that represents a single shot, the corresponding key-frame is the central image of the shot. For a node that represents more than one shot, the key-frame associated to the first shot is used.

In the temporal visualization mode (timeline), a succession of key-frames around the selected node (which appears in the center of the panel) is shown. A border between consecutive nodes is active (fully drawn) if its previous and succeeding nodes are not siblings in the tree (in the current hierarchy level). This mode allows the activation or deactivation of the borders between segments and is specially useful when new levels need to be inserted into the tree.

4. Conclusion

This paper has reviewed the Segment Description Scheme (DS) involved in MPEG-7 Description Schemes. The Segment DS is used to specify physical structures and signal properties. It may be used to define tables of contents and indexes of the documents. Low-level features such as shots, regions, color, texture, motion, etc. are described in the Segment DS. The review has highlighted the necessity to use automatic analysis tools not only for low-level descriptors but also for some of the relationships defined in the Segment DS. In particular, each segment may be decomposed to form a tree. This tree computation goes beyond the classical segmentation problem and new tools are needed.

In this paper, the strategy proposed to compute segment trees involves three steps. The first one is a classical segmentation which creates elementary segments. These elementary segments form the leaves of a BPT. The BPT itself is computed in a second step by keeping track of the merging steps resulting from a segment merging algorithm. At this stage, complex criteria can be used. Moreover the type of features used to compute the merging order may change in the hierarchy (for example color and motion in the example of Fig. 6). Finally, the BPT is restructured to create an arbitrary segment tree reflecting the signal properties. This restructuring is based on the study of the evolution of the homogeneity criterion along tree branches. As can be seen, the BPT is used as an intermediate representation. This approach is proposed because efficient algorithms exist for the computation of BPTs and moreover, the restructuring of the BPT can be performed in a robust way.

References

[1] MPEG, MPEG-7: requirements document, Technical Report ISO/IEC JTC1/SC29/WG11/w2859, MPEG, Vancouver, Canada, July 1999.

[2] P. Salembier, F. Marqués, Region-based representations of image and video: segmentation tools for multimedia services, IEEE Trans. Circuits Systems Video Technol. 9 (8) (1999) 1147–1169.

[3] R. Castagno, T. Ebrahimi, M. Kunt, Video segmentation based on multiple features for interactive multimedia applications, IEEE Trans. Circuits Systems Video Technol. 8 (5) (1998) 562–571.

[4] J.G. Choi, M. Kim, M.H. Lee, C. Ahn, Automatic segmentation based on spatio-temporal information, Technical Report, Doc. ISO/IEC JTC 1/SC 29/WG 11 MPEG97/2091, Bristol, UK, April 1997.

[5] Y. Deng, B.S. Manjunath, Netra-v: toward an object-based video representation, IEEE Trans. Circuits Systems Video Technol. 8 (5) (1998) 616–627.

[6] T. Meier, K.N. Ngan, Automatic segmentation of moving objects for video object plane generation, IEEE Trans. Circuits Systems Video Technol. 8 (5) (1998) 525–538.

[7] D. Zhong, S.F. Chang, AMOS: an active system for MPEG-4 video object segmentation, Proceedings of International Conference on Image Processing, Volume TP5, Chicago, USA, October 1998, p. 04.

[8] G. Ahanger. T. Little, A survey of technologies for parsing and indexing digital video, J. Visual Commun. Image Representation 7 (1) (1996) 28–43.

[9] B.L. Yeo, B. Liu, Rapid scene analysis on compressed video, IEEE Trans. Circuits Systems Video Technology 5 (6) (1995) 533–544.

[10] H. Zhang, A. Kankanhalli, S. Smoliar, Automatic partitioning of full-motion video, ACM/Springer Multimedia Systems 1 (1) (1993) 10–28.

[11] P. Salembier, L. Garrido, Binary partition tree as an efficient representation for image processing, segmentation and information retrieval, IEEE Trans. Image Process. 9 (4) (2000) 561–576.

[12] L. Garrido, P. Salembier, D. Garcia, Extensive operators in partition lattices for image sequence analysis, EURASIP Signal Processing 66 (2) (1998) 157–180.

[13] J. Llach, P. Salembier, Analysis of video sequences: Table of contents and index creation, Proceedings of the International Workshop on Very Low Bit-rate Video, VLBV'99, Kyoto, Japan, October 1999.

[14] O. Morris, M. Lee, A. Constantinidies, Graph theory for image analysis: an approach based on the shortest spanning tree, IEE Proc., F 133 (2) (1986) 146–152.

[15] B. Marcotegui, Segmentation algorithm by multicriteria region merging, in: P. Maragos, R.W. Schafer, M.A. Butt. (Eds.), Third Workshop on Mathematical Morphology and its Applications to Image Processing, Atlanta, USA, Kluwer Academic Publishers, Dordrecht, May 1996, pp. 313–320.

[16] L. Garrido, P. Salembier, Region based analysis of video sequences with a general merging algorithm, Proceedings of IX European Signal Processing Conference, EUSIPCO'98, Vol. III, Rhodes, Greece, September, 8–11 1998, pp. 1693–1696.

[17] J. Benois-Pineau, F. Morier, D. Barba, H. Sanson, Hierarchical segmentation of video sequences for content manipulation and adaptive coding, EURASIP Signal Process. 66 (2) (1998) 181–201.

[18] M. Yeung, B. Liu, Efficient matching and clustering of video shots. In International Conference on Image Processing, 1 (1995) 338–341.

[19] R. Mohan, Video sequence matching. International Conference on Acoustics, Speech and Signal Processing, 6 (1998) 3697–3700.

[20] H. Sawhney, S. Ayer, Compact representations of videos through dominant and multiple motion estimation, IEEE Transactions on Pattern Anal. and Mach. Intell. 18 (1996) 814–830.
[21] D.A. Adjeroh, M.C. Lee, I. King, A distance measure for video sequence similarity matching. Proceedings of the 1998 International Workshop on Multimedia Database Management Systems (1998) pp. 72–79.
[22] P. Correia, F. Pereira, The role of analysis in content-based video coding and indexing, EURASIP, Signal Processing 66 (2) (1998) 125–142.
[23] B. Marcotegui, P. Correia, F. Marques, R. Mech, R. Rosa, M. Wollborn, F. Zanoguera. A video object generaion tool allowing friendly user interaction, IEEE International Conference on Image Processing, ICIP'99, Kobe, Japan, October 1999.

About the Author—PHILIPPE SALEMBIER reveived a degree from the Ecole Polytechnique, Paris, France, in 1983 and a degree from the Ecole Nationale Supérieure des Télécommunications, Paris, France, in 1985. He received his Ph.D. degree from the Swiss Federal Institute of Technology (EPFL) in 1991. He was a Postdoctoral Fellow at the Harvard Robotics Laboratory, Cambridge, MA, in 1991.

From 1985 to 1989 he worked at Laboratories 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 the Swiss Federal Institute of Technology in Lausanne, Switzerland, to work on image processing. At the end of 1991, after a stay at the Harvard Robotics Lab, he joined the Polytechnic University of Catalonia, Barcelona, Spain, where he is currently associate professor. He lectures 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 morphhology and nonlinear filtering. He is involved in the definition of the MPEG-7 standard ("Multimedia Content Description Interface") where he chairs the "Multimedia Description Scheme" group.

He served as an Area Editor of the Journal of Visual Communication and Image Representation (Academic Press) from 1995 to 1998 and as an AdCom officer of the European Association for Signal Processing (EURASIP) in charge of the edition of the Newsletter from 1994 to 1999. He has edited (as guest editor) special issues of Signal Processing on "Mathematical Morphology" (1994) and on "Video sequence analysis" (1998). He has also co-edited a special issue of Signal processing: Image Communication on MPEG-7 proposals (2000). Currently, he is the deputy editor of Signal Processing. Finally, he is a member of the Image and Multidimensional Signal Processing Technical Committee of the IEEE Signal Processing Society.

About the Author—JOAN LLACH received the Electrical Engineering degree (specializing in Telecommunications Engineering) from the Polytechnic University of Catalonia (UPC), Barcelona, Spain, in October 1997. In November 1997, he joined the Image Processing Group at the same university where he started his Ph.D. in collaboration with Philips and worked as a Research Assistant until April 2000. Later he joined the Laboratoires d'Electronique Philips, Limeil-Brevannes, France, where he is currently working on his Ph.D. His current research interests include video sequence segmentation and analysis, the MPEG-7 standard and scene classification.

About the Author—LUIS GARRIDO received a degree in Telecommunication Engineering from the Telecommunication School of the Polytechnic University of Catalonia, Barcelona, Spain, in 1996. Later he joined the Image Processing Group at this university to work on his Ph.D. He is currently working on image and video analysis tools for indexing application in collaboration with France Telecom (CNET).