Automated High-Level Movie Segmentation for Advanced Video-Retrieval Systems

Alan Hanjalic, Reginald L. Lagendijk, *Senior Member, IEEE*, and Jan Biemond, *Fellow, IEEE*

***Abstract*—We present a newly developed strategy for automatically segmenting movies into logical story units. A logical story unit can be understood as an approximation of a movie episode, which is a high-level temporal movie segment, characterized either by a single event (dialog, action scene, etc.) or by several events taking place in parallel. Since we consider a whole event and not a single shot to be the most natural retrieval unit for the *movie* category of video programs, the proposed segmentation is the crucial first step toward a concise and comprehensive content-based movie representation for browsing and retrieval purposes. The automation aspect is becoming increasingly important with the rising amount of information to be processed in video archives of the future. The segmentation process is designed to work on MPEG-DC sequences, where we have taken into account that at least a partial decoding is required for performing content-based operations on MPEG compressed video streams. The proposed technique allows for carrying out the segmentation procedure in a single pass through a video sequence.**

***Index Terms*—Video content analysis, video data bases, video segmentation.**

I. Introduction

In recent years, technology has reached a level where vast amounts of digital information are available at a low price. During the same time, the performance-versus-price ratio of digital storage media has steadily increased. The ease and low cost of obtaining and storing digital information as well as the almost unlimited possibility to manipulate it make people eager to collect and store more and more of it. Thus, we have witnessed the rapid growth of digital archives in the professional and consumer environment; examples of this are digital museum archives and Internet archives.

The developments in the field of digital video compression have also made possible the creation of digital *video* archives. Such video libraries are already available to commercial service providers. We expect that the digital storage of video material at home will soon overtake the current analog video cassette recording systems [9], [13], [16]–[18].

A particular problem with digital libraries is the management of large amounts of information. Solutions for text-oriented databases (e.g., SQL) already exist. However, methods for browsing, querying, and organizing visual information (images, graphics, and video), and their linking to textual information are still in their infancy [11]. The MPEG-7 standardization platform [6] addresses ways to represent visual information by means of a standard set of descriptors, so that it can be used effectively in professional and consumer applications and especially in user interfaces. The question of how to obtain these descriptors automatically is becoming a research topic of increasing importance. Particularly, the automation aspect becomes highly important with steadily rising volumes of information to be processed.

To obtain descriptors for visual information, a content analysis is required. Several approaches for the content analysis of still images exist; most of them are based on an analysis and comparison of color, texture, and shape [5], [12], [14]. It is generally accepted that content analysis of video sequences requires a preprocessing procedure that first breaks up the sequences into temporally homogeneous segments called *shots* [1]–[3], [8], [15], then condenses these segments into one or a few representative frames (*key frames*) [4], [7], [20], [23], and finally determines the relationship between shots on the basis of their audiovisual characteristics (e.g., audio tracks, key frames). This last step we call video-content organization. Since most of the video streams that modern digital storage systems have to deal with are available in the MPEG compressed format, no content-related operations are possible on these streams directly. However, it is also not necessary to decode the stream completely, since all processing steps can be performed on the corresponding *DC*[1] *sequence* [19]. This has a main consequence that key frames are only available in subsampled formats, which are called DC images.

Most existing approaches perform the video-content organization by clustering shots on the basis of the similarity between the visual contents contained in their key frames. As with still images, the content is typically represented by color histograms, object shapes, and textures. Approaches in [20] and [24] show shot-based organization structures (e.g., shot cluster trees), in which a single shot is considered to be an elementary retrieval unit of the analyzed video. In addition to key frames, temporal content variations in a shot can be used. The scene-transition graphs in [22] simulate the story flow of an analyzed video sequence by temporally connecting different shot clusters.

In this paper, we concentrate on *movies* as a particularly important category of video programs. We see several problems involved in using unconstrained shot-based cluster trees to organize movie material for retrieval purposes. In the first place, the obtained structures may be very large due to the

Manuscript received January 10, 1997; revised January 27, 1999. This work was supported in part by the European Commission under ACTS SMASH project AC018. This paper was recommended by Associate Editor T. Sikora.

The authors are with the Faculty of Information Technology and Systems, Information and Communication Theory Group, Delft University of Technology, Delft 2628 CD The Netherlands (e-mail: alan@it.et.tudelft.nl).

Publisher Item Identifier S 1051-8215(99)04157-9.

[1]DC sequence consists of frames, which are formed using the DC coefficients of the discrete cosine transform (DCT). This transform is one of the steps in the MPEG-compression scheme.

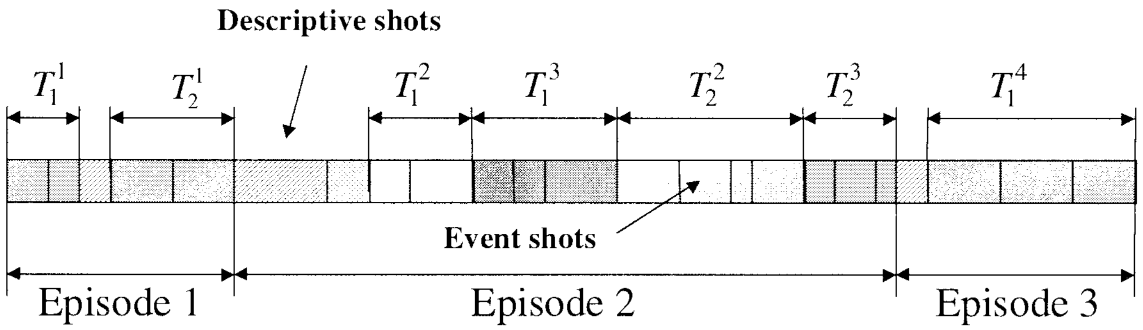


Fig. 1. A sample movie sequence consisting of three episodes. Two of them (1 and 3) cover only one event and have a simple structure. Episode 2 covers two events taking place at the same time, presented by their interchanging fragments. Descriptive shots are described as boxes with lined patterns. All other shots belonging to segments T_i^j are event shots.

huge number of shots in a full-length movie, they may not be unique, and they may not be transparent enough, especially if no natural cluster structure exists among the shots. Second, the concept of using single shots as elementary retrieval units may be useful for some video material, but certainly not for full-length movies. As already noticed in [21], it is highly unlikely that a user would think in terms of single shots when interacting with a movie-retrieval system. A far more probable scenario is that humans remember different *events* after having watched a movie, and hence they also think in terms of events during the retrieval process. Such an event can be a dialog, an action scene, or, generally, any series of shots "unified by location or dramatic incident" [21]. Instead of a single shot, the whole event should be treated as an elementary retrieval unit in advanced movie-retrieval systems. Therefore, we believe that for efficient movie-content organization and retrieval, strategies other than unconstrained shot clustering are required. In this paper, we propose a novel method to automatically segment movies into groups of successive shots, which we call *logical story units*. Each of these units is characterized by one or several temporally interrelated events, which implies that the result of our segmentation can provide a concise and comprehensive entrance level to an event-oriented movie-organization scheme. The high-level segmentation method that we propose can be carried out in parallel with the preprocessing steps of shot-change detection and key-frame extraction.

In Section II, we will give a definition of a logical story unit and discuss the justification of our segmentation approach in more detail, also in view of related approaches in recent literature. Section III proposes the novel movie-segmentation method and explains the possible differences between detected logical story units and actual movie episodes. We applied the proposed method to two full-length movie sequences and evaluated the segmentation results in Section IV. Conclusions can be found in Section V.

II. Concept of Logical Story Units

A. From an Episode to a Logical Story Unit

Each shot within a movie belongs to a certain global context built up around one movie event or several of them taking place in parallel. Thereby, a shot can either be a part of an event or serve for its "description" by, e.g., showing the scenery where the coming or the current event takes place, showing a "storytelling" narrator in typical retrospective movies, etc. In view of such a distinction, we will further refer to shots of a movie as either *event shots* or *descriptive shots*.

We can now realistically assume that a standard movie is produced as a series of meaningful segments corresponding to the event-oriented global contexts described above, which we will call *episodes*. An episode is generally a combination of the event shots and descriptive shots related to the event(s) of the episode. It can be simple, if it concentrates on only one event. However, more complex episode structures exist as well, containing several events taking place in parallel that are presented as a series of their interchanging fragments. We denote the fragment i of the event j by T_i^j and introduce a model for the movie structure (shown in Fig. 1), which takes into account episodes of different complexity.

In view of the event-based structure of an episode and the assumed limited number of episodes in a typical movie, segmenting a movie into episodes can provide a compact and comprehensive entrance level to an event-oriented movie-organization scheme. Such segmentation can be performed most precisely if the movie script is available. However, this is not the case in automated sequence analysis systems, especially in those operating at the user side [13] of a video transmission network. In such systems, all movie-content analysis, segmentation, and organization processes are done on the basis of the movie's audiovisual characteristics and their temporal variations, measured and captured by standard audio-, image-, and video-processing tools. It is realistic to assume that segmentation results obtained by means of these tools are generally different from those obtained by using the movie script, since different segmentation criteria are used. This will be explained in more detail in Section III-C. In this paper, we only use *visual features* to perform the movie segmentation. As a result, approximations of the actual movie episodes are obtained, which we will call *logical story units* (LSU's).

Various applications in digital video libraries can benefit from an LSU-based movie-organization scheme. For example, an overview of a movie can be obtained immediately if one looks at the obtained set of LSU's. Fig. 2 illustrates how a movie can be broken up into LSU's and how existing content-based clustering algorithms can be applied to all shots of an

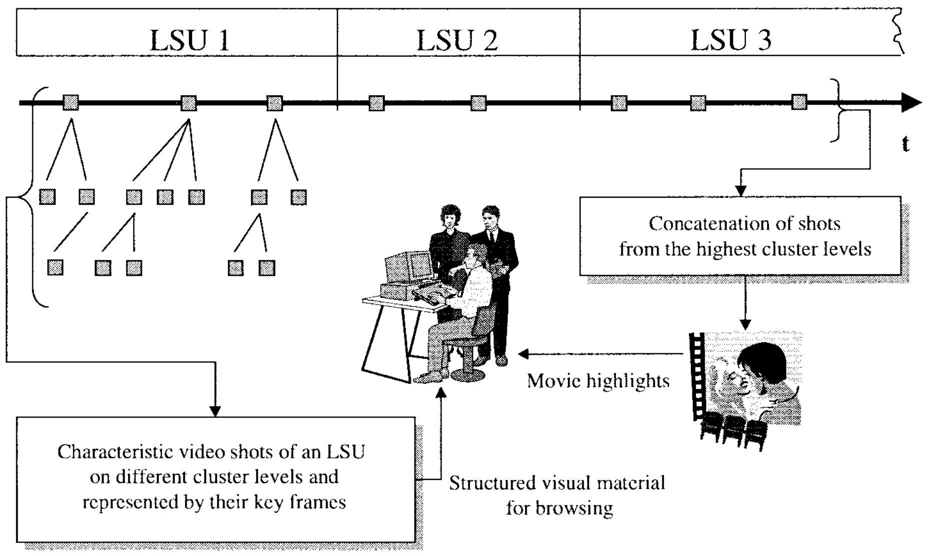

LSU 1
LSU 2
LSU 3
t
Concatenation of shots from the highest cluster levels
Movie highlights
Characteristic video shots of an LSU on different cluster levels and represented by their key frames
Structured visual material for browsing

Fig. 2. Possible scheme for movie representation for which LSU's are used.

LSU. The shots of each LSU that are most representative can be glued together and played as movie highlights. One can also use key frames to browse through each individual LSU, which is an especially important feature for LSU's having a complicated structure (e.g., containing several temporally interrelated events). The user only browses through relevant shots, e.g., those relating to the selected LSU (for instance, when searching for a particular movie character in the context of a certain event), and is not burdened with (the many) other shots of a sequence. For each granularity (cluster) level, a key-frame set is available providing video representations through pictorial summaries having different amounts of detail.

B. Previous Work

Few methods dealing with high-level movie segments can be found in the literature. In [10], characteristic temporal events like dialogs and high-motion and high-contrast segments are extracted for the purpose of making a movie trailer, but no attempt is made to capture the entire movie material. Audio as well as motion information is used for an extraction procedure. In [21], an approach is presented that is based on time-constrained clustering and assignment of labels to all shots in a sequence. Predefined models are used to analyze the resulting label sequence and to recognize patterns corresponding to dialogs, action segments, and arbitrary story units. However, the effectiveness of this method, especially for segmenting movies into story units, depends on the applicability of the model used for a story unit. We foresee several practical problems here, such as, for example, the choice of the interval for time-constrained clustering, which puts an artificial limit on the duration of an episode. Another problem is that characterizing shots by distinct labels simplifies the real interrelation among neighboring shots far too much.

The method of detecting LSU boundaries that we propose in this paper essentially finds a compromise between the totally unconstrained shot-clustering approaches from [20], [22], and [24] and the model-driven shot-clustering approach from [21].

C. Definition of LSU

Our definition of an LSU is based on the global temporal consistency of its visual content. Such a consistency can be expected in view of the realistic assumption that an event is related to a specific location (scenery) and to certain characters. It can be expected that every now and then within an event, similar *visual content elements* (scenery, background, people, faces, dresses, specific patterns, etc.) appear, and some of them even repeat. Such content matches clearly may not happen in successive shots, but it is highly probable that these occur within a certain time interval. The definition of an LSU can now be formulated as follows: an LSU is a series of temporally contiguous shots, which is characterized by overlapping links that connect shots with similar visual content elements.

An illustration of an LSU as defined above is given in Fig. 3. In the movies that we have processed, the length of an LSU varied roughly between 1 and 10 min.

The basis of the definition of an LSU given above is that a visual dissimilarity between two video shots can be measured. In the next section, we will propose a suitable intershot dissimilarity measure and discuss it in detail. For now, we assume that the dissimilarity $A(k, k+l)$ between the shots k and $k+l$ is quantitatively available. Three different cases can be distinguished, depending on the relation of the current shot k and the mth LSU.

Case 1: Visual content elements from shot k_1 reappear (approximately) in shot $k_1 + p_1$. Then, shots k_1 and $k_1 + p_1$ form a linked pair, illustrated in Fig. 3 by the arrow. Since shots k_1 and k_1+p_1 belong to the same *LSU*(m), consequently all intermediate shots also belong to *LSU*(m)

$$[k_1, k_1 + p_1] \in \mathit{LSU}(m) \qquad \text{if } p_1 \Leftarrow \min_{l=1,\cdots,c} A(k_1, k_1 + l) < M(k_1). \quad (1)$$

Here, c is the number of subsequent shots with which the current shot is compared to check the visual dissimilarity. The threshold function $M(k)$ specifies the maximum dissimilarity allowed within a single LSU. Since the visual content is

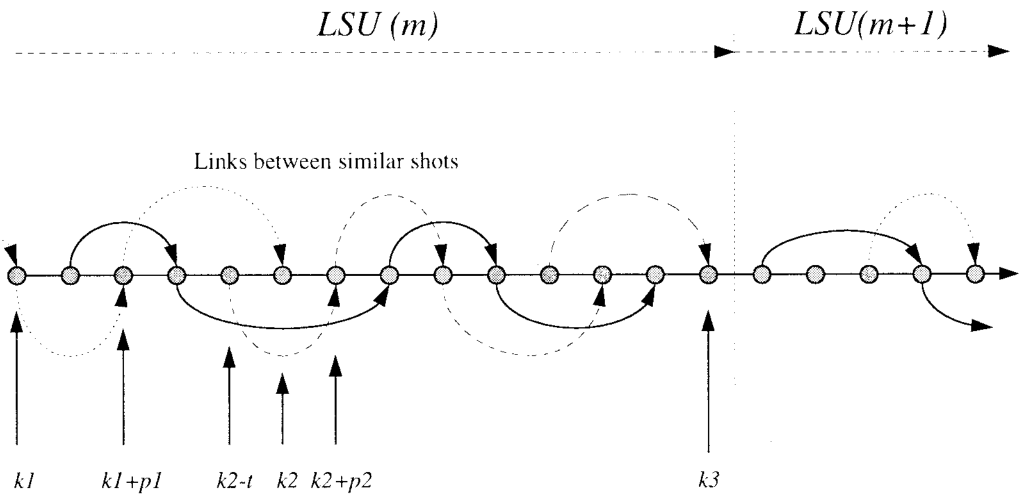


Fig. 3. Illustration of LSU's characterized by overlapping links connecting similar shots.

usually time variant, the function $M(k)$ also varies with the shot under consideration. The estimation of $M(k)$ will be discussed in Section III.

Case 2: There are no subsequent shots with sufficient similarity to shot k_2, i.e., the inequality in (1) is not satisfied. However, one or more shots preceding shot k_2 link with shot(s) following shot k_2 (see Fig. 3). Then, the current shot is enclosed by a pair of shots that belongs to *LSU*(m), i.e.,

$$[k_2 - t, k_2 + p_2] \in \mathit{LSU}(m)$$
$$\text{if } (t, p_2 > 0) \Leftarrow \min_{i=1,\cdots,r} \min_{l=-i+1,\ldots,c} A(k_2-i, k_2+l) < M(k_2). \tag{2}$$

Here r is the number of shots to be considered preceding the current shot k_2.

Case 3: If for the current shot k_3 neither (1) nor (2) is fulfilled, but if shot k_3 links with one of the previous shots, then shot k_3 is the last shot of *LSU*(m). This can also be seen in Fig. 3.

In Section III, we will discuss the calculation of $A(k, n)$ and $M(k)$, as well as an efficient way to use the above definition to detect the LSU boundaries.

III. Novel Approach for LSU Boundary Detection

A. Threshold Function

The objective is to detect the boundaries between LSU's, given the definition of an LSU and the concept of linking shots. In principle, one can check (1) and (2) for all shots in the video sequence. This, however, is computationally intensive and also unnecessary. According to (1), if the current shot k is linked to shot $k + p$, all intermediate shots automatically belong to the same LSU, so they do not need to be checked. Only if no link can be found for shot k is it necessary to check whether at least one of r shots preceding the current shot k can be linked with a shot $k + p$ [for $p > 0$, as stated in (2)]. If such a link is found, the procedure can continue at shot $k + p$; otherwise, shot k is at the boundary of *LSU*(m). The procedure then continues with shot $k + 1$ for *LSU*$(m + 1)$. The proposed LSU boundary-detection procedure is illustrated in Fig. 4.

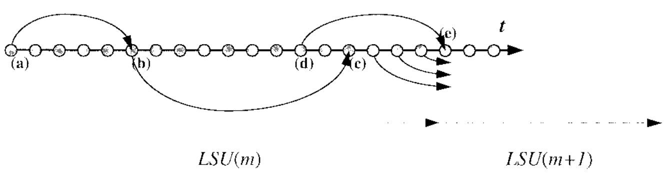


Fig. 4. Illustration of the LSU boundary-detection procedure. The shots indicated by (a) and (b) can be linked and are by definition part of *LSU*(m). Shot (c) is implicitly declared part of *LSU*(m) since the shot (d) preceding (c) is linked to a future shot (e). Shot (e) is at the boundary of *LSU*(m) since it cannot be linked to future shots, nor can any of its r predecessors.

To determine whether a link can be established between two shots, we need the threshold function $M(k)$. We compute this threshold recursively from already detected shots that belong to the current LSU. If the minimum of $A(k, n)$ found in (1) [or (2) if (1) does not hold] denotes the *content inconsistency value* $C(k)$ of shot k, then the threshold function $M(k)$ we propose is

$$M(k) = \alpha \bar{C}(k, N_k). \tag{3}$$

Here α is a fixed parameter whose value is not critical between 1.3 and 2.0. Further, $\bar{C}(k, N_k)$ is computed as

$$\bar{C}(k, N_k) = \frac{1}{N_k + 1}\left(\sum_{i=1}^{N_k} C(k - i) + C_0\right). \tag{4}$$

The parameter N_k denotes the number of links in the current LSU that have led to the current shot k, while the summation in (4) comprises the shots defining these links. Essentially, the threshold $M(k)$ adapts itself to the content inconsistencies found so far in the LSU. It also uses as a bias the last content inconsistency value C_0 of the previous LSU for which (1) or (2) is valid.

B. Intershot Dissimilarity Measure

The LSU detection algorithm and the computation of the threshold function require the use of a content-based

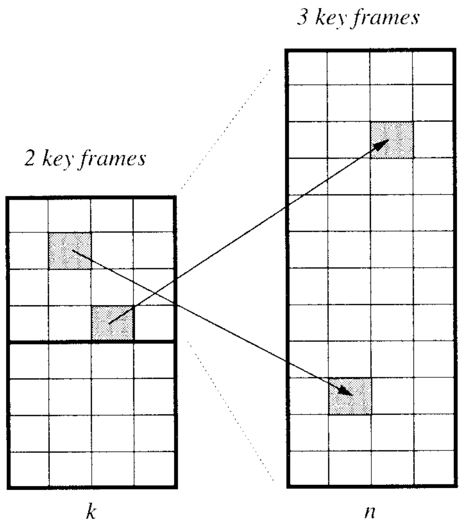


Fig. 5. Comparison of shot k with shot n by matching $H \times W$ blocks from each key frame of shot image k with shot image n. Shot k had two key frames, and shot n had three key frames.

dissimilarity function $A(k,n)$. In the following, we define our own dissimilarity measure.

We assume that the video sequence is segmented into shots, for which any of the methods found in the literature [1]–[3], [8], [15] can be used. Each detected shot is represented by one or multiple key frames so that its visual information is captured in the best possible way [4], [7], [20], [23]. Since we are using MPEG compressed video sequences, the key frames are DC images, which are typically 64 times smaller than the original frames (8 $\times$ 8 discrete cosine transform blocks are used).

For each shot, all key frames are merged in one large variable-size image, called the *shot image*, which is then divided into blocks of $H \times W$ pixels. Each block is now a simple representation of one visual-content element of the shot. Since we cannot expect an exact shot-to-shot match in most cases, and because the influence of those shot-content details that are not interesting for an LSU as a whole should be as small as possible, we choose to use only those features that describe the $H \times W$ elements *globally*. Furthermore, fairly large blocks have to be used, for instance, $H = W = 8$. In this paper, we only use the average color in the $L^*u^*v^*$ uniform color space as a block feature.

For each pair of shots (k,n), with $k < n$, we would now like to find the mapping between the blocks b_k and b_n, each being an $H \times W$ block from the shot image k and n, respectively.

- Each block b_k in a key frame of shot image k has a unique correspondence to a block b_n in shot image n. If a block b_n has already been assigned to a block b_k of a key frame belonging to shot image k, no other block of that key frame may use it. All blocks b_n are only available when a new key frame of shot k is to be matched. Fig. 5 illustrates this in more detail.
- The average distance in the $L^*u^*v^*$ color space between corresponding blocks of the two shot images is minimized

$$\min_{\text{all possible block combinations}} \sum_{\text{all blocks } b} d(b_k, b_n) \tag{5}$$

  where $d(b_k, b_n)$ is as shown in (6) at the bottom of the page and where all possible block combinations are given by the first item.

Unfortunately, this is a problem of high combinatorial complexity. We therefore use a suboptimal approach to optimize (5). The blocks b_k of a key frame of shot k are matched in the unconstrained way in shot image n starting with the top-left block in that key frame, and subsequently scanning in the line-fashioned way to its bottom-right block. If a block b_n has been assigned to a block b_k, it is no longer available for assignment until the end of the scanning path. For each block b_k, the obtained match yields a minimal distance value $d_1(b_k)$. This procedure is repeated for the same key frame in the opposite scanning fashion, i.e., from bottom-right to top-left, yielding a difference mapping for the blocks b_k and a new minimal distance value for each block, denoted by $d_2(b_k)$. On the basis of these two different mappings for a key frame of shot k and corresponding minimal distance values $d_1(b_k)$ and $d_2(b_k)$ per block, the final correspondence and actual minimal distance $d_m(b_k)$ per block are constructed as follows.

- $d_m(b_k) = d_1(b_k) \quad \text{if } d_1(b_k) = d_2(b_k).$ (7a)
- $d_m(b_k) = d_1(b_k)$ if $d_1(b_k) < d_2(b_k)$ and $d_1(b_k)$ is the lowest distance value measured for the assigned block in the shot image n (one block in shot image n can be assigned to two different blocks in a key frame of shot k: one time in each scanning direction) (7b)

$$d_m(b_k) = \infty, \quad \text{otherwise.} \tag{7c}$$

- $d_m(b_k) = d_2(b_k)$ if $d_2(b_k) < d_1(b_k)$ and $d_2(b_k)$ is the lowest distance value measured for the assigned block in the shot image n (7d)

$$d_m(b_k) = \infty, \quad \text{otherwise.} \tag{7e}$$

∞ stands for a fairly large value, indicating that no objective best match for a block b_k could be found. The entire procedure is repeated for all key frames of a shot k, leading to one value $d_m(b_k)$ for each block of a shot image k.

Last, the average of the distances $d_m(b_k)$ of the B best matching blocks [those with lowest $d_m(b_k)$ values] in the shot image k is computed as the final intershot dissimilarity value

$$A(k,n) = \frac{1}{B} \sum_{\substack{B \text{ best matching} \\ \text{blocks}}} d_m(b_k). \tag{8}$$

$$d(b_k, b_n) = \sqrt{(L^*(b_k) - L^*(b_n))^2 + (u^*(b_k) - u^*(b_n))^2 + (v^*(b_k) - v^*(b_n))^2} \tag{6}$$

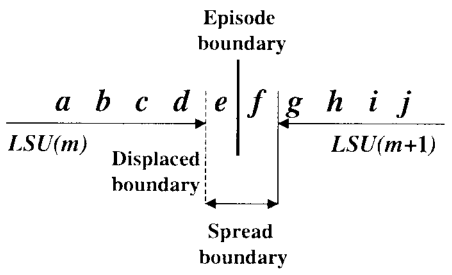


Fig. 6. Possible differences between an LSU and an episode boundary.

The reason for taking only the B best matching blocks is that two shots should be compared only on a global level. In this way, we allow for inevitable changes within the LSU, which, however, do not degrade the global continuity of its visual content.

C. LSU Versus Episode Boundaries

After the LSU boundary detection procedure has been explained, the characteristics of the obtained LSU's in view of the actual episodes will now be discussed. For this purpose, we investigate a series of shots a to j, as illustrated in Fig. 6. According to the movie script, the boundary between episodes m and $m+1$ lies between shots e and f. We now assume that shot e, although belonging to episode m, has a different visual content than the rest of the shots in that episode. This can be the case if, e.g., e is a descriptive shot, which generally differs from event shots. Consequently, the content consistency could be followed by overlapping links in the *LSU*(m) up to shot d, so that the LSU boundary is found between shots d and e. If shot e contains enough visual elements also appearing in episode $m+1$, so that a link can be established, e is assumed to be the first shot of *LSU*$(m+1)$ instead of shot f. This results in a *displaced* episode boundary, as shown in Fig. 6. However, if no content-consistency link can be established between shot e and any of the shots from episode $m+1$, another LSU boundary is found between shots e and f. Suppose that f is a descriptive shot of episode $m+1$, containing a different visual content than the rest of the shots in that episode, so again no content-consistency link can be established. Another LSU boundary is found between shots f and g. If the linking procedure can now be started from shot g, it is considered to be the first shot of the new *LSU*$(m+1)$. In this case, not a precise *LSU boundary* is found but one that is spread around the actual episode boundary, taking into consideration all places where the actual episode boundary can be defined. Consequently, shots e and f are not included in LSU's, as shown in Fig. 6. Such scenarios occur quite often and show that by investigating the temporal consistency of the visual content, only an approximation for the actual episode should be expected.

IV. Experimental Validation

To test the proposed LSU boundary-detection approach, we used two full-length movies. Both were available as DC sequences obtained from MPEG streams with (slightly modified) frame sizes, 88 × 72 and 80 × 64, respectively. We detected the shots using the method from [3] and represented each shot by two key frames taken from the beginning and the end of a shot, in order to capture most of its important visual content elements.

To get an idea about where the episode boundaries should actually be positioned, we asked unbiased test subjects to manually segment both movies. The obtained segmentation results differed mainly in the number of detected episode boundaries; this was especially noticeable in the complicated movie segments and can be explained by how each subject perceived that episode to be constructed. An example of such a complicated movie segment is a broad context of a wedding party with a lot of small events taking place in parallel—some complete, some mixed with other events—but all relating to the same scenery and movie characters. Some subjects regarded this as one episode, while others labeled each individual event.

Then, we had our algorithm perform the automatic segmentation of the movies for different values of parameters B and α. Thereby, we limited the range of the parameter α only to 1.4–1.5 while B varied in the range 40–70% of all shot-image blocks. We learned that taking less than 30% of the blocks makes the intershot comparison too rough. On the other hand, more than 70% makes the comparison too detailed. Hereby, we related the number of blocks considered in (8) to the total number of blocks in a shot image to compute the percentages. Although both parameters determine the sensitivity of the detection procedure and, consequently, also the number and positions of detected boundaries, the parameter B is more interesting since it defines the limits of intershot comparison, concerning both the amount of detail taken into account and how "global" this comparison should be. On the other hand, we left the parameters c (look-ahead distance) and r (look-back distance) constant at values $c = 8$ and $r = 3$, since the segmentation results were fairly insensitive to the setting of these parameters.

On the basis of manual segmentation results, we defined two different classes of episode boundaries:

- *probable boundaries:* registered by all test subjects;
- *potential boundaries:* registered by some of the test subjects;

and used them to classify the automatically obtained results for each parameter pair (B, α). Thereby, an automatically obtained LSU boundary was considered to be properly detected if it was close enough to the one detected manually. For this purpose and in view of the possible boundary *displacement* discussed in Section III-C, we set the maximum tolerable distance to four shots. Any other automatically detected boundary was considered to be *false*. To evaluate the quality of the automated boundary detection, we used the following expression:

$$Q = \frac{D}{1+F}. \tag{9}$$

Q denotes the quality of the boundary detection, D is the number of properly detected *probable* boundaries, and F is the number of falsely detected boundaries. Since the *probable* boundaries were those that all test subjects had selected, we

TABLE I
LSU Boundary-Detection Results for Different Parameter Settings. Bold Numbers Indicate the Parameter Combination Providing the Optimal Detection Performance. Combinations with the Same Q Values Are Assigned the Same Ranking

(B (%), α)	MOVIE 1				MOVIE 2				OVERALL QUALITY RANKING
	Detected probable boundaries (out of 19)	Detected potential boundaries (out of 17)	Falsely detected boundaries	Quality ranking	Detected probable boundaries (out of 26)	Detected potential boundaries (out of 16)	Falsely detected boundaries	Quality ranking	
(40, 1.4)	11	2	0	(2)	18	6	4	(6)	(4)
(40, 1.5)	9	1	0	(3)	18	5	3	(4)	(3)
(50, 1.4)	12	3	0	(1)	19	4	2	(3)	**(1)**
(50, 1.5)	11	1	0	(2)	18	4	3	(4)	(2)
(60, 1.4)	14	4	1	(4)	19	4	2	(3)	(3)
(60, 1.5)	12	4	1	(6)	19	5	1	(2)	(4)
(70, 1.4)	14	6	2	(7)	21	4	1	(1)	(5)
(70, 1.5)	13	4	1	(5)	20	7	4	(5)	(6)

considered them to be fundamental, and relevant for quality evaluation. This is not the case with *potential* boundaries, and they are, therefore, not considered in (9). For each parameter combination (B, α) and for each movie, the quality Q was computed, resulting in a ranking list of all pairs (B, α). The first column of Table I shows all parameter combinations (B, α) used in the experiments. The other columns show for each of the movies the number of probable and potential boundaries that were detected, the number of false alarms, and the ranking for each parameter combination according to the computed detection quality Q. In the final step, ranks of all pairs (B, α) obtained for both movies have been added up and served for the overall ranking of all parameter combinations based on the quality of their detection.

As shown by the overall ranking list in the last column of the table, the best performance for both movies is obtained when 50% of the blocks are considered for computing the overall intershot difference value and when the threshold multiplication factor α is 1.4. It can also be observed that the quality of a parameter combination decreases the more it differs from the optimal parameter set. This is mainly due to the influence of parameter B: if fewer blocks are taken into account when (8) is computed, the intershot comparison becomes too global, resulting in a low number of detected boundaries. Although only a few or even no false alarms were registered, the resulting video representation is too coarse for the first interaction level in the movie retrieval system. On the other hand, the large number of blocks considered in (8) can make the boundary detection too sensitive. This may result in an increased number of false boundaries, so that the first user-interaction level is too complex and the interaction inefficient.

For the chosen optimal parameter combination $B = 50\%$ and $\alpha = 1.4$, the average percentage of detected *probable* boundaries is 69%, with only 5% of false detections. The low number of false detections obtained for this parameter set is in line with our requirements of conciseness and comprehensiveness of the movie-retrieval interface at its highest level. These characteristics are not guaranteed if there is a high percentage of "false episodes," making the first interaction level overloaded. On the other hand, after investigating the missed 31% of *probable* boundaries, we found out that most of the episodes, which could not be distinguished from each other, belong to the same global context (e.g., a series of episodes including a wedding ceremony, a reception, and a wedding party). Therefore, a relatively high percentage of missed *probable* boundaries does not actually diminish the comprehensiveness of the LSU set obtained for $B = 50\%$ and $\alpha = 1.4$ when we take into account that more detailed information about the content of each detected LSU can be obtained by browsing through different hierarchical levels of the structure in Fig. 2.

Table I also shows that the efficiency of the algorithm concerning the detection of probable and potential boundaries is not the same. The higher percentage of probable boundaries that were detected can be explained by the fact that those boundaries were characterized by a radical change of the scenery, which could easily be recognized by all test subjects, but also by our algorithm. On the other hand, most of the potential boundaries were marked by some of the users in highly complex parts of the movies, where clearly distinguishing different episodes was a difficult task. Since our assumption about the temporal consistency of the visual content within an episode, i.e., its change at an episode boundary, was often not fulfilled in such complex movie segments, no good detection performance could be expected there.

Falsely detected boundaries in both movies were, in general, those lying in the middle of an event. One such boundary is found in a scene that takes place in a dark room where suddenly the light was turned on, ending the temporal consistency of the visual content. On the other hand, if boundaries were missed, this was most often a consequence of insufficient changes of visual features at certain episode boundaries. One example is the transition from a wedding ceremony to a wedding party, featuring the same crowd and a similar scenery.

The movies used in this paper belong to quite different categories in view of the dynamics and variety of their contents.

However, the results in Table I show that the difference in the algorithm performance is not as large, indicating a sufficient consistency of the defined detection approach and of the LSU model for different movie types. According to Table I, the number of false detections is slightly higher for the second movie. This was expected in view of the fact that most of the movie is characterized by episodes having a highly complex visual structure. On the other hand, this is compensated by the number of properly detected *probable* boundaries, which is a bit higher than in the first movie. This can be interpreted as a consequence of a larger variety of sceneries used in this movie to characterize different episodes, making their boundaries more obvious and easier to detect.

V. Conclusion

In this paper, we presented a new approach for automatically segmenting movies into units that closely approximate actual movie episodes. Our segmentation is based on an investigation of the visual information of a video sequence and the temporal variations, as well as on the assumption that the visual content within a movie episode is temporally consistent. Consequently, an LSU is defined on the basis of overlapping links, which connect shots with similar visual content. We determine whether a link between two shots exists by applying an adaptive threshold function to shot dissimilarities. Based on experimental results, we can conclude that the number of missed episode boundaries for a particular movie primarily depends on the degree to which an episode boundary corresponds to a large discontinuity in the global visual content flow. Similarly, the number of falsely detected boundaries is directly related to the global temporal consistency of the visual content within an episode. The results in Table I show that the majority of episode boundaries in both movies could be found with only a low percentage of false detections, providing a concise and comprehensive first interaction level of a movie-retrieval interface. Also, the results of applying the algorithm to the two movies belonging to quite different movie categories did not differ much, indicating that the detection performance, and therefore also the defined LSU model, are sufficiently consistent for different types of movies.

In general, our work shows that using only visual features of a movie sequence can provide satisfactory segmentation results, although the LSU boundaries only approximate the actual episode boundaries in some cases.

The results of the high-level segmentation procedure presented in this paper can be used for developing an efficient event-oriented movie-retrieval scheme. As the proposed technique computes the detection threshold recursively and only looks ahead a limited number of shots, the entire process, including the shot-change detection, key-frame extraction, and LSU boundary detection, can be carried out in a single pass through a sequence.

Acknowledgment

The authors wish to express their gratitude to Philips Research, Eindhoven, The Netherlands, for providing them with test sequences.

References

[1] G. Ahanger and T. D. C. Little, "A survey of technologies for parsing and indexing digital video," *J. Visual Commun. Image Represent.*, vol. 7, no. 1, pp. 28–43, Mar. 1996.

[2] J. S. Boreczky and L. Rowe, "Comparison of video shot boundary detection techniques," in *Proc. IS&T/SPIE Storage and Retrieval for Still Image and Video Databases IV*, Feb. 1996, vol. 2670, pp. 170–179.

[3] A. Hanjalic, M. Ceccarelli, R. L. Lagendijk, and J. Biemond, "Automation of systems enabling search on stored video data," in *Proc. IS&T/SPIE Storage and Retrieval for Image and Video Databases V*, Feb. 1997, vol. 3022, pp. 427–438.

[4] A. Hanjalic, R. L. Lagendijk, and J. Biemond, "A new method for key frame based video content representation," in *Image Databases and Multi Media Search*, A. W. M. Smeulders and R. Jain, Eds. Singapore: World Scientific, 1997, pp. 97–107.

[5] S. He and N. Abe, "A structural representation and its application to image retrieval," in *Proc. IEEE 1st Workshop Multimedia Signal Processing*, Princeton, NJ, 1997, pp. 349–354.

[6] "MPEG-7: Context and objectives (v.4)," ISO/IEC JTC1/SC29/WG11, July 1997.

[7] R. L. Lagendijk, A. Hanjalic, M. P. Ceccarelli, M. Soletic, and E. Persoon, "Visual search in a SMASH system," in *Proc. IEEE ICIP'96*, vol. III, pp. 671–674.

[8] Y. Nakajima, K. Ujihara, and A. Yoneyama, "Universal scene change detection on MPEG-coded data domain," in *Proc. IS&T/SPIE Visual Communications and Image Processing*, Feb. 1997, vol. 3024, pp. 992–1003.

[9] H. Okamoto, M. Nakamura, Y. Hatanaka, and S. Yamazaki, "A consumer digital VCR for advanced television," *IEEE Trans. Consumer Electron.*, vol. 39, pp. 199–204, Aug. 1993.

[10] S. Pfeiffer, R. Lienhart, S. Fischer, and W. Effelsberg, "Abstracting digital movies automatically," *J. Visual Commun. Image Represent.*, vol. 7, no. 4, pp. 345–353, Dec. 1996.

[11] R. W. Picard, "Light-years from Lena: Video and image libraries of the future," in *Proc. IEEE ICIP'95*, vol. I, pp. 310–313.

[12] S. Servetto, K. Ramchandaran, and T. S. Huang, "A successively refinable wavelet-based representation for content-based image retrieval," in *Proc. IEEE 1st Workshop Multimedia Signal Processing*, Princeton, NJ, 1997, pp. 325–330.

[13] SMASH project. [Online]. Available WWW: http://www-it.et.tudelft.nl/pda/smash.

[14] J. R. Smith and S.-F. Chang, "SaFe: A general framework for integrated spatial and feature image search," in *Proc. IEEE 1st Workshop Multimedia Signal Processing*, Princeton, NJ, 1997, pp. 301–306.

[15] Q. Wei, H. Zhang, and Y. Zhong, "A robust approach to video segmentation using compressed data," in *Proc. IS&T/SPIE Storage and Retrieval for Image and Video Databases V*, Feb. 1997, vol. 3022, pp. 448–456.

[16] P. H. N. de With, "Data compression systems for home-use digital video recording," *IEEE J. Select. Areas Commun.*, vol. 10, pp. 97–121, Jan. 1992.

[17] P. H. N. de With, A. M. A. Rijckaert, H.-W. Keesen, J. Kaaden, and C. Opelt, "An experimental digital consumer HDTV recorder using MC-DCT video compression," *IEEE Trans. Consumer Electron.*, vol. 39, pp. 711–722, Nov. 1993.

[18] N. Yanagihara, C. Siu, K. Kanota, and Y. Kubota, "A video coding scheme with a high compression ratio for consumer digital VCR's," *IEEE Trans. Consumer Electron.*, vol. 39, pp. 192–198, Aug. 1993.

[19] B.-L. Yeo and B. Liu, "On the extraction of DC sequence from MPEG compressed video," in *Proc. IEEE ICIP'95*, vol. II, pp. 260–263.

[20] M. M. Yeung and B. Liu, "Efficient matching and clustering of video shots," in *Proc. IEEE ICIP'95*, vol. I, pp. 338–341.

[21] M. Yeung and B.-L. Yeo, "Video content characterization and compaction for digital library applications," in *Proc. IS&T/SPIE Storage and Retrieval for Image and Video Databases V*, Feb. 1997, vol. 3022, pp. 45–58.

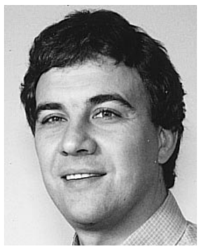
[22] M. Yeung, B.-L. Yeo, W. Wolf, and B. Liu, "Video browsing using clustering and scene transitions on compressed sequences," in *Proc. IS&T/SPIE Multimedia Computing and Networking*, Feb. 1995, pp. 399–413.

[23] H. Zhang, C. Y. Low, and S. W. Smoliar, "Video parsing and browsing using compressed data," in *Multimedia Tools and Applications*. Norwell, MA: Kluwer Academic, 1995, vol. 1, pp. 89–111.

[24] D. Zhong, H. Zhang, and S.-F. Chang, "Clustering methods for video browsing and annotation," in *Proc. IS&T/SPIE Storage and Retrieval for Still Image and Video Databases IV*, Feb. 1996, vol. 2670, pp. 239–246.

Alan Hanjalic received the Dipl.-Ing. degree in electrical engineering from the Friedrich-Alexander University, Erlangen-Nuremberg, Germany, in 1995. He currently is pursuing the Ph.D. degree in electrical engineering at the Delft University of Technology, The Netherlands.

His current research interests concentrate on video content analysis and processing for advanced video-retrieval systems. From 1995 to 1998, he was a Researcher within the European ACTS Storage for Multimedia Applications Systems in the Home (SMASH) project. From May to September 1998, he was with Hewlett-Packard Laboratories, Palo Alto, CA, where his activities were concentrated on developing efficient video segmentation and abstraction techniques.



Reginald L. Lagendijk (S'87–M'90–SM'97) received the M.Sc. and Ph.D. degrees in electrical engineering from the Technical University of Delft, The Netherlands, in 1985 and 1990, respectively.

He was a Visiting Scientist in the Electronic Image Processing Laboratories, Eastman Kodak Research, Rochester, NY, in 1991. Currently, he is a Professor in the Information and Communication Theory Group, Technical University of Delft. He is the author of *Iterative Identification and Restoration of Images* (Norwell, MA: Kluwer, 1991) and coauthor of *Motion Analysis and Image Sequence Processing* (Norwell, MA: Kluwer, 1993). At present, his research interests include signal processing and communication theory, with emphasis on visual communications, compression, analysis, searching, and watermarking of image sequences. He has been involved in the European research projects DART, SMASH, and DISTIMA. He is currently leading several projects in the field of wireless visual communications.

Prof. Lagendijk was Associate Editor of the IEEE TRANSACTIONS ON IMAGE PROCESSING. He currently is a member of the IEEE Signal Processing Society's Technical Committee on Image and Multidimensional Signal Processing.

Jan Biemond (M'80–SM'87–F'92) was born in De Kaag, The Netherlands. He received the M.S. and Ph.D. degrees in electrical engineering from Delft University of Technology, The Netherlands, in 1973 and 1982, respectively.

Currently, he is a Professor and Chairman of the Information and Communication Theory Group, Faculty of Information Technology and Systems, Delft University of Technology. His research interests include multidimensional signal processing, image enhancement and restoration, video compression (digital TV, stereoscopic TV, and HDTV), and motion estimation with applications in image coding and computer vision. He has published extensively in these fields. In 1983, he was a Visiting Professor at Rensselaer Polytechnic Institute, Troy, NY, and at the Georgia Institute of Technology, Atlanta. He was General Chairman of the Fifth ASSP/EURASIP Workshop on Multidimensional Signal Processing, Noordwijkerhout, The Netherlands, in September 1987.

Prof. Biemond received the Dutch Telecom Award "Vederprijs" in 1986 for his contributions in the area of digital image processing, in particular in image restoration and subband coding. He was a Distinguished Lecturer of the IEEE Signal Processing Society for 1993–1994. He is a former member of the IEEE-SPS Technical Committee on Image and Multidimensional Signal Processing. Since 1988, he has been a member of the IEEE-CAS Technical Committee on Visual Signal Processing and Communications.