

***ViBE*: A New Paradigm for Video Database Browsing and Search ^{*†}**

Jau-Yuen Chen, Cüneyt Taşkıran, Edward J. Delp and Charles A. Bouman
Electronic Imaging Systems Laboratory (*EISL*)
Video and Image Processing Laboratory (*VIPER*)
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907-1285

Abstract

Video Browsing Environment (ViBE) is a unique browseable/searchable paradigm for organizing video databases containing a large number of sequences. The system first segments video sequences into shots by using the Generalized Trace obtained from the DC-sequence of the compressed data stream. Each video shot is then represented by a hierarchical tree structure of key frames, and the shots are automatically classified into predetermined pseudo-semantic classes. Finally, the results are presented to the user in an active browsing environment using a similarity pyramid data structure. The similarity pyramid allows the user to view the video database at various levels of detail. The user can also define semantic classes and reorganize the browsing environment based on relevance feedback.

1 Introduction

Clearly applications for digital video are undergoing explosive growth. While these applications will generate and use vast amounts of video data, the technologies for organizing and searching this video data are in their infancy.

In recent years, tremendous attention has been given to developing content-based methods for managing video and image databases. Most of the initial work in content-based video access has concentrated on developing methods for temporal segmentation of video sequences, i.e., detecting scene changes [1, 8, 14]. Recently, however, the focus has shifted to key frame selection, shot clustering, and design of complete systems for extracting information from video sequences and presenting it compactly to the user [15]. Most

of the work done has concentrated 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 trend is to do the processing in the compressed domain. Working directly in the DCT domain has a number of advantages: First, by removing the decoding/re-encoding step and working with a much lower data rate signal, computational complexity is greatly reduced. Second, the compressed data stream contains many pre-computed features, such as motion vectors and block averages, that can be very useful in the analysis of the video stream.

The design of video databases and their user interfaces remains very much an open problem. Perhaps the most natural approach is to extract key frames and use the search-by-query techniques often applied in image database applications [18]. Others have proposed more linear database organizations which are well adapted to scanning of sequences [10, 5]. Approaches which emphasize hierarchical organization have been used on individual sequences and the automatic generation of storyboards [15].

In this paper, we describe a unique new paradigm for video database management known as *ViBE* (Video Browsing Environment). Unlike many previous approaches, *ViBE* emphasizes the scalable management of large numbers of diverse video sequences. *ViBE* provides a structure for browsing, searching and content labeling while keeping the user in the loop. Relevance feedback can be provided by the user through the dynamic reorganization of the database and the use of user-defined interest classes. *ViBE* is divided into four components: scene change detection and identification, hierarchical shot representation, pseudo-semantic shot labeling, and active browsing based on relevance feed-

* This work was supported by the AT&T Foundation, the Rockwell Foundation, and Apple Computer, Inc.

back.

We have implemented the system using compressed-domain processing techniques on MPEG video sequences. We feel that *ViBE* provides an extensible framework that will scale as video databases grow in size, and application needs increase in complexity.

2 Scene Change Detection and Identification

For our purposes, a *shot* is defined as a collection of frames $\{f_{b_i}, \dots, f_{e_i}\}$ which are similar, where b_i and e_i are the beginning and end frame numbers of the i 'th shot, respectively. Our features are extracted from DC-image sequence. Each DC-image is formed by the DC coefficients of the DCT for each frame. The DC-image is directly available for the intracoded I-frames. The DC-image for B and P-frames is estimated using the motion vectors as described in [11].

Given the DC-sequence, we extract a 12-dimensional feature vector $\vec{x}_i = [x_{i1}, \dots, x_{i12}]^T$ for the i 'th DC-image and call this vector the *Generalized Trace* (GT) [12]. The first three features are the histogram intersection values for Y, U, and V components, respectively, while the next three are the standard deviation of the DC-image. Features x_{i7} to x_{i9} contain the number of intracoded blocks, the number of forward motion vectors, and the number of backward motion vectors respectively. Note that not all of these motion features are applicable to all types of frames, e.g., feature x_{i8} and x_{i9} will be zero for I frames, and x_{i9} will be zero for P frames. This is the reason features x_{i10} to x_{i12} are binary flags indicating the type of frame. Since some features have step discontinuities at cut locations, we also use the time difference of the GT to obtain impulse discontinuities which make classification easier. To this effect, we define the components of the *differential Generalized Trace* as

$$dx_{ij} = |x_{ij} - x_{i-1j}| \quad (1)$$

The L_2 norm of the $d\vec{x}_{i,1-3}$ for the first 2000 frames of a video sequence is shown in Figure 1.

The first step in analyzing scene changes is locating the cuts. In the past, this detection step has been implemented using simple thresholding [16, 8]. Recently, however, researchers have tried to use more robust methods to detect scene changes which do not rely on thresholding, such as the K -means clustering algorithm of [7]. To avoid thresholding and make the classification more robust, we use a classification/regression tree to classify each frame into two classes: scene cut or no scene cut. The tree is first built with one training set, and then pruned with a second as described in [6].

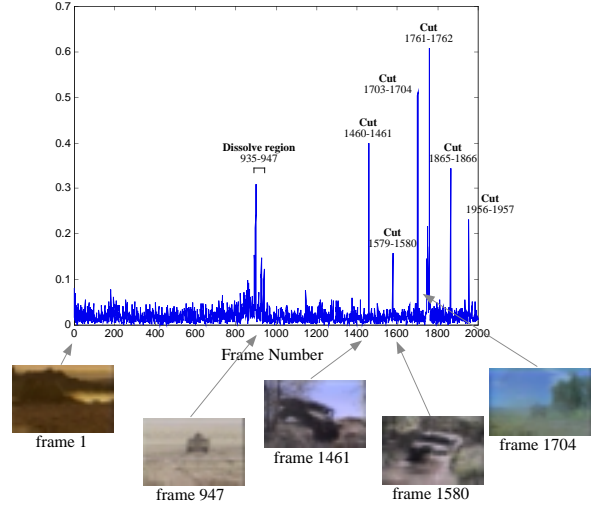


Figure 1. The L_2 norm of GT for an action sequence shown with the first frames from some shots.

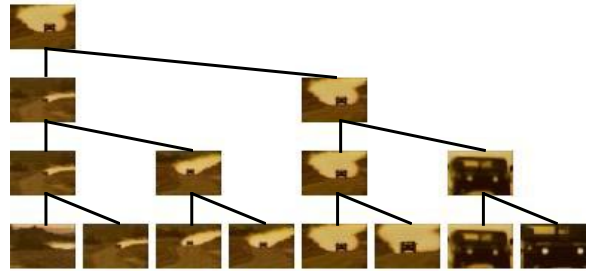


Figure 2. The representation for a shot from an action sequence. The top four levels of the binary tree are shown.

3 Shot Representation using Tree Structures

Typically, a shot may be represented by one or more sequentially selected keyframes [14]. 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. This 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 frames. We use the global color, texture and edge histograms proposed in [2] as the feature vector for each frame in a shot, and we use the L_1 norm to describe feature distances. Figure 2 illustrates such a rep-

resentation for a shot from an action sequence. This shot contains significant changes which can not be captured by a single keyframe. This example clearly shows the superiority of the shot tree representation.

The tree structure hierarchically organizes frames into similar groups, therefore allowing automatic detection of subgroups inside a shot. Each node of the tree is associated with a set of frames and contains a feature vector, z_s , which represents the cluster of frames. Generally, z_s is defined as the centroid of the image features in the cluster. We assume that the dissimilarity between two tree nodes i and j with feature vectors z_i and z_j can be measured using a symmetric function $d(z_i, z_j)$. The dissimilarity between two trees is then equal to the weighted sum of distances between corresponding nodes in the trees; so the tree distances may be computed in order N time where N is the number of nodes in the tree.

4 Pseudo-Semantic Classification

Most approaches in content-based retrieval rely on descriptions which are either problem-specific such as anchor shot models in news video [17] or low-level features 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. Ideally, one would like to query a video database using very high-level semantic descriptions such as “Michael Jordan doing a reverse slam dunk when pushing off on his left foot”. Unfortunately, automatic labeling of such categories is currently impossible. In addition, organization of video databases based on highly specialized semantic categories is probably not even desirable for browsing applications.

In order to overcome these difficulties, we adopt the use of pseudo-semantic classes which are broadly defined semantic categories that can be accurately derived from mid- and low-level features. Here the objective is to obtain high confidence as to semantic content, but not to obtain a very detailed semantic analysis. For example, Vasconcelos and Lippman [13] were able to separate pseudo-semantic classes such as “action”, “romance”, and “comedy” sequences based on shot activity and duration.

We use the features described in Section 3 together with the motion vectors and shot duration and type to extract pseudo-semantic classes corresponding to head and shoulders, indoor versus outdoor, high action, and man made versus natural. For each shot, we define a vector indicating the confidence level for each pseudo-semantic class. Each element of this vector has a continuous value in the range [0,1].

5 Active Browsing with Similarity Pyramids

In this section, we make use of the similarity pyramid we proposed in [3] 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 quad-tree 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 [4, 9]. However, our use of relevance feedback is distinct in two ways: First, we use the feedback to reorganize and prune the browsing environment, rather than 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 .

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 $z_i, i = 1 \dots N$ 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,

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

where $d_\theta(.,.)$ is the dissimilarity measure with a set of



Figure 3. An example of a similarity pyramid and its embedded quad-tree.

weightings, θ . Let $r_\theta(x_j)$ be the ranking of x_j with respect to $d_\theta(\cdot)$ 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, $x_i, i = 1, \dots, M$, which minimize the distance function $d_\theta(\cdot)$. These M shots are then organized into a similarity pyramid according to algorithm defined in [3] using the dissimilarity $d_\theta(\cdot, \cdot)$.

A user may also define a new pseudo-semantic class by selecting a set of images which are representative of that class and, in some cases, ones that are not. Based on these examples, a class model is constructed which best discriminates between the desired and undesired shots. The model is then used to label shots in the database as to their pseudo-semantic class.

6 Experimental Results

We have used a database consisting of six clips of approximately 10 minutes to run the classification experiments. These clips are from CNN Headline News, C-SPAN, a local newscast, and a promotional video. We use two sequences to train the tree classifier and the remaining four clips to measure classification performance.

Fig. 4 shows the results of applying the tree classifier to scene cut detection. From these results we see that the performance of the classifier can be considerably improved by using extra features other than the histogram intersection features, which generally are the only ones used in the literature. The motion features $x_{i,7-12}$ are particularly important for improving cut detection, while features at neighboring times x_{i+1} and x_{i-1} used in cases 4 and 5 reduce false alarms. 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 four 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] F. Arman, A. Hsu, and M.-Y. Chiu. Feature management for large video databases. In *Proc. of SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases*, volume 1908, pages 2–12, San Jose, CA, February 1993.
- [2] J.-Y. Chen, C. A. Bouman, and J. P. Allebach. Stochastic models for fast multiscale image search. In *Proc. of SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases V*, volume 3022, pages 133–144, San Jose, CA, February 13-14 1997.
- [3] J.-Y. Chen, C. A. Bouman, and J. Dalton. Similarity pyramids for browsing and organization of large image databases. In *Proceedings of SPIE/IS&T Conference on Human Vision and Electronic Imaging III*, volume 3299, San Jose, CA, January 26-29 1998.
- [4] I. J. Cox, M. L. Miller, S. M. Omohundro, and P. N. Yianilos. Pichunter: Bayesian relevance feedback for image retrieval. In *Proceedings of International Conference on Pattern Recognition*, volume 3, pages 361–369, Vienna, Austria, August 1996.
- [5] P. England, R. Allen, M. Sullivan, A. Heybey, M. Bianchi, and A. Dailanas. I/Browse: The bellcore video library toolkit. In *Proc. of SPIE/IS&T Conf. on Storage and Retrieval for Image and Video Databases IV*, volume 2670, pages 254–264, San Jose, CA, February 1996.
- [6] S. Gelfand, C. Ravishankar, and E. Delp. An iterative growing and pruning algorithm for classification tree design. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(2):163–174, February 1991.
- [7] B. Gunsel, Y. Fu, and A. M. Tekalp. Hierarchical temporal video segmentation and content characterization. In *Proceedings of SPIE Conference on Multimedia Storage and*

Case	Description	Feature Subset	Correct	Missed	FA	Misclass.
1	Hist. Intersec.	$x_{i,1-3}$	90	74	17	10
2	case 1 + motion	$x_{i,1-3}, x_{i,7-12}$	133	31	32	4
3	all features	\vec{x}_i	108	56	9	5
4	all features + time window	$\vec{x}_i, x_{i\pm 1,1-3}$	135	29	3	0
5	all features + time window + dx	$\vec{x}_i, x_{i\pm 1,1-3}, dx_{i,4}, dx_{i+1,4}$	143	21	12	1

Table 1. Results for cut detection using tree classifier listing correct detections, missed detections, false alarms, and missclassifications.

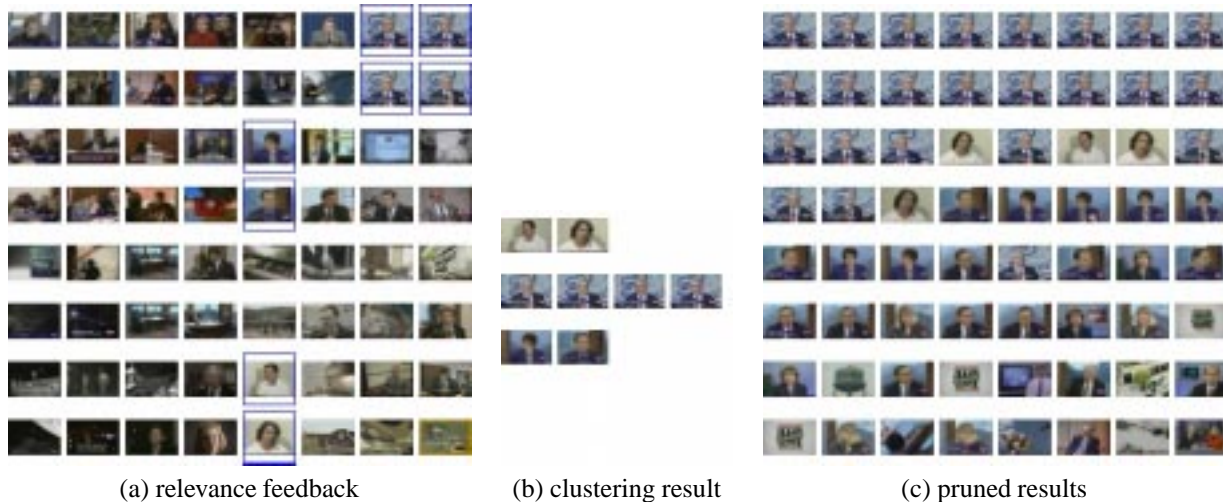


Figure 4. (a) A set of images with user selected images outlined with a box. These images form a semantic category of "head shot of speaker". (b) The result for clustering the relevant shots. Three groupings are formed to represent the category as described in Section 5. (c) The set of images which are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

- Archiving Systems II*, volume 3329, pages 46–56, November 1997.
- [8] N. V. Patel and I. K. Sethi. Video shot detection and characterization for video databases. *Pattern Recognition*, 30(4):583–592, April 1997.
- [9] Y. Rui, T. S. Huang, and S. 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*, volume 3312, San Jose, CA, January 26–29 1998.
- [10] B. Shahraray and D. C. Gibbon. Automatic generation of pictorial transcripts of video programs. In *Proceedings of SPIE Conference on Multimedia Computing and Networking*, volume 2417, pages 338–341, San Jose, CA, February 1995.
- [11] K. Shen and E. J. Delp. A fast algorithm for video parsing using MPEG compressed sequences. In *Proc. of IEEE Int'l Conf. on Image Proc.*, pages 252–255, Washington, D.C., October 26–29 1995.
- [12] C. Taskiran and E. J. Delp. Video scene change detection using the generalized trace. In *Proc. of IEEE Int'l Conf. on Acoust., Speech and Sig. Proc.*, Seattle, WA, 1998.
- [13] N. Vasconcelos and A. 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.
- [14] B.-L. Yeo and B. Liu. Rapid scene analysis on compressed video. *IEEE Trans. on Circuits and Systems for Video Technology*, 5(6):533–544, December 1995.
- [15] M. M. Yeung and B.-L. Yeo. Video visualization for compact presentation and fast browsing of pictorial content. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(5):771–785, October 1997.
- [16] H. Zhang, C. Y. Low, and S. W. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, 1(1):89–111, March 1995.
- [17] H. Zhang, S. Y. Tan, S. W. Smoliar, and G. Yihong. Automatic parsing of news video. *Multimedia Systems*, 2:256–266, 1995.
- [18] H. J. Zhang, J. Wu, D. Zhong, and S. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern Recognition*, 30(4):643–658, April 1997.