Clustering Methods for Video Browsing and Annotation

Di Zhong[1], HongJiang Zhang[2] and Shih-Fu Chang*
Institute of System Science, National University of Singapore
Kent Ridge, Singapore 0511
*Center for Telecommunication Research, Columbia University
New York, NY10027

1. INTRODUCTION

The large amount of video data makes it a tedious and hard job to browse and annotate them by just fast forward and rewind. Recent works in video parsing provide a foundation for building interactive and content based video browsing systems[1,2]. In the parsing process, video data are segmented into shots, and each shot may be represented visually by one or more key frames extracted from it. These key frames, together with temporal information of the video shots, can be used for video browsing purpose. If these key frames are extracted automatically in a content based manner[3], not just simple subsampling, key frame browsing could obtain an overview of video data without losing any important information. However, without proper organization, key frames can only be viewed in a sequential looking up manner.

How to provide efficient video browsing facilities has been addressed by many researchers. Generally, a knowledge model is needed to guide the organization of video according to semantic or syntactic contents[4]. Automatic parsing of news video programs based on syntactic model is a good example of such methods[5], where domain model is used to locate and classify video shots into different categories and build up their relations. However, such methods can only be applied to some limited applications, since in general, such domain models may not be available and automatic extraction of high level information from video data is beyond current video parsing and computer vision technologies. On the other hand, similarities among video shots can be represented by visual features, such as color, texture and temporal variation[6,7]. These features provide a variety of evidences for us to determine video shots containing similar objects, occurring in similar environments or changing in similar ways. Therefore, how to use these low level features in organizing video data for video browsing has attracted many research efforts.

The video browser developed at Siemens[8], uses Rframes (representative frames, obtained from subsampling) as the basic browsing unit to organize the visual contents of video clips. Each Rframe contains information about image features (shape and color), shot length and motion visualization. A hierarchical scene transition graph is proposed by the Princeton group[9], which tries to extract scenic structure (story structure) of videos using visual and temporal information with no prior knowledge of the content. The Berkeley Distributed VOD System[10] built a mixed-mode query interface including some basic search operations, that allows a user to select a set of videos or images and incrementally modify the answer set. The key frame based sequential and hierarchical broswers developed by the ISS group made use of video parsing result results. However, in spite of many research effort, how to organize feature data and thus visual items to facilitate video browsing still remains to be one of the most important and open issue. Also, more suitable visual features and similarity measures, especially in representing temporal and motion features of video data, need to researched.

In this paper, a generalized top-down hierarchical clustering process, which adopts partition clustering recursively at each level of the hierarchy, is studied and used to build hierarchical views of video shots. With the clustering processes, when a list of video programs or clips is provided, a browsing system can use either key-frame and/or shot features to cluster shots into classes, each of which consists of shots of similar content. After such

---

1. Current address: Center for Telecommunication Research, Columbia University, New York, NY10027.
2. Current address: HP Labs, MS 1U-17, 1501 Page Mill Road. Palo Alto, CA94304.

clustering, each class of shots can be represented by an icon, which can then be displayed at the high levels of a hierarchical browser. As a result, users can know roughly the content of video shots even without moving down to lower level of the hierarchy.

In the remaining part of this paper, we will first discuss the representation of video streams, with efforts on the definition and extraction of suitable visual features, including color histogram, temporal variance and statistical motion features. In Section 3, we will discuss the hierarchical viewing of video streams and the generalized clustering method in detail. Some experimental results are given in Section 4. Finally, conclusions are summarized in Section 5.

2. REPRESENTATION OF VIDEO CONTENT

Representation of video content is different from still images in many aspects[7]. To obtain the features we need for content-based browsing purpose, several processing steps are performed (see **Figure 1**). The first is *temporal segmentation*, which aims to detect scene changes. In *abstraction* process, one or more key frames are obtained automatically based on content variation of each shot. *Feature extraction* includes two parts: Temporal and motion features are extracted from video shots. and still image features, such as color, texture and shape, are extracted from key frames. Finally, these feature data are organized automatically using *clustering* methods, and a video stream can be represented hierarchically by its key frames. In this section, some of visual features used in video representation will be discussed. The clustering processes will be discussed in the next section.

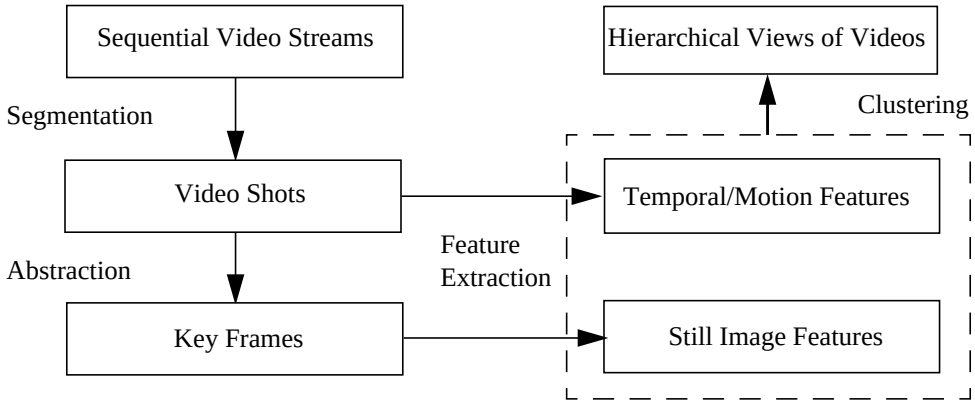

Figure 1: Parsing, representation and organization of video content

2.1 Color Features

Color is an important attribute of visual information. It is very rare that two images of totally different objects will have similar colors. Color histogram is an important color feature, commonly used to represent color information of image content similarity. It provides a robust, efficient cue for identifying multicolored objects[11].

Given a discrete color space defined by some color axes(e.g. red, green and blue), the color histogram is obtained by quantizing the image colors and counting the number of times each discrete color occurs in the image array. One direct quantization approach of color space is to evenly divide it into *N* boxes. Each box has the color of its center as its representative color. Then the *N*-bin color histogram is calculated in each of the *N* boxes. This approach is suitable for RGB color space and its linear transformations. But, for color spaces derived from RGB space by non-linear transformations, such as CIE-L*u*v* and Munsell, even division of color spaces does not make much sense since color triples converted from RGB space are not evenly distributed in these spaces. However, these non-linear color spaces are closer to human perception. In other words, the distance between two color triples in these spaces is a better approximation to the difference perceived by human. Therefore, it is more adequate to calculate color histograms in these spaces for the purpose of image content comparing. To calculate color histograms in these non-linear color space, a general color quantization process, described as below, is performed (assuming images are originally represented in RGB space)[6]:

1) RGB space is evenly divided (e.g. 16x16x16) and the color triples of the center of each boxes (e.g. 4096 color triples) are obtained;
2) Transform these color triples into the target color space (e.g. *L*u*v**);
3) Cluster converted color triples in the target color space to obtain *N* representative colors (i.e. to obtain *N* classes), where N is the number of representative bins of the color histogram;
4) Compute a lookup table that maps all the RGB color triples (e.g. 4096) into the N representative color bins in the target color space.

After building the color lookup table, color histograms can be counted in the target space by looking up the representative colors through original RGB triples of each pixel.

A possible variant of the above clustering method is to obtain RGB triples (at first step above) not from divisions of RGB space but from given images (frames) by a sampling process. For example, one can randomly choose 100 pixels (i.e. RGB triples) from each image, convert these triples to color vectors in target color spaces and cluster them. This approach uses the actual color space occupied by the images and can assign more bins to the major colors, thus color histograms derived are more suitable for the description and distinguish of given images. Obvious improvement of the clustering result is seen in our tests using this method. The method is especially effective while major colors in given images are limited or in the case of multi-feature clustering, where colors of images of low-level classes are already reduced to a certain range/number.

2.2 Temporal Variance

The motion of objects or cameras in a video shot is very important for the clustering purpose. In a news video, anchor person and interview shots always have small temporal variance, while some other shots like sports and commercial usually have large motion and variance. This indicates that temporal information can be very useful in grouping video shots with similar contents. Two types of temporal features will be discussed below and in Section 2.3 respectively.

As we discussed in Section 2.1, color histogram is a good representation of objects and backgrounds occurred in an image. Naturally, the variance of color histogram over all the frames in a shot can be used as a metric for the temporal variance of the shot.

Suppose that a video shot contains N frames, H_i *(i = 1, 2, ..., N)* are the L-bin color histogram of these frames and let $\bar{H}$ be the average color histogram, i.e., $\bar{H} = (1/N)\sum_i^N H_i$, we have the average difference and its variance defined as follows:

$$\mu = (1/N)\sum_{i=1}^{N} d_i$$

$$\sigma^2 = (1/N)\sum_{i=1}^{N} (d_i - \mu)^2$$

where

$$d_i = (1/L)\sum_{j=1}^{L}\left(H_i(j) - \bar{H}(j) \right).$$

μ and σ are then used as numerical indications of the temporal variance of a given shot.

2.3 Statistical Motion features

Although motion is the most prominent property of video data, it is now seldom used as a type of visual features in representing video content. This may because that the recognition of moving objects from videos are

still very restricted, not reliable and requiring extensive preprocessing. Currently, computer vision researchers have developed theories and method for extracting motion information independently from recognizing objects in human and computer vision systems[12]. This indicates that similarly as representing still images with low level features, we can define low level motion features as keys to describe activities inside video data.

In our approach, instead of using trajectories of motion objects, we use certain types of statistical features derived from optical flows to describe motions in video shots. Compared with motion trajectories, these statistical features are more robust and easy to be extracted from video shots. In addition, it is also easy to compare these features.

To compute optical flows, we use image sequence around each key frame of a shot since we want to assign motion features to related key frames. Since variations and motions around a key frame are usually small and continuous, it is reasonable to assume that the computed optical flow can reflect motions around a key frame properly. Based on the motion information of optical flow, including directions, speeds, sizes and positions, many statistical variables can be estimated. One can calculate statistical variables by speeds or directions alone in an entire scene or calculate them in different areas. For certain applications, we can also estimate motion directions at certain speed ranges, motion sizes at certain directions and so on. For our purpose of clustering video shot, two statistical features are used and discussed in bellow.

(1) Directional distributions

Many activities exhibit clear dominant motion directions. To express such activities, dominant directions provide valuable evidence. An M-dimension feature vector of motion directions is defined as follows.

$$d_i = \frac{N_i}{N_{mt}}, \quad i = 1, \ldots, M$$

where i represents one of the M equally divided directions, N_i is the number of moving points in direction i and N_{mt} is the number of total moving points at all directions. Since N_{mt} is used as the dominator, we consider not only the dominant directions but also the sizes of moving components in different directions. It can be replaced by the number of total points in the optical flow so as to consider how large the moving area is in the scene.

(2) Speed distributions

Similar to calculating the number of moving points in different directions, we can estimate speed distributions at different directions as below:

$$\bar{s}_i = \frac{1}{N_i}\sum_{j=1}^{N_i} s_{ij}, \qquad \sigma^2 = \frac{1}{N_i - 1}\sum_{j=1}^{N_i}\left(s_{ij} - \bar{s}_i\right)^2, \qquad i = 1, \ldots, M$$

where N_i is the number of moving points in direction i, s_{ij} is the speed of j'th moving point in direction i. As one can see, $\bar{s}_i$ is the average speed and σ_i is the standard deviation in direction i. The feature vector consists of both $\bar{s}_i$ and σ_i and they provide a generalized description of speed distributions.

3. HIERARCHICAL BROWSING OF VIDEO STREAM

As we mentioned before, without a proper organization, key frames or video shots can only be browsed in a sequential manner, even in the hierarchical browser[3]. On the other hand, if we pre-cluster video shots or key frames, then, a class of video shots can be represented visually by an icon of the class at each level of the hierarchical browser, with the key frames belonging to the class of shots listed at the lower level. As a result, users can get a sense of the video content roughly even without moving down to lower level of the hierarchy. To build such a class-based hierarchical video browser, a hierarchical abstraction process is needed to group shots of similar contents and to get representative icons of each class. In this section, we presented a generalized hierarchical clustering method used in the visual icon abstraction process.

3.1 A Generalized Hierarchical Clustering Method

Clustering is sometimes called "unsupervised learning" because it classifies objects into subsets only by their actual observations. No presumed classes are needed in the classification process, which is different from pattern recognition methods. There are mainly two types of clustering: partition clustering arranges data in separate clusters; and hierarchical clustering leads to a hierarchical classification tree.

These two types of methods have different characteristics. Hierarchical methods give a more detail description about the relation among data items at different levels. They are more efficient with small data sets and less prone to various types of noise. In the case of large data set, partition methods are more suitable since instead of aiming at optimal partition at each level as hierarchical methods, they find optimal clustering directly at a certain level[13]. Therefore, partition clustering methods are more suitable to obtain an abstraction of given data items. Since the abstraction will be done hierarchically in our application, either top-down or bottom-up, the entire abstraction process is actually a generalized hierarchical clustering which adapting partition at each level. This generalized clustering method is very flexible, different feature data and measure metrics, as well as clustering methods, can be applied at different level.

There are many kinds of partition clustering algorithms, among which, the mostly common used one is the iterative algorithm. The basic idea of an iterative clustering algorithm is to start with an initial partition and assign patterns to clusters so as to reduce certain optimum functions. *K*-means and ISODATA are two typical iterative clustering algorithms. *K*-Means algorithm aims at classifying data items into a fixed number of classes starting from an initial partition. ISODATA algorithm is more general than *K*-means when the desired clustering number is not certain or the initial partition is poor. In ISODATA algorithms, the number of clusters can be adjusted by merging and splitting existing clusters or by removing small clusters.

To realize better and more robust clustering while the amount of data is large, we have also developed an cluster algorithm based on the unsupervised learning neural network[14], Self-Organization Map(SOM), which has been used widely in different areas. Its learning ability without prior knowledge and good classification performance have been extensively proved by many researches. Another benefit of using SOM is that the similarities among the extracted classes can be seen directly from the two dimensional map. This will allow horizontal exploring as well as vertical browsing of the video data, which is very useful while we have a large amount of classes at lower levels.

3.2 Fuzzy Partition Classification

The above mentioned partition clustering methods can be made more practical by using fuzzy classification. In the fuzzy classification algorithm, the membership functions of a data item to all the classes is calculated and thus it can assign data items at the boundary of two classes to both classes[15]. This is useful especially at high levels of a hierarchical browser, where users expect all the similar data item are under one same node so they need not to browse large amount of items in other nodes. At the same time, outliers, i.e. data items whose membership functions to all classes are very small, can be detected and put into a miscellaneous class.

As an example, we give the calculation of membership functions in the common *K*-Means algorithm as follows:

(1) Get N classes using the *K*-Means algorithm.

(2) For every data items v_i, $i=1,..., M$

- calculate its similarity with each class c ($c=1,..., N$) as

$$s_{ic} = \frac{d_{ic}^{-2/(\phi-1)}}{\sum_{j=1}^{N} d_{ij}^{-2/(\phi-1)}}$$

where d_{ij} is the distance between data item v_i and the reference vector of class j. ϕ is the fuzzy exponent ($\phi>1.0$).

- if $s_{ij} \geq \rho$, where ρ is a threshold set by users ($0 < \rho < 1$); add item v_i into class c.

- if v_i is not assigned to any class in above step, assign it to the miscellaneous class.

4. Experimental Results

4.1 Clustering of Similar Scenes

In this experiment, a 10 minutes documentary video about Singapore was used. The sequence contains 71 shots and is represented by 71 key frames after video parsing. Our aim was to build a hierarchical view with five (sub)classes at each level of the hierarchy. Clustering of these 71 key frames were based on their color histograms, 64-bin color histogram in L*u*v* space as we discussed in Section 2.1.

The 5 classes at the top level were formed based color histograms by using the fuzzy *K*-Means clustering algorithm in Section 3.2, where the desired clustering number *N* was set to 4, leaving one class to miscellaneous images. The fuzziness exponent ϕ was set to 2.0 and the threshold ρ was set to 0.3. Successive clustering of lower level views were done by the conventional *K*-Means clustering method with *N=5*. This method generated a very good hierarchical view of the video sequence as shown in **Figure 2** and **Figure 3**.

a. Hierarchy: level 1

b. Hierarchy: level 2 (from the first class at level 1)

c. Hierarchy: level 3 (from the first class at level 2)

d. Hierarchy: level 4 (from the third class at level 3)

Figure 2. Class-based hierarchical view of a video sequence

In **Figure 2**, the top level contains five classes, each represented by an icon, which is the frame closest to the centroid of the class. An overview of the video content can obtained by looking at the top level. For the first

class, five subclasses are listed in the second level that are similar in color (i.e. green and white). Again, under the first iconic image at the level two, five shots of green trees and houses are shown at the level three. Under the second class at this level, there are still two more key frames, which are further explored at the bottom level.

Figure 3. Images in the miscellaneous class

From the miscellaneous class (the last one in the top level in **Figure 2**), we see three images that are very different from others (**Figure 3**). Two of them are maps with large area of uniform blue. The other one is a light view of a building with large area of many uniform white. It is reasonable that these shots were not classified to any other classes by color.

4.2 News Video Parsing

In news video parsing system developed by Zhang, *et al*, detection of some special shots (e.g. anchorperson and transient shots) are very important in order to build the story structure of a program[5]. *A priori* models of video structure are adopted for identify these shots for a given news program. We apply the clustering method presented above to improve the parsing and annotation process of news videos. As one can see below, this method can be easily applied to different news programs since no special *a priori* models are needed.

In this experiment, our objective was to annotate all the anchorperson shots so as to build the story structure of a news video stream. A CNN news program of ten minutes was used, which were segmented into 80 shots by scene cut detection. Visual features, including temporal variance, statistical motion features and color histograms as discussed previously, were derived for each shot respectively and used in a hierarchical clustering process, and a hierarchical view of the news video as follows was obtained.

a. Level 1: Clustering of all the video shots by temporal variances

b. Level 2: Clustering of shots under the second class above by color histograms (Lu*v* space)

c. Anchorperson shots under the third class at level 2

Figure 4. Clustering based annotation of anchorperson shots

As shown in **Figure 4**, in the first level, temporal variances were utilized in the clustering process (using K-Means algorithm). The 80 shots were classified into 6 classes. Since all anchorperson shots have small temporal variance of same order, they are classified into the same class, i.e. the second one at the first level, which contains 30 shots. At the second level, these 30 shots are further classified according to their color histograms. Again, six classes were generated. Among them, the second, third and sixth classes are consisted of three different kinds of anchorperson shots respectively. The five anchorperson shots in the third class at the second level are shown as the bottom level in **Figure 4**. With such a hierarchical viewer of news video, one can annotate all the desired anchorperson shots easily. Automatic annotation is also possible if some specify features (e.g. face) or special structure of anchor person shots are available.

Similar methods can be applied for the annotation of transient shots and sports shots (the first and the last class at the first level in **Figure 4**), which usually have large temporal variance. For sport shots, statistic speed features (see section 2.3) are more useful than temporal variance. This method can be further developed for the grouping, annotation and browsing of news programs from different days, e.g. in one week. Repetitively reported news will be clustered together, so we can browse news across different dates.

5. CONCLUSIONS

Organization of video data for efficient browsing and annotation is an important issue in content-based video access. In this paper, we presented our approaches to class-based video browsing and annotation, and some useful visual features for video content representation. It is shown that the generalized top-down hierarchical clustering method can be easily adopted to different demands. With the selection of suitable visual features, automatic or semi-automatic organization of video data can be realized for browsing and annotation.

6. REFERENCES

1. H. J. Zhang and S. W. Smoliar, Developing Power Tools for Video Indexing and Retrieval, Proc. IS&T/SPIE. on Storage and Retrieval for Image and Video Databases II, San Jose, CA, 1994, pp.140-149.
2. J. Meng, Y. Juan and S.-F. Chang, Scene Change Detection in a MPEG Compressed Video Sequence, *IS&T/ SPIE Conf. on Digital Video Compression: Algorithms and Technologies*, San Jose, Feb. 1995.
3. H. J. Zhang, S. W. Smoliar, and J. H. Wu, Content-Based Video Browsing Tools, *Proc. IS&T/SPIE Conf. on Multimedia Computing and Networking 1995*, San Jose, CA, 1995.
4. D. Swanberg, C.-F. Shu, and R. Jain, Knowledge Guided Parsing in Video Databases, *Proc. IS&T/SPIE Conf. on Storage and Retrieval for Image and Video Databases*, San Jose, CA, 1993.
5. H. J. Zhang *et al*., Automatic Parsing and Indexing of News Video, *Multimedia Systems*, 2 (6), pp. 256-266, 1995.3.
6. W. Niblack, et, al, The QBIC Project: Querying Images by Content Using Color, Texture and Shape", *Proc. IS&T/SPIE. on Storage and Retrieval for Image and Video Databases II*, San Jose, CA, February 1993.
7. H. J. Zhang, *et al*, "Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution", *Proc. of ACM Multimedia'95*, San Francisco, Nov.7-9, 1995, pp.15-24.
8. F.Arman, *et al*, "Content-Based Browsing of Video Sequences", *Proceedings of ACM international Conference on Multimedia '94*, Oct. 15-20, San Francisco, CA.
9. M. Yeung, *et a*l, Video Browsing using Clustering and Scene Transitions on Compressed Sequences, *Multimedia Computing and Networking*, San Jose, Feb. 1995.
10. L. A. Rowe, J. S. Boreczky, and C. A. Eads, Indexes for User Access to Large Video Databases, *Proc. IS&T/ SPIE Conf. on Storage and Retrieval for Image and Video Databases II*, San Jose, CA, 1994, pp. 150-161.
11. M. J. Swain and D. H. Ballard, Color Indexing, *International Journal of Computer Vision*, Vol 7, No 1, 1991, pp.11-32.
12. R. Polana and R.C. Nelson, Recognition of motion from temporal texture, Proc. IEEE Conference on Computer Vision and Pattern Recognition, Champaign, IL

13. A. K. Jain and R. C. Dubes, Algorithms for Clustering Data, Englewood Cliffs, Prentice Hall, NJ. 1988.

14. H. J. Zhang and D. Zhong, "A scheme for visual feature based image indexing", Proc. IS&T/SPIE Conf. on Storage and Retrieval for Image and Video Databases III, February 1995, San Jose, pp. 36-46.

15. J. J. De Gruijter and A.B. McBratney, A Modified Fuzzy K-Means Method for Predictive Classification, *Classification and Related Methods of Data Analysis: Proc. of the First Conference of the International Federation of Classification Societies (IFCS)*, North-Holland, 1988.