Using Content Analysis for Video Compression

Sergio Benini, Aldo Bianchetti, and Riccardo Leonardi

DEA-SCL, University of Brescia, Via Branze 38, 25123 Brescia, Italy
Phone number: +39 030 3715434 - Fax: +39 030 380014
Email: {name.surname}@ing.unibs.it

Abstract. This paper suggests the idea to model video information as a concatenation of different recurring sources. For each source a different tailored compressed representation can be optimally designed so as to best match the intrinsic characteristics of the viewed scene. Since in a video, a shot or scene with similar visual content recurs more than once, even at distant intervals in time, this enables to build a more compact representation of information. In a specific implementation of this idea, we suggest a content-based approach to structure video sequences into hierarchical summaries, and have each such summary represented by a tailored set of dictionaries of codewords. Vector quantization techniques, formerly employed for compression purposes only, have been here used first to represent the visual content of video shots and then to exploit visual-content redundancy inside the video. The depth in the hierarchy determines the precision in the representation both from a structural point of view and from a quality level in reproducing the video sequence. The effectiveness of the proposed method is demonstrated by preliminary tests performed on a limited collection of video-data excerpted from a feature movie. Some additional functionalities such as video skimming may remarkably benefit from this type of representation.
Index Terms Content Analysis, Vector Quantization *(VQ)*, Hierarchical Clustering, Long-term Recurrence

1 Introduction

As we are living in the multimedia era, tremendous amounts of video data have been made available to the normal users. This condition determines a twofold effect: on one side the increasing need for efficient retrieval of desired information; on the other hand this has led to the development of algorithms that enable compression of large video data.

This paper addresses the problem of video coding suggesting the idea that a sound analysis on visual content can improve existing coding schemes.

Concerning video content analysis, the segmentation into shots is now commonly considered as the prior step for performing effective content-based indexing, summarization [4] and retrieval [7]. However, a shot separation often leads to a far too fine temporal segmentation. So building upon this, efforts are invested towards grouping shots into more compact structures (clusters) sharing chromatic consistency and often common semantic threads ([12], [8]). Providing a compact representation of a video, clustering of shots results to be useful for generating video summaries. Moreover, since in a sequence, a shot or a scene with similar visual content may occur more than once, even at distant intervals in time, clusters enables to model video information as a concatenation of different recurring sources. In addition to the already investigated spatial and temporal redundancies, this long-term recurrence of video content can be considered another redundancy to be exploited by coding architectures.

We are aware that the vector quantization technique [3] we employ in this work has been widely adopted in the past for coding, and that new more performing methods have been proposed recently (see for example [9]). However, the aim of this paper is only to show that even a preliminary content analysis performed on video can be extremely helpful for improving generic video coding schemes. In this context vector quantization is a rather peculiar technique, since it can be used for two different purposes. On one side *VQ* codebooks are used as low-level features to represent the shot content. Building upon this, similar shots are grouped together in a hierarchical clustering scheme, and a segmentation into *Logical Story Units (LSU)* [5] can be obtained once chosen a specific level of clustering. On the other side, exploiting the recurrence of similar shots even distant in time, *VQ* codebooks built on clusters can be used for coding.

The paper is organized as follows: in section 2 video content is analyzed from *VQ* codebook similarities; section 3 deals with the structuring of the video content, while section 4 shows how the gained knowledge on video structure can be exploited in a generic coding scheme; finally, in sections 5 and 6 experimental results and conclusions are discussed.

2 Video Content Analysis

Starting from an already given shot decomposition, each shot is further analyzed in terms of low-level features, determining its *vector quantization* codebook on color information.

2.1 *VQ* for Video Shot Representation

The procedure is functionally scalable to the case when more than one frame per shot is needed to represent its visual content. So five frames per second are first extracted from the video stream. Then, for each shot, a *tree-structured vector quantization (TSVQ)* codebook [3] 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 for a *TSVQ* algorithm by using the *Generalized Lloyd Algorithm (GLA)* for codebook design 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 pre-determined maximum distortion (or a maximum given codebook size) is reached. Then, an iterative attempt is made to reduce the number of codewords in the interval $[2^n, 2^{n-1}]$ without exceeding the pre-determined distortion limit. Finally the algorithm returns the code vectors and the *TSVQ* codebook final dimension for each investigated shot. Note that in this phase, the dimensions of each codebook could be different for each single shot. The objective of this approach is to produce codebooks for each keyframe with close distortion values, so as to allow for a further comparison between different codebooks.

2.2 Shot-to-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:

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

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.*:

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

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 [10] between shot S_i and shot S_j can, thus, be defined as:

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

where $D_{K_i}(S_i)$ is the distortion obtained when shot S_i is quantized using its associated codebook. The smaller $\phi(S_i, S_j)$ is, the more similar the two shot contents 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 the cross-effect of codebooks on the two shots is needed to obtain a sound similarity measure.

2.3 Clustering of Video Shots

Now that is possible to evaluate similarity between shots, the next step is to identify clusters of shots. Suppose we have a sequence with N_s shots. At the beginning of the process each shot belongs to a different cluster (level-N_s). 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.*:

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

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

2.4 Dendrograms

The ouput resulting from a clustering process can be graphically rendered by a dendrogram plot. A dendrogram consists of many ⊓-shaped lines connecting objects in a binary 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 ⊓-branch can be adopted so as to represent the similarity between the two clusters it connects, 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.

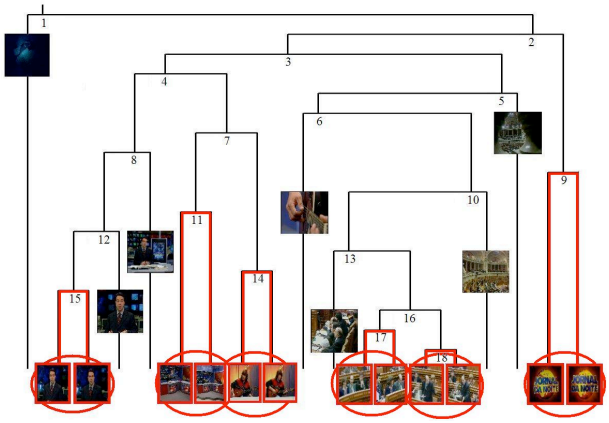

Fig. 1. Dendrogram with the first shots that will be clustered together that have been highlighted.

Fig. 2. Hierarchical clustering for *Pulp Fiction*.

3 Structuring Video Content

The dendrogram representation allows to adopt different strategies for arresting the growth of each cluster at different levels of aggregation. For example, the method recently proposed in [2] arrests each cluster growth by evaluating an internal distortion based on the similarity between clusters of equation 4. Applying this scheme, clusters of shots can be finally parsed into a hierarchical structure, each level containing a compact overview (i.e. a *summary*) of the video at different granularity. As an example, this method is applied to a short sequence from the movie *Pulp Fiction* and the resulting output is shown in Figure 2.

At the top of the hierarchy the (4^{th}) *summary* is a unique cluster containing all the shots; the 3^{rd} *summary* distinguishes among three different settings (a room, a car and a kitchen). Then, on lower *summaries* the hierarchical decomposition continues with increasing levels of granularity (distinguishing different characters inside the car, etc.).

In order to find out the best level of summarization (i.e. the best cluster accuracy), the *cluster validity analysis* as in [6] can be used. Independently from the chosen clustering method, the depth of the arrest level in the cluster aggregation determines the precision in the representation both from a structural point of view (*i.e.* the accuracy of clusters) and, later, from a quality level in reproducing the video sequence (*i.e.* the *PSNR* at the decoder side).

3.1 Segmentation into *Logical Story Units*

Once chosen a level of cluster representation, the *Scene Transition Graph (STG)* as proposed in [11] can be conveniently employed to detect the *Logical Story Unit (LSU)* (which can be considered the best computable approximation to a semantic scene [5]), thus extracting the story structure without a priori knowledge of the video.

In the *STG* the clusters and the shot-transitions between clusters, form a directed graph: it comprises nodes (i.e. the clusters) that contain a number of visually similar and temporally close shots

(see [1] for details), and the edges between nodes (i.e. the transitions between shots) that represent the time evolution of the story. A special type of transition between two nodes is the so-called *cut-edge*, which, if removed, leads to the decomposition of the graph into two disconnected sub-graphs. Therefore, a *cut edge*, being a one-way transition between clusters with unrelated video content, defines a *Logical Story Unit* boundary. In Figure 3 one example of *Scene Transition Graph* formed on a clusterization level of a *Pulp Fiction* sequence is given. The excerpt contains 2 *Logical Story Units* and the *LSU* boundary is placed in correspondence of the *cut-edge* transition which has been highlighted.

4 Exploting Video Structure for Video Coding

As a result of such content analysis, the video is segmented into a sequence of logical story units (*i.e.* $\|Lsu1\|Lsu2\|\ldots\|LsuN\|$), where $\|$ stands for a *LSU* boundary. Each logical story unit contains a number of shots, and each shot belongs to a cluster (denoted by a label A, B, C, ...) node of the *STG* [11] at the chosen level of clustering.

4.1 Recurrent Video Sources

Following the temporal transitions between shots on the transition graph, each video sequence can then be described by a sequence of shot-patterns between clusters separated by *LSU* boundaries (such as $\|ABCBCAB\|DFGHIHIHLD\|MNAM\|\ldots$).
It has to be noted that shots with the same label (*i.e.* belonging to the same cluster) can occur more than once not only in the same story unit, but also distant in time in different *LSU*. In fact, as described in [1], when two shots belonging to the same cluster are very distant in time, they cannot be considered as part of the same *LSU*.

These long-term recurring similar video shots are the recurrent video sources, whose redundancy will be exploited in the proposed coding scheme.

4.2 *VQ* for Video Shot Coding

Since each cluster (A, B, C, ...) contains similar shots, it is possible to represent the entire cluster visual content by a single *VQ* codebook computed on all the shots belonging to the cluster. This means that the VQ codebooks have to be redesigned; in this case, the capacity of the codebook of well-representing the shot visual content is exploited not with the purpose of clustering, but with the aim of coding the sequence. In fact the re-designed cluster codebooks can be used to code each frame belonging to a shot included in the cluster itself.

The flexibility offered by the subdivision into *LSU* could be helpful for some additional functionalities such as video skimming or the user requested streaming of a specific video clip. In fact a particular *LSU* (for example a "cult" scene of a movie) can be coded independently from the others, once the codebooks of its constituent shots are sent to the decoder.

4.3 Coding Scheme

Thus, in order to code a generic video sequence, the following scheme can be defined.

1. At the beginning of the stream the codebooks of the clusters belonging to the requested *LSU* are transmitted to the receiver;
2. At the beginning of each shot, the index of the belonging cluster is sent. This determines which *VQ* cluster codebook has to be used;
3. Each frame of the shot is then coded using the associated *VQ* cluster codebook, each frame block being so represented by the index of the most similar codeword (i.e. the one at the minimum distance) only.

Consequently a *VQ*-compressed video is simply a concatenation of different addresses, collected for all the video frame blocks. Since all the needed *VQ* codebooks are available at the receiver side, a *VQ*-compressed video can be easily decoded by filling in the frame blocks, according to the addresses received by the decoder.

5 Experimental Results

In order to evaluate the coding performance of an architecture adopting a preliminary content analysis on the video, we carried out some experiments only by comparing the proposed scheme with simple vector quantization based approaches. These experiments only want to demonstrate that existing compression methods, when combined with a sound content analysis, offer promising performance at least for certain video programme categories. The

proposed idea, if optimized, could outperform traditional video compression methods, where spatial and temporal redundancy is typically exploited only between adjacent frames.

For our preliminary test we used a video segment taken from one feature movie ("Pulp Fiction") for a total of 4804 frames (at 25 frame/s). The segment contains a total of 44 shots, that can be grouped into 6 clusters and then subdivided into 2 *LSU* showing classic dialogue shot-pattern. The obtained 6 clusters and the segmentation into the 2 *LSU* is shown in Figure 3.

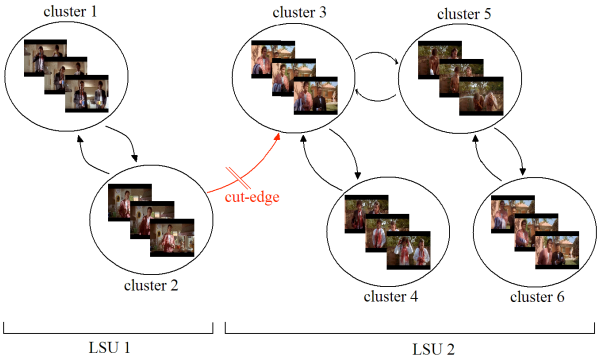

Fig. 3. The six resulting clusters and the sub-division into two *logical story units* for the sequence extracted from *Pulp Fiction*.

For the codebook generation we adopted the same procedure described in section 2.1 for the video content representation, except that it is conducted at CIF resolution. In order to code the examined sequence, different coding scenarios have been compared.

Generic Codebook. The entire sequence has been coded using a *VQ* codebook built starting from a generic set of image sequences, not related to the test sequence excerpted from "Pulp Fiction".

Sequence Codebook. In this second scenario the sequence has been coded using a *VQ* codebook built starting from the frames belonging to the whole test sequence. In particular one frame every four seconds has been used for the codebook generation.

Shot Codebooks. In this case the sequence has been coded using a different *VQ* codebook for each single shot of the test sequence. Each codebook has been generated exploiting the specific shot video content only. In particular five frames per second of each shot have been employed as the training set for the generating process.

Cluster Codebooks. The last coding solution tries to exploit the results of the preceding video content analysis, and one codebook per each obtained cluster has been extracted. Concerning our sequence, all the shots have been grouped into six distinct clusters. For each of them, the codebook has been generated using five frames per second of all the shots belonging to the cluster.
In all the four examined scenarios, codebooks of different sizes have been extracted. Of course as the codebook dimensions increase, the bit-rate becomes larger and lower distortion values are obtained at the decoder side. Concerning the bit-rate computation, no optimization has been applied in order to efficiently transmit the codebooks and the codeword indexes, since the main objective was only to demonstrate that such a content-based coding can outperform traditional compression techniques.

In Figure 4 the rate-distortion curves related to the four coding scenarios are showed. Due to the grouping of similar shots and the computation of only one codebook per cluster, the method relying on a preliminary analysis of the video content outperforms other coding solutions showing the highest rate-distortion curve. It has to be noted that the cost for sending all codebooks has been considered in the computation of the rate-distortion performance.

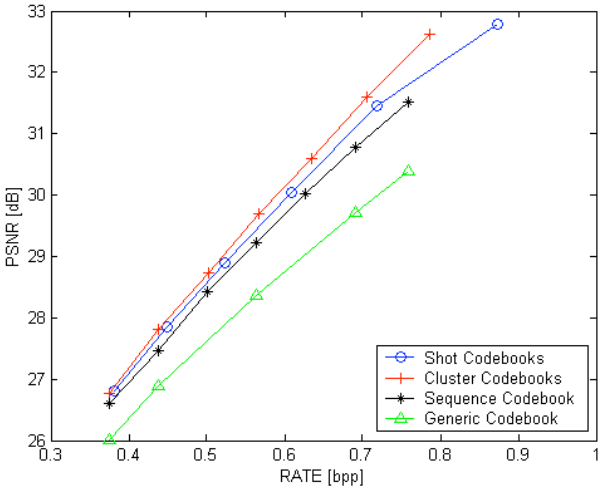

Fig. 4. Rate-distortion curves for the four scenarios.

6 Conclusions

The paper introduces the idea of a content-based coding for video, where the exploited redundancy is given by the (even long-term) recurrence of shots showing similar video content. Preliminary tests performed on a feature movie demonstrate that, if optimized, such a scheme could outperform traditional compression methods and can be an inviting research topic.

References

1. S. Benini, L.-Q. Xu and R. Leonardi , "Identifying video content consistency by vector quantization", WIAMIS'05, Montreux, Switzerland, Apr 2005.
2. S. Benini, A. Bianchetti, R. Leonardi and P. Migliorati, "Video shot clustering and summarization through dendrograms", WIAMIS'06, Seoul, South Korea, Apr 2006.
3. A. Gersho and R. M. Gray, "Vector Quantization and Signal Compression", Kluwer Academic Publishers, 1992.
4. Y. Gong and X. Liu, "Video summarization and retrieval using Singular Value Decomposition," ACM MM Systems Journal, Vol. 9, No. 2, pp. 157-168, Aug 2003.
5. A. Hanjalic and R. L. Lagendijk, "Automated high-level movie segmentation for advanced video retrieval systems," IEEE Trans. on CSVT, Vol. 9, No. 4, Jun 1999.
6. A. Hanjalic and H. J. Zhang, "An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis," IEEE Trans. on CSVT, Vol. 9, No. 8, Dec 1999.
7. C.-W. Ngo, T.-C. Pong and H.-J. Zhang, "On clustering and retrieval of video shots", ACMMM'01, Ottawa, Canada, 2001.
8. J-M. Odobez, D. Gatica-Perez and M. Guillemot, "Video shot clustering using spectral methods", CBMI'03, Rennes, France, Sep 2003.
9. I. E. G. Richardson, "H.264 and MPEG-4 Video Compression", John Wiley & Sons, Sep 2003.
10. 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.
11. 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.
12. D. Q. Zhang, C. Y. Lin, S. F. Chang and J. R. Smith, "Semantic video clustering across sources using bipartite spectral clustering," ICME'04, Taiwan, Jun 2004.