Shot Boundary Determination on MPEG Compressed Domain and Story Segmentation Experiments for TRECVID 2003

Masaru Sugano, Keiichiro Hoashi, Kazunori Matsumoto, Fumiaki Sugaya, and Yasuyuki Nakajima

KDDI R&D Laboratories Inc.
2-1-15 Ohara, Kamifukuoka, Saitama 356-8502, JAPAN
{sugano,hoashi,matsu,fsugaya,nakajima}@kddilabs.jp

1. INTRODUCTION

KDDI R&D Laboratories has been participating in the past TREC conferences for text retrieval tasks. In this year we are newly participating in TRECVID 2003, namely the shot boundary determination and story segmentation tasks. In shot boundary determination task, we applied our proprietary shot segmentation algorithm originally proposed in [1] and slightly upgraded for this task. In our methods, statistics such as histogram as well as motion vector information from MPEG coded bitstream are used to adaptively determine various types of shot boundaries. For the story segmentation task, we conducted experiments under the conditions "Audio/Video" and "ASR Only". Our officially submitted results were based on the ASR Only condition, where we implemented a story segmentation method based on the TextTiling algorithm. This paper also describes our Audio/Video story segmentation experiments conducted after official result submission.

2. SHOT BOUNDARY DETERMINATION

This section describes our shot boundary determination methods and experiments.

2.1 Partial MPEG decoding

DCT DC coefficients give the lowest frequency component of image and at the same time they represent spatially scaled image since DC component is a block averaged value [2]. Furthermore, in I-pictures these coefficients are directly obtained during VLD (Variable Length Decoding) process without time consuming process such as Inverse DCT. In [2], more than 90% of abrupt scene changes are detected using DCT DC information on I-picture interval. However, this low temporal resolution may limit detection accuracy; for example, a scene with a very fast panning may change whole scene after one GOP period, which leads to false shot boundaries since the current I-picture is completely different from the previous one. Therefore in order to enhance temporal resolution of shot boundary determination, coded frame information on P- and B-picture is required. DC components in these pictures can be obtained after some manipulation. In P- and B-pictures, although some of macroblocks may be intra coded, most of the coded blocks are inter coded where only prediction error after motion compensation is coded using DCT. In addition, there may be skip blocks and MC no Coded blocks where no DCT coefficient is coded.

DCT DC image is a reduced size image by 1/8 both horizontally and vertically. Therefore DC components of P- and B-pictures are obtained using motion compensation in reduced size image domain. There are two ways to obtain DCT DC image for P-/B-pictures. One is to apply motion compensation (MC) using reduced size motion vectors in 1/8. The other is to apply weighted motion compensation reflecting contribution of all the blocks used for motion compensation [9][14]. Figure 1 shows a block diagram of the latter scheme. Subjectively, it is found that the latter has less visible noise due to motion compensation mismatch. Therefore we use the latter method to obtain DCT DC images for P- and B-pictures.

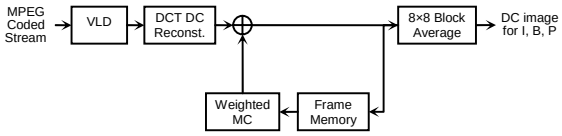


Figure 1. DC image with weighted MC

2.2 Shot boundary determination methods

2.2.1 Abrupt shot boundary determination

By incorporating the MC operation mentioned above, P- and B-pictures are roughly reconstructed so that temporal resolution can be greatly improved. Previously a good deal of research work has been reported on shot boundary determination [2-13]. The major technique includes pixel differences, histogram comparison, edge differences statistical differences, compressed data amount differences, and motion vectors. Although either one of the above techniques achieves relatively high accuracy, each has its own disadvantage [1].

We proposed shot boundary determination from I-picture sequence of MPEG coded video in 1994 [2]. We use both pixel differences and histograms methods to overcome problems when either one of them is used. Here, we extend this approach to detect shot boundaries in one frame unit.

Pre-processing

To exclude undesired false detection mainly due to camera motion and object movement, only frames with high inter-frame difference are picked up for the succeeding shot boundary determination. The inter-frame difference is obtained by:

$$D_n = \sum_{i=0}^{M}\sum_{j=0}^{N}\left|Y_n(i,j) - Y_{n-1}(i,j)\right| \quad (1)$$

, where M and N are total number of 8×8 blocks in a frame for vertical and horizontal direction, respectively. For example, in MPEG-1 in SIF size (352×240), M=30 and N=44. $Y_n(i,j)$ is the luminance block average at block (i,j) in the n th frame. Since DCT DC component of each 8×8 block is obtained from section 2, $Y_n(i,j)$ for each frame is directly given from this value. Then the following equation is used as pre-processing:

$$D_n > Th\_pre \quad (2)$$

Only those frames which satisfy the above conditions are further investigated in abrupt shot boundary detection in the following.

Shot boundary determination using luminance and chrominance change

Both luminance and chrominance characteristics dramatically change at shot boundaries. Thus ordinary shot boundary are detected when both the luminance and chrominance information greatly change. We use temporal peak detection of both inter-frame luminance difference and chrominance histogram correlation [2]. A frame is declared as a shot boundary when:

$$D_n > D_{n-1}, D_{n+1} \text{ and } \rho_n > \rho_{n-1}, \rho_{n+1} \quad (3)$$

Here, is a weighting factor for the detection. ρ_n is chrominance histogram correlation obtained by:

$$\rho_n = \frac{\sum_{k,l} H_{n,k,l} H_{n-1,k,l}}{\left(\sum_{k,l} H_{n,k,l}^{\,2} \sum_{k,l} H_{n-1,k,l}^{\,2}\right)^{1/2}} \quad (4)$$

, where $H_{n,k,l}$ is a chrominance histogram matrix. The histogram is obtained classifying DC chrominance Cb and Cr data in a frame into hc classes for each chrominance component. Then two dimensional hc×hc histogram matrix in the n-th frame $H_{n,k,l}$ ($k, l = 0, 1, 2... hc-1$) is obtained.

When shot boundary exists on scenes with large motion, it is very difficult to find temporal peak using frame difference since frame difference may be very large all the way due to motion so that Eq. (3) may not detect such shot boundaries. Therefore, only chrominance correlation is used to detect such shot boundary for those frames which don't satisfy Eq. (3).

$$\rho_n > Th\_ac \quad (5)$$

, where Th_ac is a threshold value for determination of temporal peak in ρ_n.

Furthermore, when consecutive two shots are different only in camera angle, color histogram will be similar and thus it is difficult to detect shot boundary by the above conditions such as Eq. (3) and (5). However, since pixel difference usually has a very large peak at these shot boundaries, peak detection of luminance difference are applied. When either of the following equation is satisfied for those frames which are not declared as scene change in the above process, the frame is declared as shot boundary.

$$D_n > D_{n-1}, D_{n+1} \quad (6)$$

$$D_n - Th\_ad > D_{n-1}, D_{n+1} \quad (7)$$

, where and Th_ad are a weighting factor and a threshold value for detecting a temporal peak in D_n, respectively. Basically, Eq. (6) will detect shot boundary in similar scenes. However, Eq. (7) is also used for such cases when motion is involved since all of the inter-frame differences are kept relatively high and the ratio of D_n to D_{n-1} or D_{n+1} may not be significantly high enough to find the shot boundary using Eq. (6).

2.2.2 Dissolve shot boundary determination

Basic detection algorithm of dissolve and fade

In gradual transition such as dissolve and fade in/out, two different shots are usually synthesized in the course of transition. For example, in dissolve transition, gradual change from one shot to another occurs with simultaneous decrease and increase of intensities of preceding and following shots. Since both shots are synthesized during transition, activity of the each frame shows U-shape curve surrounded by flat shoulders when dissolve occurs [13]. In the case of fade in/out, activity curve shows monotonous increase/decrease. The frame activity for n-th frame FA_n is described as:

$$FA_n = \sum_{i=0}^{M}\sum_{j=0}^{N}\left\langle Y_n(i,j)^2 - \left\langle Y_n(i,j)\right\rangle^2\right\rangle \quad (8)$$

In [13], positive peak before dissolve and negative peak during dissolve are used to detect U-shape variance curve. It assumes that only single pair of positive and negative peaks with a large peak to peak difference exists during dissolve period. However, in the actual video sequences, it rarely shows these shapes due to motion and local fluctuations. However it is difficult to find real positive and negative peaks of dissolve region even if the variance shows U-shape curve [1]. Furthermore, peak to peak difference may not always be large due to picture flatness or motion.

In order to detect these shapes avoiding false detection, we have applied filtering process as noise reduction for the DCT DC activity data. Since dissolve and fade processes take long duration, temporal filtering with long tap is suitable to absorb spontaneous fluctuations and examine long duration variation. As a temporal filtering, we use a moving average of activities MA_n for a period of frames VF which includes current and previous (VF -*1*) frames:

$$MA_n = \frac{1}{VF} \sum_{t=n}^{n-VF+1} FA_t \qquad (9)$$

After temporal filtering, temporal peak or monotonous increase/decrease can be detected. However, since duration of dissolve and fade depends on how the shots are edited, such a technique as simple peak detection may result in false detection. Furthermore, very flat U-shape curve will be expected when a dissolve transition occurs in between relatively flat shots. Therefore it is necessary to contrast these curves with others. We use first order derivative of the filtered activity DA_n in order to detect these curves. It is obtained as:

$$DA_n = MA_n - MA_{n-1} \qquad (10)$$

In TRECVID data, the derivative curve tends to be negative in our preliminary experiment. Therefore dissolve period are found when the derivative curve continuously takes negative values during a certain period. Fade in/out period can also be found when only positive/negative period is detected. In order to exclude undesired detection in such scenes as motion, we use chrominance correlation between *n*-th and (*n*-*dd*)-th frames to confirm that the region is a shot boundary candidate. Therefore dissolve sequence candidates are detected using the following equations.

$$DA_n < -Th\_dis1 \text{ and } \rho_{n,\,n-dd} < Th\_dis2 \qquad (11)$$

Between *n*-th and (*n*-*dd*)-th frame, if the number of frames satisfying Eq. (11) is larger than *Th_dis3*, a dissolve transition is determined in this period. In order to avoid detecting motion scenes, the following equation should be considered:

$$k < Th\_dis4 \qquad (12)$$

, where k is number of non-intra blocks. Dissolve detection is carried out for those frames which are determined as non abrupt scene change in the previous section.

Although the above equations can detect most of the dissolving, there are two problems in terms of detection accuracy. One is that it is difficult to detect those dissolve transitions in similar color shots or in shots with large flat areas, since conditions in Eq. (11) assumes that two shots have different color distributions with non-flat regions. The other is that it may also detect panning or motion scenes as dissolving since these scenes may have similar activity curve in such cases when scenes with large flat object appear during panning. In the following, countermeasures for these errors in the detection are described.

Dissolve determination in shots with flat areas

As for the first problem described above, it is necessary to have more detailed observation of activity variation for those frames which are determined as non-dissolve in Eq. (11). Since negative period is continuing in dissolve as described earlier, closer investigation of these characteristics is carried out as follows:

$$\forall i \in n\_p, \quad DA_i > -Th\_ea \quad \& \quad \sum_{i \in n\_p} DA_i > -Th\_sa \qquad (13)$$

where $n\_p = \langle n-dm, n-dm-1, \cdots, n-dm-dh \rangle$ (14)

Here, detection of negative period is carried out by observing the derivative of activity DA_i and sum of DA_i in a period *n_p* are greater than threshold values -*Th_ea* and -*Th_sa*, respectively as shown in Eq. (13).

Although the above equations can detect dissolving which has relatively small variations in activity during dissolving, they may also detect such scenes as very slow panning since both characteristics will show relatively flat activity curve. In order to distinguish dissolve from such non-dissolve scenes, we have also used prediction error information obtained from coded bitstream. In the scenes with very small motion, most of the blocks are successfully motion compensated and prediction error in the MC block is relatively small. On the other hand, in the case of dissolve, inter-frame difference may be as small as that of small motion case. However, prediction error is large in dissolve transition since motion compensation is usually ineffective in the course of synthesizing of two shots. The normalized prediction error NPE_n in *n*-th frame is obtained as following equation.

$$NPE_n = \left(\sum_{i,j \in non\_intra}^{k} DC_n(i,j) \right) MN/k \quad (15)$$

Here, $DC_n(i,j)$ is DC component of prediction error which is directly obtained from (0,0) element of DCT coefficients. *MN* is total number of blocks in a frame and *k* is number of non-intra blocks. Since NPE_n in the dissolve period has a large value, the numbers of frames which have a large prediction error around dissolving are compared with threshold values which are shown in the following equations.

$$\sum_{l \in bd\_p} PE_{fl} > Th\_dbd \quad \& \quad \sum_{l \in dd\_p} PE_{fl} > Th\_ddd \quad (16)$$

where

$$dd\_p = \langle n, n-1, \cdots, n-df \rangle$$

$$bd\_p = \langle n-df, n-df-1, \cdots, n-df-db \rangle \quad (17)$$

Here PE_{fl} is "1" if normalized prediction error NPE_l in *l*-th frame is larger than the threshold value *Th_pe*. Using predetermined values of *df* and *db*, *dd_p* and *bd_p* are periods during dissolve and before the dissolve, respectively. Therefore, dissolve in flat region is detected when Eq. (13) and Eq. (16) are satisfied.

Exclusion of panning/motion scenes in dissolve determination

Although panning/motion scenes when flat object is appeared have very similar activity curves to normal dissolve case as described earlier, panning/motion scenes have different characteristics concerning motion. In panning/motion scenes, most of all the macroblocks will be motion compensated and inter-frame difference will be very large, whereas in the case of dissolving the number of motion compensated block is usually small and inter-frame difference is also small. Therefore, we use the number of motion compensated blocks and inter-frame difference in order to exclude these false scenes from detected dissolve frames. As for number of motion compensated blocks, the following condition is applied since it has a large value in the case of panning and motion scenes.

$$MVC, PMVC > Th\_mvc \quad (18)$$

, where *MVC* and *PMVC* are numbers of motion compensated blocks in the most recent P-picture and its previous P-picture, respectively. In order to exclude motion compensated blocks which are not real motion involved, only blocks which have motion vector size larger than threshold value *Th_mv* are counted in Eq. (18). We have also applied several conditions described in the following since above equation may also exclude dissolve in the panning/motion scenes.

Motion scenes are characterized as large inter-frame difference whereas panning scenes are characterized that most of all the motion vectors are in the same direction. Therefore the following conditions are used.

$$D_n, D_{n-1} > Th\_bm \quad (19)$$

$$D_n > Th\_mm \ , \ |<mvx>| \ or \ |<mvy>| > Th\_am \quad (20)$$

Eq. (19) corresponds to motion scenes where consecutive motion is detected using inter-frame difference. Eq. (20) corresponds to panning where frame average horizontal/vertical motion vectors are compared with threshold value. Therefore if either Eq. (19) or Eq. (20) along with Eq. (18) are satisfied, the frame is declared as panning/motion scenes.

2.2.3 Wipe shot boundary determination

A wipe is a scene transition where a new shot appears and at the same time current shot disappears changing their spatial positions. Although wipe in TV program are found mostly in TV news and may not be found in other programs like commercials and film, wipe tends to be recognized more easily than other scene changes due to its rather long transition duration and therefore it usually plays an important semantic role in the program.

Several examples of wipe transitions are depicted in the Figure 2. Figure 2(a) shows most typical wipe where a new shot ***B*** translated to the right direction over the current shot ***A***. Figure 2(b) is modified version of Figure 2(a) where shot ***B*** expands horizontally whereas shot ***A*** shrinks accordingly. Figure 2(c) and (d) are modifications of wipe model in Figure 2(d) where a new shot ***B*** expands over shot ***A*** in vertical direction. Figure 2(e) is a page-turn type wipe where a new shot ***B*** appears as if a current *page* ***A*** is turned.

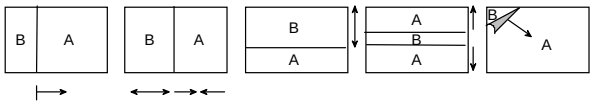


Figure 2. Wipe models

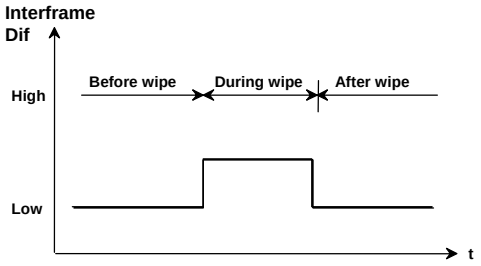
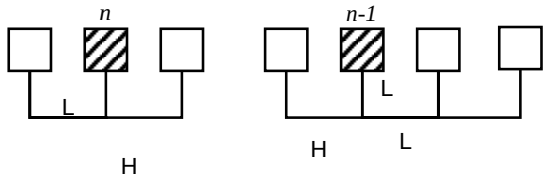


Figure 3. Wipe model using inter-frame difference



(a) Flashlight/subliminal at *n*-th frame

(b) Flashlight/subliminal at *n-1*-th frame

Figure 4. Single flashlight/subliminal effect models

As can be seen from these patterns in Figure 2, it is difficult to use motion information for wipe determination since various kinds of motion models are required for corresponding wipe patterns. Although moving patterns are completely different depending on wipe models, spatial positions of two shots are always moving during wipe periods in any types of wipe and each shot before/after wipe period is usually still and stable unless large motion is involved in shots. Furthermore, moving speed of shots in wipe is slow and steady during wipe period. Therefore, when inter-frame difference is used as determination measure, each wipe can be represented by the simple model as shown in Figure 3. Then a wipe is declared when the following equations are satisfied for those frames which are not designated as abrupt nor dissolve scene change.

$$BW > Th\_bw\ ,\ DW > Th\_dw,\ AW > Th\_aw \qquad (21)$$

Here *BW*, *DW*, and *AW* are number of frames which are recognized as periods before wipe, during wipe and after wipe, respectively. These values are obtained by:

$$BW = \sum_{k=prw+1}^{ws} DL(k),\quad DW = \sum_{k=ws+1}^{we} DH(k),\quad AW = \sum_{k=we+1}^{pow} DL(k) \qquad (22)$$

where *DL(k)* and *DH(k)* are flags which show that *k*-th frame has low and high inter-frame difference D_k, respectively. These flags are determined by the following conditions.

$$\text{if } D_k > Th\_wp \text{ then } DL(k)=0,\ DH(k)=1$$
$$\text{else } DL(k)=1,\ DH(k)=0 \qquad (23)$$

2.2.4 Flashlight and subliminal effect detection

A flashlight scene is spontaneous frame change due to flashlight in a shot. For example, in TV news sequence, a flashlight scene appears while an important person gives a speech in a press conference. Also a subliminal effect (simply subliminal, hereafter) may be inserted into TV programs or films with a certain intention. Since a flashlight frame and a subliminal frame are quite different from preceding and following frames, frames with flashlight/subliminal and after flashlight/ subliminal are often falsely detected as scene change. Luminance and chrominance distributions in flashlight/ subliminal frame are completely different from those in the previous frames. However, unlike shot boundary, these distributions in flashlight return to the previous states after one or a few frames. Therefore by investigating frames before and after flashlight scene, flashlight scene can be excluded from scene change points. Single flashlight model is depicted in Figure 4. For example, when *n*-th frame is flashlight scene, correlation between *n*-th and *n-1*-th is low whereas correlation between *n+1* and *n-1* is high as shown Figure 4(a). In the same way, especially consecutive flashlight scenes can be easily modeled by extending single flashlight model. We use chrominance histogram correlation as correlation measure in order to distinguish flashlight from other shot boundary. Therefore flashlight/ subliminal effect at *n*-th frame is detected when:

$$\rho(n\ ,\ n-1) < Th\_fl\ ,\ \ \rho(n+1,\ n-1) > Th\_fh \qquad (24)$$

2.3 Evaluation results

We applied the above mentioned shot boundary determination to TRECVID 2003 test data (totally 12 sequences). All the parameters used in the above equations are determined through a 20 minutes TV sequence encoded by MPEG-1, not in TREC test data.

Table 1 shows the results of shot boundary determination; recall (Re.) and precision (Pr.) for total, recall and precision for abrupt shot boundaries, and

recall, precision, frame-recall and frame-precision for gradual transition boundaries. These scores are calculated using TREC shot boundary evaluation program provided by NIST. As shown in Table 1, most of abrupt shot boundaries are successfully detected. However, in spite of incorporating flashlight exclusion algorithm, most of the false detections for abrupt shot boundaries are flashlights. In addition, sudden changes of brightness such as shining are falsely determined as abrupt shot boundaries. As for un-detection, the abrupt shot boundaries between fields are not detected since the test data is encoded in frame structures. Also the shot boundaries where the frame is only partly changed are not detected.

As for gradual transitions, about half of the shot boundaries are detected in our algorithm. The cause of false detection is roughly categorized in two cases; one is that a scene is falsely determined as wipe or dissolve when a large object slowly comes into a frame, and the other case is when an object suddenly starts to move very fast from still mode. These false detections require more detailed observation of motion of the object. As for un-detection, many of wipe transitions cannot be determined. In addition, dissolve transitions between very similar shots in terms of color, texture, etc. are not detected. Therefore more detailed analysis is needed for enhancing the gradual transition detection accuracy, which corresponds to future challenge.

As for computational cost, our method achieves very fast operation, about 24 times faster than real-time playback on the normal Windows PC with Pentium 4 1.8GHz CPU, since all the processes are performed on compressed data domain.

Table 1. Shot boundary determination results for TRECVID 2003 test sequences

Sequence	All		Abrupt		Gradual			
	Re.	Pr.	Re.	Pr.	Re.	Pr.	F-Re.	F-Pr.
19980203	0.758	0.786	0.928	0.830	0.479	0.672	0.590	0.524
19980222	0.846	0.812	0.961	0.855	0.495	0.625	0.404	0.624
19980224	0.800	0.844	0.952	0.867	0.458	0.750	0.440	0.508
19980412	0.815	0.810	0.973	0.850	0.416	0.633	0.514	0.626
19980425	0.776	0.785	0.942	0.810	0.505	0.716	0.593	0.533
19980515	0.826	0.832	0.957	0.879	0.541	0.689	0.491	0.576
19980531	0.873	0.842	0.974	0.868	0.537	0.716	0.542	0.595
19980619	0.839	0.868	0.984	0.915	0.472	0.681	0.538	0.540
19990303	1.000	0.684	1.000	1.000	0.000	0.000	-	-
19990308	0.960	0.857	0.960	1.000	0.000	0.000	-	-
20010614	1.000	0.470	1.000	1.000	0.000	0.000	-	-
20010702	1.000	0.937	1.000	1.000	0.000	0.000	-	-

2.4 Conclusion

In this Section, firstly a preprocessing for shot boundary determination is described. By using motion vectors and DCT DC information, DC image in 1/64 of original coded sized has been obtained directly from MPEG bitstream for P- and B-pictures as well as I-pictures. Shot boundary determination algorithm not only for abrupt scene change but also for gradual transitions is proposed. In our methods, statistics like histogram as well as motion vector from coded bitstream are used to adaptively detect various types of shot boundaries. In addition, exclusion algorithms for panning and flashlight/subliminal scenes have also been proposed. In the experiment around 95% of abrupt shot boundaries are successfully detected for the TRECVID test data. As for gradual transitions, about half of shot boundaries are detected. Since its process is very fast and only less than 5% of normal playback time is required, the proposed method well realizes efficient shot boundary determination used for higher level processing such as content base video analysis.

3. STORY SEGMENTATION

Our story segmentation methods and experiments are described in this section. As mentioned in Section 1, experiments based on two conditions, i.e., “ASR only” and “Audio/Video”, were conducted. Due to delays in the development of our audio/video feature extraction programs, we were only able to submit the “ASR only” results as our official submission. Therefore, the “Audio/Video” experiments were conducted after the official submission.

3.1 Story segmentation based on ASR results

For our “ASR only” experiments, we implemented a story segmentation method based on the TextTiling algorithm [15], where the similarity of adjacent text sequences are calculated, and story boundaries are drawn at points where similarity decreases. In our implementation of the TextTiling algorithm, we made a vector space model of ASR results per shot, and calculated the similarity of ASR results in adjacent shots. Each shot vector was constructed by calculating the TF*IDF value of all words occurring within a shot. Prior to this process, we removed all general stopwords, and applied Porter stemming.

The outline of the TextTiling algorithm is illustrated in Figure 5, where the vertical axis indicates the calculated similarity between documents (shots) at the time indicated by the horizontal axis. Similarity decrease points, i.e., candidates of story boundaries, are the points where similarity stops decreasing. A score for each candidate is calculated based on the “depth” of the decrease point. For example, in Figure 5, the score of point d_0 is calculated by adding $diff_1(d_0)$ and $diff_2(d_0)$.

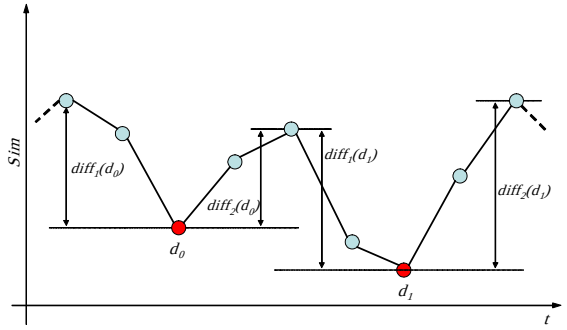


Figure 5. Outline of TextTiling algorithm

Since the amount of text information within a single shot is low, some deceiving similarity decrease points may occur due to the lack of essential words which describe the story of the regarding shot. In order to cope with this problem, we also implemented a document expansion method, based on Rocchio's algorithm [16]. Essentially, this process adds information of words which do not occur within a shot.

Document expansion is conducted by the following procedures. First, a collection of documents similar to the regarding document is constructed by calculating the similarity between the regarding document and all other documents within the program, and extracting the top N documents based on similarity. Next, a score for each word is calculated by the following formula:

$$Score(W_k) = \sum_{i \in Simshots} w_{ik} \qquad (25)$$

where the vector of shot *i* is expressed as the following formula:

$$\vec{S}_i = (w_{i1}, w_{i2}, \ldots, w_{in}) \qquad (26)$$

All words are ranked based on the above score, and the top *M* words and their regarding scores are added to the original shot vector. However, words which occur in the original shot are not added in this process. The expanded shot vectors are then used to calculate similarities for the TextTiling algorithm.

3.2 Story segmentation based on audio-video features

For our "Audio/Video" story segmentation experiments, we focused on the development of a story segmentation algorithm based purely on content-independent low-level features, instead of the widely popular and heuristic approach to detect significant "cues" of story boundaries, i.e., news anchor shots in broadcast news programs.

The general flow of our story segmentation process is as the following. First, the video is divided into individual shots. For this process, we used the TRECVID common shot boundaries. Next, low-level audio-video features are extracted from each shot. These features are used to generate a vector expression of each individual shot. Each shot vector is then input into a SVM-based story boundary determinator, which determines whether or not a story boundary occurs within the shot. A flowchart of the proposed method is illustrated in Figure 6.

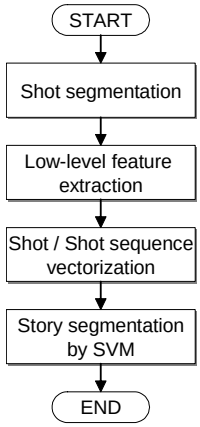


Figure 6. Outline of story segmentation algorithm

3.2.1 Audio-video feature extraction

The low-level features extracted from each shot can be roughly divided into four types: audio, motion, color, and temporal related features.

The audio related features consist of the average RMS of the shot, average RMS of the first *n* frames of the shot, and the frequency of four audio classes (silence, speech, music, noise) per shot. The average RMS of the first *n* frames is extracted mainly to detect silent periods at the beginning of a shot, which are assumed to occur at story boundaries. For the following experiments, *n* was fixed to 10.

Frequency of audio class is extracted by classifying the audio of each frame based on an audio classification algorithm by Nakajima et al [17]. This algorithm classifies incoming MPEG audio into the previously mentioned four classes, by analyzing characteristics such as temporal density, and bandwidth/center frequency of subband energy on compressed domain. The frequency of each audio class is derived by calculating the number of class occurrences within a shot.

The motion of a shot is calculated based on motion vector features of the video. Motion vectors can be directly extracted from the P-frames of MPEG-

encoded video. Total motion of the shot is obtained from the absolute sum of motion vector amplitudes. Motion intensity, which indicates the intuitional amount of motion in a shot, is defined in MPEG-7 Visual [18].

The color layout features, also defined in MPEG-7 Visual, are extracted based on the algorithm of Sugano et al[19]. Simply said, the color layout features specify spatial distribution of colors within a frame. This information is extracted from the DC image, which corresponds to a (horizontally and vertically) downscaled version of the original image. For our method, the color layout features are extracted from DC images generated from the first, center, and last frame of the regarding shot. Extracting color layout features from these three frames is assumed to be useful to detect the stability of a shot. For example, the color layout features of a static shot, such as an anchor shot, are expected to be similar throughout the shot. On the contrary, color layouts within a dynamic shot are assumed to be of wide variance.

The temporal features are shot duration and shot density. Both of these features are also general low-level features which express important characteristics of a shot. For example, anchor shots are expected to be relatively longer than other types of shots, and the density of commercial shots are expected to be higher than other shots.

All of the above features are completely independent from the video content, and can be efficiently extracted from any video data.

3.2.2 Story segmentation based on SVM

Based on the audio-video features described in the previous section, each shot of the regarding video is expressed as a vector, where each element of the vector expresses the value of each audio-video feature described in the previous section. These “shot vectors” are used as input information for a classifier based on support vector machines (SVM)[20], which is a widely implemented and effective algorithm for classification. In the proposed method, SVM is utilized to discriminate shots which include a story boundary.

Two methods are tested to define the input vector for the SVM-based classifier. One method is to simply use the shot vector as a representation of a single shot. To train the SVM based on this method, all shots that include a story boundary are labeled positive, and all other shots are labeled as negative. The resulting SVM will be able to discriminate shots with a story boundary, from all other shots in the test data. This method will be referred to as the “1-shot method".

The other method is to use a single vector to represent a sequence of adjacent shots. This is accomplished simply by connecting each shot vector to generate a large vector which expresses the features extracted from all shots within the sequence. For example, if each shot vector is *k*-dimensional, the vector of a shot S_x can be expressed by the following formula:

$$\vec{S}_x = (s_{x1}, s_{x2}, \ldots, s_{xk}) \qquad (27)$$

where s_{xm} expresses the value of the *m*-th feature. The vector of a shot sequence consisting of two shots S_1 and S_2 is a (*2*k*)-dimensional vector, which can be expressed by the following formula:

$$\vec{S}eq = (s_{11}, s_{12}, \ldots s_{1k}, s_{21}, s_{22}, \ldots s_{2k}) \quad (28)$$

In other words, the shot sequence vector is generated simply by concatenating the vectors of the two shots within the sequence.

In order to prepare training data for the SVM classifier for the shot sequence method, a shot sequence is labeled positive when the first shot of the sequence includes a story boundary. All other shot sequences are labeled negative. Figure 7 illustrates this labeling scheme of the shot sequence method.

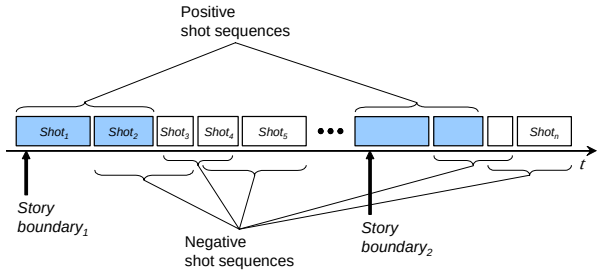


Figure 7. Outline of shot sequence labeling method

The hypothesis regarding the shot sequence method is that, while a single occurrence of a shot indicating a story boundary may be effective to detect story boundaries, this approach may also raise many false alarms, such as anchor shots which do not initiate a new story. The aim to increase the amount of utilized information from features of a single shot to plural shots, is to reduce story segmentation errors which may be caused by such false alarms, and discriminate distinct sequences of shots which occur at the beginning of a new story, which may be a better indicator of a story boundary.

3.3 Experiments

In this section, we present the results of our story segmentation experiments. Note that the officially submitted results are not discussed here, due to errors in the results that became apparent after submission.

3.3.1 ASR only experiments

As mentioned in Section 3.1, there are mainly two parameters for document expansion: the number of documents to extract additional word information (*N*), and the number of words to add to each document (*M*). These parameters were set to 5, 10, 15, and 20. Precision, recall, and F-measure of all experiments are listed in Tables 2 to 4, respectively.

Table 2. Precision of TRECVID story segmentation experiments (ASR only)

	# of words (*M*)			
# of docs	5	10	15	20
N=5	0.194	0.202	0.203	0.204
10	0.190	0.201	0.205	0.205
15	0.189	0.193	0.197	0.203
20	0.186	0.194	0.198	0.199

Table 3. Recall of TRECVID story segmentation experiments (ASR only)

	# of words (*M*)			
# of docs	5	10	15	20
N=5	0.309	0.320	0.322	0.323
10	0.301	0.318	0.322	0.323
15	0.300	0.307	0.313	0.322
20	0.295	0.307	0.314	0.315

Table 4. F-measure of TRECVID story segmentation experiments (ASR only)

	# of words (*M*)			
# of docs	5	10	15	20
N=5	0.238	0.248	0.249	0.250
10	0.233	0.246	0.251	0.251
15	0.232	0.237	0.242	0.249
20	0.228	0.238	0.242	0.244

Table 5. Results of TRECVID story segmentation experiments (Audio/Video)

	Precision	Recall	F-measure
1-shot	0.545	0.551	0.548
2-shot	0.554	0.560	0.557

Results in Tables 2 and 3 show that there is a small correlation between the number of additional words and the improvement of both precision and recall. On the contrary, the increase of documents to extract words for expansion causes the decrease of precision and recall. However, the difference between results of all parameter settings in these Tables are insignificant, as can be observed from the F-measure results in Table 4. Furthermore, the results themselves are generally low.

3.3.2 Audio/Video experiments

Next, we present the story segmentation experiment results.

In our experiments, we constructed a separate SVM model for ABC and CNN. All shots or shot sequences in the test data set are inputted to the SVM based story boundary determinator. The distance from the SVM hyperplane was used as the score for each input shot or shot sequence. All shots (shot sequences) are ranked based on this score, and the top *K* shots are extracted as story boundaries. Since the average number of story boundaries in the TREDCVID development data was 19.0 for ABC and 34.8 for CNN, we set the default number of computed story boundaries for this experiment to 19 for ABC, and 35 for CNN.

Table 5 shows the precision, recall, and F-measure of the audio/video experiments, for the 1-shot and shot sequence (2-shot) methods.

As clear from the results in Table 5, the results of the Audio/Video experiments are significantly higher than those of the ASR only experiments. These results indicate that audio/video features are more effective for story segmentation of news video.

Furthermore, the shot sequence (2-shot) method has shown better results than the single shot method. This indicates that utilizing information from two shots is effective for accurate story segmentation.

Moreover, comparison of the results in Table 5 to the official results of other participants show that these results are competitive to other TRECVID participants, even though most of the approaches of other participants make use of high-level analysis of video, such as shot classification. This shows that our simple approach to use only low-level audio/video features to directly model story boundary occurrence is quite effective. Since our method only uses content-independent features, we believe our method is easily applicable to various video content other than broadcast news.

3.4 Conclusion

We conducted two story segmentation experiments, based on the “ASR only” and “Audio/Video” conditions. Overall comparison of the two experiments shows that audio/video features are more effective than the ASR results to determine story boundaries. Future work within the story segmentation framework includes implementation of high-level audio/video features, and experiments on story labeling.

4. ACKNOWLEDGMENT

The authors would like to thank Dr. Tohru Asami, Dr. Shuichi Matsumoto, and Dr. Masahiro Wada for their continuous support and also wish to thank to the original proponents of our shot boundary determination algorithm, Ms. Kiyono Watanabe and Mr. Akio Yoneyama. Furthermore, the authors also appreciate Mr. Torbjorn Duner of Uppsala University, Sweden, for his efforts in the story segmentation experiments.

5. REFERENCES

[1] Y. Nakajima, K. Ujihara, and A. Yoneyama, "Universal scene change detection on MPEG-coded data domain," in Proceeding SPIE Visual Communications and Image Processing, vol. 3024, pp. 992-1003, 1997.

[2] Y.Nakajima, "A Video Browsing Using fast scene change detection for an efficient networked video database access," IEICE Transactions on Information & Systems, vol.E-77-D, No.12, pp.1355-1364, Dec.1994.

[3] Y.Tonomura, A.Akutsu, Y.Taniguchi, and G.Suzuki, "Structured video computing," IEEE Multimedia, pp.34-43, Fall 1994.

[4] K.Otsuji and Y.Tonomura, "Projection detecting filter for video cut detection", Proceedings of First ACM International Conference on Multimedia, pp.251-257, Aug.1993

[5] H.J.Zhang, A.Kankanhalli, and S.W.Smoliar, "Automatic parsing of news video," Proc. IEEE Int'l Conf. Multimedia Computing and Systems, May. 1994.

[6] S.W.Smoliar and H.J.Zhang, "Content-based video indexing and retrieval," IEEE Multimedia, pp.62-72, 1994.

[7] A.Nagasaka and Y.Tanaka, "Automatic video indexing and full-motion search for object appearances", Visual database systems, vol.II, E.Knuth and L.M.Wegner, eds, Elsevier, Amsterdam, pp.113-127, 1992.

[8] F.Arman, R.Depommier, A.Hsu, and M.Y.Chiu, "Image processing on compressed data for large video databases", Proceedings of First ACM International Conference on Multimedia, pp.267-272, Aug.1993.

[9] B.L.Yeo and B.Liu, "Rapid scene analysis on compressed video", IEEE Transactions on Circuits and Systems for Video Technology, Dec.1995.

[10] B.Shahrarary, "Scene change detection and content-based sampling of video sequences", Digital Video Compression: Algorithms and Technologies, SPIE, Vol.2419, pp.2-13, 1995.

[11] A.Hampapur, R.Jain and T.Weymouth, "Digital Video Segmentation", Proc. ACM Multimedia 94, pp.357-364, 1994.

[12] K.Shen and E.J.Delp, "A Fast Algorithm for Video Parsing Using MPEG Compressed Sequences", Proceeding of IEEE ICIP '95, pp.252-255, 1995.

[13] J.Meng, Y,Juan and S-F Chang, "Scene Change Detection in a MPEG Compressed Video Sequence", Digital Video Compression: Algorithms and Technologies, SPIE, Vol.2419, pp.14-25, 1995.

[14] Y.Nakajima, K.Ujihara, and T.Kanoh, "Video structure analysis and its application to creation of video summary", IEICE 2nd Joint Workshop on Multimedia Communications, pp.3-2, Oct.1995.

[15] Hearst: "TextTiling: Segmenting text into multi-paragraph subtopic passages", Computation Linguistics, 23(1), pp 33-64, 1997.

[16] Rocchio: "Relevance feedback in information retrieval", in "The SMART Retrieval System – Experiments in automatic document processing", Prentice Hall Inc., pp 313-323, 1971.

[17] Nakajima, Lu, Sugano, Yoneyama, Yanagihara, Kurematsu: "A fast audio classification from MPEG coded data", Proc. ICASSP '99, Vol.6, pp 3005-3008, 1999.

[18] ISO/IEC 15938-3, "Information Technology --- Multimedia content description interface – Part 3: Visual", 2002.

[19] Sugano, Nakajima, Yanagihara: "MPEG content summarization based on compressed domain feature analysis", Proc. SPIE Int'l Symposium ITCom2003, Vol 5242, pp 280-288, 2003.

[20] Vapnik: "Statistical learning theory", A Wiley-Interscience Publication, 1998.