Scene Detection in Videos Using Shot Clustering and Symbolic Sequence Segmentation

Vasileios Chasanis, Aristidis Likas and *Nikolaos Galatsanos*
Dept. of Computer Science, University of Ioannina,
GR 45510, Ioannina, Greece
e-mail: {vchasani, arly, galatsanos}@cs.uoi.gr

Abstract— *Video indexing requires the efficient segmentation of the video into scenes. In the method we propose, the video is first segmented into shots and key-frames are extracted using the global k-means clustering algorithm that represent each shot. Then an improved spectral clustering method is applied to cluster the shots into groups based on visual similarity and a label is assigned to each shot according to the group that it belongs to. Next, a method for segmenting the sequence of shot labels is applied, providing the final scene segmentation result. Numerical experiments indicate that the method we propose correctly detects most of the scene boundaries while preserving a good trade off between recall and precision.*

Keywords—scene segmentation; spectral clustering; key-frames; shot similarity

Topic area—multimedia analysis.

I. Introduction

In recent years the extended use of videos in several applications such as internet-TV and video on demand, as well as the thousand TV-series and movies produced every year, has led to a significant increase in the availability and the amount of video information. Video indexing, retrieval and analysis seem quite difficult due to this huge amount of data constantly produced. Video scene segmentation provides the most efficient solution so far. However to proceed with scene segmentation, low level segmentation of the video must be implemented.

The smallest physical segment of a video is the shot and is defined as an unbroken sequence of frames recorded from the same camera. The visual content of each shot of the video can be represented by one or multiple frames, called key-frames. A further analysis demands grouping shots into scenes. A scene can be regarded as a series of shots semantically correlated. The term scene usually refers to a group of shots taken in the same physical location describing objects or events. A more compact representation of a video could be the merging of scenes into logical story units that correspond to chapters that describe the different subthemes of the movies.

There are several approaches to scene segmentation problem. In [7] the authors transform this task into a graph partitioning problem and they apply *normalize cuts* [8] to partition the graph. In [2], a method is proposed for detecting boundaries of the logical story units by linking similar shots and connecting overlapping links. A similar approach is presented in [9], where a first *scene transition graph* is constructed to represent the video and then the *complete-link* method of hierarchical clustering is applied to split the graph into subgraphs representing the scenes. A different approach is presented in [6] where a two-pass algorithm is proposed. In the first pass shots are clustered by computing *backward shot coherence*, while in the second pass oversegmented scenes are merged based on the computation of *scene dynamics*.

Most of the approaches presented above, apart from calculating shot similarities based on visual similarity, they consider temporal distance of shots as an extra feature for shot clustering into scenes. Due to the absence of prior knowledge concerning the video content and the duration of scenes, it is difficult to determine an appropriate parameter value that will account for the contribution of time distance in the computation of the overall similarity between shots.

In our approach shots are clustered into groups using an improved version of the typical spectral clustering method [4] that employs the global k-means algorithm [3] in the clustering stage after eigenvector computation. Moreover, in contrast to other clustering-based scene segmentation methods [7], shot similarity is based only on visual features, while time adjacency of shots is taken into account in a second processing stage. More specifically, cluster labels are assigned to shots according to their visual content and next, label pairs of successive shots are compared to identify patterns in the sequence of shot labels. When a change in such patterns occurs, a scene boundary is detected. The proposed approach

This research project (PENED) is co-financed by E.U.-European Social Fund (75%) and the Greek Ministry of Development-GSRT (25%).

achieves high correct detection rates while preserving a good trade off between the number of missed scenes and the number of false detected scenes.

The rest of the paper is organized as follows: In section II the procedure followed for computing shot similarity is described. In section III the scene detection algorithm is proposed. In section IV we present numerical experiments and compare our method with the one proposed in [7]. Finally in section V we conclude our work and present suggestions for further study.

II. SHOT DETECTION AND SHOT SIMILARITY

The first level of video segmentation is shot detection. We implemented the most widely used method for shot detection [10] that is based on color histograms. For each frame a 16-bin HSV normalized histogram is used, with 8 bins for *hue* and 4 bins for each of *saturation* and *value*.

A. Key-Frame Extraction

In order to compute shot similarity, first one or multiple key-frames must be extracted as representatives of the shot content. The number of key-frames cannot be predetermined because content variation may be different for each shot. For example for a static shot where there is little object motion, one key-frame may represent the shot quite adequately, whereas when there is high camera and object motion more key-frames are needed for a good representation.

Several approaches have been proposed for key-frame extraction. In [11] the authors detect multiple frames using unsupervised clustering based on the visual variations in shots. A main problem is the selection of the appropriate number of key-frames to represent each shot.

In our approach we have used the efficient global k-means algorithm [3] in order to cluster the frames into groups. Then the frame closest to the centroid of each group is considered as key-frame. As feature for frame clustering we have considered the color histogram of each frame which is represented by a vector with 16 components. Global k-means in an incremental deterministic clustering algorithm that overcomes the important initialization problem of the typical k-means approach. This initialization problem was found to be severe in the case of frame clustering, significantly affecting the quality of the key-frames. Using the global k-means, the obtained key frames usually provide a sensible representation of shot content. Since global k-means is an incremental clustering algorithm (one cluster centroid (key-frame) is added at each step), we stop adding key-frames when the improvement in clustering criterion was below a threshold value T_c.

B. Video Shots Similarity

In order to define shot similarity first the color similarity *ColSim* between two frames must be defined:

$$ColSim(x,y) = \sum_{h \in bins} \min\left(H_x(h), H_y(h)\right) \quad (1)$$

where H_x and H_y are the HSV normalized color histograms of frames x and y respectively.

As explained earlier shots that belong to the same scene often have similar color content. As the authors suggest in [7] the visual similarity between a pair of shots i and j can be computed as the maximum color similarity (*ColSim*) of all a possible pairs of their key-frames:

$$VisSim(i,j) = \max_{p \in K_i, q \in K_j} ColSim(p,q) \quad (2)$$

where K_i and K_j are the set of key-frames of shots i and j, respectively.

III. SCENE DETECTION

In order to perform scene detection a clustering of shots is needed into groups taking into account visual similarity (*VisSim*) and time adjacency. Supposed there is a set S of N shots to be segmented. In [7] the *normalized cut* method has been proposed which is equivalent to *spectral clustering*. In order to implement these methods an NxN similarity matrix is used as input. In [7,5] shot similarity takes into account both visual similarity and time adjacency, while in our method we have considered only visual similarity and time is taken into account in a second processing stage:

$$w(i,j) = VisSim(i,j), \quad i,j \in S \quad (3)$$

A. Spectral Clustering

After the similarity matrix W has been computed, the spectral clustering algorithm [4] is used to group shots into clusters. Spectral clustering (into k groups) involves two main computational steps: i) Compute the k principal eigenvectors of the similarity vectors to build an Nxk matrix X whose columns are those eigenvectors. ii) Cluster the rows of X into k groups using k-means.

Since, as previously emphasized, k-means is sensitive to initialization, in our approach the global k-means algorithm is implemented in the second step of the spectral clustering algorithm. In contrast with other methods that take into consideration time [7, 5], we do not, but cluster shots into groups only using the visual similarity of shots.

B. Scene Segmentation

Once the spectral clustering algorithm generates k clusters $\{C_1, C_2, \ldots, C_k\}$, a label is assigned to each shot according to the cluster that it belongs, thus producing a symbolic sequence of labels. For a representative example:

$$C_1C_1C_1C_2C_2C_2C_3C_5C_3C_5C_3C_5C_4C_4C_2C_4C_4$$

A scene change occurs when the pattern of symbols changes. In the above example scenes correspond to shots with time indices 1-3, 4-6, 7-12 (repetitive pattern C_3C_5) and 13-17. The number of clusters (k) is not the number of the scenes, but groups of shots with respect to their visual similarity. Considering the example above the third scene consists of shots that belong to two different clusters (C_3, C_5).

In order to account for noise in the sequence (shot C_2 in last scene above) and for the case of repetitive patterns (scene 7-12 above) we form a new sequence containing *pairs of successive labels* (this is analogous to *two-grams* in traditional string processing). In this sequence of pairs, successive similar pairs are considered to belong to the same scene. More specifically, in order to merge successive pairs in the same scene we check for the existence of at least one similar label in both pairs. If this is not the case, a scene boundary is identified and we consider that a new scene starts from the next pair in the sequence.

The algorithm proceeds in two passes. In pass one the first two shots are regarded as the first pair, whereas in pass two the second and the third shots are regarded as the first pair. As we can see in "*Fig. 1*," during the first pass the scene boundary (denoted as $\downarrow$) between clusters 3 and 4 is not detected since shots that belong to different scenes form a pair. However, in the second pass this boundary is clearly detected. The final boundaries are the union of the boundaries detected during both phases.

Original sequence $\rightarrow C_1 C_1 C_1 C_2 C_2 C_2 C_3 C_5 C_3 C_5 C_3 C_5 C_4 C_4 C_2 C_4 C_4$

Shot pairs sequence (first pass) $\rightarrow (C_1 C_1)(C_1 C_2)(C_2 C_2)\downarrow(C_3 C_5)(C_3 C_5)(C_3 C_5)\downarrow(C_4 C_4)(C_2 C_4)\, C_4$

Shot pairs sequence (second pass) $\rightarrow C_1\, (C_1 C_1)\downarrow(C_2 C_2)(C_2 C_3)(C_5 C_3)(C_5 C_3)(C_5 C_4)(C_4 C_2)(C_4 C_4)$

Fig. 1: The two phases of the scene detection algorithm

EXPERIMENTS

C. Data

The video sequences used for our data set were taken from TV-series and movies. Ten videos were used consisting of 5051 shots and 177 scenes. The average was 505 shots and 18 scenes per video. The ground truth of the scene boundaries was generated by a human observer.

D. Performance criteria

To evaluate the performance of our method we used the following measures [1]:

$$RECALL = \frac{N_c}{N_c + N_m},\ PRECISION = \frac{N_c}{N_c + N_f},$$
$$F_1 = \frac{2 * RECALL * PRECISION}{RECALL + PRECISION} \tag{4}$$

where N_c stands for the number of correct detected scene boundaries, N_m for the number of missed ones and N_f the number of false detections.

E. Results and Comparison

In our experiments we implemented the proposed algorithm for different values of k, where k is the number of clusters (shot labels) produced by the spectral clustering algorithm. In *Tables I* and *II* we present the recall and the precision as defined above for different values of k. It is obvious that our approach achieves high correct detection rate while keeping small the number of false detections. It seems that the choice $k=8$ results in good scene segmentation performance in all videos

To compare the effectiveness of our approach we have also implemented the method proposed in [7]. This method calculates both color and the motion content for each shot and the final shot similarity matrix is weighted by a decreasing function of the temporal distance between shots. Then the Normalized cuts method [8] is applied iteratively till the *Ncut* value reaches a pre-specified threshold T. We have tested the method in [7] using the same video set and for different values of the threshold parameter T. In *Table III* the values of the F_1 measure are presented for both methods. The obtained results indicate that our algorithm provides better F_1 values for all videos, and in general the performance of our method is superior as compared to the other method.

IV. CONCLUSIONS

In this paper a new method for video scene segmentation has been proposed. First key-frames are extracted using the global k-means algorithm. Then shots are clustered into groups using only visual similarity as feature and are labeled according to the group they belong. Shot clustering is achieved using an adaptation of the typical spectral clustering approach [4] that employs the global k-means algorithm in the second phase of spectral clustering methodology.

In our method we treated time adjacency of shots separately from visual similarity by developing an algorithm that assigns to same scene adjacent shots with the same label. To deal with noise and with the frequently encountered case of scenes containing a repetitive sequence of two different shots, we considered pairs of shots which are compared for similar labels. Our approach takes into account (enforces) time adjacency of shots in a post-processing step while existing methods incorporate temporal distance between shots into the clustering procedure. The presented experimental results on several videos indicate that the proposed method accurately detects most of the scene boundaries, while providing a good trade off between recall and precision. A drawback of all current algorithms including the one proposed herein is the oversegmentation that occurs in cases where there is a continuous change in the visual content of shots in a scene. However, oversegmentation is preferable over under-segmentation, since splitted scenes can be combined by further analysis [7]. In our future work, we plan to focus on addressing the problems that appear in videos where the visual content of shots continuously changes.

REFERENCES

[1] A. Del Bimbo, *Visual Information Retrieval*, Morgan Kaufmann Publishers, Inc, San Francisco, California, 1999.

[2] A. Hanjalic, R. L. Lagendijk, and J. Biemond, "Automated high-level movie segmentation for advanced video-retrieval systems,"*IEEE Trans. Circuits and Systems for Video Technology*, vol. 9, no. 4, pp. 580–588, June 1999.

[3] A. Likas, N. Vlassis and J. J. Verbeek: The global k-means clustering algorithm. *Pattern Recognition* Vol 36, No 2, 2003.

[4] A.Y. Ng, M.I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. *Proc. Neural Info. Processing Systems* (NIPS 2001), 2001.

[5] J.-M. Odobez, D. Gatica-Perez, and M. Guillemot, "Spectral structuring of home videos," *Proc. Int. Conf. Image and Video Retrieval*, pp. 310-320, 2003.

[6] Z. Rasheed and M. Shah, "Scene detection in Hollywood movies and TV shows," *Int. Conf. Computer Vision and Pattern Recognition*, 2003.

[7] Z. Rasheed and M. Shah, "Detection and representation of scenes in videos," *Multimedia, IEEE Transactions on* On page(s): 1097-1105, Volume: 7, Issue: 6, Dec. 2005.

[8] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888-905, Aug. 2000.

[9] M. Yeung, B. Yeo, and B. Liu, "Segmentation of videos by clustering and graph analysis," *Comput. Vis. Image Understand.*, vol. 71, no. 1,pp. 94–109, Jul. 1998.

[10] H.J. Zhang, A. Kankanhalli, S.W. Smoliar, Automatic partitioning of full-motion video, *Multimedia Systems*, vol.1, no. 1, 10-28, 1993.

[11] Y. Zhuang, Y. Rui, T. S. Huang, and S. Mehrotra, "Adaptive key frame extraction using unsupervisedclustering," *in Proc. of IEEE Int Conf on Image Processing*, pp.866-870.1998.

TABLE I. RECALL OF OUR METHOD

VIDEO	RECALL in %							
	k=2	*k=3*	*k=4*	*k=5*	*k=6*	*k=7*	*k=8*	*k=9*
V1	60	66.7	66.7	80	93.3	80	86.6	86.6
V2	55.6	72.2	77.7	55.6	77.7	77.7	93.3	88.8
V3	68.8	75	68.8	93.8	87.5	81.3	81.3	81.3
V4	23.1	23.1	46.2	100	100	84.6	100	100
V5	71.5	57.1	71.5	100	85.7	73.5	92.5	84.6
V6	40	53.3	66.6	80	80	80	80	80
V7	26.6	40	46.6	66.6	86.6	73.3	73.3	73.3
V8	48	40	64	68	72	76	72	52
V9	40	44	44	68	64	80	68	72
V10	31.6	68.8	63.2	77.5	81.5	78.5	82.1	80.3

TABLE II. PRECISION OF OUR METHOD

VIDEO	PRECISION in %							
	k=2	*k=3*	*k=4*	*k=5*	*k=6*	*k=7*	*k=8*	*k=9*
V1	100	90.1	100	75	82.4	70.6	72.2	72.2
V2	76.6	46.5	73.7	62.5	70	82.4	88.2	80
V3	91.7	85.7	73.3	75	73.7	68.4	65	62
V4	75	37.5	60	62	68.4	64.7	72.2	62
V5	100	100	71.5	73.7	70.6	66.6	76.4	61.1
V6	75	66.6	66.6	70	57	52	60	60
V7	100	42.8	47.1	62.5	68.4	68.8	91.6	68.8
V8	100	100	94.1	100	90	82.6	85.7	92.9
V9	76.9	57.9	68.8	54.8	59.3	58.4	58.6	56.3
V10	85.7	84.6	92.3	92.3	86.7	88.2	84.2	84.2

TABLE III.COMPARATIVE RESULTS USING F_1 MEASURE

METHOD	PARAMETER	VIDEO									
		V1	*V2*	*V3*	*V4*	*V5*	*V6*	*V7*	*V8*	*V9*	*V10*
OUR METHOD	***k=2***	75	64.4	78.6	35.3	83.4	52.2	42	64.9	52.6	46.2
	k=3	76.7	56.6	80	28.6	72.7	59.2	41.4	57.1	50	75.9
	k=4	80	75.6	71	52.2	71.5	66.6	46.8	76.2	53.7	75
	k=5	77.4	58.8	**83.4**	76.5	**84.9**	**74.7**	64.5	**81**	60.7	**84.3**
	k=6	**87.5**	73.6	80	81.2	77.4	66.6	76.4	80	61.6	84
	k=7	75	80	74.3	73.3	69.9	63	71	79.2	**67.5**	83.1
	k=8	78.7	**90.7**	72.2	**83.9**	83.7	68.6	**81.4**	78.3	63	83.1
	k=9	78.7	84.2	70.4	76.5	71	68.6	71	66.7	63.2	82.2
METHOD IN [7]	***T=0.05***	33.3	22.2	29.6	72.6	74	23	22.2	40	30	26.1
	T=0.1	38.5	41	40	61.4	72.8	32.2	47.6	41.8	43.1	40
	T=0.2	38.7	44	44.5	58.8	57.1	30	48	50	52.3	56.3
	T=0.3	46.2	54.3	43.7	60	45.2	35	51.9	56.6	51.2	61.1
	T=0.4	48.8	41.1	37.1	56.5	37.5	44.4	57.1	62.7	50.1	63.2
	T=0.5	48	39	41.1	47.3	33.7	40.7	60	57.5	48.9	68.1