

A Shot Clustering Based Algorithm for Scene Segmentation

Xuejun Wang
Jilin University
Changchun China
xjwang@jlu.edu.cn

Shigang Wang
Jilin University
Changchun China
Wangshigang@vip.sina.com.cn

Hexin Chen
Jilin University
Changchun China
chx@jlu.edu.cn

Moncef Gabbouj
Tampere University
of Technology
FIN-33101 Tampere
Moncef.Gabbouj@utu.fi

Abstract

A scene segmentation method utilizing both visual features and motion features of video is presented in this paper. Not only the visual similarity but also the motion consistency of shots within a scene is considered in clustering shots into scenes. In addition, a method to merge the over-segmented scenes is presented also. And the experimental results show the effectiveness of the proposed algorithms.

1. Introduction

With the rapid development of information technology, video data has become more and more important part of everyday life of human beings. So the content-based video analysis and retrieval has been developed to help people deal with the huge amount of video data. And the scene segmentation is the key problem for semantic video analysis.

A scene is consisted of several shots that are semantically related and temporally closer. As a high level unit, a scene has two characteristics: the first is visual similarity and the second is time locality. That is, shots within the same scene are likely to be visually similar and will be closer to each other temporally. Note that visually dissimilar shots are also likely to belong to the same scene as long as they are not far from each other; on the other hand, two visually similar shots will not be grouped into the same scene if the temporal distance between them is greater than a threshold. As we will see, these two attributes are important elements of most scene segmentation algorithms.

The first step of scene segmentation is shot detection. After the key frames are extracted and compared, the similar shots are merged to be a scene[1]. Then, the content of a scene can be denoted by several key frames which are simpler processed and needs less data.

Now, Most scene segmentation algorithms employ shot similarity comparison to extract scenes[2]. And, color histograms of key frames are most frequently used to compute shot similarity. In addition to color, motion content is also an important feature of shots.

The time-constrained clustering and the adaptive time grouping are two representative methods. In time-constrained clustering method[3], the shot similarity comparison is constrained within a fixed time window; the shot similarity of two shots is considered zero if their temporal distance exceeds the length of this time window. In adaptive time grouping method[4], the shot similarity is a varying function related the distance of two shots. Further more, a shot neighborhood coherence method is presented [5][6]. Firstly, divide the frames into several subblocks, then obtain the number of the most matched blocks, and the neighborhood coherence is defined in terms of the average smallest distance among the matched subblocks. According the coherence values, an overlapping links connecting similar shots is formed for scene segmentation. The disadvantages of the proposed methods above are that only the local color features and the DC elements of video are used.

In this paper, by adopting the overlapping links, the proposed algorithm employs both global color features and motion features in shot similarity comparison. And a scene merging method to handle over-segmented scenes is presented also. The experimental results show the effectiveness of the proposed algorithms.

2. Shot clustering based scene segmentation algorithm

2.1 Shot boundary detection and key frame extraction

We adopt the shot boundary detection method that utilizes macroblock type information in MPEG

This work is supported by national natural science foundation projects(60672100 , 60572068) and international corporation project(2005DFA10300)

compressed domain [7]. The idea is that at a shot boundary the shot content changes so that the motion compensation is not effective. Since the B frames immediately before the shot boundary are not similar to the referenced frame after the shot boundary, most macroblocks are forward compensated. Since the B frames immediately after the shot boundary is not similar to the referenced frame before the shot boundary, most macroblocks in the B frames are backward compensated. Likewise, most macroblocks in the P frames before the shot boundary are intra coded. By summing the number of predominant macroblock types of frames, the shot boundary can be accurately determined accordingly. In the case of key frame extraction, we choose the first and last frame as key frames of a shot.

The shot motion content represents the extent to which the shot content changes. Generally, the more motion there is in a shot, the more dissimilar the frames of this shot are; on the contrary, if a shot contains less motion, its frames tend to be less dissimilar. So, the motion of a shot can be expressed in terms of the dissimilarity of its frames. The motion feature Mot_z of shot z is defined as:

$$Mot_z = \frac{1}{b-a} \sum_{i=a}^{b-1} (1 - Sim(f_i, f_{i+1})) \quad (1)$$

Here, shot $z = \{f_a, f_{a+1}, \dots, f_b\}$, a and b are the first frame and last frame of z . Sim is defined as the similarity between two image frames.

$$Sim(x, y) = \frac{1}{width \times height} \times \sum_{k=0}^{L-1} \min(H_x(k), H_y(k)) \quad (2)$$

Here, $H_x(k)$ and $H_y(y)$ are color histograms of image frame x and y , respectively, and width and height are the sizes of a image frame.

Since similar shots within a scene have similar visual features and consistent motion characteristics, the shot similarity can be defined based on them. The shot similarity between shot i and j is defined as:

$$ShotSim(i, j) = W_v \times VisSim(i, j) + W_m \times MotSim(i, j) \quad (3)$$

Here, $VisSim(i, j)$ is the visual similarity between shot i and j , $MotSim(i, j)$ the motion similarity between them, and α and β are two weigh coefficients. So the visual similarity $VisSim(i, j)$ is defined as

$$VisSim(i, j) = \max[Sim(b_i, e_i), Sim(b_i, b_j), Sim(b_i, e_j), Sim(e_i, e_j)] \quad (4)$$

Here, b_i and e_i are the begin and end frame of shot i .

The motion similarity $MotSim(i, j)$ is defined as

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

Thus, if two shots are similar, the value of $MotSim$ will be very large.

In equation (3), shot similarity is defined as the sum of shot visual similarity and shot motion similarity; however, since the visual feature and the motion feature are two different attributes, it is not proper to combine them directly without normalizing them first. The normalization process ensures that entities from different domains are normalized to the different range. The normalizing process is as follows. First, compute the mean and standard deviation of $VisSim$:

$$\mu_v = \frac{1}{\sum_{k=1}^F (N-k)} \sum_{k=1}^F \sum_{i=0}^{N-k-1} VisSim(i, i+k) \quad (6)$$

$$\sigma_v = \sqrt{\frac{1}{\sum_{k=1}^F (N-k)} \sum_{k=1}^F \sum_{i=0}^{N-k-1} (VisSim(i, i+k) - \mu_v)^2} \quad (7)$$

Here, N is the number shots of a video, and F is the forward search range. The mean μ_m and standard deviation σ_m of $MotSim$ can be computed similarly.

After normalization, the values of color similarity and motion similarity are in the same range. To compute the combination similarity, two weights are set blow:

$$W_v = \frac{\sigma_v}{\sigma_v + \sigma_m} \quad (8)$$

$$W_m = \frac{\sigma_m}{\sigma_v + \sigma_m} \quad (9)$$

So, the final corresponding shot similarity is defined as:

$$ShotSim(i, j) = \frac{\sigma_v}{\sigma_v + \sigma_m} \times \frac{VisSim(i, j) - \mu_v}{\sigma_v} + \frac{\sigma_m}{\sigma_v + \sigma_m} \times \frac{MotSim(i, j) - \mu_m}{\sigma_m} \quad (10)$$

2.2 Proposed overlapping links method

Based on the above definitions, the proposed overlapping links method for scene segmentation is followed as:

Input: F (forward search range), R (backward search range), $F \geq R$.

Output: scene boundary.

$CtPreShot$ is the shot number before the current shot.

$CtFutureShot$ is the shot number from the current shot to the last shot.

Start: the first shot of video is set the current shot.

Step 1. Forward search

$$r = \min(R, PreShot)$$

$f = \min(F, CtFutureShot)$, find the nearest matching shot within F future shots. If such a shot exists, group the matched pair shots into the same scene, set the matched shot the current shot, and go to Step1. Otherwise go to step 2.

Step 2. Backward search

Set current shot number i, find the nearest matching shot within R preceding shots. If such a shot exists, group the matched pair shots into the same scene, set the matched shot the current shot, and go to Step1. Otherwise set $i=i-1$, repeat step 2, until the rth preceding shot is the current shot.

Step 3. Declare a scene boundary right after the current shot. The next shot becomes the current shot, and go to Step1.

An example of above steps is shown in Figure 1.

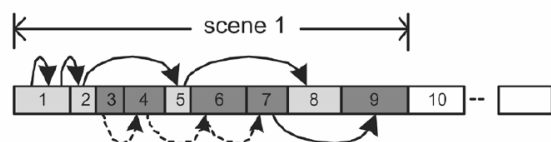


Figure 1. A scene segmentation($F=3, R=1$)

2.3 Scene segmentation postprocessing

The larger the values of F, R and shot similarity threshold are, the more scenes are obtained in our method, and hence an over-segmentation is resulted. On the contrary, if their values are set smaller, some real scenes might not be detected at all, so a under-segmentation is caused. Generally, we prefer over-segmentation to under-segmentation since the undetected scenes are difficult to recover. Therefore, we set the parameters to larger values and employ postprocessing to merge the over-segmented scenes afterwards. In merging scenes, we assume that a real scene shall consist of at least three shots; any detected scene that contains fewer than three shots is considered the false positive and merged into other scenes. From our observation of scenes of several Hollywood movies, we find our assumption is reasonable.

The process of scene merging is as follows. Find a scene C that has fewer than three shots, the immediate preceding scene C_p and the immediate subsequent

scene C_f of scene C. Scenes C_p and C_f shall each contain more than three shots.

Compute the similarities between C and C_p, C_f .

If $ShotSim_F \geq ShotSim_B$, merge C to C_f , otherwise merge C to C_p .

This process is operated until all scenes consist of at least three shots.

3. Experimental results

To prove the effectiveness of our method, we extract two video clips as test videos from two movies “Sleepless in Seattle” and “A Beautiful Mind”. In experiments, F is given the value of 3, and R the value of 2. The recall and precision is defined as:

$$Recall = \frac{N_c}{N_c + N_m} \times 100\%, \quad (11)$$

$$Precision = \frac{N_c}{N_c + N_f} \times 100\%$$

Here, N_c is the number of correctly detected shots, N_f the number of falsely detected shots, and N_m the number of missed shots.

The characteristics of test videos are summarized in Table 1.

Table 1 Characteristics of test videos

Video title	Sleepless in Seattle	A Beautiful Mind
Duration(min:sec)	32: 31	37: 43
Number of scenes	15	19
Number of shots	217	449
Number of abrupt shots	217	431
Number of gradual shots	0	18
Genre	Comedy/ Romance/ Drama	Drama/Mystery

The experimental results are shown in table 2 and 3. And Figure 2 and Figure 3 show the related shots in the scene.

As can be seen from the tables, as the value of threshold increases, more real scenes are detected, and the value of recall grows; but the falsely detected scenes also increases, and the value of precision decreases accordingly. We can also see that the segmentation result for “A Beautiful Mind” is not as good as that for “Sleepless in Seattle”; one of the reasons is that the former contains many gradual shot transitions and some scene boundaries overlaps with these scene boundaries, but in our shot boundary

detection stage, only abrupt shot boundaries are handled, and the gradual shot transitions are ignored.

Table 2 Scene segmentation results(“Sleepless in Seattle”)

Threshold	N_c	N_m	N_f	Precision (%)	Recall (%)
0.3	13	2	4	76.5	86.7
0.2	13	2	2	86.7	86.7

Table 3 Scene segmentation results(“A Beautiful Mind”)

Threshold	N_c	N_m	N_f	Precision (%)	Recall (%)
0.3	16	3	16	50.0	84.2
0.2	14	5	12	53.8	73.7

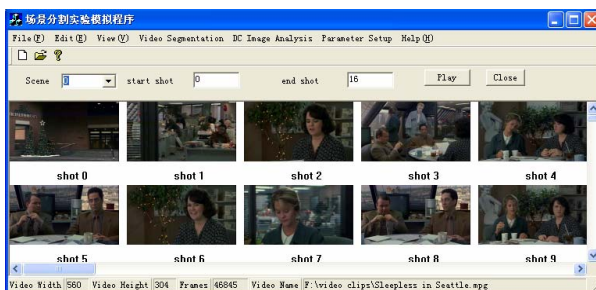


Figure 2. Scene segmentation results (“Sleepless in Seattle”)

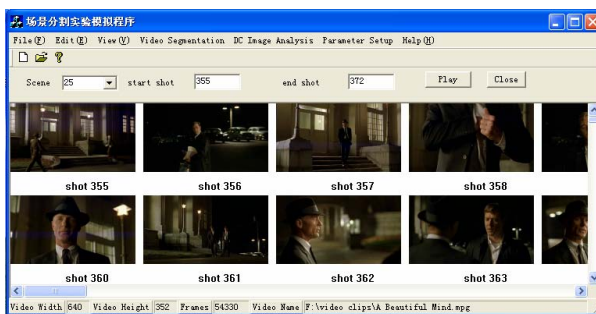


Figure 3. Scene segmentation results (“A Beautiful Mind”)

For comparison with literature[6], under the same conditions, the average recall and the average precision all raised greatly. The reason is that the numbers of false-detection and under-detection decreased after the

post-processing for the scenes which has no more than 3 shots.

4. Conclusion

In this paper we present a scene segmentation algorithm using visual and motion features. We also propose a scene merging method to handle over-segmentation. The experimental results show the effectiveness of our algorithm. However, a potential problem is that there might be still some false scenes even after scene merging using our method, so a better alternative is needed. Another point is that video properties alone are not adequate in scene segmentation; audio features should also be considered to improve the performance of scene segmentation algorithms further.

References

- [1]. Chong-Wah Ngo, Hong-Jiang Zhang, and Ting-chuen Pong, “Recent Advances in Content Based Video Analysis” [J]. International Journal of Image and Graphic, 2001, 1(3), pp. 445-469.
- [2]. Jeroen Vendrig, and Marcel Worring, “Systematic evaluation of Logical Story Unit Segmentation” [J]. IEEE Transactions on Multimedia, December 2002, 4(4), pp. 492-499.
- [3]. Minerva Yeung and Boon-Lock Yeo, and Bede Liu, “Segmentation of Video by Clustering and Graph Analysis” [J]. Computer Vision and Image Understanding, 1998,71(1),pp 94-109.
- [4]. Yong Rui, Thomas S. Huang, and Sharad Mehrotra, “Constructing Table-of-content for Videos” [J]. Multimedia Systems 1999,7(5),pp 359–368.
- [5]. Alan Hanjalic, Reginald L. Lagendijk, and Jan Biemond, “Automatic High-Level Movie Segmentation for Advanced Video-Retrieval Systems” [J]. IEEE Transactions on Circuits and Systems for Video Technology, June 1999, 9(4),pp 580 – 588.
- [6]. Wallapak Tavanapong, Junyu Zhou, “Shot Clustering Techniques for Story Browsing”[J]. IEEE Transactions on Multimedia, August 2004, 6(4),pp 517–527.
- [7]. Soo-Chang Pei, Yu-Zuog, “Efficient MPEG Compressed Video Analysis Using Macroblock Type Information” [J]. IEEE Transactions on Multimedia, December 1999, 1(4),pp 312-33.