SIViP (2010) 4:435–450
DOI 10.1007/s11760-009-0151-2

ORIGINAL PAPER

Hierarchical structuring of video previews by Leading-Cluster-Analysis

Sergio Benini · Pierangelo Migliorati · Riccardo Leonardi

Received: 28 April 2009 / Accepted: 16 August 2009 / Published online: 6 January 2010

Abstract Clustering of shots is frequently used for accessing video data and enabling quick grasping of the associated content. In this work we first group video shots by a classic hierarchical algorithm, where shot content is described by a codebook of visual words and different codebooks are compared by a suitable measure of distortion. To deal with the high number of levels in a hierarchical tree, a novel procedure of *Leading-Cluster-Analysis* is then proposed to extract a reduced set of hierarchically arranged previews. The depth of the obtained structure is driven both from the nature of the visual content information, and by the user needs, who can navigate the obtained video previews at various levels of representation. The effectiveness of the proposed method is demonstrated by extensive tests and comparisons carried out on a large collection of video data.

Keywords Hierarchical video summarization · Vector quantization · Leading-Cluster-Analysis

1 Introduction

With the proliferation of digital broadcasting, internet websites, private recording of home video, a large amount of audio-visual information is becoming available to the final users. However, this massive explosion in the availability of digital videos has not been accompanied by a parallel increase in its accessibility.

In this context, video abstraction techniques may represent a key components of a practical video management system: indeed a condensed video may be effective for a quick browsing or retrieval tasks. A commonly accepted type of abstract for generic videos does not exist yet, and the solutions investigated so far depend usually on the nature and the genre of video data.

For unscripted content videos, such as sports and home-videos, events occurs spontaneously and not according to a given script. Previous work on video abstraction for this type of content mainly focused on the extraction of *highlights* [1]. For scripted videos instead, e.g., movies, dramas, news and cartoons, basically two families of abstraction methods have been adopted so far, namely *video summarization* and *video skimming* [2]. The first one, also known as *static video summarization*, is a process that selects a set of salient key-frames to represent the video-content in a compact form, often called *preview*. On the other hand, video skimming, also known as *dynamic video summarization*, tries to condense the original video in the form of a shorter video clip [3]. Recently also the problem of the summarization of raw and unedited video material (called *rushes*) is attracting more and more research interests [4–6,50].

For most of summarization techniques, the decomposition into *shots* (i.e., the basic video segments filmed in one single camera take), and the following key-frame extraction from shots, are commonly considered as the prior steps for the automatic generation of summaries. However, if we consider that there are usually several hundreds of shots for an hour long video, shot decomposition often leads to a huge amounts of key-frames, thus making the browsing impractical.

S. Benini (✉) · P. Migliorati · R. Leonardi
Department of Information Engineering,
University of Brescia,
via Branze 38, 25123 Brescia, Italy
e-mail: Sergio.Benini@ing.unibs.it

P. Migliorati
e-mail: Pierangelo.Migliorati@ing.unibs.it

R. Leonardi
e-mail: Riccardo.Leonardi@ing.unibs.it

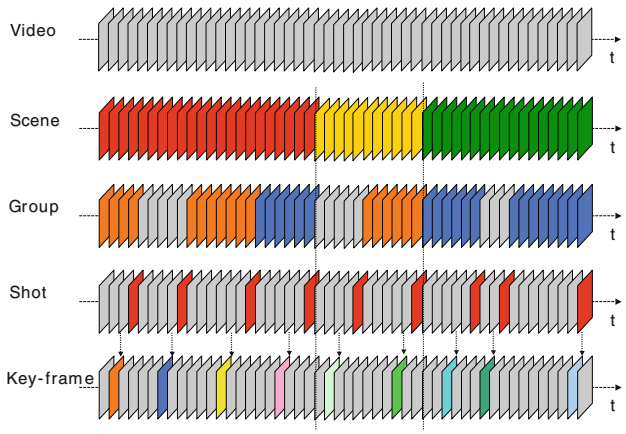

Fig. 1 Hierarchical decomposition for scripted video-content

Significant research efforts have been therefore directed towards effective clustering of visually similar shots into structures called *clusters* (or *groups*) (see Fig. 1).

Several works have shown how an accurate grouping of similar shots can for example facilitate the access to video-content [7], and can help in understanding associated high-level semantics (as in [8] and [9]). According to the decomposition shown in Fig. 1, groups of shots can be considered as intermediate entities between physical shots and semantic scenes, and can serve as a bridge to reduce the gap between the two representation levels. Building upon shot clusters, a method to produce effective video summaries is easily developed. Moreover, a number of other tasks such as video retrieval and semantic annotation, can largely benefit from accurate clustering of similar shots.

2 Paper aims and organization

The main objective of this work is to propose a novel solution for shot clustering used for the generation of a hierarchical video summary (preview). As we discuss in Sect. 3.1, the success of a clustering method is based on an effective combination of the following factors:

- the chosen clustering algorithm;
- the low-level feature to represent the visual-content;
- the employed similarity measure between shots and clusters of shots.

Concerning the clustering method, a classic *Hierarchical Agglomerative Clustering* (*HAC*) based on average-linkage criterion has been adopted, and it has been tailored to the specific case of visual data. In unsupervised clustering, when we explore a data set with unknown properties, a recurring problem is that of deciding how many clusters are present. Traditional approaches usually impose an a priori criterion to stop the clustering, for example by setting in advance the final number of clusters, or by imposing a minimum-variance partition. Anyway, these global criteria often fail in preserving the visual coherence of clusters.

To automatically determine the number of natural clusters, in this work we propose a criterion [10] which is local to each growing cluster, thus avoiding the drawbacks of global criteria. This method, which we name *Leading-Cluster-Analysis* (*LCA*), identifies the merging steps in which the visual-content of each cluster changes significantly, thus determining a likely change also in the associated semantics.

In addition to this, *LCA* provides also a natural hierarchical solution to the issue of summarization, by organizing the video in a tree structure, where the number of layers and the number of clusters in each layer depend only on the video-content, and are not a priori assigned.

The most important achievement of the proposed method is the solution to the problem of the high number of levels usually provided by a classic hierarchical method. By *LCA* a reduced set of levels is extracted from the hierarchy, and a video preview ready to be used in a browsing solution is built, as shown in [50].

Since semantic clustering is often not applicable, for example in case of not annotated content (e.g. rushes), one feasible way to represent and group visual-content is according to low-level video properties. Concerning the low-level features used, and the metric adopted to estimate shot and cluster similarities, a variety of methods exists (see Sect. 3.1). However, in most cases video shots have been described by means of color information, often simply relying on histograms or on color spatial distribution. In this work instead, we describe each shot in terms of a dictionary of visual words obtained from a *Tree-Structured Vector Quantization* (*TSVQ*) process. Traditionally adopted for coding purposes [11], Vector Quantization has been successfully proposed in [12] also as an effective low-level feature for video indexing, suitable to overcome some limitations of color layout or color histogram. A related similarity measure has also been introduced with the goal to assess at best the visual coherence of each cluster.

The rest of the paper is organized as follows. In Sect. 3 previous work on shot clustering and video summarization is discussed. The hierarchical algorithm and its related dendrogram representation are presented in Sect. 4. Section 5 introduces the *Leading-Cluster-Analysis* and the method for the generation of the hierarchical video preview. The extraction of the low-level feature based on *Tree-Structured Vector Quantization* is described in Sect. 6. Experiments and performance evaluation are provided in Sect. 7, while conclusions are drawn in Sect. 8.

3 Related work on video summarization

A significant number of applications, ranging from semantic annotation to video summarization and skimming, can

largely benefit from effective clustering of similar shots. In this section some video summarization methods adopting an initial clustering of shots are discussed.

3.1 Shot-based clustering

Data clustering methods are generally classified into *supervised* and *unsupervised* ones (the reader can refer to [13] for an accurate overview on data clustering).

Supervised methods for shot clustering have been proposed for example in [14] and [15]. In these cases, basic low-level features, such as dominant color and motion intensity, are extracted from selected key-frames, and the training data are labeled by hand. Then Hidden Markov Models and Neural Networks are used for statistical training and classification.

In general, supervised methods are more accurate and efficient than the unsupervised ones, but the work of human labeling requires a lot of time. Moreover, these classifiers can only be applied on the same types of video, and different classifiers should be trained for different video sets. To overcome these problems, clustering methods based on unsupervised learning have been proposed. They can be applied directly on unknown data without any hand-labeling and, in case of video data, represent a universal solution for different sets of video programmes.

Among unsupervised methods two main categories can be distinguished, namely, *partitional* and *hierarchical* [13]. In partitional algorithms the final grouping is obtained by partitioning the entire data set into a certain number of clusters. Hierarchical methods instead, starting from a data set, produce a hierarchy of nested clusters at different granularity.

A well known method of partitional clustering is the *k-means* algorithm, which is popular due to its simple implementation and low computational complexity. The major drawback of such an approach is that the number of the k final clusters must be set a priori, or automatically estimated, often without enough knowledge of the data set.

For video data, many variants of the *k-means* algorithm have been proposed in literature. Hanjalic and Zhang [16] introduced a partitional clustering of all video frames by utilizing the *YUV* histogram as feature vector. Initially, frames are assigned to a random partition, and then they are reassigned to the k clusters on the basis of the similarity between the frame and the cluster centroid, until a predefined convergence criterion is met. Each cluster is finally represented in the video summary by a key-frame. The proposed algorithm is basically a *k-means* clustering, where the optimal number of clusters is automatically determined by cluster validity analysis. The adopted objective function (the mean square error) is more effective in the case of isolated and compact clusters, which is not often the case for the feature vectors extracted from videos. The main problem of this kind of algorithms is that they are highly sensitive to the initial partition, so that a local minimum can be likely reached. For these reasons hierarchical algorithms are considered more versatile and have recently captured much research attention. By allowing a representation of data on different levels of abstraction, hierarchical solutions are considered interesting particularly for video data.

Hierarchical algorithms can be further distinguished into *agglomerative* and *divisive*. Specifically, agglomerative algorithms initially assign each item to a different cluster, and iteratively merge the two most similar clusters until a suitable stopping criterion is met (as in [17]), or only one cluster is left. On the other hand, divisive approaches start by considering all data in a single cluster, and then proceed by means of iterative splitting until the stopping condition is reached.

In [9] a probabilistic hierarchical agglomerative clustering algorithm has been used to discover the structure in home-videos. In particular, a statistical model of visual similarity is considered (using Gaussian Mixture Models), taking into account also the duration and the video segment adjacency.

In general, the key point of agglomerative methods is the criterion adopted to measure the distance between clusters [18]. The most used ones are the *complete-linkage*, *single-linkage* and *average-linkage* [13].

A hierarchical agglomerative clustering based on a complete-linkage, i.e., in which the distance between clusters is given by the maximum distance between all couples of items belonging to the clusters, has been adopted for example in [19]. In this work the authors obtain interesting results clustering shots according to a similarity measure based on color and pixel correlation between different key-frames. The described time-constrained method takes also into account the temporal vicinity of shots, and allows the further application of the *Scene Transition Graph* framework to automatically segment the video into story units. A similar time-constrained approach is applied in [20], where *Principal Component Analysis* is adopted to reduce the dimensionality of color and motion features extracted from video frames.

In general a complete-linkage algorithm produces numerous tightly bound and compact clusters. By contrast, a single-linkage algorithm, in which the distance between clusters is given by the minimum distance between all couples, suffers from a chaining effect: it has a tendency to produce less clusters, that are straggly or elongated. Both methods are therefore sensitive to outliers, since the distance measure relies on minimum or maximum operations. To overcome these problems more robust average-linkage approaches have been introduced, where the distance between two clusters is defined as the average distance between elements in the two groups.

An interesting example of hierarchical agglomerative clustering using a variant of the average-linkage criterion is the semi-automatic approach proposed in [21], whose final aim

is to extract a hierarchical summary. In this work similarity between shots is defined on arbitrarily chosen key-frames in terms of *HSV* color histogram and texture features. Then groups of temporally related shots are detected by measuring shot correlation between adjacent shots. Obtained clusters are manually annotated by labels from a set of ontologies, and annotations are automatically propagated to shots and frames. The similarity between two clusters is here defined as the average similarity between shots in one group and the correspondingly most similar shots in the other group, using both the above visual features and the semantic labeling. Finally the hierarchical summary is created on four predefined levels (that are *video*, *scene*, *group* and *shot*). The described system facilitates browsing and retrieval of the desired piece of video, with the drawbacks of the need for manual annotation and the rigidity of the four-level hierarchy.

The extraction of hierarchical summaries (first proposed in [22]) is performed in more stages even in [23]. In this algorithm each shot is characterized by its α-trimmed average luminance histogram, and each frame is characterized by the normalized distance between its histogram and that of the shot which it belongs to. Then fuzzy *k-means* clustering is performed on the frames in each shot with increasing k, until a decreasing value of *cluster validity* is detected. For each cluster, the most representative frame is selected as a key-frame, based either on maximal proximity to the cluster centroid or on the maximal fuzzy cluster membership. The same merging approach is also used on adjacent clusters to extract the hierarchy of summaries.

The main innovation of the approach presented by Koprinska et al. in [24] is the ability to form a hierarchy where the number of layers and the number of clusters in each layer depend on the video-content and are not set in advance. In this case clustering is achieved by the *Growing Cell Structures* neural algorithm, and key-frames are selected according to the color histogram of *DC*-images of MPEG *I*-frames. Although the method has been tested on a limited set of sequences and requires the user to set numerous parameters, it captures well salient video-content, while offering a flexible method for hierarchical browsing.

An attempt to combine the advantages of hierarchical algorithms with the lower complexity of partitional methods, is made by Ngo et al. in [25]. It adopts a two-level hierarchical approach where the *k-means* algorithm is employed to cluster shots at each level of the hierarchy independently. Color features (*YUV* histogram on *DC*-images) are adopted at the top level, while motion features (i.e., the tensor histogram) are used at the bottom level. The experimental results demonstrated that such an approach is more suitable for shot retrieval rather than for video summarization, since it suffers from the same limitations described for [16].

A completely different class of shot clustering algorithms [26,27] adopts short-term memory-based models. In [26], the detection of local minima in the continuous measure of coherence based on *RGB* color histogram, allows robust and flexible segmentation of the video into scenes. Then a one-pass on-the-fly shot clustering algorithm is derived. A similar procedure has been employed in [28], with the goal of high-level movie segmentation. The dissimilarity between shots is estimated by the correlation between *DC* key-frame images by block matching. Finally this method groups together contiguous and interconnected shots sharing a common semantic thread into *Logical Story Units* by building upon temporal relations between similar shots.

Lately *spectral clustering* demonstrated to be effective in capturing perceptual features and grouping similar shots [29, 30]. In spite of its name, spectral clustering realizes grouping by partitioning data graphs and it constitutes a favorable choice for visual data, since it works well with high-dimensional features and when distributions are not necessarily convex or Gaussian.

3.2 Other work on video summarization

Apart from shot clustering, alternative solutions to the problem of static summarization are typically based on a two-step approach: first identifying video shots from the video sequence and then, without grouping them, directly selecting key-frames according to different criteria. Most key-frames extraction techniques are based only on visual-information, except some approaches (e.g., [31]) where also motion is considered. In other approaches like [32] and [33], audio, linguistic information, and MPEG-7 metadata have been also considered in order to build the related summaries.

Some other related works apply sophisticated mathematical tools to the summarization process. For instance in [34] and [35] video-content is represented by a curve in a high-dimensional feature space, and key-frames are selected in relation to the high curvature points. Sundaram et al. instead, in [36], use the Kolmogorov complexity as a measure of the video shot complexity, and compute the video summary according to both video shot complexity and some additional semantic information under a constrained optimization formulation.

As an interesting evolution of their previous work proposed in [19], Yeung and Yeo in [37] organize the most representative images of a story unit into a *poster*, i.e., a single regular-sized image following a predefined visual layout. This approach has been adopted by other authors too (see [33,38,39]), who aim to arrange semantically relevant key-frames in a bi-dimensional plane according to different criteria such as temporal time-stamps.

However, most of these poster methods use only low-level features and do not consider the semantic content, and also the time length of the summary and the number of the key-frames to be displayed can not be changed freely. Moreover,

since the generated summary is not semantically structured, users still have to view the whole video to search for a specific scene. As an attempt to overcome some of these drawbacks, Uchihashi et al. in [17] present a method of making video posters in which the key-frame sizes are changed according to an extracted importance measure.

Since regular key-frames cannot effectively represent the underlying video dynamics, researchers have also looked for an alternative way to display the shot content using a synthesized panoramic image called *mosaic*. Following this approach, various types of mosaics such as static background mosaics and synopsis mosaics can be found in [40] and [41], while an interchangeable use of regular key-frames and mosaic images has been investigated in [42].

Finally, other schemes based on sophisticated temporal frame sampling [43], fuzzy classification [44] and singular value decomposition [45] have been also studied with encouraging results.

In the next sections, the proposed method of hierarchical shot clustering is described in detail.

4 Hierarchical agglomerative clustering

The choice of a *Hierarchical Agglomerative Clustering* (*HAC*) as a basis for our further analysis is here motivated. An *unsupervised* method was preferred to overcome the limitations imposed by a manual labeling process. *Hierarchical* clustering was chosen since it allows a natural representation of data at different levels of abstraction. Finally, the *agglomerative average-linkage* approach was adopted for its intrinsic robustness to outliers, as discussed in Sect. 3.1.

In general a hierarchical organisation supports fast understanding since it splits its content into smaller subsets on different levels and emphasises the relationships between different sets. For example, an ideal arrangement of news videos as in Fig. 2, with each node labelled with one *semantic* category, would enable a fast access and a complete understanding of the video structure.

Unfortunately semantic clustering is not always applicable to videos, especially when this content is un-annotated. Therefore one feasible approach when dealing with such material, is to hierarchically arrange content by performing a visual clustering on a set of key-frames extracted from the data.

Even if the grouping of similar content is based on visual similarity rather than semantics, the proposed arrangement helps in reducing the semantic gap between low-level features and high-level concepts familiar to the user. In fact, if the video has been produced according to a script, the director often conveys a persistent semantics through the association with an implicit temporal continuity of some low-level features (at least in the chromatic composition and lighting [27]).

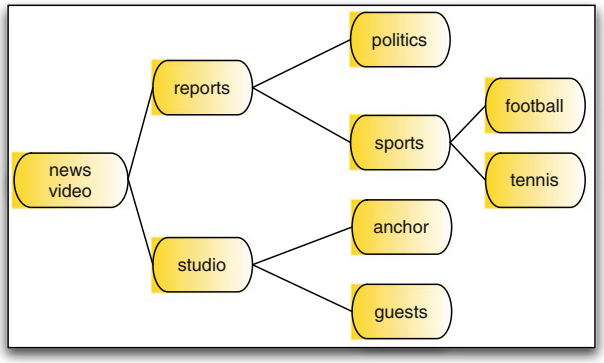

Fig. 2 Hierarchical arrangement of news based on semantic labels. Unfortunately this approach remains unfeasible in case of un-annotated video material

This has been observed experimentally by many authors, by analysing film-making rules and experimental results in the psychology of audition.

4.1 Shot detection and low-level representation

Our approach assumes to start from an accurate shot segmentation of the video into N_s separate shots (see [46] and [47] for surveys on shot detection). Key-frames are then extracted at regular intervals along the shot duration. Especially in case of low-motion shots, we have observed that it is sufficient to extract only one key-frame from each shot (e.g., the central one) to have an adequate representation of its visual-content. In any case, the procedure has been designed in a scalable fashion in the sense that it is able to handle more than one key-frame per shot, so as to adequately cover the visual-content, for example in case of high-motion shots. The choice of which low-level feature to extract from key-frames represents a fundamental step to perform the content analysis [47], and it strongly influences the overall clustering performances. In our work, to describe the visual-content of shot key-frames, we compare two low-level features based on color. The first one, the histogram on *HSV* color space, has been widely used, while the second one, the *Tree-Structured Vector Quantization* (*TSVQ*), is here proposed as an extension of the method introduced in [12]. However, it is important to note that the proposed clustering method can ideally employ *any* low-level visual feature, provided with a suitable method of similarity estimation.

4.2 Shot-to-shot and cluster-to-cluster similarities

Let $\phi_f(S_i, S_j)$ be the measure of similarity[1] between two shots S_i and S_j, based on the chosen feature f extracted from

[1] Note that we consider here a shot-to-shot "similarity" and not a "distance" since, although a measure can be symmetric, in most cases it is no longer a distance, for the reasons exposed in [26].

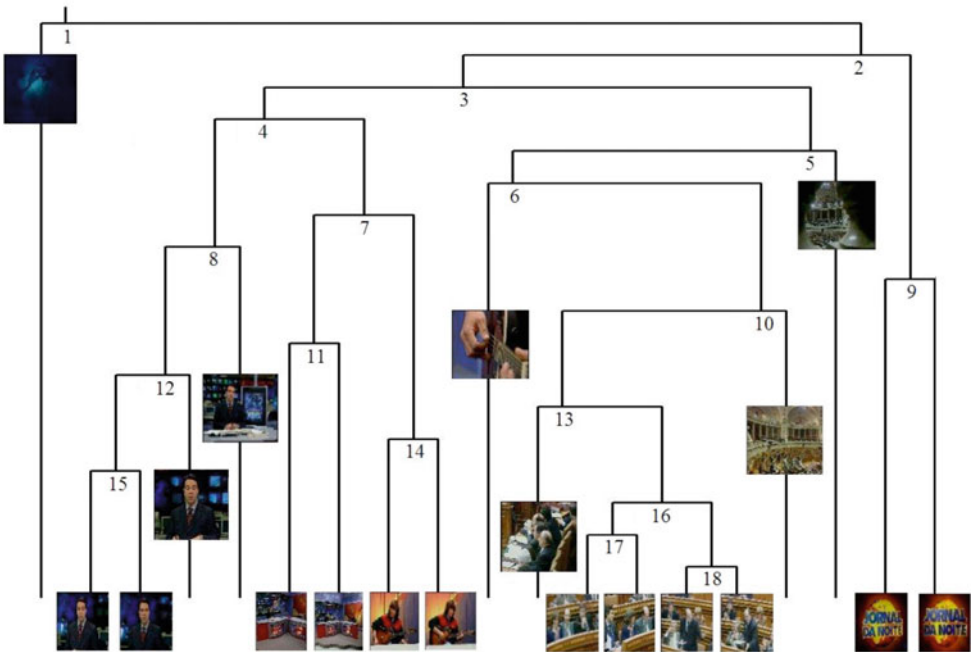

Fig. 3 Dendrogram representation of a clustering process on 19 shots extracted from the *Portuguese News* programme. Note that the first two clusters to be merged are the ones connected by the shortest ⊓-branch

shot key-frames. According to the agglomerative approach, at the beginning of the clustering process each of the N_s shots is assigned to a different cluster. Then, iteratively on each *level-i* (where $i \in I = \{N_s, N_s - 1, \ldots, 1\}$ indicates the number of the remaining clusters at the current stage), the *HAC* merges the two most similar clusters, from *level-*N_s (one shot per cluster) up to *level*-1 (all shots in the last remaining cluster).

Similarity $\Phi(C_h, C_k)$ between two clusters is computed according to the average-linkage approach, i.e., by averaging the similarities between all the shots of the two clusters, that is:

$$\Phi(C_h, C_k) = \frac{1}{N_h N_k} \sum_{S_i \in C_h} \sum_{S_j \in C_k} \phi_f(S_i, S_j), \tag{1}$$

where N_h (respectively N_k) is the number of shots belonging to cluster C_h (respectively C_k).

4.3 Dendrogram representation

It is well known that any hierarchical agglomerative clustering can be graphically represented by a binary-tree called dendrogram [13]. An example dendrogram, built on a short sequence of 19 shots excerpted from the *Portuguese News* programme, is shown in Fig. 3.

It consists of many ⊓-shaped branches, where each ⊓-branch represents the fusion between two clusters. The height of the branch is proportional to the similarity between the two clusters, so that short (respectively long) connections correspond to similar (respectively dissimilar) clusters. This means that the first two clusters to be merged are those connected by the shortest ⊓-branch, as can be seen in Fig. 3. The clustering process then continues on higher levels with the fusion of the clusters connected by progressively longer ⊓-branches.

5 Leading-Cluster-Analysis

If no criterion is used to stop the clustering of the N_s key-frames, a classic hierarchical approach produces a tree which is $(N_s - 1)$ levels deep. The main objective of the *Leading-Cluster-Analysis* is to generate a hierarchical preview $\mathcal{P} = \{L_1, L_2, \ldots, L_w\}$ organised on a reduced number of w levels, where $w << N_s$.

As shown in Fig. 4, each level L_i of the hierarchy contains the whole set of N_s key-frames organised in a number of visually similar clusters. Such organisation enables structured exploration: once the user identifies an interesting key-frame, he can interactively request more similar content from the same cluster, or refine his search by descending into the hierarchy, thus restricting the scope of his quest.

In unsupervised clustering, a formal approach to the issue of deciding how many clusters are present, is to devise some measure of goodness of fit that expresses how well a given clustering matches the data. The *Chi-Squared* and *Kolmogorov–Smirnov* statistics [13] are the traditional measures of goodness of fit, but the curse of dimensionality usually demands the use of simpler measures. Therefore, a common approach is to repeat the clustering procedure for an increasing (or decreasing) number of clusters, and to see how a criterion function J changes.

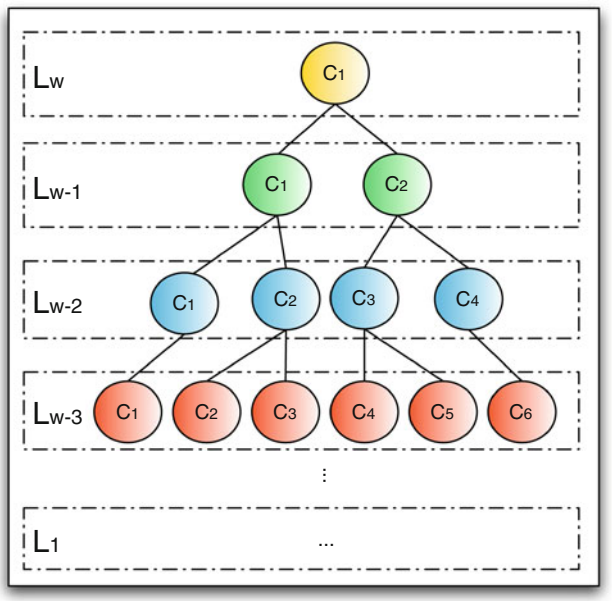

Fig. 4 Structure of the hierarchical preview

Classic *HAC* approaches usually adopt a global criterion function in order to stop the clustering. This can be the widely used *minimum variance* criterion, or a criterion based on the cluster dimensions [17], or on the desired final number of clusters, or again a constraint on a measure of global distortion [48]. For example, when using a minimum variance criterion, if the N_s shots are naturally grouped into k clusters, one would expect to see J increase slowly from $J = 0$ (when each cluster is a singleton) until we have k clusters, and increasing much more rapidly thereafter. However this might not be true in case the derivative of J is not characterized by a pronounced global maximum, since the mean of the squared error is averaged on all the items. Again, by setting in advance the desired number of clusters, the unique final partition is obtained without any control on the visual-content coherency of each cluster. Similar arguments can be advanced for all the methods used to stop the clustering which are based on a global criterion function.

The proposed *Leading-Cluster-Analysis* (*LCA*) provides a more robust solution to the issue of determining the number of natural clusters by using a criterion function which is local to few clusters, thus avoiding the drawbacks of global criteria and reducing the computational time. Observing the dendrogram, there are only few main clusters around which the merging takes place (that we name *leading clusters*). Since a large disparity in the branch heights on these clusters indicates the presence of natural grouping, *LCA* can identify when irrelevant shots start being wrongly incorporated into a coherent cluster.

Traditional flat algorithms cannot easily produce nested partitions, unless by iterating the clustering on different levels. The proposed *LCA* instead, provides a natural solution to this issue by automatically building a hierarchy of partitions where the number of layers and the number of clusters in each layer depend only on the video-content, and are not assigned a priori. By adopting such a scheme, the video content is progressively condensed, from bottom to top, at decreasing levels of granularity.

5.1 Identification of leading clusters

To perform the *Leading-Cluster-Analysis*, it is useful to distinguish on a dendrogram two different categories of clusters, namely the *leading clusters* and the *solitary* ones. The *leading clusters* can be considered the main clusters around which the agglomerative process takes place. The underlying idea is that the *leading clusters* are the first formed clusters, while *solitary clusters* are singletons which join a *leading cluster* on a higher level of the dendrogram.

In particular, by observing the bottom level of the dendrogram, where each cluster contains a single shot, it is easy to single out the *leading clusters* as the ones originally formed by the fusion of two singletons (see Fig. 5, where the *leading clusters* have been highlighted, together with their connecting ⊓-branches). On the contrary, all the other singleton clusters (which in Fig. 5, for the sake of clarity, are floating on a higher level than the *leading clusters*) can be referred to as the *solitary clusters*.

Let C_k^L be a *leading cluster* originally formed at *level-k*, where $k \in I$ (in Fig. 5 for example, the *leading cluster* C_{15}^L is the one formed by the two "anchorman" shots at the

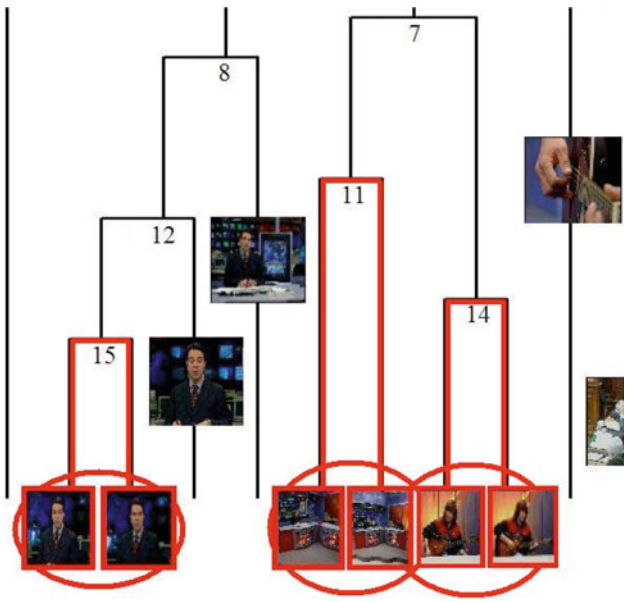

Fig. 5 Zoom of the dendrogram in Fig. 3 with the *leading clusters* highlighted at the bottom level, and the *solitary clusters* floating on higher levels

bottom-left of the dendrogram). Let C_h^S be a generic *solitary cluster* instead, where $h \in I$ denotes that such cluster participates in a merging for the first time at *level-h*.

Climbing from the bottom to the top of the dendrogram, on each *level-i* ($i \in I$) a *leading cluster* C_k^L and another cluster C_h (where C_h can either be a *leading cluster* or a *solitary* one) are involved in a merging operation. As a result of the fusion, if C_h is a *solitary cluster* (*i.e.*, $C_h = C_h^S$), the *leading cluster* C_k^L will include all the shots of C_h^S. However, if C_h is also a *leading cluster* (i.e., $C_h = C_h^L$), after the merging the older *leading cluster* will include all the shots of the more recently formed one, i.e., the *leading cluster* C_k^L (respectively C_h^L) will include all the shots of C_h^L (respectively C_k^L) if $k > h$ (respectively $h > k$). In both cases, after each merging, one *leading cluster* will include all the shots of the two merged clusters. Since only one *leading cluster* survives each merging operation, it is possible to perform a faster (but complete) analysis on the clustering process by only monitoring what happens to *leading clusters*, while ignoring other ones.

5.2 Local distortion in leading clusters

Once identified a *leading cluster*, we are interested in detecting abrupt changes in its visual content when it is involved in merging operations.

Let $C_k^L(i)$ be a *leading cluster* when observed at *level-i* ($i \in I = \{N_s, N_s - 1, \ldots, 1\}$). Since C_k^L is not involved in a merging operation on each step $i \in I$, let $I_{C_k^L} = \{i_1, i_2, \ldots, i_n\} \subseteq I$ be the sub-set of levels of I in which C_k^L actually takes part in a merging (note that $i_1 = k$). To evaluate the changes in the visual-content as the cluster grows bigger, we introduce the *cluster local distortion* as a measure of the cluster visual coherence. The *local distortion* Θ of the cluster C_k^L at level i_j is the similarity Φ between the two clusters being merged:

$$\Theta(C_k^L(i_j)) = \Phi(C_k^L(i_{j-1}), C_h), \tag{2}$$

where C_h is the cluster (that can be *leading* or not) merged with C_k^L at level-i_j. An example of *local distortion* $\Theta(C_k^L(i_j))$ for a *leading cluster* C_k^L at merging steps i_j is shown in Fig. 6.

5.3 Freeze levels

From the observation of the local distortion Θ, a criterion to stop the growth of each *leading cluster* is derived. In Fig. 6 significant increases in the slope of local distortion Θ correspond to the merging steps in which the cluster visual-content drastically changes. By analyzing $\Theta(C_k^L(i_j))$ of each *leading* cluster, we are able to automatically determine for each *leading* cluster the merging steps which determine abrupt changes in its visual-content coherence. In particular by analyzing the

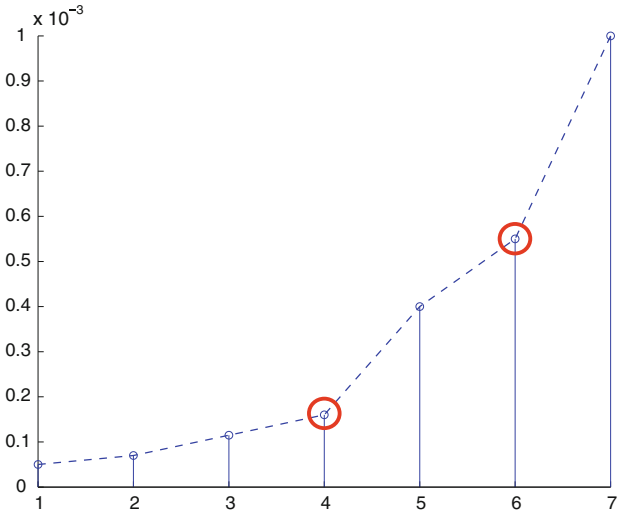

Fig. 6 Local distortion Θ for a *leading cluster* C_k^L with respect to the merging steps i_j; the significant increases in the slope are caused by substantial changes in the cluster visual-content

discontinuities of the local distortion of each *leading cluster*, and by setting a threshold δ on the discrete derivative Δ of the local distortion:

$$\Delta(C_k^L(i_j)) = \Theta(C_k^L(i_j)) - \Theta(C_k^L(i_{j-1})) < \delta, \tag{3}$$

we determine the so-called *freeze levels*. These are the set $F_{C_k^L} = \{i_{f_1}, i_{f_2}, \ldots, i_{f_n}\}$ of levels belonging to $I_{C_k^L}$ (i.e., $F_{C_k^L} \subseteq I_{C_k^L}$) for which $\Delta(C_k^L(i_j))$ exceeds δ.

Since persistent semantics is often conveyed by an implicit continuity of low-level features, the collection of these levels indicates the instants in the growing process of C_k^L that are likely related to a change in the cluster semantics too. In general low values of δ, by allowing only the merging of clusters with strong visual similarity, determine a large number of *freeze levels*, while higher values, by allowing the fusion of also visually different clusters, reduces the cardinality of $F_{C_k^L}$.

5.4 Hierarchy of partitions

Once extracted the *freeze levels* for each *leading* cluster, the obtained partitions are organized into a hierarchical preview $\mathcal{P} = \{L_1, L_2, \ldots, L_w\}$. The number of summarization layers w depends on the nature of the video-content and it is given by the maximum cardinality among the sets $F_{C_k^L}$, that is:

$$w = \max_k \left| F_{C_k^L} \right|. \tag{4}$$

To extract the mth-layer of the hierarchical summary ($m = 1, 2, \ldots, w$), the algorithm lets each *leading cluster* C_k^L grow until its mth-*freeze level* (i.e., until *level-*i_{f_m}). This means that in order to obtain the first-layer of the video

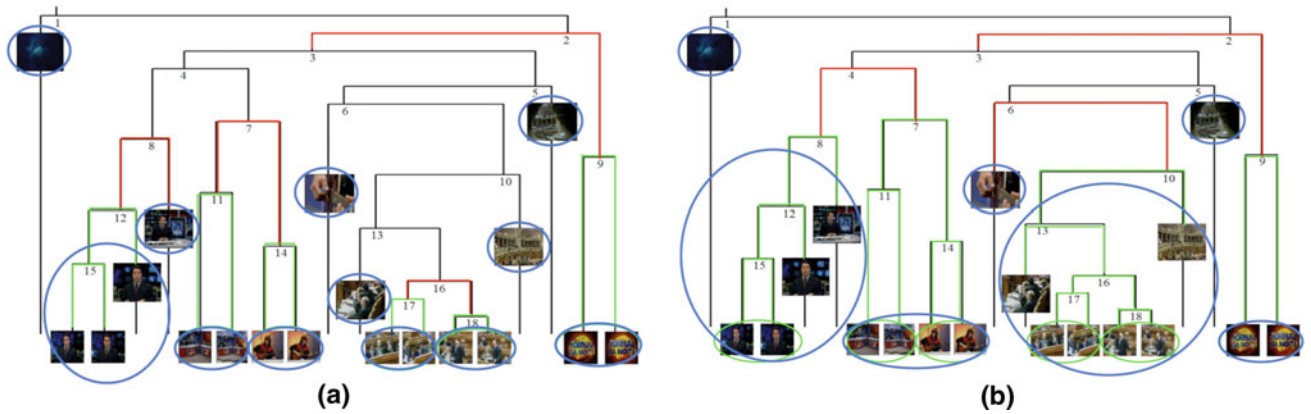

Fig. 7 An example of the extraction of the first two layers of the hierarchical summary for the news programme *Portuguese News*. **a** The clusters (*in balloons*) belonging to the first-layer of the summary of *Portuguese News*, shown on the dendrogram. **b** The clusters of the second-layer of the summary. Notice that some clusters have grown in size with respect to the first-layer.

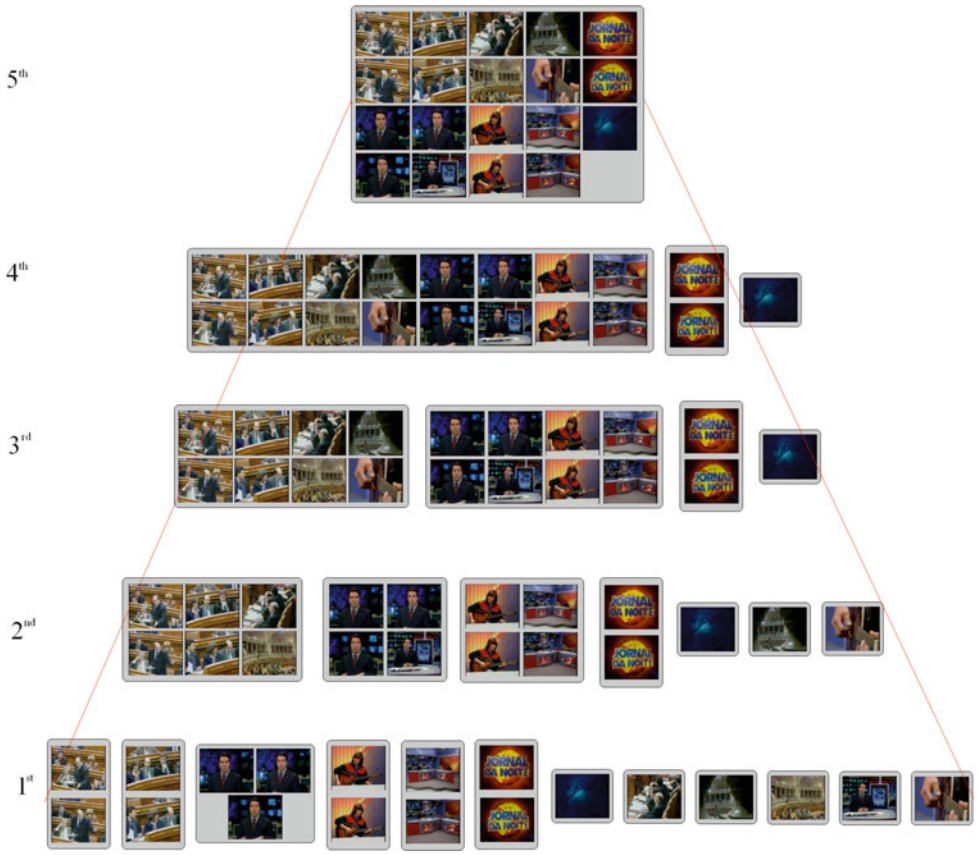

Fig. 8 Hierarchical summary obtained from a short *Portuguese News* sequence by using the *LCA* approach

summary all the *leading clusters* are allowed to grow until their *first-freeze level*. Similarly, in order to obtain the second-layer of the summary, all the *leading clusters* are allowed to grow until their *second-freeze level*, and so on.

An example of the extraction of the first two layers of a hierarchical summary for the news programme is given in Fig. 7, while the whole related hierarchy is shown in Fig. 8. The first extracted partition (Fig. 7a) provides the layer with the finest granularity, while from the second-layer (Fig. 7b) on, clusters grow in size and their number reduce.

5.5 Dependency condition

When a *leading cluster* C_k^L is allowed to grow until its mth-*freeze level* ($i_{f_m}^k$), it may happen that it has to merge with

another *leading cluster* C_h^L which has already reached its mth-*freeze level* ($i_{f_m}^h$), thus being prevented from any further merging.

So at each step i_j^k (where $i_j^k \in \{i_1^k, \ldots, i_{f_m}^k\} \subseteq I_{C_k^L}$), when C_k^L merges with another *leading cluster* C_h^L, we should verify that the cluster C_h^L has not been already arrested on a previous level (lower in the dendrogram), that is:

$$i_{f_m}^h \leq i_j^k. \tag{5}$$

If this condition is not satisfied, the growth of C_k^L must be stopped iteratively at lower levels on the dendrogram (i.e., going back to level $i_{(j-1)}^k$) until the dependency condition with the corresponding merging cluster is verified.

5.6 Depth of the hierarchy

By tuning the value of δ, the user influences the number of the extracted layers, obtaining a more or less refined hierarchical summary. The use of δ might be erroneously interpreted as an a priori criterion to stop the clustering; in this case, it would suffer from the same limitations discussed for the other a priori criteria. It is instead a tuning parameter offered to the end-user to generate hierarchical summaries with different granularity.

If the end-user is interested in a rough granularity and a reduced set of layers (for example if he/she is interested in distinguishing only *"daylight"* shots from *"night-time"* ones) a higher-value of δ will allow to detect only relevant visual-changes of content (in terms of chromaticity and lighting). On the contrary, if the user is interested in having a deeper hierarchy of finer granularity (e.g., to have single actors of a dialogue in separate clusters) he/she can reduce the value of δ. By doing so, all minimum changes in the visual-content of each cluster (reflected by the discontinuities in the leading cluster distortion) will be registered.

Therefore, once we have set δ, we have not chosen the final number of clusters on one summarization layer, but this depends only on the number of natural clusters which are present in the video at the requested granularity. In the same way, for a given value of δ, also the final number of summarization layers is not set: in the case of very coherent visual-content, we will obtain a shallow hierarchical summary. On the contrary, for the same value of δ, if the visual-content of the video is very incoherent, the depth of the hierarchy will naturally increase.

6 Tree-structured vector quantization

Regarding the low-level features, the proposed procedure for creating the hierarchical summary is very flexible since it may use any type of real valued low-level feature to compute pairwise similarity between shots. In order to improve the performance of the clustering with respect to traditionally employed low-level features, we propose to represent the shot visual-content in terms of *tree-structured vector quantization* (*TSVQ*) code-books.

In the past, the color histogram has been the most adopted feature in order to perform these operations on images, due to its simplicity and low computational cost. However the histogram often does not match perceptually well different images, since it does not take into account the local spatial distribution of color in images.

Although the main contribution of this paper deals with the clustering procedure, in this section details on the code-book design and an effective shot-to-shot similarity measure based on such code-books are discussed.

6.1 Code-book design

For each extracted key-frame, a *TSVQ* code-book is designed so as to reconstruct each frame within a certain distortion limit. After having been sub-sampled in both directions at *QCIF* resolution, every key-frame is divided into non-overlapping blocks of $N \times N$ pixels. Block color components in the *LUV* color space are then used as training vectors to a *TSVQ* algorithm [11] which uses the *Generalized Lloyd Algorithm*. Code-books of increasing size 2^n ($n = 0, 1, 2, \ldots$) are then computed, until a pre-determined constraint on distortion is satisfied (or a maximum code-book size is reached). Then, an attempt is made to reduce the number of code-words in the interval $[2^{n-1}, 2^n]$ without exceeding the imposed distortion.

Finally the algorithm returns the code-words and the *TSVQ* code-book final dimension for each investigated shot. Note that the dimensions of each code-book could be different for each single shot. In fact the objective of this approach is to produce code-books with close distortion values, so that a code-book can represent the content of visually similar key-frames with a comparable distortion. This allows measuring the visual similarity between shot key-frames through a sound comparison between their related code-books.

6.2 Shot-to-shot dissimilarity based on *TSVQ*

The dissimilarity between two shots is then measured by crossing the code-books on the visual content of shot key-frames. Let S_i be a shot, and let K_j be a generic code-book; if V_i is the number of vectors of S_i, when a vector of S_i is quantized to a code-word of K_j, a quantization error occurs. This error may be measured by the average distortion $D_{K_j}(S_i)$:

$$D_{K_j}(S_i) = \frac{1}{V_i} \sum_{p=0}^{V_i - 1} \|s_{ip} - k_{jq}\|^2, \tag{6}$$

Table 1 Video data set

Video (genre)	Length	Shots
Portuguese News (news)	47:21	476
Notting Hill (movie)	2:04:00	1,029
A Beautiful Mind (movie)	2:15:42	1,202
Pulp Fiction (movie)	2:34:30	1,476
Camilo and Filho (soap)	38:12	140
Riscos (soap)	27:37	423
Misc. (basket/soap/quiz)	38:30	195
Don Quixotte (cartoon)	15:26	188

where k_{jq} is the word of K_j with the smallest Euclidean distance from s_{ip} ($q = \arg\min_z \|s_{ip} - k_{jz}\|^2$).

Furthermore, given two code-books K_i and K_j, the value:

$$\varphi_{i,j}(S_i) = |D_{K_i}(S_i) - D_{K_j}(S_i)| \tag{7}$$

can be interpreted as the dissimilarity between the two code-books, when applied to the same shot S_i.

The dissimilarity between two shots S_i and S_j is defined as a symmetric form of the measure used in [12], i.e.:

$$\phi_{vq}(S_i, S_j) = \varphi_{i,j}(S_i) + \varphi_{i,j}(S_j) \tag{8}$$

where $\varphi_{i,j}(S_j)$ is the dissimilarity between code-books K_i and K_j when are both applied to shot S_j. The smaller ϕ_{vq} is, the more similar the shots are. Note that the similarity is based on the cross-effect of the two code-books on the two considered shots. In fact, it may be possible that the majority of blocks of one shot (for example S_i), can be very well represented by a subset of code-words of code-book K_j representing the other shot. Therefore K_j can represent S_i with a small average distortion, even if the visual-content of the two shots is only partly similar. On the other hand, it is possible that code-book K_i does not lead to a small distortion when applied to S_j. So the cross-effect of code-books on the two shots is needed.

6.3 *TSVQ* computational complexity

The traditional technique for searching in a code-book the best code-word to associate to a certain vector, is to compare the vector to be quantized with all the possible code-words, performing then an exhaustive search. For big sets of vectors and large code-books however, an exhaustive search would require an enormous number of operations, due to elevate number of needed comparisons.

For these reasons, in this work the techniques for a fast search returning the same results of an exhaustive one have been implemented as in [49]. These methods obtain a reduced computational complexity due to geometric considerations: each vector is compared only with the code-words belonging to a specific region, which has been determined by observing the spatial distribution of the code-words with respect to the vector to be quantized.

Supposing to have N_s shots, we have to compute $N_s(N_s - 1)/2$ shot-to-shot dissimilarities and store the results in a table (whose complexity is then $O(N_s^2)$). In the case of a movie of 90 min length with about 1,000 shots, on an Xserve Quad Xeon 64-bit server, the code-book generation and the table population require a computational time which is in the order of the real time.

7 Experimental results

In order to objectively evaluate the clustering accuracy, we focused our experiments on scripted-content videos (in Table 1) taken from one news programme, three movies, two soap operas, one miscellaneous and one cartoon, for a total of about 10 h of video and several thousand shots.

Extensive tests on unscripted material have been also carried over with the purpose of visualisation of *rushes* preview, and are documented in [50]. As a general note to the examples and results presented in this paper, we remark that we have not taken into account the visualisation issues regarding video summarization, which have been instead deeply analysed in [50]. Therefore, while the processing and the evaluation are performed on the whole videos, the shown examples present only short excerpts.

To better understand the output of the proposed scheme, a summary of a short sequence from *Pulp Fiction* is shown in Fig. 9. The obtained partitions are parsed into a hierarchical structure, and each layer of the hierarchy contains a preview of the video at a different granularity. Looking at the top of the hierarchy, the fourth-layer is a unique cluster containing all the shots; the third-layer distinguishes among three different settings that a human may label as: *"the corridor"*, *"the car"* and *"the apartment"*. Then, the hierarchical decomposition continues on the second and on the first-layer, separating for example the shots with actor *"J. Travolta"* from the ones with *"S.L. Jackson"*.

7.1 Evaluation criterion and performance measures

Since common ground-truth sets to evaluate summarization results are not yet available in literature, we need to estimate the content representational value of each obtained summary in terms of metrics that try to answer legitimate questions such as "how accurate is the summary?" or "how concise is the summary?". To this purpose it has to be pointed out that, if a human observe the same shot on different layers of the hierarchy, the semantics of the cluster containing the shot change depending on the summarization layer. For example

Fig. 9 Hierarchical summary obtained on a short *Pulp Fiction* sequence by using the *LCA* approach together with the *TSVQ* color feature (color figure online)

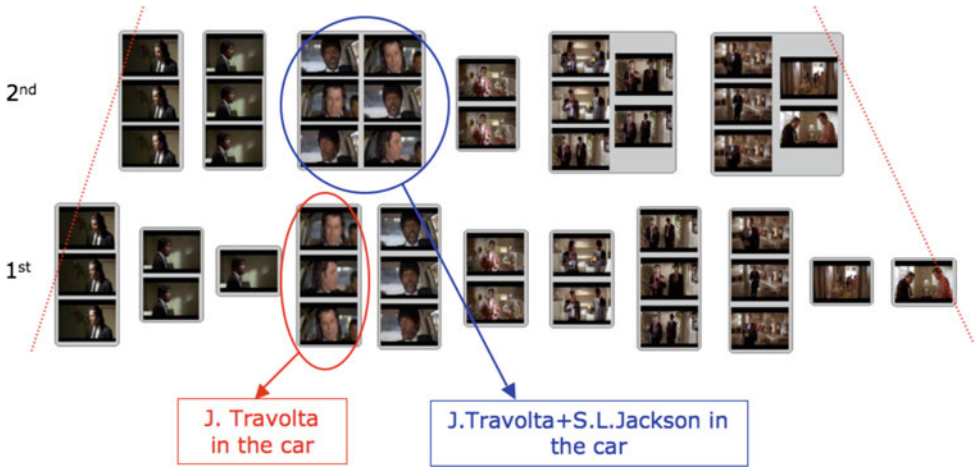

Fig. 10 The semantics of the clusters change depending on the summarization layer

in the first-layer of the *Pulp Fiction* summary (see Fig. 10) we have a cluster containing only shots sharing the semantics *"J. Travolta in a car"*. But if we climb on the higher level of abstraction, the same shot is clustered together with those showing S.L. Jackson in the same car, so that a human would label the cluster semantic as *"Man in a car"*.

So, in order to evaluate the quality of the detected results, the following criterion has been applied:

"A cluster belonging to the mth-layer is judged to be correctly detected for that summarization layer if and only if all the shots in the cluster share a common semantic meaning given a posteriori by a human observer. Otherwise it is judged as wrongly detected."

The two measures proposed in [21] are used for evaluating each layer of the preview. The first one measures the clustering *precision P* (answering the question "how accurate is this summary layer?") and it is defined as:

$$P = \frac{C_c}{N_c}, \tag{9}$$

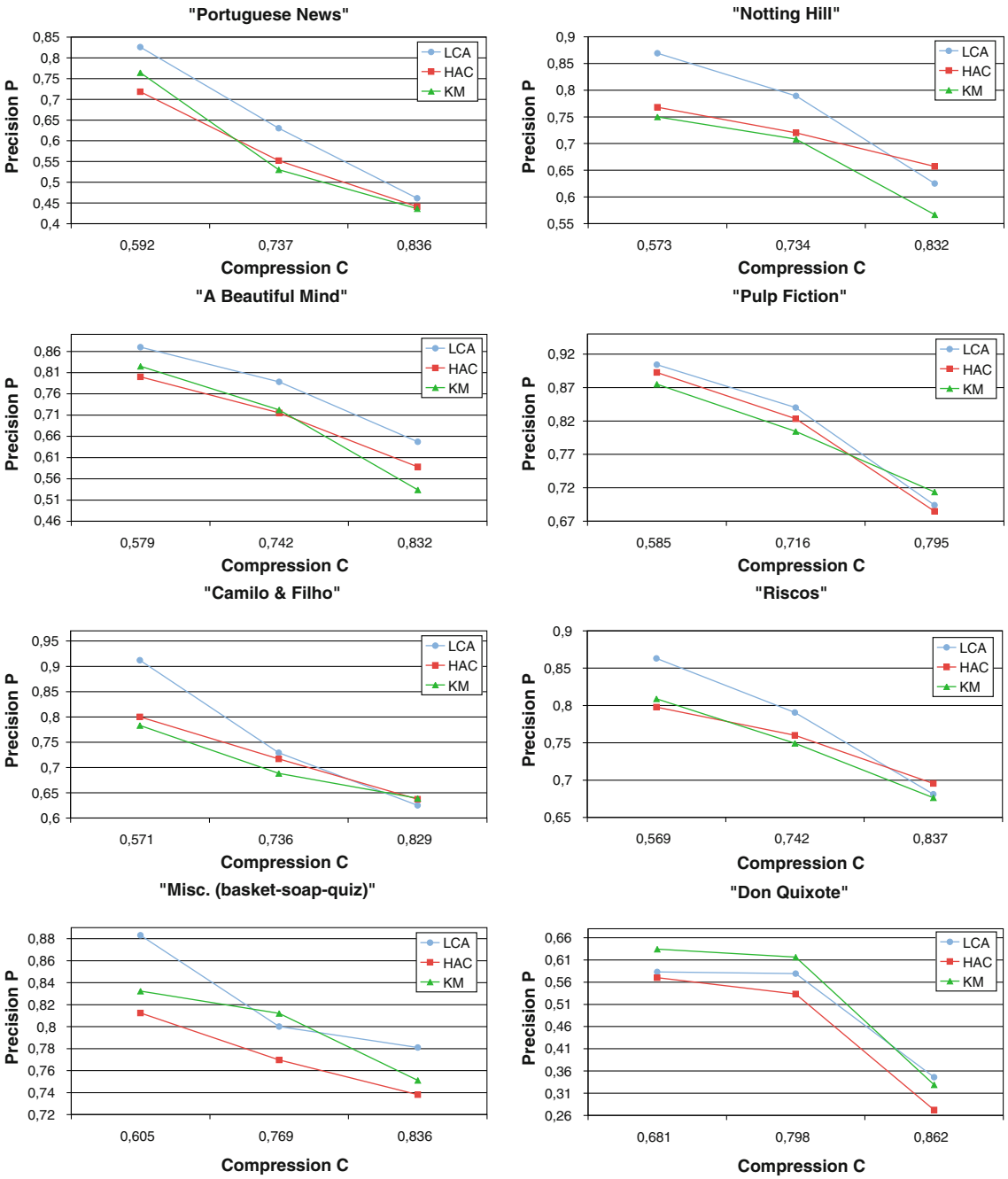

Fig. 11 Piecewise-linear approximations of the *"Compression–Precision"* curves comparing the *LCA*, *HAC* and *KM* clustering methods using the same *HSV* histogram feature

where C_c is the number of correctly detected clusters in the summarization layer, and N_c is the total number of detected clusters at the same layer.

Clearly, at the top summarization layer (all shots belonging to one cluster), the cluster *precision* would be $P = 1$ (according to the semantic *"whole video"*). Since the same value ($P = 1$) would be obtained at the beginning of the clustering when we treat each shot as a cluster, another measure is needed to express the achieved compression factor ("how concise is the summary ?"). The measure of *compression* C, is given by:

$$C = 1 - \frac{N_c}{N_s}, \tag{10}$$

where N_s is the total number of shots. When each cluster is a singleton, the measure of *compression* C is zero. On the contrary *compression* increases as clusters grow bigger and their number reduces.

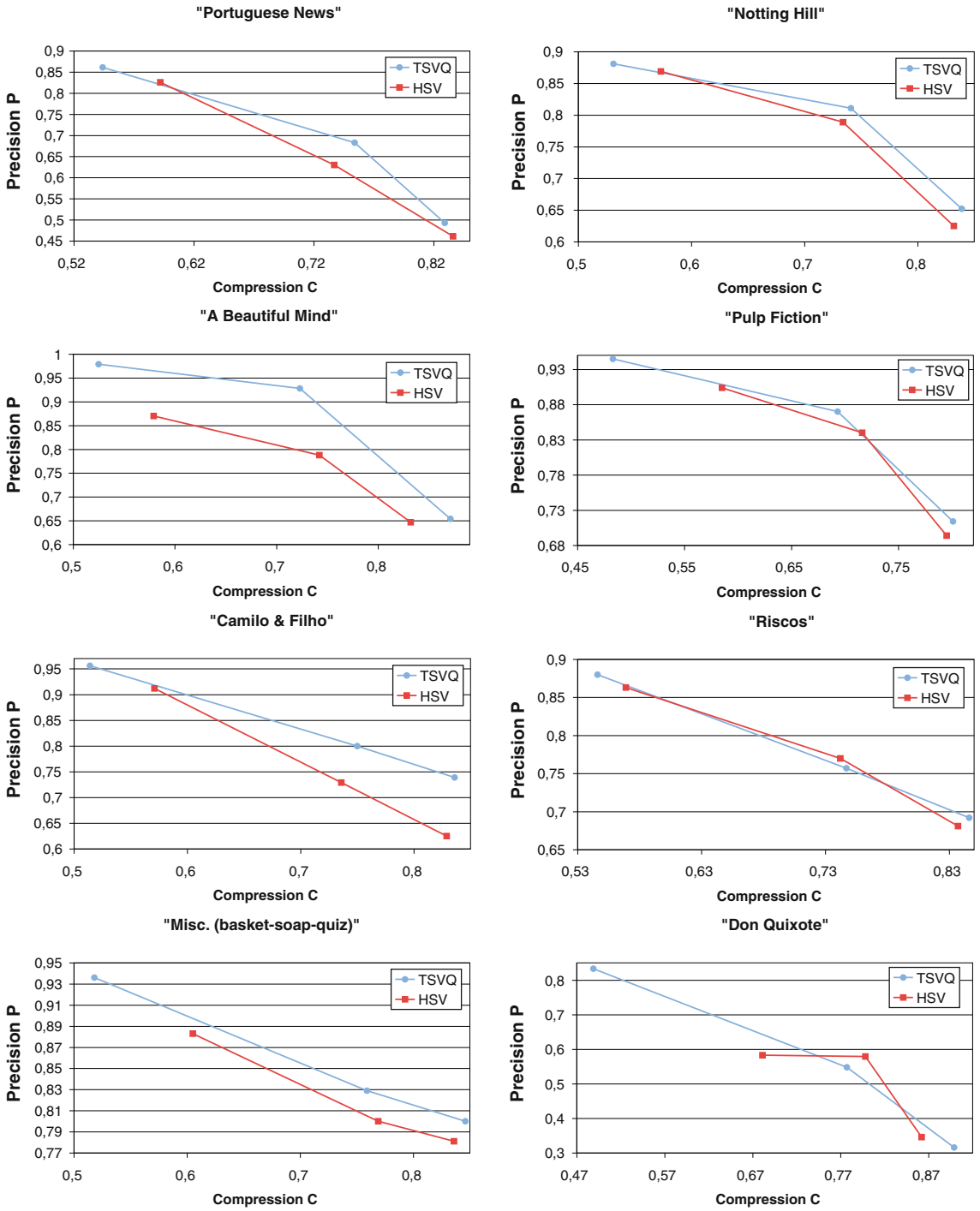

Fig. 12 Piecewise-linear approximations of the *"Compression–Precision"* curves comparing the *TSVQ* code-book with the *HSV* color histogram using the same *LCA* algorithm (color figure online)

7.2 Clustering effectiveness: *LCA* versus other algorithms

Our experimental trials are divided into two main parts to distinguish the contributes provided by the *LCA* approach from those yielded by the *TSVQ*.

In the first part of the experiments, we compare the *Leading-Cluster-Analysis* (*LCA*) with two state-of-the-art approaches: a pure *Hierarchical Agglomerative Clustering* (*HAC*) and a partitional approach based on a *k-means* (*KM*) like the one used in [16]. Tests are performed on the three algorithms by using the same low-level feature, that is the widely employed color histogram on *HSV* space.

Results obtained by the three clustering algorithms *LCA*, *HAC* and *KM* are compared in terms of *precision* and

compression. *P* is computed as the average scores assigned by ten human observers who visually verified the semantic coherence of each cluster at three different levels of *compression* (the first three layers of the summaries found by the *LCA*, with $\delta = 1.5$). Piecewise-linear approximations of the *"Compression–Precision"* curves for all the test video sequences are sketched in Fig. 11. In most cases the *LCA* approach outperforms the other methods in clustering together similar shots showing shared a-posteriori labeled semantics.

7.3 Low-level feature efficiency: *TSVQ* versus *HSV* histogram

In the second set of experiments we compare the *HSV* histogram (using the intersection of color histograms as pairwise similarity measure) with the proposed *TSVQ* feature and its related metric. For the generation of all the code-books we allow a maximum distortion $D_{\max} = 1{,}500$ and a block dimension of 4×4 pixels.

TSVQ code-book and *HSV* histogram are compared in terms of *precision* and *compression*, by using the same clustering algorithm, i.e., the *LCA*, on all the video data sequences. As before, values of *precision* are obtained as the average scores assigned by ten human observers verifying the semantic coherence of each cluster on the first three layers of the summaries found by the *LCA*, with $\delta = 1.5$. Once δ is fixed, the hierarchy depth and the values of *compression* obtained for the same summarization layers can change from video to video.

Piecewise-linear approximations of the *"Compression–Precision"* curves for all the test video sequences are sketched in Fig. 12. In most cases, except for the cartoon, which is characterized by a reduced set of homogenous colors, the *TSVQ* feature outperforms the use of *HSV* color histogram in representing the shot visual-content.

7.4 *LCA* computational complexity

The computational complexity of the *LCA* is comparable to that of any hierarchical algorithm.

Once computed all pairwise dissimilarities, finding the minimum distance pair of shots requires that we scan through the complete table. Thus, for the first agglomerative step, the total complexity is given by $O(N_s(N_s - 1)/2) = O(N_s^2)$.

For an arbitrary merging step instead (i.e., from a number of i clusters to $i - 1$), we need to scan through the $(N_s(N_s - 1)/2 - i)$ unused distances in the table and keep trace of the smallest one. This is again $O(N_s^2)$. If we want to stop the clustering procedure when only one cluster is left, the full time complexity is thus $O(N_s^2 \cdot N_s) = O(N_s^3)$.

In our implementation, the presence of a reduced set of clusters to be monitored (the *leading clusters*) speed up the search for the minimum, thus reducing the overall complexity.

On our test benchmark (video of 90 min with 1,000 shots, Xserve Quad Xeon 64-bit server), the building of the hierarchy by *LCA*, requires less than 5 s.

8 Conclusions

In this paper we have proposed an algorithm for the automatic clustering of video shots, by tailoring a hierarchical clustering method to the specific application of visual-content summarization. To prevent irrelevant shots from being wrongly incorporated into a cluster, we propose a local measure of visual coherence by using the *Leading-Cluster-Analysis*. *LCA* provides a robust solution to the problem of determining how many natural clusters are present at different levels of content representation, and reduces the computational time for the generation of summaries.

The performance of the proposed approach has been evaluated using objective metrics that estimate the content representational value of the obtained hierarchical summaries. In particular, extensive tests have been carried out on a large collection of video data taken from different genres of programmes, by using and comparing different visual features and different clustering algorithms.

References

1. Peng, W.-T., Chiang, Y.-H., Chu, W.-T., Huang, W.-J., Chang, W.-L., Huang, P.-C., Hung, Y.-P.: Aesthetics-based auto-matic home video skimming. Lecture Notes in Computer Science—Advances in Multimedia Modeling, vol. 4903, pp. 186–197 (2008)
2. Trong, B.T., Venkatesh, S.: Video abstraction: a systematic review and classification. ACM Trans. Multimedia Comput. Commun. Appl. **3**, 1–37 (2007)
3. Li, Y., Lee, S.-H., Yeh, C.-H., Kuo, C.-C.J.: Techniques for movie content analysis and skimming. IEEE Signal Process. Mag. **23**, 79–89 (2006)
4. Wang, F., Ngo, C.-W.: Rushes video summarization by object and event understanding. In: Proceedings of the ACM TVS-2007, pp. 25–29, Augsburg, Germany, September (2007)
5. Byrne, D., Kehoe, P., Lee, H., Conaire, C.O., Smeaton, A.F., O'Connor, N., Jones, G.J.F.: A user-centered approach to rushes summarization via highlight-detected keyframes. In: Proceedings of the ACM TVS-2007, pp. 35–39, Augsburg, Germany, September (2007)
6. Koskela, M., Sjoberg, M., Laaksonen, J., Viitaniemi, V., Muurinen, H.: Rushes summarization with self-organizing maps. In: Proceedings of the ACM TVS-2007, pp. 45–49, Augsburg, Germany, September (2007)
7. Rui, Y., Huang, T.: A Unified Framework for Video Browsing and Retrieval. Image and Video Processing Handbook, pp. 705–715. Academic Press, New York (2000)

8. Yeung, M.M., Yeo, B.L., Liu, B.: Segmentation of video by clustering and graph analysis. In: Proceedings of Computer Vision and Image Understanding, vol. 71, no. 1, pp. 94–109, July (1998)
9. Gatica-Perez, D., Loui, A., Sun, M.-T.: Finding structure in home video by probabilistic hierarchical clustering. IEEE Trans. Circuits Syst. Video Technol. **13**, 539–548 (2003)
10. Benini, S., Bianchetti, A., Leonardi, R., Migliorati, P.: Extraction of significant video summaries by dendrogram analysis. In: Proceedings of ICIP'06, Atlanta, GA, USA, 8–11 October (2006)
11. Gersho, A., Gray, R.M.: Vector Quantization and Signal Compression. Kluwer Academic Publishers, Dordrecht (1992)
12. Saraceno, C., Leonardi, R.: Indexing audio-visual databases through a joint audio and video processing. Int. J. Imaging Systems Technol. **9**, 320–331 (1998)
13. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. Wiley-Interscience, New York (2001)
14. Xie, L., Chang, S.-F., Divakaran, A., Sun, H.: Structure analysis of soccer video with hidden markov model. In: Proceedings of ICASSP'02, Orlando, Florida, USA, May (2002)
15. Kijak, E., Oisel, L., Gros, P.: Hierarchical structure analysis of sport videos using hmms. In: Proceedings of ICIP'03, pp. 1025–1028, Barcelona, Spain, September, (2003)
16. Hanjalic, A., Zhang, H.J.: An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis. IEEE Trans. Circuits Syst. Video Technol. **9**, 1280–1289 (1999)
17. Uchihashi, S., Foote, J., Girgensohn, A., Boreczky, J.: Video manga: Generating semantically meaningful video summaries. In: Proceedings of 7th ACM International Conference on Multimedia'99, pp. 383–392, Orlando, Florida, USA, October (1999)
18. Zhao, Y., Karypis, G.: Evaluation of hierarchical clustering algorithms for document datasets. In: Proceedings of the International Conference on Information and Knowledge Management CIKM'02, pp. 515–524, November (2002)
19. Yeung, M.M., Yeo, B.-L.: Time-constrained clustering for segmentation of video into story units. In: Proceedings of ICPR'96, vol. 3, p. 375, Vienna, Austria, August (1996)
20. Sahouria, E., Zakhor, A.: Content analysis of video using principal components. IEEE Trans. Circuits Syst. Video Technol. **9**(8), 1290–1298 (1999)
21. Zhu, X., Fan, J., Elmagarmid, A.K., Wu, X.: Hierarchical video summarization and content description by joint semantic and visual similarity. ACM Multimedia Syst. **9**(1), 31–53 (2003)
22. Ratakonda, K., Sezan, M.I., Crinon, R.: Hierarchical video summarization. In: Proceedings of SPIE'99, vol. 3653, pp. 1531–1541, January (1999)
23. Ferman, A., Tekalp, A.: Two-stage hierarchical video summary extraction to match low-level user browsing preferences. IEEE Trans. Multimedia **5**, 244–256 (2003)
24. Koprinska, I., Clark, J., Carrato, S.: Video gcs - a clustering-based system for video summarization and browsing. In: Proceedings of 6th COST 276 Workshop, pp. 34–40, Thessaloniki, Greece, May (2004)
25. Ngo, C.-W., Pong, T.-C., Zhang, H.-J.: On clustering and retrieval of video shots through temporal slices analysis. IEEE Trans. Multimedia **4**, 446–458 (2002)
26. Kender, J.R., Yeo, B.-L.: Video scene segmentation via continuous video coherence. In: Proceedings of CVPR'98, pp. 367–373, Santa Barbara, CA, USA, May (1998)
27. Sundaram, H., Chang, S.-F.: Computable scenes and structures in films. IEEE Trans. Multimedia **4**, 482–491 (2002)
28. Hanjalic, A., Lagendijk, R.L., Biemond, J.: Automated high-level movie segmentation for advanced video retrieval systems. IEEE Trans. Circuits Syst. Video Technol. **9**, 580–588 (1999)
29. Odobez, J.-M., Gatica-Perez, D., Guillemot, M.: Video shot clustering using spectral methods. In: Proceedings of CBMI'03, Rennes, France, September (2003)
30. Zhang, J., Sun, L., Yang, S., Zhong, Y.: Joint inter and intra shot modeling for spectral video shot clustering. In: Proceedings of ICME'05, Amsterdam, The Netherland, July (2005)
31. Divakaran, A., Radhakrishnan, R., Peker, K.: Motion activity-based extraction of key-frames from video shots. In: Proceedings of ICIP'02, Rochester, NY, USA, September (2002)
32. Smith, M.A., Kanade, T.: Video skimming and characterization through the combination of image and language understanding techniques. In: Proceedings of CVPR'97, pp. 775–781, Puerto Rico, June (1997)
33. Takahashi, Y., Nitta, N., Babaguchi, N.: Video summarization for large sports video archives. In: Proceedings of ICME'05, Amsterdam, The Netherlands, July (2005)
34. DeMenthon, D., Kobla, V., Doermann, D.: Video summarization by curve simplification. In: ACM Multimedia '98, pp. 211–218, Bristol, England, September (1998)
35. You, J., Liu, G., Sun, L., Li, H.: A multiple visual models based perceptive analysis framework for multilevel video summarization. IEEE Trans. Circuits Syst. Video Technol. **17**, 273–285 (2007)
36. Sundaram, H., Chang, S.-F.: Constrained utility maximization for generating visual skims. In: Proceedings of IEEE Workshop Content-Based Access of Image and Video Library '01, pp. 124–131, Kauai, HI (2001)
37. Yeung, M.M., Yeo, B.-L.: Video visualization for compact presentation and fast browsing of pictorial content. IEEE Trans. Circuits Syst. Video Technol. **7**, 771–785 (1997)
38. Chiu, P., Girgensohn, A., Liu, Q.: Stained-glass visualization for highly condensed video summaries. In: Proceedings of ICME'04, Taipei, Taiwan, June (2004)
39. Wang, T., Mei, T., Hua, X.S., Liu, X.L., Zhou, H.Q.: Video collage: A novel presentation of video sequence. In: Proceedings of ICME-2007, pp. 1479–1482, Beijing, China, July (2007)
40. Iran, M., Anandan, P.: Video indexing based on mosaic representation. IEEE Trans. Pattern Anal. Mach. Intell. **86**, 905–921 (1998)
41. Vasconcelos, N., Lippman, A.: A spatio-temporal motion model for video summarization. In: Proceedings of International Conference on Computer Vision and Pattern Recognition CVPR'98, pp. 361–366, Santa Barbara, CA, USA, June (1998)
42. Taniguchi, Y., Akutsu, A., Tonomura, Y.: Panorama excerpts: extracting and packing panoramas for video browsing. In: Proceedings of ACM Multimedia '97, pp. 427–436, November (1997)
43. Sun, X.D., Kankanhalli, M.S.: Video summarization using r-sequences. Real-Time Imaging **6**, 449–459 (2000)
44. Doulamis, A.D., Doulamis, N.D., Kollias, S.D.: A fuzzy video content representation for video summarization and content-based retrieval. Signal Process. **80**, 1049–1067 (2000)
45. Gong, Y., Liu, X.: Video summarization and retrieval using singular value decomposition. ACM MultiMedia Syst. J. **9**, 157–168 (2003)
46. Cotsaces, C., Nikolaidis, N., Pitas, I.: Video shot detection and condensed representation. IEEE Signal Process. Mag. **23**, 28–37 (2006)
47. Koprinska, I., Carrato, S.: Temporal video segmentation: a survey. Signal Process. Image Commun. **16**, 477–500 (2001)
48. Benini, S., Xu, L.-Q., Leonardi, R.: Identifying video content consistency by vector quantization. In: Proceedings of WIAMIS'05, Montreux, Switzerland, 13–15 April (2005)
49. Huang, C.-M., Bi, Q., Stiles, G.S., Harris, R.W.: Fast full search equivalent encoding algorithms for image compression using vector quantization. IEEE Trans. Image Process. **1**, 413–416 (1992)
50. Janjusevic, T., Benini, S., Leonardi, R., Izquierdo, E.: Random assisted browsing of rushes archives. J. Multimedia (2009, accepted)