HIERARCHICAL SUMMARIZATION OF VIDEOS BY TREE-STRUCTURED VECTOR QUANTIZATION

S. Benini, A. Bianchetti, R. Leonardi, P. Migliorati

DEA-SCL, University of Brescia, Via Branze 38, I-25123, Brescia, Italy

ABSTRACT

Accurate grouping of video shots could lead to semantic indexing of video segments for content analysis and retrieval. This paper introduces a novel cluster analysis which, depending both on the video genre and the specific user needs, produces a hierarchical representation of the video only on a reduced number of significant summaries. An outlook on a possible implementation strategy is then suggested. Specifically, vector-quantization codebooks are used to represent the visual content and to cluster the shots with a similar chromatic consistency. The evaluation of the codebook distortion introduced in each cluster is used to stop the procedure on few levels, exploiting the dependency relationships between clusters. Finally, the user can navigate through summaries at each hierarchical level and then decide which level to adopt for eventual post-processing. The effectiveness of the proposed method is validated through a series of experiments on real visual-data excerpted from different kinds of programmes.

1. INTRODUCTION

As the amount of digital video continues to grow, efficient algorithms that enable automated analysis of large multimedia databases are becoming more and more important. While retrieval queries on videos are usually formulated at a higher level, the analysis of such documents is feasible only at the algorithmic level, in terms of low-level features.

For videos, the segmentation into shots and the key-frame extraction are commonly considered as the prior steps for performing effective content-based indexing, browsing and retrieval. However a shot separation often leads to a far too fine segmentation of the sequence. So, building upon this, efforts are invested towards grouping shots into more compact structures that can share common semantic threads.

Methods dealing with data clustering are widely reported in literature [1]. Concerning video shots, in [8] and [6] some approaches based on a time-constrained clustering are presented. In [8] a visual similarity between shots is measured by means of color or pixel correlation between key-frames. In [3] the dissimilarity between shots is examined by estimating correlations between key-frames block matching. Other algorithms calculate a short term memory-based model of shot-to-shot *coherence* [4]. Lately, spectral methods [5] resulted to be effective in capturing perceptual organization features.

The aim of this paper is to propose the use of a tree-structured vector-quantization *(VQ)* codebook as an effective low-level feature for representing each video shot content. Then shots with long-term chromatic consistency are grouped together. The proposed distortion measure allows to stop the clustering process only on few levels, according to the nature of data and to the specific user needs. The goal of such analysis is to generate few hierarchical summaries of the video document, which provide the user with a fast non-linear access to the desired visual material. The obtained results can be useful for further post-processing, such as semantic annotation and story unit detection [3].

The paper is organized as follows: in section 2 vector quantization on shots is introduced; sections 3 and 4 present an effective shot-clustering algorithm which allows a video representation on a reduced number of hierarchical summaries; finally, in sections 5 and 6 experimental results and conclusions are discussed.

2. LOW-LEVEL REPRESENTATION

Assuming to start from an already given shot decomposition, each shot is further analyzed in order to determine its *vector quantization* codebook.

2.1. Key-frame vector quantization

First the central frame of each shot is chosen, even if the procedure is functionally scalable to the case when more than one frame per shot are needed. Then, for each extracted frame, a *tree-structured vector quantization (TSVQ)* codebook is designed so as to reconstruct each frame with a certain distortion with respect to the original one. In the specific, after having been sub-sampled in both directions at *QCIF* resolution, and filtered with a denoising gaussian filter, every frame is divided into non overlapping blocks of $N \times N$ pixels, scanning the image from left to right and top to bottom. All blocks are then represented using the *LUV* color space and used as the training vectors to a *TSVQ* algorithm [2] by using the *Generalized Lloyd Algorithm (GLA)* for codebooks of size $2^n (n = 0, 1, 2, \ldots)$. Each increase in the size of

the codebook is done by splitting codewords from the next smallest codebook (perturbed versions of the old most populated codewords). The *GLA* continues to run until a predetermined maximum distortion (or a maximum codebook size) is reached. Finally the algorithm returns the code vectors and the *TSVQ* codebook final dimension for each investigated shot. Note that the dimensions of each codebook could be different for each single shot. The objective of this approach is to produce codebooks for each key-frame with close distortion values, so as to allow for a further comparison between different codebooks.

2.2. Shot similarity

The similarity between two shots can be measured by using the codebooks computed on respective shots.

Let S_i be a shot, and let K_j be a generic codebook; when a vector $s \in S_i$ is quantized to a vector $k \in K_j$, a quantization error occurs. This quantization error may be measured by the average distortion $D_{K_j}(S_i)$, defined as:

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

where V_i is the number of vectors s_{ip} of shot S_i (the number of $N \times N$ blocks in the shot), and k_{jq} is the code vector of K_j with the smallest euclidean distance from s_{ip}, *i.e.*:

$$q = \arg\min_z \|s_{ip} - k_{jz}\|^2 \quad (2)$$

where $k_{jz} \in K_j$. Furthermore, given two codebooks (K_i and K_j), the value $|D_{K_i}(S_i) - D_{K_j}(S_i)|$ can be interpreted as the distance between the two codebooks, when applied to shot S_i. A symmetric form of the similarity measure used in [7] between shot S_i and shot S_j can, thus, be defined as:

$$\phi(S_i, S_j) = |D_{K_j}(S_i) - D_{K_i}(S_i)| + |D_{K_i}(S_j) - D_{K_j}(S_j)| \quad (3)$$

where $D_{K_i}(S_i)$ is the distortion obtained when shot S_i is quantized using its associated codebook. The smaller ϕ is, the more similar the shots are. It should be noticed that the similarity is based on the cross-effect of the two codebooks on the two shots. In fact, it may happen that the majority of blocks of one shot (for example S_i), can be very well represented by a subset of codewords of codebook 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 codebook K_i doesn't lead to a small distortion when applied to S_j. So cross-effect of codebooks on the two shots is needed to obtain a sound similarity measure.

3. CLUSTERING OF VIDEO SHOTS

Once a shot similarity measure has been defined, the next step is to identify clusters of shots. Suppose we have a sequence with N_s shots. At the beginning of the iterative process each shot belongs to a different cluster (level-N_s).

3.1. Cluster similarity and *Dendrograms*

At each new iteration, the algorithm sets to merge the two most similar clusters, where similarity between clusters C_i and C_j, $\Phi(C_i, C_j)$, is defined as the average of the similarities between shots belonging to C_i and C_j, *i.e.*:

$$\Phi(C_i, C_j) = \frac{1}{N_i N_j} \sum_{S_i \in C_i} \sum_{S_j \in C_j} \phi(S_i, S_j) \quad (4)$$

where N_i (N_j) is the number of shots of cluster C_i (C_j).

The results of the clustering process can be graphically rendered by a dendrogram plot. A dendrogram consists of many $\sqcap$-shaped lines connecting objects in a binary-hierarchical tree. For our scope, a dendrogram represents the whole clustering process of N_s shots, from the level-N_s (each cluster containing one single shot) up to level-1, where a single cluster contains all the shots of the sequence (as in Figure 1). Moreover the height of each $\sqcap$-branch represents the similarity between the two clusters being connected, so that low (high) connections correspond to similar (dissimilar) merged clusters. Through a dendrogram, it is therefore possible to follow the clustering process at each iteration step, every level providing a different representation of the video sequence.

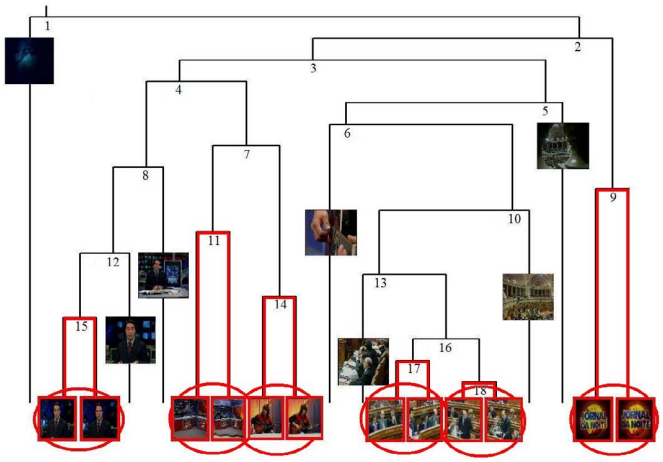

Fig. 1. Dendrogram with the *leading* clusters highlighted.

4. HIERARCHICAL SUMMARIES

Observing the clustering process at each step is almost of no use for a multimedia content consumer, due to the large number of levels. Our principal aim is to automatically determine few significant levels able to offer the user a number of different semantic *summaries* on the observed video sequence.

A news program, for example, can be described at various levels of granularity, but only very few of them are *semantically* significant. For example, the top level *summary* can be the whole programme; on a lower level, it may be helpful to discriminate between the "studio" shots and the "reports"; then, inside the "studio", to distinguish between the "anchorman" shots and ones of the "guest", and so on. With such a scheme, the video content can be expressed progressively, from top to bottom in increasing levels of granularity.

4.1. Leading Clusters

Looking at the dendrogram, it is straightforward to single out the *leading* clusters as the ones originally formed by the fusion of two single shots (see Figure 1). In the upper branches of the dendrogram, each time a *leading* cluster merges with another one, it propagates its property of being a *leading* cluster to the new formed one. Since at each merging step at least one of the two merged cluster is a *leading* cluster, only by following the evolution of the *leading* clusters it is possible to have a complete overview of the entire clustering process.

Let C_k^* be a *leading* cluster, and let us call $C_k^*(i)$ the cluster at level-i, where $i \in I = \{N_s, N_s - 1, \ldots, 1\}$. Following the evolution of C_k^* from level-N_s to level-1 it is possible to evaluate the cluster's internal distortion introduced as the cluster grows bigger. In particular, let $I_k^* = \{i_1, i_2, \ldots, i_n\} \subseteq I$ be the sub-set of levels of I in which $C_k^*(i)$ actually takes part in a merging operation, the internal distortion of cluster C_k^* at level i_j can be expressed as:

$$\Psi(C_k^*(i_j)) = \Phi(C_k^*(i_{j-1}), C_h) \tag{5}$$

where C_h is the cluster (which can be *leading* or not) merged with C_k^* at level-i_j (*i.e.* the internal distortion is given by the cluster similarity between the two clusters being merged).

4.2. Determining *Summaries*

Following the internal distortion of each *leading* cluster C_k^* on each level belonging to I_k^*, it is possible to automatically determine some significant *summaries* over the generated cluster levels. Each *summary* is a particular representation of the video, obtained by using a different grade of cluster aggregation. Observing the internal distortion of each *leading* cluster, $\Psi(C_k^*)$, and setting a threshold on its discrete derivative

$$\Psi'(C_k^*(i_j)) = \Psi(C_k^*(i_j)) - \Psi(C_k^*(i_{j-1})) \tag{6}$$

the user is able to stop the *leading* cluster C_k^* growth at levels $D_k^* = \{i_{d_1}, i_{d_2}, \ldots, ..., i_{d_n}\} \subseteq I_k^*$. These levels indicate meaningful moments in the growth evolution of C_k^* (*i.e.* when the height of the $\sqcap$-branch of the dendrogram varies significantly with respect to the previous steps and the visual content of C_k^* significantly changes).

Once obtained all the sets D_k^* for each C_k^*, all the significant *summaries* for the investigated sequence are obtained. The number of the available *summaries* will be given by $w = \max_k |D_k^*|$, where w is the maximum cardinality among sets D_k^*. If we want to obtain the m^{th} summary ($m = 1, 2, \ldots, w$), the algorithm lets each *leading* cluster C_k^* grow until $C_k^*(i_{d_m})$. Since at each level $i_j^k \in I_k^*$ with $i_1^k \leq i_j^k \leq i_{d_m}^k$ the cluster C_k^* merges with another cluster C_h, if C_h is a *leading* cluster, the condition $i_{d_m}^k \leq i_j^h$ must be met. This condition verifies the dependency relationship between the merging clusters, *i.e.* the case when the cluster C_h has been already arrested at a previous level with regard to that of the merging with C_k^*. If the condition is not fulfilled, the growth of C_k^* must be stopped iteratively at level $i_{(j-1)}^k$ until the dependency condition is verified. The resulting set of all the obtained clusters determines the m^{th} *summary* of the video.

5. EXAMPLES AND RESULTS

Applying this scheme, for example to the *Portuguese News* sequence, data can be parsed into a hierarchical structure, with each level containing a different video *summary*. Looking at Figure 2, at the top (5^{th}) *summary*, a unique cluster contains all shots; at the 4^{th} *summary*, the algorithm distinguishes between a "news programme" cluster and the opening and closing "jingle". Then, on the 3^{rd} *summary*, the "report" shots, the "studio", and the "jingle" shots are presented in separated clusters. The hierarchical decomposition continues on lower *summaries* at increasing levels of granularity. For each *summary*, the user can evaluate the quality of the decomposition with respects to his/her own desires. After that, he/she can recursively descend the *summaries* until a satisfactory result is achieved.

In order to objectively evaluate the cluster decomposition accuracy, we carried out some experiments using video segments from one news programme and three feature movies, for a total time of about 2 hours.

To judge the quality of the detected results, the following rule is applied: *a cluster belonging to the m^{th} summary is judged to be correctly detected if and only if* ***all*** *shots in the current cluster share a common semantic meaning. Otherwise the current cluster is judged to be falsely detected.* Clustering *Precision* P is used for performance evaluation, where P is defined as the number of rightly detected clusters over all detected ones. Clearly, at the top level summary (all shots belonging to one cluster), the cluster detection precision would be 100%. And the same happens if we treat each shot as a cluster. Hence, in order to discriminate the representative power of a given summary, another measure is needed to express the *Compression* factor of the summary, i.e. one minus the ratio between the number of clusters of the given summary and the total number of shots in the video.

The experimental results of cluster detection at different summarization levels for all our video data set are given in Table 1 in terms of *Precision P* and *Compression C*.

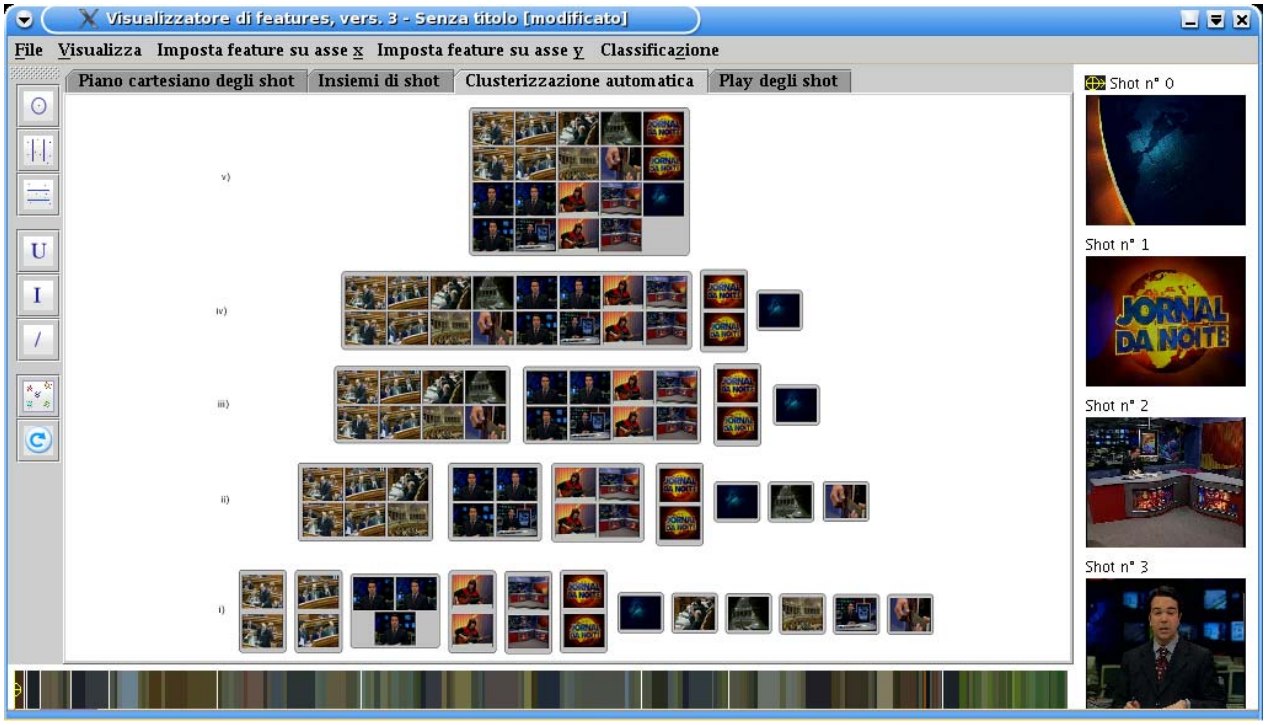

Fig. 2. Hierarchical *summaries* for the *Portuguese News* programme.

6. CONCLUSIONS

The work describes the issue of clustering shots by using a tree-structured vector quantization representation. The proposed hierarchical clustering is suitable for expressing video content progressively at increasing levels of granularity. Obtained clusters allows to extract semantic *views* only on few levels of clustering and provide the user with fast access to the desired video material for eventual post-processing.

Video (genre)	Summary	*C (%)*	*P (%)*
Portuguese News (news)	1^{st} *(217 clusters)*	54.4	86.1
476 shots	2^{nd} *(117 clusters)*	75.4	68.3
47:21	3^{rd} *(81 clusters)*	82.9	49.3
Notting Hill (movie)	1^{st} *(201 clusters)*	53.1	88.1
429 shots	2^{nd} *(111 clusters)*	74.1	81.1
30:00	3^{rd} *(69 clusters)*	83.9	73.9
A Beautiful Mind (movie)	1^{st} *(98 clusters)*	53.4	93.1
210 shots	2^{nd} *(60 clusters)*	71.4	81.1
17:42	3^{rd} *(41 clusters)*	80.4	69.1
Pulp Fiction (movie)	1^{st} *(91 clusters)*	48.3	94.5
176 shots	2^{nd} *(54 clusters)*	69.3	87.0
20:30	3^{rd} *(35 clusters)*	80.1	71.4

Table 1. For each video the first three *summaries* are presented in terms of *Compression C* and *Precision P*.

7. REFERENCES

[1] R. O. Duda, P. E. Hart and D. G. Stork, "Pattern Classification", Wiley-Interscience, 2^{nd} ed., New York, 2001.

[2] A. Gersho and R. M. Gray, "Vector Quantization and Signal Compression", Kluwer Acad. Publishers, 1992.

[3] A. Hanjalic and R. L. Lagendijk, "Automated high-level movie segmentation for advanced video retrieval systems," IEEE Trans. on CSVT, Vol 9, No 4, June 1999.

[4] J. R. Kender and B.-L. Yeo, "Video scene segmentation via continuous video coherence", CVPR'98, pp. 367-373, Santa Barbara, USA, 1998.

[5] J-M. Odobez, D. Gatica-Perez and M. Guillemot, "Video shot clustering using spectral methods", CBMI'03, Rennes, France, Sept 2003.

[6] E. Sahouria and A. Zakhor, "Content analysis of video using principal components," IEEE Trans. CSVT, Vol. 9, No. 8, pp. 1290-1298, 1999.

[7] C. Saraceno and R. Leonardi, "Indexing audio-visual databases through a joint audio and video processing", Int. Journal of Imaging Systems and Technology, Vol. 9, No. 5, pp. 320-331, Oct 1998.

[8] M. M. Yeung and B.-L. Yeo, "Time-constrained clustering for segmentation of video into story units," ICPR'96, Vol III-Vol 7276, p.375, Vienna, Austria, Aug 1996.