A Compressed Video Database Structured for Active Browsing and Search *

Cüneyt Taşkiran, Jau-Yuen Chen, Charles A. Bouman and Edward J. Delp
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285
{taskiran,jauyuen,bouman,ace}@ecn.purdue.edu

June 1, 1999

Abstract

We describe a unique system called *ViBE* (video browsing environment) for browsing and searching large databases of video sequences. The system first computes the DC sequence for a given MPEG sequence. It then detects and identifies shot boundaries by using the generalized trace. A hierarchical tree structure is constructed for shot comparison and keyframe extraction. In addition to low-level image features, the system also uses pseudo-semantic features to characterize the frames. Finally, the results are presented to the user in an active browsing environment which we call a similarity pyramid. The users can also prune and reorganize the environment using relevance feedback methods.

1 Introduction

Due to rapid advances in compression technology, the expansion of low-cost storage media, and the explosive growth of the internet, the availability of video has greatly increased. However, the technology for organizing and searching vast video sources is still in its infancy.

In recent years tremendous attention has been given to developing content-based methods to access video sequences and finding better ways of presenting the results to the user. Most of the initial work has concentrated on developing reliable methods for scene change detection [1], [2]. Recently, however, the focus has shifted to key frame detection, shot clustering, and to design of complete systems for extracting information from video sequences and presenting it compactly to the user [3]. However, most previous approaches have focused on processing a single video sequence. This approach has the drawback of not exploiting shot clustering *across* sequences, which can substantially limit the search power of the video database. Another problem is that techniques designed to analyze single sequences may fail to scale in performance when dealing with hundreds of hours of video of different types.

Another important trend is the use of algorithms which operate directly on the compressed video stream. Such approachs are important for a number of reasons. First, it often is not practical to decompress video data because of the large computational demands. But in addition, the compressed video stream contains a rich set of precomputed features, such as motion vectors, that can be used to analyze and organize the data.

In this paper we describe a complete video database management system to be used on a database of MPEG compressed video sequences. The system includes content based analysis of video and supports an active environment for browsing and searching the database. An essential component of the approach is the use of compression domain processing. In particular, all our processing is based on the DC image sequence which can be extracted from the DC components of the DCT transformations. The DC images form a "thumbnail" representation of the video sequences which are then used to perform the four major functions of the ViBE system: scene change detection and classification; shot representation using a hierarchical keyframe structure; pseudo-semantic classification; and active browsing using a hierarchical data structure which we call a similarity pyramid.

*Appeared in ICIP '98, vol. 3, pp. 133-137, Chicago, IL, Oct. 4-7, 1998.

2 Scene Change Detection and Identification

We will refer to each continuous video sequence as a shot. Our first task is to segment out shots by detecting and classifying scene changes. Scene changes can take a variety of forms ranging from abrupt cuts to dissolves, fades, and special effects such as wipes and page turns. Our objective is to both detect and classify each of these scene change types so that the associated shot may be segmented out of the sequence.

The feature vector for performing scene change detection and classification is extracted from the compressed video stream and the associated DC sequence. The DC sequence is formed from the DC coefficients of the DCT transform blocks. While the DC image is directly available for I frames, we have used the method of [4] to estimate the DC image for B and P frames.

From the DC image and the compressed video stream, we extract a 12 dimensional feature vector which we call the *generalized trace* (GT) [5]. The GT for frame i of the sequence is denoted by $\vec{x}_i$ where $x_{i,j}$ is its j^{th} component. The elements of the vector are as follows:

$x_{i,1-3}$ - Histogram intersection between frames i and $i-1$ for the Y, U, and V color components.
$x_{i,4-6}$ - Absolute difference between variance of i^{th} DC image and variance of $(i-1)^{th}$ DC image for the Y, U, and V color components.
$x_{i,7}$ - Number of intracoded macroblocks.
$x_{i,8}$ - Number of forward predicted macroblocks.
$x_{i,9}$ - Number of backward predicted macroblocks.
$x_{i,10-12}$ - Flags which identify the frame type as I, P, or B.

Figure 1 illustrates how the components of the GT vary with frame number for a typical sequence. Importantly, the information contained in the GT must be interpreted differently for different frames of the sequence. For example, the number of intracoded macroblocks per frame ($x_{i,7}$) will differ for I, P, and B frames. In order to use this information effectively, we include a binary flag for each of the possible frame types, I, P, or B.

A conventional approach to scene change detection would be to compute the norm of the GT and threshold the result. However, from Fig. 1 it is clear that the time-varying and diverse nature of the GT's components makes this type of direct thresholding approach unworkable. To address this problem, we use a regression tree [6] to estimate the probability of a frame cut given the GT. The regression tree is first designed in a training processes that uses sequences with known scene cut locations [7]. Once designed, the tree may be applied to any sequence. The regression tree has the advantages that it normalizes the output to the range $[0, 1]$ for consistent thresholding, and it accounts for the specific frame type being considered.

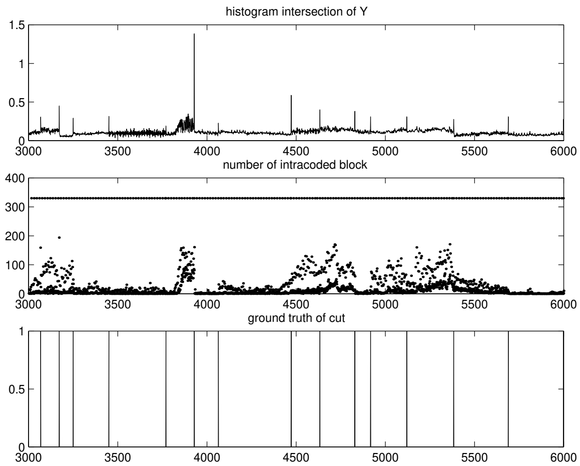


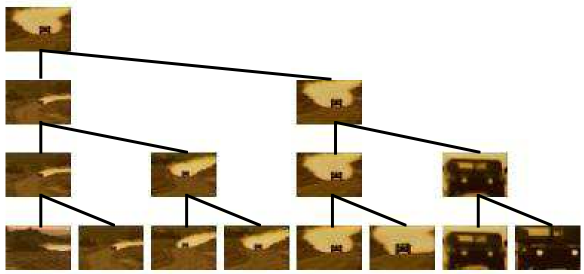
Figure 1: Plots of (1-histogram intersection), number of intracoded macroblocks per frame, and ground truth for scene cut locations.

3 Shot Representation

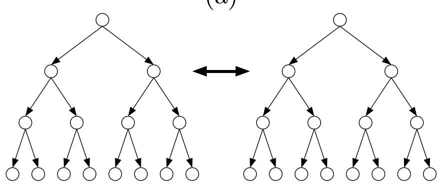
Typically, a shot may be represented by one or more sequentially selected keyframes [2]. However, these representations have been linear and lack a hierarchical structure. This section describes a novel shot representation based on a tree structure formed by clustering the frames in a shot. The tree structure allows important operations, such as similarity comparisons, to be obtained efficiently using tree matching algorithms.

This tree representation is obtained through agglomerative clustering of the shot's frames [8]. We use the global color, texture and edge histograms proposed in [9] as the feature vector for each frame in a shot, and we use the L_1 norm to measure feature distances. Figure 2(a) illustrates such a representation for a shot from an action sequence. In this example, the shot contains significant changes which can not be captured by a single keyframe; so the tree structure can better represent the diverse aspects of the shot. The tree structure hierarchically organizes frames into similar groups, therefore allowing automatic detection of subgroups inside a shot.

The dissimlarity between two shots is measured by computing the distance between the two corresponding trees as shown in Figure 2(b). Each node, s, of the tree is associated with a set of frames and contains an



(a)



(b)

Figure 2: (a) Each shot is represented by a hierarchy of key frames which are extracted with agglomerative clustering; (b) Shot dissimilarity is then determined by computing the distance between two trees.

associated feature vector, z_s, for the cluster of frames. Generally, z_s is computed to be the centroid of the feature vectors in the cluster. The distance between the trees is then given by the weighted sum of distances between nodes for the best correspondence mapping between trees. For the trees of depth three used in this paper, there are eight distinct correspondences which must be checked.

4 Shot Description

Most approaches in content-based retrieval rely on descriptions which are either problem-specific such as anchor shot models in news video [10] or low-level models such as color and edges. While they are very easy to derive, low-level features severely limit the capability of user interaction for non-expert users. However, the extraction of the truly semantic features required currently is not possible.

To overcome this difficulty a number of pseudo-semantic features, that bridge the gap between low level and semantic features, have been proposed. Interesting work in this area includes [11] where it has been shown that two pseudo-semantic features, average shot activity and duration correlate well with semantic features such as "action", "romance", or "comedy", and [12] where models of human bodies and audio features are coupled to detect violence in a given sequence.

Considering this, we have chosen to complement the statistical features for a shot described in Section 3 with a number of pseudo-semantic features, which are easy to derive from low level image features. A feature vector is associated with each shot that includes the following attributes: {head and shoulders, indoor/outdoor, action}. Each of these attributes has a continuous value in the range [0,1] which indicates the belief that the shot has the corresponding attribute.

5 Active Browsing

In this section, we make use of the similarity pyramid we proposed in [13] to organize the video database. This pyramid is created by clustering the keyframes of the shot representation described in Section 3. Figure 3 illustrates a similarity pyramid and its embedded quadtree for a video database. The similarity pyramid is created by first hierarchically clustering the shots into a quadtree structure, and then mapping the quadtree's clusters onto the 2-D grid at each level of the pyramid. The shots are clustered using the distance metric of Section 3.

Each cluster of the pyramid is represented by a single icon keyframe chosen from the cluster; so as the user moves through the pyramid they have a sense of database content at various scales. The similarity pyramid has a user interface which allows movement up and down through levels. It also allows users to pan across a single level of the database to locate video shots of interest.

The relevance feedback has long been used to improve the performance of search-by-query [14, 15]. However, our use of relevance feedback is distinct in two ways: First, we use the feedback to reorganize and prune the browsing environment, rather then simply modifying a search criterion. Second, for the browsing problem, we can investigate more sophisticated and non-iterative methods to extract information from relevance feedback, and then use this information to define pseudo-semantic classes.

Our approach consists of two major components. In the first phase, the relevant shots are clustered to find natural groupings inside the set of relevant shots. This clustering is based on a standard agglomerative clustering algorithm and forms a dendrogram (binary tree) of shots. We then cut the dendrogram to produce N clusters. The main problem is to determine the number of clusters (N) which best fits the data. We solve this problem using a cross validation strategy. We methodically leave out one shot, cluster the remaining shots into N clusters, and then compute the ranking of the left-out shot. By minimizing this ranking with respect to N, we can find the optimal cluster order, N.

Figure 3: An example of a similarity pyramid and its embedded quadtree.

After determining the groupings among the relevant shots, we apply the feature weighting optimization to separate the relevant shots from the irrelevant shots. The cost function we used is based on the ranking of each relevant shot among the domain of feedback with respect to every other relevant shot. Define D as the index set of the shots displayed on the screen, and $R \subset D$ as the index set of the relevant shots, and $z_i, i = 1 \cdots N$ as the centroid of clusters of the relevant shots. The dissimilarity measure between shot x_j and the set of relevant shots R is defined as,

$$d_\theta(x_i) = \min_{j=1,\cdots,N} d_\theta(x_i, z_j)$$

where $d_\theta(.,.)$ is the dissimilarity measure with a set of weightings, θ. Let $r_\theta(x_j)$ be the ranking of x_j with respect to $d_\theta(.)$ among the set D. The cost of selecting weighting set θ is then defined as

$$c(\theta) = \sum_{j \in R} \log r_\theta(x_j)$$

The optimal θ is then computed by minimizing $c(\theta)$.

With a set of clusters of relevant shots and a set of optimal weighting, we then find the M most similar shots, $x_i, i = 1, \cdots, M$, which minimize the distance function $d_\theta(.)$. These M shots are then organized into a similarity pyramid according to algorithm defined in [13] using the dissimilarity $d_\theta(\cdot,\cdot)$.

6 Experimental Results

We have used a database consisting of 8 ground-truthed video clips and 7 additional video clips. The clips are each approximately 10 minutes long, and come from sources such as CNN Headline News, C-SPAN, local newscasts, and a Hummer promotional video. We use two sequences to train the tree classifier and the remaining six ground-truthed clips to test classification performance.

Figure 4 shows the results of applying the tree classifier to scene cut detection. The category "misclassification" is the number of dissolves or fade scene changes that are misclassified as scene cuts. From these results we see that the performance of the classifier can be considerably improved by using additional features other than the histogram intersection. The motion statistic features, $x_{i,7-12}$, are particularly useful for reducing false alarms, and a temporal window of features seems to substantially improve overall accuracy. An important advantage of the tree classifier is that it automatically uses information which best detects the scene cut, even when this information varies from frame-to-frame.

In Figure 4, we show an example of relevance feedback for similarity pyramid pruning and design. Figure 4a) shows 8 images that have been selected by a user. Although these images have dissimilar attributes, they all form a single semantic category, "head shot of speaker". Figure 4b) shows that clustering of these selected images results in three groupings to represent the members of the semantic class. Figure 4c) shows the 64 shots most similar to the centroid of one of the three clusters after feature weight optimization. The images are listed in order of similarity. Notice that the pruned set contains images which closely match the user selected set.

References

[1] Ishwar K. Sethi and Nilesh V. Patel, "A statistical approach to scene change detection," in *Proc. of IS&T/SPIE Conf. on Storage and Retrieval for Image and Video Databases III*, San Jose, CA, February 1995, vol. 2420.

[2] Boon-Lock Yeo and Bede Liu, "Rapid scene analysis on compressed video," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 5, no. 6, pp. 533–544, December 1995.

[3] Minerva M. Yeung and Boon-Lock Yeo, "Video visualization for compact presentation and fast browsing of pictorial content," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 7, no. 5, pp. 771–785, October 1997.

[4] Ke Shen and Edward J. Delp, "A fast algorithm for video parsing using mpeg compressed sequences," in

Case	Description	Feature Subset	Correct	Missed	FA	Misclass.
1	Hist. Intersec.	$x_{i,1-3}$	140	85	62	14
2	case 1 + variance	$x_{i,1-6}$	152	73	50	12
3	case 2 + motion blocks	$x_{i,1-12}$	154	71	19	6
4	case 3 + time window	$\vec{x}_{i-1}$, $\vec{x}_i$, $\vec{x}_{i+1}$	208	17	11	0

Table 1: Results for cut detection using tree classifier. The table lists correct detections, missed detections, false alarms, and misclassifications of fades and dissolves as cuts.

(a) relevance feedback (b) clustering result (c) pruned results

Figure 4: (a) A set of images with user selected images outlined with a box. These images form a semantic category of "head shot of speaker". (b) The result for clustering the relevant shots. Three groupings are formed to represent the category as described in Section 5. (c) The set of images which are most similar to the new category listed in order of similarity.

Proc. of IEEE Int'l Conf. on Image Proc., Washington, D.C., October 26-29 1995, pp. 252–255.

[5] Cuneyt Taskiran and Edward J. Delp, "Video scene change detection using the generalized trace," in *submitted to ICASSP'98*.

[6] L. Breiman, J. H. Friendman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, Belmont, CA, 1984.

[7] Saul Gelfand, C. Ravishankar, and Edward Delp, "An iterative growing and pruning algorithm for classification tree design," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 163–174, February 1991.

[8] Anil K. Jain and Richard C. Dubes, Eds., *Algorithms for Clustering Data*, Prentice Hall, New Jersey, 1988.

[9] Jau-Yuen Chen, Charles A. Bouman, and Jan P. Allebach, "Multiscale branch and bound image database search," in *Proc. of SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases V*, San Jose, CA, February 13-14 1997, vol. 3022, pp. 133–144.

[10] HongJiang Zhang, Shuang Y. Tan, Stephen W. Smoliar, and Gong Yihong, "Automatic parsing of news video," *Multimedia Systems*, vol. 2, pp. 256–266, 1995.

[11] Nuno Vasconcelos and Andrew Lippman, "Towards semantically meaningful feature spaces for the characterization of video content," in *Proc. of IEEE Int'l Conf. on Image Proc.*, Santa Barbara, CA, October 1997.

[12] Stephan Fischer, "Automatic violance detection in digital movies," in *Proc. of IS&T/SPIE Conf. on Multimedia Storage and Archiving Systems II*, San Jose, CA, February 1996, vol. 2916, pp. 212–223.

[13] Jau-Yuen Chen, Charles A. Bouman, and John Dalton, "Similarity pyramids for browsing and organization of large image databases," in *Proc. of SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases III*, San Jose, CA, January 26-29 1998, vol. 3299.

[14] Ingemar J. Cox, Matt L. Miller, Stephen M. Omohundro, and Peter N. Yianilos, "Pichunter: Bayesian relevance feedback for image retrieval," in *Proceedings of International Conference on Pattern Recognition*, Vienna, Austria, August 1996, vol. 3, pp. 361–369.

[15] Yong Rui, Thomas S. Huang, and Sharad Mehrotra, "Relevance feedback techniques in interactive content-based image retrieval," in *Proc. of SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases VI*, San Jose, CA, January 26-29 1998, vol. 3312, pp. 25–36.