Available online at www.sciencedirect.com
ScienceDirect
Pattern Recognition 41 (2008) 1056–1065
PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY
www.elsevier.com/locate/pr

Movie scene segmentation using background information

Liang-Hua Chen[a,*], Yu-Chun Lai[b], Hong-Yuan Mark Liao[c]

[a]*Department of Computer Science and Information Engineering, Fu Jen University, Taipei, Taiwan*
[b]*Department of Computer Science, Chiao Tung University, Hsinchu, Taiwan*
[c]*Institute of Information Science, Academia Sinica, Taipei, Taiwan*

Received 15 June 2006; received in revised form 24 July 2007; accepted 31 July 2007

Abstract

Scene extraction is the first step toward semantic understanding of a video. It also provides improved browsing and retrieval facilities to users of video database. This paper presents an effective approach to movie scene extraction based on the analysis of background images. Our approach exploits the fact that shots belonging to one particular scene often have similar backgrounds. Although part of the video frame is covered by foreground objects, the background scene can still be reconstructed by a mosaic technique. The proposed scene extraction algorithm consists of two main components: determination of the shot similarity measure and a shot grouping process. In our approach, several low-level visual features are integrated to compute the similarity measure between two shots. On the other hand, the rules of film-making are used to guide the shot grouping process. Experimental results show that our approach is promising and outperforms some existing techniques.
© 2007 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.

Keywords: Video content analysis; Video segmentation; Shot clustering; Scene extraction

1. Introduction

The advances in low cost mass storage devices, higher transmission rates, and improved compression techniques have led to the widespread use and availability of digital video. Video data offer users of multimedia systems a wealth of information and also serve as a data source in many applications including digital libraries, publishing, entertainment, broadcasting and education. However, because of the large amount of data and unstructured format, efficient access to videos is not an easy task. To make the original video data in a database available for browsing and retrieval, it must be analyzed, indexed and reorganized. The suitably organized video data have the right structure for non-linear browsing and for content-based retrieval through large amount of data.

To derive a structured representation of a video, each video sequence is usually decomposed into a hierarchical structure using shots and scenes as construction units. A shot is a sequence of frames that were continuously captured by the same camera. The definition of a scene is not as straightforward and usually refers to a common environment that is shared by a group of consecutive shots [1]. For example, we could see many consecutive shots (taken by different cameras) share the similar visual content because they are produced in the same environment such as a meeting room or a sports field. Generally, a video scene is basically a story unit that shows the same objects and allows one event to be fully presented. Shots in a video are analogous to words in a language in that they convey little semantic information in isolation. On the other hand, scenes reflect the dramatic and narrative structure of a video. One scenario is that humans remember different events after having watched a digital movie. Such an event can be a dialogue, an action scene, or a group of shots unified by location. Therefore, scene extraction is the first step toward greater semantic understanding of video content. The objective of video scene extraction is to cluster video shots into several groups, such that the shots within each group are related by some common aspects.

Nowadays, there exist various types of videos, including movies, news casts, sitcoms (situation comedies), commercials,

* Corresponding author. Tel.: +886 2 2905 3706; fax: +886 2 2341 5838.
E-mail address: lchen@csie.fju.edu.tw (L.-H. Chen).

0031-3203/$30.00 © 2007 Pattern Recognition Society. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.patcog.2007.07.024

sports, and documentary videos. Some of them have "story units" such as movies, while others (e.g., sports) do not. In this work, we concentrate on movies. Usually, there are two steps to extract video scene structures after shot boundary detection. The first step is to represent visual content of one shot and to define the similarity measure between two shots. The second is to group correlated consecutive shots into a scene. The compact representation of video shot content for shot similarity measure remains one of the most challenging issues. In our approach, the similarity measure is based on the background information obtained through a mosaic technique. By aligning all images of a video sequence onto a common reference frame, the mosaic technique is able to generate the static background of a video scene. Although the background image alone is not an effective video content representation for some tasks such as video retrieval, it is sufficient for the task of scene extraction. It is also worth noting that the way shots are grouped into a scene generally depends on the type of scene under analysis as well as on the video genre. The scenes of a TV-news report are different from the scenes of a basketball game, a documentary, or a movie. Hence, it is important to aggregate the shots by considering a model for the scene. In our approach, we exploit cinematic rules to devise a shot grouping algorithm. The rest of this paper is organized as follows. In the next section, we review some related works and explain the motivation for our approach. In Section 3, we describe how to generate the mosaic image of a video shot. Then, a new similarity measure between two shots is introduced in Section 4. The proposed video shot grouping algorithm is described in Section 5. The performance evaluation of our approach is reported in Section 6. Finally, some concluding remarks are given in Section 7.

2. Background and motivation

Numerous methods for shot boundary detection have been proposed [2]. While shots are marked by physical boundaries, scenes are marked by semantic boundaries. Hence, scene boundary detection is far more difficult than shot boundary detection. Current techniques for video scene extraction can be broadly classified into three categories. The first groups shots that are visually similar and temporally close into a scene [3–12]. In these approaches, video shots are mostly represented by a set of selected key-frames. Low level features such as color, texture, motion, and shape are extracted directly from key-frames. Then, a classic clustering algorithm or simple peak detection is used to detect scene boundaries. However, the limitation of key-frame-based shot representation is that a frame taken from a shot often fails to represent the dynamic contents within the shot. When a sequence of shots is considered as a scene, it is often because the shots are correlated by the same environment rather than by visual similarity in terms of key-frames. In the second category, emphasis is put on the integration of audio and visual information. Various methods have been proposed to determine a shot boundary as a scene boundary if the visual and audio content change simultaneously [13,14]. However, how to determine scene boundary efficiently still remains a difficult issue since the relationship between audio segments and visual shots is complicated. The third category exploits characteristics embedded in specific video domains, such as sports and news casts [15–18]. Since this approach is based on specific application models, it normally achieves high accuracy. The main drawback is that an *a priori* model needs to be constructed beforehand for each application. The modeling process is time consuming and requires good domain knowledge and experience.

In this paper, we present a scheme for automatic video scene extraction based on visual information only. When a scene is composed of several shots, there will be at least one aspect in common between these shots. To measure the common aspects between two shots, we define the shot similarity using background information. Our approach is based on the following observations: (i) each frame of a video can be divided into foreground objects and background scenery; (ii) in most cases, while objects may move, appear and disappear, the background in a scene does not change significantly. These observations suggest that it would be more effective to focus on the background to detect a scene, which is a collection of shots unified by a common locale. However, in a single frame, most of the physical location information is invisible since it is concealed by the foreground objects. One solution to this problem is to use a mosaic technique. A mosaic is a panoramic image obtained by aligning all images of a video sequence onto a common reference frame [19,20]. The resulting mosaic is an efficient and compact representation of a collection of frames [21]. In the case of a static mosaic, the moving objects blur out or disappear. Only the stationary objects and the background are displayed in the constructed mosaic. Therefore, in our approach, we use a mosaic technique to reconstruct the background scene of each video frame. Then, we compute the color and texture distribution of all the background images in a shot to determine the shot similarity. Since our method does not depend on the content of key-frames only, it is able to fully exploit the spatiotemporal information contained in video sequences.

3. Mosaic construction

A shot is the basic unit for video indexing. To facilitate subsequent video analysis, in our system, the original video sequences are segmented into shots [22]. We have also proposed an algorithm to align frames from a video shot to build the static mosaic of the background [23]. Here, we briefly describe the generation of mosaics, and refer the reader to Ref. [23] for further details.

We first need to derive the transformation between partially overlapping frames. Assuming background motion (due to camera motion) is the dominant motion in the video, the image motion of the majority of scene points can be approximated by the following transformation model:

$$u(x, y) = a_1 x + a_2 y + a_3, \tag{1}$$

$$v(x, y) = -a_2 x + a_1 y + a_4, \tag{2}$$

where $(u(x, y), v(x, y))$ is the motion vector at image position (x, y). This model is a special case of an affine model

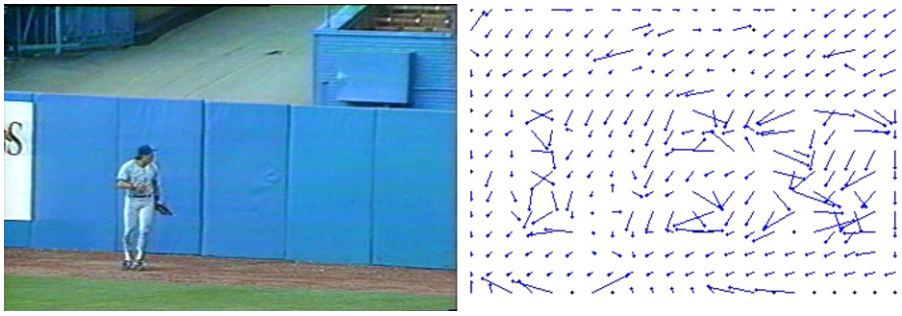

Fig. 1. A frame of a baseball game sequence and its corresponding motion vectors.

Fig. 2. Sampled frames of a "Terminator II" sequence.

with six parameters. However, experimental results show that our model is better than the general affine model. This is because we impose more constraints on the model's parameters to avoid some undesirable transformations implied in affine model such as skewing. In this step, we need to obtain the motion vectors (or displacements) between successive frames. Existing approaches are based on feature matching or optical flow computation. These techniques are computationally intensive. To reduce the processing time, we use the motion vectors encoded in the MPEG-1 [24] video stream directly. Fig. 1 shows a frame of a baseball game sequence and its corresponding motion vectors.

Given the motion vector, the image motion parameters can be estimated by a robust regression technique called least-median-squares [25]. Let the motion vectors of each frame be denoted as $\{[u_1(x_1, y_1), v_1(x_1, y_1)], \ldots, [u_n(x_n, y_n), v_n(x_n, y_n)]\}$. The least-median-squares method can be described as

$$\min_{\hat{a}} \{\text{median}[(a_1 x_i + a_2 y_i + a_3 - u_i)^2 + (-a_2 x_i + a_1 y_i + a_4 - v_i)^2]\}, \quad (3)$$

where $\hat{a} = (a_1, a_2, a_3, a_4)$ is the parameter vector. One distinctive property of the algorithm is that it can tolerate up to 50% of the outliers in the data set, i.e., half of the data set can be arbitrary without significantly effecting the regression result. Therefore, this technique can robustly estimate the motion of the majority of scene points (background) and will not be biased by the minority scene points (moving object).

Once the transformations between successive frames have been determined, the transformations can be composed to obtain the alignment between any two frames of the video sequence, and in particular, between the current frame and the reference frame. In most cases, we choose the first frame of the sequence as a reference. After all the frames have been aligned to a reference frame, the next step is the selection of pixels to be put into the resulting mosaic. The gray value of each pixel of the mosaic is computed by applying an appropriate temporal operator to the aligned frames. The temporal average operator is effective in removing temporal noise, but the moving objects appear blurred, with "ghost-like" traces in the resulting mosaic. The temporal median operator can remove the moving objects and produce a static mosaic of the background, but it is computationally expensive and is an off-line process. Therefore, we propose a novel scheme that is both effective in deleting moving objects and feasible for the on-line creation of panoramic images. Our approach is based on the observation that each overlapping pixel of the aligned frames will fall into one of the two categories: background or moving object. Since background motion is the dominant motion of video and we want to build the mosaic of background, we select the pixel that appears most frequently in the temporal domain. In Fig. 2, some sampled frames of "Terminator II" sequence are shown. We observe that the camera has large movement in this sequence. Fig. 3 shows the static mosaic where only the background of

Fig. 3. The static mosaic of the "Terminator II" sequence.

the scene is visible. Using the background mosaic, the background scene image of each video frame is reconstructed. In the subsequent processing, we focus on these background image sequences.

4. Shot similarity measure

This section describes the color and texture features used in our work and how these features are taken into account in the formulation of shot similarity measure. A major requirement for shot similarity measure is to define a content representation that captures the common aspects or characteristics of the shot. One common method is to select one key-frame from the shot and use the image features of that key-frame as an abstract representation of the shot. For shots with fast changing content, one key-frame per shot is not adequate. Besides, the content description it provides varies significantly with the key-frame selection criterion. To avoid these problems, a more feasible approach is to consider the visual content of all the frames within a shot for shot representation.

Color is one of the most widely used visual features in video content analysis. Most scene extraction algorithms compare color histograms between key-frames to determine the shot similarity measure. The histogram-based approach is relatively simple to implement and provides reasonable results. However, due to its statistical nature, the color histogram cannot capture the spatial layout information of each color. When the image collection is large, two different content images are likely to have quite similar histograms. To remedy this deficiency, in our approach, the distribution state of each color in the spatial (image) domain is also taken into account.

The color histogram of an image is constructed by counting the number of pixels of each color. The main issues regarding the construction of color histograms involve the choice of color space and quantization of the color space. The *RGB* color space is the most common color format for digital images, but it is not perceptually uniform. Uniform quantization of *RGB* space yields perceptually redundant bins and perceptual holes in the color space. Therefore, non-uniform quantization may be needed. Alternatively, *HSV* (hue, saturation, intensity) color space is chosen since it is nearly perceptually uniform. Thus, the similarity between two colors is determined by their proximity in the *HSV* color space. When a perceptually uniform color space is chosen, uniform quantization may be appropriate. Since the human visual system is more sensitive to hue than to saturation and intensity [26], H should be quantized finer than S and V. In our implementation, the hue is quantized into 20 bins. The saturation and intensity are each quantized into 10 bins. This quantization provides 2000 ($=20\times10\times10$) distinct colors (bins), and each bin with non-zero count corresponds to a *color object*.

Since we are interested in the whole shot rather than single image frame, only one histogram is used to count the color distribution of all background images within a shot. Then, each bin of the resulting histogram is divided by the number of frames in a shot to obtain the average histogram. Next, several spatial features are calculated to characterize the distribution state of each color object in each image frame. Assuming a set of pixels $S=\{(x_1, y_1), \ldots, (x_n, y_n)\}$ belong to color object c_i, k is the image size, and m is the total number of 4-connected pixels in S. Then, we define

(i) the density of distribution as

$$f_{i1}=\frac{n}{k},$$

(ii) the compactness of distribution as

$$f_{i2}=\frac{m}{n},$$

(iii) the scatter as

$$f_{i3}=\frac{1}{n\sqrt{k}}\sum_{j=1}^{n}\sqrt{(x_j-x_\mu)^2+(y_j-y_\mu)^2},$$

where $x_\mu=(1/n)\sum_{i=1}^{n}x_i$ and $y_\mu=(1/n)\sum_{i=1}^{n}y_i$.

To define the fourth feature, the image is partitioned equally into p blocks of size 16×16. A block is *active* if it contains some subsets of S. Let the number of active blocks in the image frame be q, we define

(iv) the ratio of active block as

$$f_{i4}=\frac{q}{p}.$$

After the spatial features of all images are computed, we take average of these values, respectively. Let $\overline{f_{i1}}$, $\overline{f_{i2}}$, $\overline{f_{i3}}$, and $\overline{f_{i4}}$ be the average feature values of a color object c_i in a shot, for two color objects c_i and c_j, the difference in the spatial distribution within a shot is defined as

$$D_s(c_i,c_j)=\tfrac{1}{4}(|\overline{f_{i1}}-\overline{f_{j1}}|+|\overline{f_{i2}}-\overline{f_{j2}}|+|\overline{f_{i3}}-\overline{f_{j3}}|+|\overline{f_{i4}}-\overline{f_{j4}}|). \quad (4)$$

Texture refers to the visual patterns that have properties of homogeneity that do not result from the presence of only a single one color or intensity only. It contains important information about the structural arrangement of objects and their relationship to the surrounding environment. We define the coarseness of an image's texture in term of the distribution density of the edges. The Canny edge detector is used to extract edges from an image. The edge location indicates sharp intensity variation. Psychophysical experiments have shown that the human visual system is sensitive to the high-frequency regions of an image such as edges. The detected edge image is partitioned into a set of 16×22 blocks. A block is *textured*, if the number of edge points in the block is greater than a threshold (=30, in our setting). Then, we can compute the ratio of the textured block of each image and its average value over a shot. The texture similarity between two shots is determined by the minimum of the two average values. In Fig. 4, two images with different level of texture coarseness are shown. Fig. 5 shows the detected edge image partitioned into a set of 16×22 blocks.

Fig. 4. Two images with different texture coarseness.

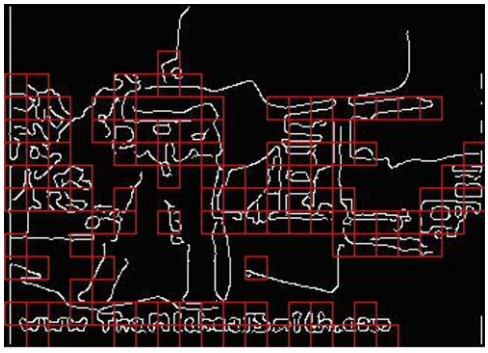

Fig. 5. The detected edge image is partitioned into a set of 16×22 blocks.

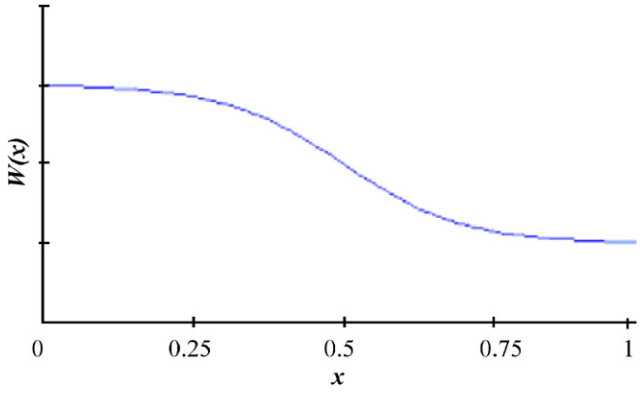

Fig. 6. Sigmoid function with parameters $a = 10$ and $b = -5$.

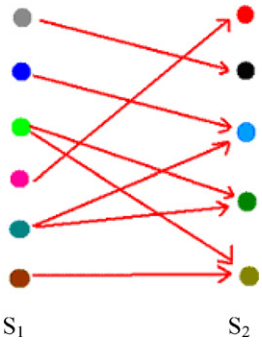

Fig. 7. Finding similar color object pairs.

Histogram intersection is a popular similarity measure used for color-based image matching [27]. It yields the number of pixels that have same color in two images. In our work, we extend this idea to shot similarity measure. Let A, B be the set of all color objects in shot S_1 and S_2, respectively, for a given $u \in A$, its similar color object in B is some $v \in B$ such that $\|u - v\| < \varepsilon$, where $\|u - v\|$ denotes the Euclidean distance between u and v in the *HSV* color space, and ε is a threshold (=3, in our setting). Then, (u, v) is called a *similar color pair*. Let $\Omega = \{(u, v) | (u, v) \in A \times B, (u, v)$ is a similar color pair$\}$, the shot similarity measure between S_1 (with the average histogram $\overline{H_1}$) and S_2 (with the average histogram $\overline{H_2}$) is defined as

$$\text{ShotSim}(S_1, S_2) = \frac{1}{k} \sum_{(u,v)\in\Omega} \{W(D_s(u, v)) \min(\overline{H_1}(u), \overline{H_2}(v))\} + w_t \times \min(t_1, t_2), \quad (5)$$

where k is the image size; t_1 and t_2 are the average ratios of textured block for shot S_1 and S_2, respectively; w_t is the weight of texture feature; D_s is the difference in spatial features as defined in Eq. (4); and W is a weight function defined as

$$W(x) = \frac{1}{1 + e^{a \times x + b}}.$$

The weight function W is the general form of the sigmoid function which is frequently used in neural networks computation [28], where a and b are parameters. In our work, it is used to fuse the spatial distribution information with a histogram. The construction of this weight function is motivated by the psychophysical observation that the effect of spatial distribution on human perception is progressive [29]. Only when the difference in spatial features is greater than a threshold, humans perceive significant visual variation. The property of the sigmoid function fulfills this requirement. In our system, we set $a = 10$ and $b = -5$. As shown is Fig. 6, the function's value becomes significantly small for $x > 0.75$.

It is noted that a given color object in shot S_1 may have more than one similar color objects in shot S_2 as illustrated in Fig. 7. To avoid the overlapping contribution in calculating shot similarity, after each step of $\min(\overline{H_1}(u), \overline{H_2}(v))$, $\overline{H_1}(u)$ and $\overline{H_2}(v)$ are all subtracted by $\min(\overline{H_1}(u), \overline{H_2}(v))$.

5. Scene extraction

This section describes the details of applying the cinematic model to group correlated consecutive shots into one scene. Movies directors, while filming scenes, also control the pace of a film in order to sustain the viewers interest. One important factor known to influence the pace of a movie is the Montage. Montage usually refers to a model that defines the usage of editing effects to assemble the shots into a smooth sequence in physical time and/or space and in the psychological association of ideas [30]. In order to convey an idea that has a strong resonance with viewers, Montage is widely used as the basis to model scenes. In most situations, Montage can be simplified as a set of cinematic rules. Commonly used rules include [31,32]:

- Parallel rule: It is used to compose scenes involving multiple themes, where shots from different themes are shown alternately. This rule is frequently used to model interactions between two parties such as conversations, hunting, and chasing.
- Concentration rule: It starts with long distance shot, and progressively zooms into close-up shots of the main objects. It is used to introduce the main objects and their context.
- Enlargement rule: It is the reverse of the concentration rule. It is used to introduce the context of the current main object before switching to other objects that possibly share a similar context. Thus, it typically signals the transition to a new scene.
- Serial Content rule: It is used to model scenes that preserve the continuity of location, time, space, and topic.

Together, these rules can be used to model most types of scenes. We use this knowledge to develop a two-pass algorithm for scene boundary detection suitable for feature movies.

The first pass of the algorithm deals with the detection of potential scene boundaries. This is achieved by computing the shot similarity between two consecutive shots (see Fig. 8). Given a sequence of shots $\{S_1, \ldots, S_n\}$, if ShotSim $(S_{i+1}, S_i) < T_\mu$ then a potential scene boundary is detected. The threshold T_μ is empirically set to be the mean of all shot similarities, i.e.,

$$T_\mu = \frac{1}{n-1} \sum_{i=1}^{n-1} \text{ShotSim}(S_{i+1}, S_i). \quad (6)$$

Any two adjacent potential scene boundaries delineate a *candidate scene*. Thus, all shots are grouped into a set of candidate scenes (see Fig. 9).

The above algorithm assumes that all shots in a scene take place in the same location and share many common backgrounds. This is true for most of the scenes composed using the serial content rule and parallel rule (such as a conversation in a studio). However, the algorithm tends to oversegment the more complex scenes composed using parallel or concentration/enlargement rules. In the outdoor chasing scene (also modeled by parallel rule), the escapee and pursuer shots are shown alternatively. But the background of both types of shots may be different. One important component of a scene defined by the concentration/enlargement rule is the close-up shot where more than half of the frame is occupied by the foreground object. Because of the limitations of the mosaic technique, the background information can not be recovered completely. Thus, the close-up shot will be identified as belonging to another scene. To handle such scenes, we need to merge the candidate scenes further.

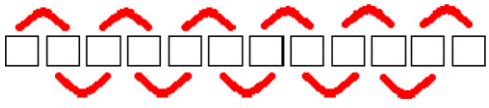

Fig. 8. Computing the similarity between every two adjacent shots.

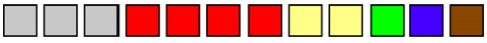

Fig. 9. Shots are grouped into several candidate scenes.

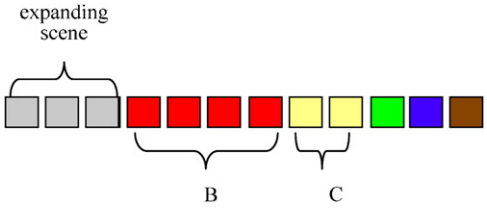

Fig. 10. An expanding scene and the two subsequent scenes.

Let $G_1 = \{S_{11}, \ldots, S_{1m}\}$ and $G_2 = \{S_{21}, \ldots, S_{2n}\}$ be two candidate scenes consisting of m and n shots, respectively. The scene similarity between G_1 and G_2 is

$$\text{SceneSim}(G_1, G_2) = \frac{1}{m \times n} \sum_{i=1}^{m} \sum_{j=1}^{n} \text{ShotSim}(S_{1i}, S_{2j}). \quad (7)$$

Two scenes G_1 and G_2 are *visually similar*, if SceneSim $(G_1, G_2) > T_\mu - \sigma/2$, where T_μ (defined in Eq. (6)) and σ are, respectively, the mean and variance of all similarity measures between every two adjacent shots. Because several simple scenes are merged into one complex scene, this threshold should be smaller than that of the first pass. Given a sequence of candidate scenes, the second pass of the scene extraction algorithm mainly consists of the following merging process:

Step 1: Set the *expanding scene* to be the first scene.

Step 2: Compare the expanding scene with two subsequent scenes B and C (see Fig. 10).

Step 3: If the expanding scene and scene C are visually similar, then

- merge the expanding scene, scene B and scene C into one scene;
- set the expanding scene to be the merged scene;
- go to Step 2.

Step 4: If the expanding scene and scene B are visually similar, then

- merge the expanding scene and scene B into one scene;
- set the expanding scene to be the merged scene;
- go to Step 2.

Step 5: Set the expanding scene to be scene B and go to Step 2.

This process is repeated until no more scenes can be merged. It is noted that B and C always refer to the two scenes immediately following the current expanding scene and they are always updated in Step 2.

6. Experimental results

Six test videos in MPEG-1 format were used to evaluate our scene extraction algorithm: one home video and five full-length movies. The home video "Lgerca_lisa_1" is an MPEG standard test video with ground truth from original video provider. The genres of movies include action, drama, comedy, thriller, and music. The testing with five different genres of movies would ensure that the overall performance of the algorithm is not biased toward a specific movie kind. To get the ground truth of other videos, two graduate students were invited to watch the movies and then asked to give their own scene boundaries. The intersection of their segmentation was used as the ground truth for the experiments. In movies, there is usually not a concrete or clear boundary between two adjacent scenes due to editing effects. Therefore, we follow Hanjalic's evaluation criterion [5]: if the detected scene boundary is within four shots from the boundary detected manually, this boundary is counted as a correct boundary. Basic information about the test videos and the experimental results are shown in Table 1.

As shown in Table 1, our algorithm correctly extracts all the six scenes of the first video (Lgerca_lisa_1) without any missed or false detection. Fig. 11 shows the shot grouping result of the first video, where the first frame of each shot is displayed. Fig. 12 shows one extracted scene (consisting of 4 shots) from another video "Little Voice". This scene is composed by the parallel rule and has different backgrounds in the first and second shot. Our shot group algorithm is able to identify both shots as belonging to the same scene. However, our algorithm has some false and missed detections in the other test videos. The false detection is due to the significant change of lighting such as explosions and flashing lights. This could perhaps be improved by a more sophisticated visual similarity measure. The missed detection is mainly caused by inappropriate setting of the merging threshold. As the threshold depends on the variance (σ) of all similarity measures between every two adjacent shots, a too large σ results in under-segmentation of the video. This type of error occurs in some video scenes with very inconsistent pace.

For performance comparison, we implement the well-known scene extraction algorithm proposed by Yeung et al. [3]. In their approach, a video sequence is first segmented into shots. Then, a time-constrained clustering algorithm is used to group visually similar and temporally adjacent shots into clusters. The visual similarity between two shots is measured by comparing color histograms of the respective key-frames. Finally, a scene transition graph is constructed based on the clusters, and cutting edges are identified to construct the scene structure. For fair comparison, the parameters of Yeung's algorithm are tuned to achieve the best performance. To measure the performance quantitatively, two metrics are used:

$$\text{recall} = \frac{D}{D + MD}, \quad \text{precision} = \frac{D}{D + FD},$$

where D is the number of scene boundaries detected correctly, MD is the number of missed detection and FD is the number of false detection. Table 2 shows the performance comparison.

As "Lgerca_lisa_1" is an MPEG standard test video, some related works have also used it as test video. According to the reports of Lin [7] and Ngo [8], their approaches have three and two false detections, respectively.

7. Conclusion

We have proposed a mosaic-based algorithm for extracting scene structures from digital movies. Our approach is based on the idea that shots belonging to one particular scene often have similar backgrounds. Using a mosaic technique, the background of each video frame can be recovered. The color feature and texture feature of each background image are integrated to compute shot similarities. Based on the movie making model, our algorithm is able to group correlated shots into one scene. The computation is costly, but the spatiotemporal information

Table 1
Accuracy measures for the six test videos

Video title	Genre	Duration (in min)	No. of Scenes (ground truth)	Correct detection	Missed detection	False detection
Lgerca_lisa_1	Home video	15	6	6	0	0
Dungeons & Dragons	Action	107	66	54	12	6
Little Voice	Drama	96	141	110	31	24
Hot Chick	Comedy	104	104	71	33	52
Bugs	Thriller	82	76	58	18	26
Walk the Line	Music	136	118	70	48	35

Fig. 11. Shot grouping result of "Lgerca_lisa_1".

of videos is fully exploited to achieve scene extraction. Experimental results show that the proposed approach works reasonably well in detecting most of the scene boundaries. Compared with some existing techniques [3,7,8], our approach is promising. Our approach can be applied directly to organize videos and can be utilized to provide browsing/retrieval facilities to the users. As scene is a subject concept to reflect human perception, our future work will focus on investigating an adaptive technique to perform user-oriented scene extraction. The proposed shot similarity measure does not use motion feature to capture temporal variation in a video. Thus, another future research issue will be the integration of motion information into the proposed shot similarity measure for other tasks such as video retrieval.

Fig. 12. A scene extracted from “Little Voice”.

Table 2
Performance comparison for scene extraction

Video title	Our approach		Yeung’s approach	
	Recall (%)	Precision (%)	Recall (%)	Precision (%)
Lgerca_lisa_1	100	100	100	85.7
Dungeons & Dragons	81.8	90.0	74.2	81.6
Little Voice	78.0	82.1	72.3	72.3
Hot Chick	68.3	57.7	54.8	50.9
Bugs	76.3	69.0	59.2	64.3
Walk the Line	59.3	66.7	47.5	56.0

References

[1] F. Beaver, Dictionary of Film Terms, Twayne Publishing, New York, 1994.
[2] I. Koprinska, S. Carrato, Temporal video segmentation: a survey, Signal Process.: Image Commun. 16 (2001) 477–500.
[3] M. Yeung, B.L. Yeo, Segmentation of video by clustering and graph analysis, Comput. Vision Image Understanding 71 (1) (1998) 97–109.
[4] J.R. Kender, B.-L. Yeo, Video scene segmentation via continuous video coherence, in: IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, 1998, pp. 367–373.
[5] A. Hanjalic, R.L. Lagendijk, J. Biemond, Automated high-level movie segmentation for advanced video-retrieval systems, IEEE Trans. Circuits Syst. Video Technol. 9 (4) (1999) 580–588.
[6] Y. Rui, T.S. Huang, S. Mehrotra, Constructing table-of-content for video, Multimedia Syst. 7 (5) (1999) 359–368.
[7] T. Lin, H.-J. Zhang, Q.-Y. Shi, Video content representation for shot retrieval and scene extraction, Int. J. Image Gr. 1 (3) (2001) 507–526.
[8] C.-W. Ngo, T.-C. Pong, H.-J. Zhang, R.T. Chin, Motion-based video representation for scene change detection, Int. J. Comput. Vision 50 (2) (2002) 127–142.
[9] B.T. Truong, S. Venkatesh, C. Dorai, Scene extraction in motion picture, IEEE Trans. Circuits Syst. Video Technol. 13 (1) (2003) 5–15.
[10] W. Tavanapong, J. Zhou, Shot clustering techniques for story browsing, IEEE Trans. Multimedia 6 (4) (2004) 517–527.
[11] Z. Rasheed, M. Shah, Detection and representation of scenes in videos, IEEE Trans. Multimedia 7 (6) (2005) 1097–1105.
[12] Y. Zhai, M. Shah, A general framework for temporal video scene segmentation, in: International Conference on Computer Vision, Beijing, China, October 2005, pp. 1111–1116.
[13] J. Juang, Z. Liu, Y. Wang, Integration of audio and visual information for content-based video segmentation, in: International Conference on Image Processing, Chicago, 1998, pp. 526–530.
[14] H. Sundaram, S.-F. Chang, Video scene segmentation using video and audio features, in: International Conference on Multimedia and Expo, New York, 2000, pp. 1145–1148.
[15] A. Merlino, D. Morey, M.T. Maybury, Broadcast news navigation using story segmentation, in: ACM International Conference on Multimedia, Seattle, WA, November 1997, pp. 381–391.
[16] Y. Ariki, M. Kumano, K. Tsukada, Highlight scene extraction in real time from baseball live video, in: ACM International Workshop on Multimedia Information Retrieval, Berkeley, CA, November 2003, pp. 209–214.
[17] L. Xie, P. Xu, S.-F. Chang, A. Dirakaran, H. Sun, Structure analysis of soccer video with domain knowledge and hidden Markov models, Pattern Recognition Lett. 25 (7) (2004) 767–775.
[18] Y. Zhai, A. Yilmaz, M. Shah, Story segmentation in news using visual and text cues, in: International Conference on Image and Video Retrieval, Singapore, July 2005, pp. 92–102.
[19] M. Irani, P. Anandan, J. Bergen, R. Kumar, S. Hsu, Efficient representations of video sequences and their applications, Signal Process.: Image Commun. 8 (4) (1996) 327–351.
[20] M. Irani, P. Anandan, Video indexing based on mosaic representations, Proc. IEEE 86 (5) (1998) 905–921.
[21] A. Aner, J. Kender, Video summaries and cross-referencing through mosaic-based representation, Comput. Vision Image Understanding 95 (2) (2004) 201–237.
[22] L.-H. Chen, C.-W. Su, H.-Y. Liao, C.-C. Shih, On the preview of digital movies, J. Visual Commun. Image Representation 14 (3) (2003) 357–367.
[23] L.-H. Chen, Y.-C. Lai, C.-W. Su, H.-Y. Liao, Extraction of video object with complex motion, Pattern Recognition Lett. 25 (11) (2004) 1285–1291.
[24] D.L. Gall, MPEG: a video compression standard for multimedia applications, Commun. ACM 34 (4) (1991) 46–58.
[25] P.J. Rousseeuw, A.M. Leroy, Robust Regression & Outliers Detection, Wiley, New York, 1987.

[26] X. Wan, C.-C. Jay Kuo, A new approach to image retrieval with hierarchical color clustering, IEEE Trans. Circuits Syst. Video Technol. 8 (5) (1998) 628–643.
[27] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comput. Vision 7 (11) (1991) 11–32.
[28] S. Haykin, Neural Networks: A Comprehensive Foundation, Prentice-Hall, New Jersey, USA, 1999.
[29] B.B. Boycott, Color Vision, Cambridge University Press, Cambridge, UK, 2001.
[30] L. Giannetti, Understanding Movie, Prentice-Hall, Englewood Chiffs, NJ, 1987.
[31] G. Davenport, T.A. Smoth, N. Princever, Cinematic primitives for multimedia, IEEE Comput. Graphics Appl. 11 (4) (1991) 67–74.
[32] T.S. Chua, L.Q. Ruan, A video retrieval and sequencing system, ACM Trans. Inf. Syst. 13 (4) (1995) 373–407.

About the Author—LIANG-HUA CHEN received the B.S. degree in information engineering from National Taiwan University, Taipei, Taiwan in 1983. After completing two years military service, he joined the Institute of Information Science, Academia Sinica, Taipei, Taiwan, as a research assistant. He received the M.S. degree in computer science from Columbia University, New York, in 1988, and the Ph.D. degree in computer science from Northwestern University, Evanston, IL, in 1992. From March 1992 to July 1992, he was a senior engineer at Special System Division, Institute for Information Industry, Taipei, Taiwan. He is currently a full professor in the Department of Computer Science and Information Engineering at the Fu Jen University, Taipei, Taiwan. Dr. Chen's research interests include computer vision and pattern recognition.

About the Author—YU-CHUN LAI received the B.S. degree in information science from Chinese Culture University, Taipei, Taiwan, in 2002, and the M.S. degree in computer science and information engineering from Fu Jen University, Taipei, Taiwan, in 2004. He is currently pursuing the Ph.D. degree in the Department of Computer Science, National Chiao-Tung University, Hsinchu, Taiwan. His current research interests include video analysis and image processing.

About the Author—HONG-YUAN MARK LIAO received the B.S. degree in physics from National Tsing-Hua University, Hsin-Chu, Taiwan, in 1981, and the M.S. and Ph.D. degrees in electrical engineering from Northwestern University, Illinois, in 1985 and 1990, respectively. Currently, he is a research fellow of the Institute of Information Science, Academia Sinica, Taiwan. He is also jointly appointed as a professor of the Computer Science Department of National Chiao-Tung University, Taiwan. Dr. Liao's current research interests include multimedia signal processing, wavelet-based image analysis, content-based multimedia retrieval, and multimedia protection. He also served as the program chair of the International Symposium on Multimedia Information Processing (ISMIP'1997), the program cochair of the second IEEE Pacific-Rim Conference on Multimedia (2001), and conference cochair of the Fifth IEEE International Conference on Multimedia and Exposition (ICME2005). He was an associate editor of the IEEE Transactions on Multimedia during 1998–2001. Dr. Liao is on the Editorial Boards of Journal of Visual Communication and Image Representation; the Acta Automatica Sinica; and the Journal of Information Science and Engineering. He is a senior member of the IEEE Computer Society.