A Graph Theoretic Approach for Scene Detection in Produced Videos

Zeeshan Rasheed
Computer Vision Lab
University of Central Florida
Orlando, Fl.
zrasheed@cs.ucf.edu

Mubarak Shah
Computer Vision Lab
University of Central Florida
Orlando, Fl.
shah@cs.ucf.edu

ABSTRACT

Efficient video browsing requires indexing of videos so that the users can quickly locate the segments of their interests. While browsing a video, the users often prefer to skim through the video by scenes rather than frames or shots. In general, a scene can be defined by the continuity in the visual contents of shots due to fixed physical setting or by the continuity of the ongoing actions. We exploit this fact and propose a novel approach for clustering shots into scenes by transforming this task into a graph partitioning problem. This is achieved by constructing a weighted undirected graph called a *shot similarity graph, SSG*, where each node represents a shot and the edges between the shots are weighted by their similarities. Both color and motion information are utilized to compute shot similarities. The SSG is then split into smaller *s*tory units by applying the normalized-cut technique for graph partitioning. The proposed approach is robust and produces meaningful temporal segmentation of video, which is useful for applications such as "video on demand". Experiments are presented with promising results on two Hollywood movies.

Keywords

video segmentation, normalized cuts, graph partitioning, key frames.

1. INTRODUCTION

The ever growing amount of multimedia data increases the need for developing new and efficient techniques of multimedia data storage and retrieval. The recent years have seen increasing research interest in this field due to the boost in the computation power and storage capabilities of computers. Likewise, the exponential growth of video delivery systems, such as video on demand via cable and the Internet, further entails new designs of multimedia browsing and retrieval systems - video in particular. For example, DVDs currently are made with options to view a particular scene in the movie by providing a "chapter selection" menu. To obtain such a representation, a human observer is required to sequentially watch the video and locate the scene boundaries. However, due to the huge number of produced videos being generated every year, manual content analysis is impractical, as it is slow and expensive.

A shot is the basic unit of a video. Therefore, splitting the complete visual track of a video into shots provides a presentation of the video one level higher than browsing a sequential tape. However, the number of shots may vary from several hundred to thousands in a typical full length feature movie. Therefore, a meaningful segmentation requires clustering the shots into scenes, as these scenes provide the semantic meaning of the video. Yeung et al. [10] proposed a graphical representation of videos by constructing a *s*cene transition graph, STG. Each node in an STG represented a shot, and the edges represented the transitions between the shots based on the visual similarity and temporal locality. The STG was then split into several subgraphs using the *complete-link* method of hierarchical clustering. Each subgraph satisfied a similarity constraint based on the color and represented a *scene*. Hanjalic et al. [4] used a similar approach for shot clustering using a graph and found *logical story units*. The linking of shots was done by defining an *inter-shot dissimilarity measure* among the shots. Their method used MPEG compressed video sequences and utilized DCT images. The number of key frames for each shot varied with the dynamics of that shot. Each shot was represented by combining all key frames within the shot. An average distance between the shots was computed in the $L^*u^*v^*$ color space. This determined whether or not two shots were part of one logical story unit. If two shots were found to be similar, all the shots in between them were also merged to construct one logical story unit. Rui et al. [6] proposed the construction of table-of-contents for videos. A time-adaptive grouping of shots was done by finding the visual similarities between them. This similarity was a function of color and activity features of the shots, weighted by their temporal locality. Shots were merged together to form groups by defining a *group threshold* and groups were merged together to form scenes by defining a *scene threshold*. They also suggested a method to determine these thresholds automatically. Recently, Zhou et al. [11] exploited film editing techniques and discovered that certain regions in the video frame are more robust to noise for computing shot similarities. The clustering method is, however, similar to [4]. The

improvement made in their work was the use of different frame regions for establishing links between the shots.

The decision of whether a scene boundary exists in the aforementioned graph-based methods is often based on the similarity between two arbitrary shots, thresholded by an empirical value. However, it may happen by chance that in any typical feature movie, two different scenes will have a pair of visually similar shots. If the decision about the scene boundary is made on an individual basis, they will both be considered as parts of one scene. All shots between them will also be erroneously merged together to form one scene. This is shown in Fig. 1. Consider two scenes, 1 and 2. Scene 1 consists of shots A and B, and scene 2 consists of shots C, D and A'. If shot A happens to be similar to any shot in scene 2, such as A', an erroneous link will be formed between scene 1 and 2 (indicated by a dotted link). This will result in the undersegmentation of the video, i.e., the number of resulting scenes will be less than expected. We believe that the detection of scenes should not be based on one-on-one shot similarity. Instead, all the shots should be considered in order to maximize the intra-scene similarity between the shots of one scene and to minimize the inter-scene similarity between the shots of two different scenes.

Adams et al. [1] also parsed videos by computing the *tempo* in feature movies. Their approach was inspired by the existing cinematic conventions known as film grammar. Camera motion parameters and the shot lengths in the video were used to construct a *tempo* plot. The proposed method detected edges in the tempo function and identified instances where the tempo of the movie changed with time. This information was further used as a cue for detecting story sections and events. However, their method did not detect logical boundaries between the scenes in which the *tempo* was consistent. We believe that the visual similarities of shots can also be combined with the motion and shot length features to improve the temporal segmentation of videos.

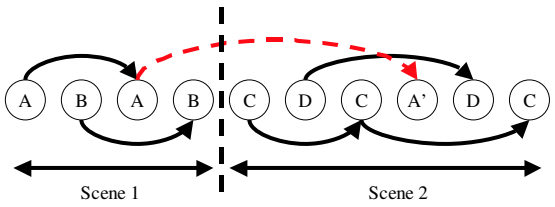

Figure 1: Erroneous shot linking indicated by the dotted link. A valid scene boundary will be missed resulting in the undersegmentation of videos.

2. OUR APPROACH

A scene can be defined as a subdivision of a movie in which the physical setting is fixed. It may also be defined by the continuity of the action even when the physical setting is not consistent. In scenes with fixed physical setting, several cameras capture video from different viewing angles where the background remains the same. For example, a scene shot inside a studio with multiple cameras. As the cameras switch back and forth, they repeatedly show similar views of the actors and the background. Therefore, the visual contents of the shots are highly correlated. However, physical setting may not be fixed for certain scenes. For example, an outdoor travelling scene is generally shot by cameras mounted on a truck or a trolley. In this case, a scene may be defined by the continuity of ongoing action performed by the actor(s). Taking these factors into account, we have developed a framework to find scene boundaries which exploits both the color and motion similarities of shots. In our approach, we construct a weighted undirected graph called a *shot similarity graph*, SSG, and transform the problem of scene boundary detection into a graph partitioning problem. The undirected graph $G = (V, E)$ consists of N nodes such that each node represents a shot and edges connect the nodes. A weight $W(i, j)$, which is proportional to the shot similarities, is associated with every edge connecting nodes i and j. Sec. 2.3 discusses the computation of shot similarities as a function of shot color and motion contents. Finally, the scene boundaries are detected by partitioning the SSG into subgraphs that maximize the intra-subgraph similarities and minimize the inter-subgraph similarities. It should be noted that our algorithm considers the *global* similarities of shots to detect boundaries. Therefore, it is not affected by any *local* mismatch between two shots that belong to two different scenes as outlined earlier in Sec. 1. The detection of scenes is explained in Sec. 2.5. We present our experiments in Sec. 3 and Sec. 4 concludes our work.

2.1 Shot Detection and Key Frame Selection

The video is first divided into shots. Much work has been reported in this area and highly accurate results have been obtained. We use a modified version of the color histogram intersection method of shot detection, which was proposed by Haering [3]. For each frame, a 16-bin HSV normalized color histogram is estimated with 8 bins for hue and 4 bins each for saturation and value. Shots that are shorter than 10 frames are merged with the following shots as being erroneous. Each shot is represented by one or more key frames depending upon the shot activity. We use the method proposed by Rasheed and Shah [5] that selects a variable number of frames to represent the shot contents. Thus a shot z is a set of frames $S_z = \{f^a, f^{a+1}, ..., f^b\}$ where a and b are the indices of the first and the last frames of the shot, and is also represented by a set of key frames K_z. Note that the index of the middle frame of the shot is expressed as $m_z = \lfloor \frac{a+b}{2} \rfloor$.

2.2 Shot Motion Content Feature

We associate a feature, Mot_z with each shot z, which is the motion content of the shot normalized by the number of frames in it, that is:

$$Mot_z = \frac{1}{b-a} \sum_{f=a}^{b-1} \left(1 - ColSim(f, f+1)\right), \qquad (1)$$

where $ColSim$ is defined as the color similarity between two image frames, that is:

$$ColSim(x, y) = \sum_{k \in bins} min(H_x(k), H_y(k)), \qquad (2)$$

where H_x and H_y are the HSV color histogram of frames x

and y respectively. Due to the temporally changing visual content in an action scene, we have observed that the motion content value of the shots from an action scene is much higher than for the shots from a dialogue scene. Therefore, it can be used as a suitable feature for shot similarity estimation as explained in the next section.

2.3 Finding Color and Motion Similarities Between the Shots

As explained in Sec. 2, shots that belong to one scene often have similar visual (color) and/or action (motion) contents. Generally, dialogue shots span a large number of frames and are often filmed with fixed physical setting. Due to the repetitive transitions between the fixed camera views, the shots in this scene category have higher visual correlation. On the other hand, shots of fight and chase scenes change rapidly and last for few frames (Arijon [2]). In a similar fashion, the motion content of shots also depends on the nature of the scene. The dialogue shots are relatively calm (neither actors nor the camera exhibit large motion). Although camera pans, tilts and zooms are common in dialogue shots, they are generally smooth. In fight and chase shots, the camera motion is jerky and haphazard with larger movements of actors. For a given scene, these two attributes are kept consistent over time to maintain the pace of the movie. Thus, we compute the similarities between shots as a function of their visual and motion content features. That is, the similarity between shot i and j will be:

$$ShotSim(i,j) = VisSim(i,j) + MotSim(i,j). \quad (3)$$

The $VisSim$ between shots i and j is now defined as follows:

$$VisSim(i,j) = \max_{p \in K_i, q \in K_j} \left(ColSim(p,q) \right), \quad (4)$$

where p and q are the key frames of shot i and j respectively. Thus the $VisSim$ for any arbitrary pair of shot is the maximum $ColSim$ of all possible pairs of their key frames.

It is more likely that the consecutive shots of a particular scene will have similar motion contents (consider an action scene for example). We compute the motion similarity, $MotSim$, between two shots as follows:

$$MotSim(i,j) = \frac{2 \times min(Mot_i, Mot_j)}{Mot_i + Mot_j}. \quad (5)$$

Thus, if two shots have similar motion content, their $MotSim$ will have a higher value. Note that both $VisSim$ and $MotSim$ are in the range 0-1.

2.4 Constructing the Shot Similarity Graph, SSG

Given N shots, we construct a weighted undirected graph called a *shot similarity graph*, $G = (V, E)$, such that each shot i is represented by a node, v_i, where i is the shot index.

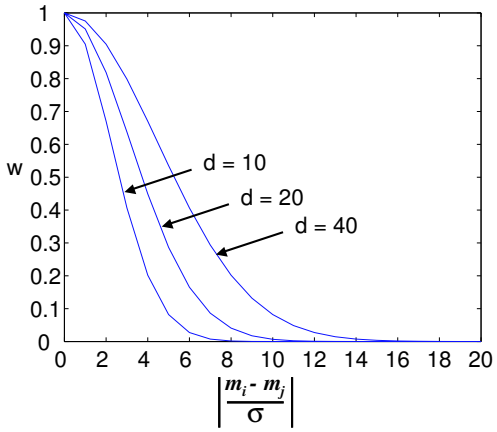

Figure 2: Computation of $w(i,j)$, which is an exponentially decreasing function of temporal distance between two arbitrary shots, i and j. Note that with a greater d, the weight decreases with a slower rate.

Also, an edge is present between every pair of nodes. Let $e(i,j) \in E$ be the edge between the nodes i and j with an associated weight $W(i,j)$ which reflects the likelihood that two shots belong to one scene. It is less likely that two shots farther apart in time will belong to one scene. That is, the higher the temporal distance between them, the lower the probability that the shots constitute one scene. Therefore, the weight $W(i,j)$ is proportional to the $ShotSim(i,j)$ and temporal proximity of the shots. This is formulated as:

$$W(i,j) = w(i,j) \times ShotSim(i,j), \quad (6)$$

where $w(i,j)$ is a decreasing function of the temporal distance between the shots. Rui et al. used a linearly decreasing function of the temporal distance to compute the *temporal attraction* between frames [6]. However, we found that an exponentially decreasing function performs better. Thus the weight $w(i,j)$ decays with the distance between the middle frames of the shots under consideration, that is:

$$w(i,j) = e^{-\frac{1}{d} \cdot |\frac{m_i - m_j}{\sigma}|^2}, \quad (7)$$

where σ is the standard deviation of the shot durations in the entire video. The rate of decay is controlled by a factor d. We set $d = 20$ in our experiments, which was kept constant throughout the experiments. Fig. 2 shows the plot of w against the temporal distance between shots.

Fig. 3 shows the *shot similarity graph* constructed for 36 minutes of the movie "A Beautiful Mind". There are 219 shots in the video. The similarities between the nodes are represented by pixel intensities such that lower intensity means higher similarity. A human observer identified the scene boundaries present in the video which are indicated by lines on the bottom-right side of the image. Note that the shots that belong to a particular scene form distinct clusters as seen in the zoomed-in section of the SSG.

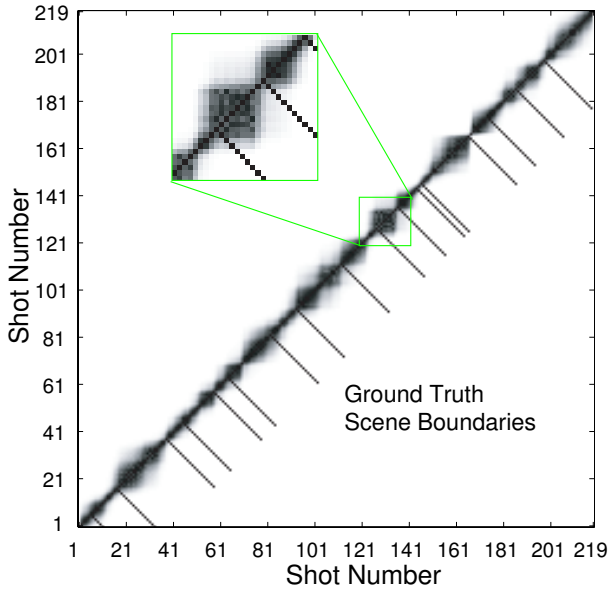

Figure 3: A *shot similarity graph* for 36 minutes of the movie "A Beautiful Mind". Darker pixels represent higher similarities. The ground truth scene boundaries are indicated with lines. Note that shots that belong to a particular scene form distinct clusters as seen in the zoomed-in section of the SSG.

2.5 Scene Detection Using Graph Cut

Graph partitioning techniques are known for effective perceptual grouping. Several algorithms have been proposed for segmenting images based on pixel proximity and color intensity similarity, for example, [8], [9] and [7]. Generally, the partitioning solution is achieved by recursive bipartitioning, that is, at each step the graph is divided into two parts based on a partitioning measure. We employ the graph partitioning technique proposed by Shi and Malik [8] called *normalized cuts*. Starting with an initial SSG, $G = (V, E)$, we seek a partitioning into two disjoint subgraphs, $G' = (V', E')$ and $G'' = (V'', E'')$ such that $V' \cup V'' = V$ and $V' \cap V'' = \emptyset$. Such a partition is achieved by removing the edges connecting subgraphs G' and G''. In particular, there exist an exponential number of such partitions. However, for videos, we seek a partition such that all shots that belong to a particular subgraph are time continuous. That is, the following condition holds:

$$(i < j \text{ or } i > j) \text{ and } i \neq j \text{ for all } \ v_i \in V', v_j \in V''.$$

Thus, the complexity of partitioning an SSG into two subgraphs is of order N which is the number of shots present in the segment of the video. In graph theory literature, the summation of weights associated with the edges being removed is called a *cut* and it reflects the degree of dissimilarity between the two parts, that is:

$$cut(V', V'') = \sum_{i \in V', j \in V''} W(i, j). \tag{8}$$

The normalized cut value for such a partitioning is expressed as:

$$Ncut(V', V'') = \frac{cut(V', V'')}{assoc(V', V)} + \frac{cut(V', V'')}{assoc(V'', V)}, \tag{9}$$

where $assoc(X, V)$ is the summation of weights associated with the edges connecting all nodes in X to all nodes in V, that is:

$$assoc(X, V) = \sum_{c \in X, d \in V} W(c, d). \tag{10}$$

We apply a recursive algorithm of graph partitioning such that the intra-subgraph similarities are maximized and the inter-subgraph similarities are minimized; that is, the shots in each subgraph will have higher visual (color) and activity (motion) similarities. This approach results in the clustering of shots that are more likely to be in one scene. It should be noted that there are no specific thresholds that control the segmentation which is the basis of several graph-based video segmentation approaches. Hence, our method does not suffer in accuracy due to any mismatch between the shots of different scenes and therefore it is more robust to noise. Fig. 4 shows the scene detection for the movie "A Beautiful Mind". The detected scene boundaries are identified with lines on the upper left side of the image. Please refer to Sec. 3, which discusses the experiments on video segments taken from both *action* and *non-action* Hollywood movies. Experiments show that our method produces results with high recall/precision values when applied to two very different genres of feature movies.

Table 1: Summary of Data Set and Experimental Results.

Movie	A Beautiful Mind	Terminator-II
Duration	36 min	55 min
#Frames	65122	98505
#Shots	219	994
G. Truth Scenes	18	36
Detected Scenes	28	70
Correct Scene	15	32
False negative	3	4
False Positive	13	38
Recall	0.833	0.889
Precision	0.536	0.457

3. EXPERIMENTS

To evaluate the performance of proposed algorithm, we performed experiments on two sets of videos each digitized at 29.97 frames per second. The first video segment was 36 minutes long, taken from a slow paced movie, "A Beautiful Mind". The second video segment was taken from the movie "Terminator II", an action movie which is very different from the first one and consists of several action scenes. The purpose of experimenting with two different genres of movies was to demonstrate the robustness of the algorithm regardless of the movie genre. A human observer identified the scene boundaries in both videos which were used as

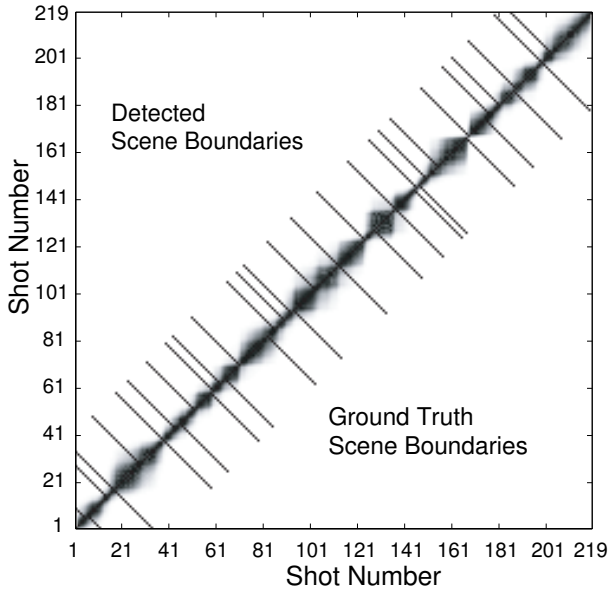

Figure 4: Scene detection for 36 minutes of the movie "A Beautiful Mind". Detected scene boundaries are indicated with lines in the upper left side of the graph. Out of 18 scene boundaries, 15 scene boundaries are identified correctly.

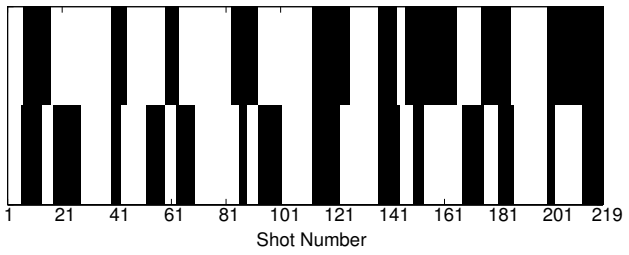

Figure 5: Ground truth scenes vs. detected scenes. The upper row represents the scenes identified by a human observer. Consecutive scenes are shown with alternating black and white patterns. The bottom row shows the scenes detected by our algorithm for the movie "A Beautiful Mind".

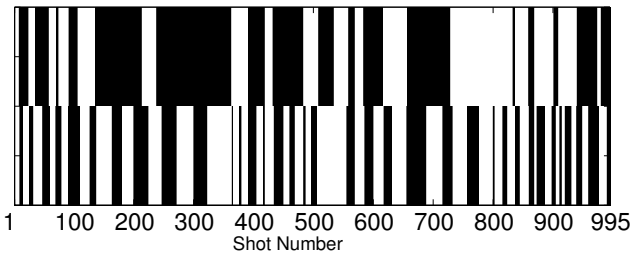

Figure 6: Ground truth scenes vs. detected scenes. The upper row represents the scenes obtained from the movie's DVD chapter selection menu. Consecutive scenes are shown with alternating black and white patterns. The bottom row shows the scenes detected by our algorithm for the movie "Terminator II".

Table 2: Scene Boundary Detection for the movie "A Beautiful Mind". Correctly identified scenes are marked as $\surd$. '*' represents the scenes identified by the human observer and did not appear in the movie's DVD chapter selection menu.

No.	DVD Chap./Human Observer*	Scene Detected?
1	Mathematicians	$\surd$
2	Reflections *	$\surd$
3	Princeton Dorm *	$\surd$
4	Drinking *	$\surd$
5	A Challenge	$\surd$
6	The Need to Focus	$\surd$
7	The Bar *	$\times$
8	Princeton *	$\surd$
9	Dorm Room *	$\surd$
10	Governing Dynamics	$\surd$
11	Research *	$\times$
12	With the Principle *	$\surd$
13	Celebrations *	$\surd$
14	The Pentagon	$\surd$
15	Wheelers Defense Labs *	$\times$
16	Teacher and Students	$\surd$
17	Code Breaker	$\surd$
18	Laboratory *	$\surd$

the ground truth. We have also incorporated DVD chapter information in ground truth construction. We found that in the movie "Terminator-II", the chapters coincided with the human observer's suggested scene boundaries (see Tab. 3). For the movie "A Beautiful Mind", we found that the DVD chapters were a superset of what the human observer had suggested. These scenes are identified by a '*' in Tab. 2. The boundaries detected by the proposed method were compared against the ground truth. A 30-second sliding window was swept over the detected boundaries as the tolerance factor. The overall results are presented in Tab. 1. There are two issues involved in evaluating any segmentation algorithm; *oversegmentation* and *undersegmentation*. We believe that a slight oversegmentation is preferable to undersegmentation, since split scenes can be combined by further analysis. That is, a higher recall value is preferred. Fig. 5 and 6 show the scene detection results. The ground truth scenes are indicated by alternating black/white patterns w.r.t. the shot numbers in the upper row. The bottom row shows the detected scenes. We have observed that the algorithm works better for slow paced scenes, such as dialogue scenes, than for fast paced scenes. This is due to the fact that slow paced scenes are often well structured. Action scenes, on the other hand, are poorly structured and appears as multiple clusters in the graph. We believe that the use of audio information as a similarity measure can be incorporated to improve the segmentation task.

4. CONCLUSION

We presented a method of high level segmentation of video into scenes. We exploited the fact that shots that belong to one particular scene often have similar visual (color) and action (motion) attributes and transformed the scene segmen-

Table 3: Scene Boundary Detection for the movie "Terminator-II". Correctly identified scenes are marked as √.

No.	DVD Chapter	Scene Detected?
1	Meet John Conner	√
2	Sarah Connor	√
3	T-1000 Visits The Voights	√
4	Easy Money	√
5	Sanity Review	×
6	Cyberdyne Systems	√
7	Model Citizen	√
8	Target Acquired	√
9	The Galleria	√
10	Zeroed In The Corridor	√
11	Into The Streets	√
12	Canal Chase	√
13	Time Out	√
14	Never This Nice	√
15	Photos	√
16	Mission Parameter	√
17	Pescadero State Hospital	√
18	Lewis The Guard	√
19	Sarah Breaks Out	×
20	215 Bones	√
21	I Swear	√
22	Syringe Point	√
23	Come With Me If You Want To Live	√
24	Escape From Pescadero	√
25	Security Car	√
26	Nice Bike	√
27	Night Repairs	×
28	Head South	√
29	No Problemo	×
30	Detailed Files	√
31	Scalcedas Camp	√
32	Weapons Cache	√
33	Fathers And Sons	√
34	Nuclear Nightmare	√
35	Sarahs Decision	√
36	No Fate	√

tation task into a graph partitioning problem. The proposed method is superior to other graph-based approaches in that it considers all the shots to cluster shots into scenes and captures the global similarities of shots rather than the local similarities. It produces semantically meaningful scenes and it is robust to noise.

5. REFERENCES

[1] B. Adams, C. Dorai, and S. Venkatesh. Towards automatic extraction of expressive elements from motion pictures: tempo. In *IEEE International Conference on Multimedia and Expo*, pages 641–644, 2000.

[2] D. Arijon. *Grammar of the Film Language*. Hasting House Publishers, NY, 1976.

[3] N. Haering. A framework for the design of event detections, (Ph.D. thesis), 1999. School of Computer Science, University of Central Florida.

[4] A. Hanjalic, R. L. Lagendijk, and J. Biemond. Automated high-level movie segmentation for advanced video-retrieval systems. *IEEE Transaction on Circuits and Systems for Video Technology*, 9(4):580–588, June 1999.

[5] Z. Rasheed and M. Shah. Scene detection in Hollywood movies and TV shows. In *IEEE Computer Vision and Pattern Recognition*, 2003.

[6] Y. Rui, T. S. Huang, and S. Mehrotra. Constructing table-of-content for videos. *ACM Multimedia Systems Journal, Special Issue Multimedia Systems on Video Libraries*, September 1999.

[7] S. Sarkar and P. Soundararajan. Supervised learning of large perceptual organization: Graph spectral partitioning and learning automata. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(5):504–525, May 2000.

[8] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8), August 2000.

[9] Z. Wu and R. Leahy. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993.

[10] M. M. Yeung, B.-L. Yeo, and B. Liu. Segmentation of video by clustering and graph analysis. *Computer Vision and Image Understanding*, 71(1), 1998.

[11] J. Zhou and W. Tavanapong. Shotweave: A shot clustering technique for story browsing for large video databases. In *International Workshop on Multimedia Data Document Engineering*, March 2002.