Determining Structure in Continuously Recorded Videos *

Yun Zhai
School of Computer Science
University of Central Florida
Orlando, Florida 32816
yzhai@cs.ucf.edu

Mubarak Shah
School of Computer Science
University of Central Florida
Orlando, Florida 32816
shah@cs.ucf.edu

ABSTRACT

In this paper, we present a scene detection framework on continuously recorded videos. Conventional temporal scene segmentation methods work for the videos composed of discrete shots, where shot boundaries are clearly defined. The proposed method detects scene segments by the spectral clustering technique and fuzzy analysis. The detected scenes are represented by the corresponding representative feature values of the feature clusters, rather than abrupt temporal boundaries. The feature clusters are generated using the spectral clustering technique. The video units have the fuzzy memberships to the feature clusters, which are generated using the Hyperbolic tangent fuzzy function. The scenes are collected from the candidate scenes from each cluster. The proposed method has been tested on several video sequences, and very promising results have been obtained.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing

General Terms

Algorithm, Performance.

Keywords

Scene Detection, Spectral Clustering, Fuzzy Representation.

1. INTRODUCTION

Temporal video scene segmentation/detection is one of the important and fundamental problems in the fields of video indexing, retrieval and analysis. It provides semantically meaningful segments of the video, so that further analysis of the video can be applied. For instance, in TV news programs, temporal video segments correspond to the coherent news topics. In feature films, video scenes provide chapters that correspond to the sub-themes of the story. In talk shows, scene segmentation separates the regular program from the commercials. Most of the above mentioned domains use videos that are composed of video shots, which are created by the camera operations in the production process, such as camera on/off and switching between cameras. Many previously developed scene segmentation methods usually consider the shots as the basic computational units and produce hard boundaries between scenes.

Many efforts have been devoted to this problem. Hanjalic *et al.* [1] proposed a scene segmentation method for the movies based on the block matching of the key-frames. The segmentation process is performed by linking the similar shots. Rasheed *et al.* [5] proposed a two-pass algorithm for the scene segmentation in feature films and TV shows. The video is first segmented into initial scenes using color-similarity, which are further refined based on the motion analysis. Sundaram *et al.* [6] used the audio-visual features in the segmentation of the movie scenes. Two audio and video scenes are first detected separately, and the correspondences between these two types of scenes are then established using a temporal nearest-neighbor algorithm. Yeung *et al.* [7] proposed a graph-based representation of the video by constructing a Shot Connectivity Graph. The graph is split into sub-graphs using the complete-link method of hierarchical clustering such that the they satisfy the color similarity constraint. In the domain of news videos, Hsu *et al.* [2] proposed an approach based on the discriminative models for the story segmentation. They have developed the BoostME utilizing the Maximum Entropy classifiers and the associated confidence scores in the boosting process. The above mentioned methods consider the video shots as the basic processing units. However, in some domains, it is not feasible to find the shot boundaries. For instance, for many home videos, the videos are often recorded in a continuous fashion. There is no abrupt scene boundary. In this type of scenarios, the scenes separately by the abrupt boundaries would contain the information of the transition periods, and un-expected output would be produced in the future video content analysis due to this defect.

In this paper, we propose a temporal scene segmentation method for the continuously recorded videos, where video shots are not well defined. Instead of finding the abrupt scene boundaries, video scenes are represented by their corresponding representative feature values such as color statistics, and each portion of the video is indicated by a fuzzy

*This material is based upon the work funded in part by the U.S. Government. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the U.S. Government.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
MM'05, November 6–12, 2005, Singapore.
Copyright 2005 ACM 1-59593-044-2/05/0011 ...$5.00.

number computed based on the membership functions with respect to the representative feature values. These representative feature values are obtained by applying spectral clustering technique. The scene segments are later determined by the desired criteria. Unlike shot-based method, the proposed method finds the scene boundaries not only based on the data (video shots), but also based on the user preference, Therefore, it provides more flexibility to the users. The rest of this paper is organized as follows: Section 2 describes the proposed framework in details. Section 3 presents the experimental results, and Section 4 concludes our work.

2. PROPOSED FRAMEWORK

In this section, we discuss the construction of the fuzzy representations of the video scenes and the procedure of obtaining them. Taking home videos as an example, each scene in the video is recorded in a specific environment setting. Thus, in our approach, we assume the relatively similar visual pattern in each of the scenes in the video. The video is first broken into smaller chunks, which can be treated as the "artificial shots" of the video. In our experiments, the size of each chunk is one second long. The visual features then are extracted from all the chunks. These features are clustered into k feature groups using the spectral clustering technique, and the mean vectors of the feature clusters are considered as the representation of the video scenes. The entire video is represented by a fuzzy set, whose memberships values are determined by the corresponding representative features.

2.1 Spectral Clustering

Based on the above described visual consistency assumption, we compute the 3-dimensional color histograms in LUV color space for every frame in the video. The chunks can be viewed as the sampled version of the video. Assume that in total there are n chunks $\mathbb{T} = \{t_1, t_2, ..., t_n\}$ in the video, we compute the average color histogram of chunk t_i to be the feature vector f_i of that chunk.

Given a set of feature points $F = f_1, ..., f_n$ in space $\mathbb{R}^m$, the spectral clustering algorithm used to partition F into k clusters is described as follows [4]:

1. Compute the similarity matrix $S \in \mathbb{R}^{n\times n}$, where $S(i,j) = exp(-\frac{d^2(f_i,f_j)}{2\sigma^2})$, if $i \neq j$, and $S(i,i) = 0$.
2. Define D to be the diagonal matrix of S, where $D_{i,i} = \sum_{j=1}^{n} S(i,j)$, and construct the Laplacian matrix $L = D^{-1/2}SD^{-1/2}$.
3. Find x_1, x_2, ..., x_k, the eigenvectors of L that correspond to the k largest eigenvalues, and form the new matrix $X = [x_1, x_2, ..., x_k] \in \mathbb{R}^{n\times k}$ by arranging the eigenvectors in the column fashion.
4. Normalize each row in X into unit length.
5. Treating each row of X as a new feature point in space $\mathbb{R}^k$, cluster them into k clusters using K-means algorithm. Let $\mathbb{C} = \{C_1, \cdots, C_k\}$ denotes the cluster set.
6. Assign the original feature points f_i to cluster C_j if and only if row i of X was assigned to C_j during the previous clustering procedure.

In this algorithm, $d(f_i, f_j)$ is the distance between feature points f_i and f_j, and σ^2 is the scaling parameter. We have incorporated the Bhattacharya distance between two feature vectors (average histograms) as the distance measure,

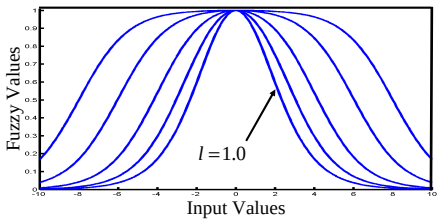

Figure 1: Plots of the Hyperbolic tangent functions with different shape parameter, l_c, varying from 1.0 to 8.0. The spread the function is $d_c = 2.5$, and the mean is 0.0.

$d(f_i, f_j) = -ln\big(\sum_{b=1}^{m}\sqrt{f_i^b f_j^b}\big)$, where f^b is the b-th element in the feature vector. Therefore, the similarity measure $S(i,j)$ is represented as, $S(i,j) = \big(\sum_{b=1}^{m}\sqrt{f_i^b f_j^b}\big)^{\frac{1}{2\sigma^2}}$.

One important issue here is how to select the number of clusters, k, in the clustering process. Several criteria could be used. We have chosen the *Sufficient Coverage*. Implied by its name, *Sufficient Coverage* is the method for selecting k, such that the overall distance between the feature points and their nearest cluster centers is under certain tolerance. The cluster center is the mean of each cluster calculated based on the assigned features in the original feature space $\mathbb{R}^m$. Let $D(k)$ be the goodness measure for clustering into k partitions,

$$D(k) = \sum_{i=1}^{n} min\Big(d(f_i, c_j)\Big), \ for\ j = 1, \ldots, k, \qquad (1)$$

where $d(f_i, c_j)$ represents the distance between feature point f_i and the mean of cluster c_j in the original feature space. The simplest way to compute $d(f_i, c_j)$ is to use the Euclidean distance. For the purpose of consistency, we also use the Bhattacharya distance here. Assume there are two or more clusters, and let $D(1) = 0$. The goodness is defined as,

$$D(i) \leq \mu_1, \quad or \quad D(i) - D(i-1) \leq \mu_2, \qquad (2)$$

where μ_1 and μ_2 are two positive constants. The first part of the goodness criteria captures how well the feature points are clustered. It is the overall distance between feature points to the cluster centers. The later part captures the rate of the overall distance change.

2.2 Fuzzy Representation

Unlike some video sequences which contain shots, continuously recorded videos do not contain any abrupt boundaries between scenes. In this scenario, it is very difficult and not meaningful to represent the video scenes in terms of their temporal intervals defined by the hard boundaries. For example, a tourist is taking videos around a park. Then, keeping the camera on, he walks to a nearby building and records another sequence. In this example, the scene is gradually transiting from one to another. Thus, it is very difficult to find an abrupt point to separate these two scenes. However, there exists a scale for the viewers to confidently distinguish one scene from another. This is called the fuzziness, which is the confidence measure of one value belonging to a certain pattern and is represented by the fuzzy numbers with range of [0, 1]. In our situation, each video chunk is assigned a sequence of numbers to indicate how confidently it belongs to the different feature clusters. This can be viewed as the membership of the feature value of the video chunk to the cluster. Several functions are often used to convert the distance values into fuzzy membership values [3]. To compute

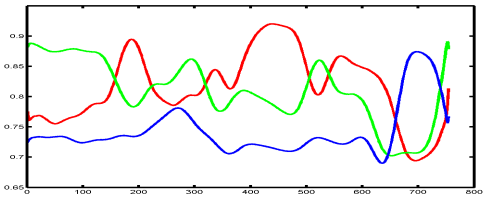

Figure 2: Fuzzy memberships of an example video. It consists of three clusters. The corresponding membership values are represented by red, green and blue colors.

the membership values, we use the Hyperbolic tangent set functions. The fuzzy function takes the distance between the feature point and the cluster center as the input and maps it to a value in the range of [0, 1].

Given a cluster C_i with mean c_i, the membership, $C_i(f_j)$, of feature f_i belonging to this cluster is computed as,

$$M_i(f_j) = \frac{tanh\left(\frac{f(j)-c(i)+l_c}{d_c}\right) - tanh\left(\frac{f(j)-c(i)-l_c}{d_c}\right)}{tanh\left(\frac{l_c}{d_c}\right)}, \quad (3)$$

where $tanh(\cdot)$ is the Hyperbolic tangent function, d_c is average distance between the centers of the clusters, and l_c is the spread of the function which controls the shape. While l_c decreases, the shape of the function is more concentrated to the function mean. If the user prefers a lose condition, l_c can be set larger, such that the feature that is relatively far from the cluster center is still having high membership value. Several plots of the Cauchy functions with different l_c is shown in Figure 1 with $d_c = 2.5$ and the mean to be 0.0. The variation of the cluster distances, d_c, is computed as the average distance between cluster centers,

$$d_c = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{k=i+1}^{N} \parallel c_i - c_k \parallel . \quad (4)$$

For each small chunk in the video, its membership values to all the feature clusters are computed. As the end result, the entire video eventually possesses k membership plots corresponding to the k feature clusters, and the scene segmentation is later performed based on these membership plots. Some examples of the fuzzy membership plots are shown in Figure 2. For this particular video sequence, there are three feature clusters, and the membership values of the sequence corresponding to the clusters are represented in red, green and blue colors.

2.3 Video Scene Segmentation

The fuzzy membership value indicates how confidently a feature point belongs to a feature cluster. Based on the application need, the users have the flexibility of selecting the scene boundaries by applying the preferred thresholds. Higher threshold results in more precise and coherent scene content. On the other hand, lower threshold results in larger scenes with more inter-scene transition content. From the empirical experience, one of the good thresholds for selecting video chunks belonging to a cluster C_i is defined as, $\tau_i = c_i + 0.5 \times \sigma_i$, where c_i is the mean of the cluster C_i, and σ_i is the standard deviation of the cluster. If a chunk has a membership value above the preferred threshold, it is classified as a *scene chunk* with respect to the corresponding feature cluster. The adjacent *scene chunk* determined by the same feature cluster form the initial candidate video scenes. The candidate scenes are later refined by a series of

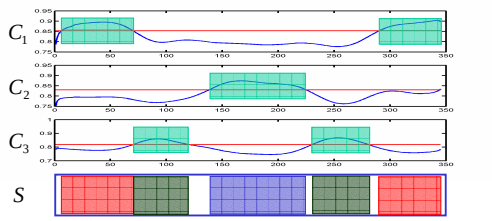

Figure 3: An example for the fuzzy representation and the resulting scene segments by applying thresholds $\tau_i = c_i + 0.5 \times \sigma_i$. The top three plots are the membership plots to three feature clusters, and the last figure shows the resulting scenes. The horizontal axis represents time. The blues lines in the membership plots represents the cutting-edge thresholds. Scenes with the same color are generated from the same feature cluster.

scene merge, scene deletion and scene competition actions. Final output is composed of the scenes determined by all the feature clusters.

The detailed scene segmentation algorithm is as follows:

1. Apply the threshold, τ_i, to all the video chunks for every feature cluster. If $M_i(f_j) \geq \tau_i$, then label chunk t_j to be a *scene chunk* and assign it a label i; otherwise, label it as a *non-scene chunk*.
2. Apply 1-D connected component method to the video chunks with the same labels to create the initial candidate scenes. The scenes are labelled as $\{S_1^1, \cdots, S_{n_1}^1, \cdots, S_1^k, \cdots, S_{n_k}^k\}$, where the superscript represents to which cluster it belongs to, and the subscript represents the temporal order of the scene.
3. Many times, it is possible that the same video chunk is classified as a *scene chunk* with respect to two or more clusters. In this case, a decision needs to be made to select the most appropriate one. That is called the scene competition. The competition/selection is carried on as follows:
   - For two scenes S_1 and S_2 with different labels, compute their overlapping region with length p.
   - Compute the fractions, $F_1 = \frac{p}{|S_1|}$ and $F_2 = \frac{p}{|S_2|}$, of the overlapping regions to scenes S_1 and S_2, respectively, where $|\cdot|$ represents the scene length.
   - If $F_1 < Th$ and $F_2 < Th$, it means that S_1 and S_2 do not seriously conflict with each other. Therefore, they are kept the same. Otherwise, if $F_1 \geq Th$ or $F_2 \geq Th$, the one with lower fraction is thought to be dominant, and the other one is deleted. In our experiments, we use $Th = 0.5$.
4. Repeat 4 until no more competition is needed.
5. Collect the resulting scenes from all the clusters after the refinement to form the final output.

One segmentation example is shown in Figure 3 with the membership plots and the resulting final scenes.

3. EXPERIMENTAL RESULTS

The proposed approach has been tested on several continuously recorded home videos. To demonstrate the generality of the proposed method, scenes with or without abrupt video shots are both tested. Each video sequence contains the

Table 1: Accuracy measures of seven continuously recorded video sequences. The results were obtained using thresholds $\tau_i = c_i + 0.5 \times \sigma_i$. Desired segmentations can be obtained by tuning τ to the preferred level.

Measures	*clip1*	*clip2*	*clip3*	*clip4*	*clip5*	*clip6*	*clip7*
Video Length	12:42	06:53	07:31	15:04	27:41	14:26	15:34
Num. of True Scenes	8	5	5	7	3	6	5
Num. of Clusters	6	3	9	3	3	4	3
Num. of Detected Scenes	5	5	4	7	4	6	6
Match (in seconds)	511	395	373	561	1302	711	779
Precision	0.6962	0.9875	0.8289	0.7835	0.8527	0.8767	0.8964
Recall	0.7197	0.9564	0.8271	0.8644	0.8235	0.8989	0.8842

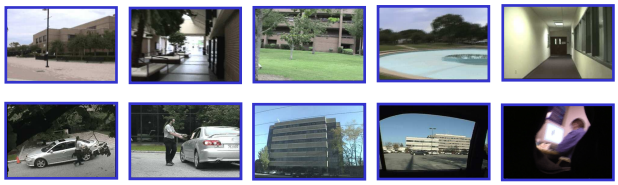

Figure 4: Some key-frames of the testing videos.

scenes recorded from several physical sites, and some of the inter-scene transitions are gradual. Some of the key-frames of the testing videos are shown in Figure 4. Clearly, due to the absence of the abrupt shot boundaries, shot-based scene segmentation is not applicable in this situation. Furthermore, due to the gradualness of the inter-scene transitions, conventional scene matching criteria based on the boundary comparison is not practical and less meaningful. To solve the evaluation problem, we used a "recover" method. Suppose there are reference (ground truth) scenes $\{T_1, T_2, \cdots, T_{t_n}\}$ and the scenes $\{S_1, S_2, \cdots, S_{s_n}\}$ detected by the proposed method. A reference scene T_i is said to be recovered, if one of the detected scenes S_j overlaps with it. Similarly, a detected scene S_j is said to be matched if one of the reference scenes T_i overlaps with it. This matching is a 1-to-1 relationship, i.e., one reference scene can be matched with at most one detected scene, and vice versa. Differing from the conventional evaluation systems, which are based on the number of matched pairs, we use the length of the overlapping region as the system performance. The overall performance on each video is computed as the total length of the overlapping regions in that video, since longer overlapping region indicates better segmentation.

We use two accuracy measures, precision and recall, measuring both of how precise the system performs and how well it recovers the ground truth. They are defined as,

$$Precision = M/D, \quad Recall = M/G, \tag{5}$$

where M is the total length (in time) of the matched regions between the detected scenes and the reference scenes; D is the total length of the detected scenes, and G is the total length of the ground truth scenes. The detailed statistics of the system performance is shown in Table 1.

To further demonstrate the effectiveness of the proposed method, we comparee its performance with the BSC method that is proposed in [5]. This method contains two steps: initial scene segmentation using color information and refinement using motion similarity. In our experimental setup, since there is no shot structure in the general sense, we computed the corresponding features on each of the small chunks ("artificial shots") in the video, and then applied the regular scene segmentation method. The performance comparison between the BSC approach and the proposed fuzzy method is shown in Table 2. The proposed method has boosted the accuracy for 10% to 20% in average.

Table 2: Performance comparison between the BSC method and the proposed fuzzy method.

Method	**BSC**		**Fuzzy**	
Accuracy	Precision	Recall	Precision	Recall
clip001	0.6299	0.6761	0.6962	0.7197
clip002	0.7732	0.7676	0.9875	0.9564
clip003	0.6341	0.6341	0.8289	0.8271
clip004	0.7490	0.6482	0.7835	0.8644
clip005	0.5931	0.6348	0.8527	0.8235
clip006	0.5076	0.6361	0.8767	0.8989
clip007	0.7093	0.5994	0.8964	0.8842

4. CONCLUSIONS

In this paper, we have presented a framework for the temporal scene detection in continuously recorded videos. The proposed method employs the spectral clustering technique to cluster the feature points into feature groups, which are used to determined the video scenes. Each video chunk has a sequence of fuzzy numbers representing its memberships to the corresponding clusters. The initial scenes are detected by applying user defined thresholds and are further processed with a series of refining actions. The major contribution in this paper is that instead of finding the abrupt scene boundaries, which are not meaningful in many situation, the proposed method provides the users with the flexibility of selecting the preferred scene content. The proposed method has been tested on several video sequences, which are continuously recorded, with or without well defined video shots. Highly accurate and competitive results have been obtained and presented.

5. REFERENCES

[1] A. Hanjalic, R.L. Lagendijk, and J. Biemond, "Automated High-Level Movie Segmentation for Advanced Video-Retrieval Systems", *CSVT*, 1999.

[2] W. Hsu and S.F. Chang, "Generative, Discriminative, and Ensemble Learning on Multi-Model Perceptual Fusion Toward News Video Story Segmentation", *ICME*, 2004.

[3] S. Mitaim and B. Kosko, "The Shape of Fuzzy Sets in Adaptive Function Approximation", *Fuzzy Systems*, 2001.

[4] A.Y. Ng, M.I. Jordan and Y. Weiss, "On Spectral Clustering: Analysis and an Algorithm", *NIPS*, 2002.

[5] Z. Rasheed, M. Shah, "Scene Detection In Hollywood Movies and TV Shows", *CVPR*, 2003.

[6] H. Sundaram and S.F. Chang, "Video Scene Segmentation Using Video and Audio Features", *ICME*, 2000.

[7] M. Yeung, B. Yeo, and B. Liu, "Segmentation of Videos by Clustering and Graph Analysis", *CVIU*, 1998.