Electronic Letters on Computer Vision and Image Analysis 9(1): 32-46, 2010

A Robust Multi-Feature Cut Detection Algorithm for Video Segmentation

Gianluigi Ciocca

DISCo - Department of Informatics, Systems and Communication,
Università degli Studi di Milano-Bicocca,
Viale Sarca 336, Milano, Italy

Received 25th May 2009; accepted 24th August 2010

Abstract

Video segmentation is the first task in almost all video analysis applications. It consists in identifying the boundaries of the meaningful video units (shots). Without a doubt, cuts are the most common among production effects that characterize the shot boundaries. In this paper we propose an algorithm for cut detection exploiting an innovative, robust frame difference measure. The measure is based on a combination of different visual features. To improve the precision of the cut detection algorithm, a temporal pattern analysis model, and a flashes removal are also proposed. Experimental results to prove the effectiveness of the proposed measure coupled with the temporal pattern analysis model on very heterogeneous and complex sets of videos are critically reported.

Key Words: Video Segmentation, Cut Detection, Visual Feature Extraction, Frame Difference Measure, Flash Removal

1 Introduction

The increase in computing power and electronic storage capacity has greatly expanded the potential of video libraries. As the size of video collections mounts to thousands of hours, there is a growing demand for automatic video analysis tools to help manage and access video contents effectively and efficiently. Video segmentation is one of such tools and is an essential task of almost all video-contents analysis applications, as well as video browsing and retrieval. It involves detecting temporal boundaries, and identifying meaningful segments of video (shots) [1][2]. The video boundaries are identified by production effects such as cuts, fades, dissolves, and wipes. Any error introduced by the video segmentation phase will make subsequent tasks more difficult or even impossible. In this paper, we focus our attention on the detection of cuts that, with fades and dissolves, are the most common editing effects [3]. A cut is defined as an abrupt change in the visual flow that separates two frame sequences. Since cuts are visual discontinuities, intuitively they can be detected by computing frame differences between consecutive frames and then searching for the highest differences in the computed values.

Following the taxonomy in [4], we can broadly classify the cut detection algorithms into several categories based on the underlying strategy adopted to extract information from the frames. The simplest algorithms directly use the information of the pixel values. These are the pixel-based algorithms and one of

Correspondence to: <ciocca@disco.unimib.it>

Recommended for acceptance by <Jon Sporring>
ELCVIA ISSN: 1577-5097
Published by Computer Vision Center / Universitat Autonoma de Barcelona, Barcelona, Spain

the first cut detection methods is described in [5]. Cuts are detected by simple pixel difference metrics. The authors evaluate different metrics computed on gray level images and color images. Cuts are identified if the difference value is above a threshold. In [6] pixels of two successive frames are compared using a Boolean formulation. Each pair of pixels is compared and if the difference is above a threshold, the corresponding Boolean value is set to one. If the overall amount of different pixels is above a given threshold then a cut is determined. The most relevant drawback of a pixel-based approach is that it is sensitive to small variations in the pixel values and thus the frame differences is very noisy and, as in the case of low quality videos, difference thresholds cannot be reliably set. To limit this problem, a more compact representation of the frame content must be used. One way is to compute a histogram on the whole frame to capture some image characteristic. Gargi et al. [7] analyzed the performances of eight different color spaces coupled with four histogram-based frame difference measures. The difference measures investigated are: bin to bin difference, chi square test histogram, histogram intersection and color average. Global and local thresholds are also investigated. A similar analysis, made on intensity histograms and statistical tests, can be found in [8] and [9]. In Hanjalic et al. [10] histograms computed on the YUV color space are used to compute the frame differences. Cuts are detected by analyzing the differences within a sliding window using a Gaussian model. If a difference exceeds a locally computed threshold then a cut is identified. Similar approaches can also be found in [11], and [12]. The majority of the histogram-based algorithms exploit only the color information of the images thus the problem is to find a suitable color space in which compute the color histogram. These algorithms perform well if the cuts separate two shots with very different color content. Moreover, the histograms are also computed on the whole frame and thus spatial arrangement of the colors. The block-based algorithms try to overcome this problem by spatially subdividing the frame into regions and the comparison between two frames is performed on a region by region basis. Lee and Ip [13] developed a block-based algorithm using a selective HSV histogram comparison. For each block, pixels are classified as gray pixels or color pixels depending on the HSV components. Two histograms are computed for the two classes, and a difference measure that combines the two histograms is used to detect the cuts. In [14] regional color histograms are used: eight element histograms are calculated for the Y, C_R and C_B color components over nine image's sub-blocks. The distance between two frames (not necessarily contiguous) is the median of the L_1 distances computed on the nine sub-blocks. A global threshold is used to identify possible cuts. A cut is definitely detected if the computed distance is greater than the 16 distances before and the 16 distances after.

The previous algorithms mostly use very basic approaches based on color information. As described in [4], within the feature-based algorithms, we can find all those algorithms that extract more sophisticate pictorial features than, for example, color histograms. In [15] an entropy-based approach is described. The cross-entropy measure computed on the intensity histograms is used as frame difference measure. The difference is then compared with a fixed threshold. A similar approach is used in [16] where the mutual information between two successive frames is calculated on each of the RGB components. A small value in the mutual information identifies a possible cut. An adaptive threshold, computed within a temporal window, is used to select the cuts. A more recent approach [17], proposed by the same authors, is based on the information theory principles. The mutual information between two consecutive frames is computed on the grey levels values of the frames and is used as cut detector index. The measure is sensible to fades and thus another index, the Joint Entropy is used to distinguish between the two. Li and Wei [18] based their cut detection algorithm on the computation of joint probability images between frames. These images consist of the frequencies of the co-occurrences of intensity values. A frame difference measure is defined on the joint probability images considering only values near the diagonal of each image. In [19] the Haar wavelet transform is used to compute a wavelet signature from each frame. The signature is constructed considering the most significant wavelet coefficients. A score is computed from the signatures by weighting the differences between the coefficients. A cut is detected if the score exceeds a global threshold. In Zabith et al. [20] edges extracted from the frame are used to characterize the frame contents. Edges are extracted from two successive frames, and the entering edges (that is the edges that appear in the second frame) and exiting edges (the edges that disappear from the first frame) are used to compute the edge change ratio. An abrupt change is determined when a peak in the sequence of the ratios is identified. In [21] the tracking of feature points (e.g. corner points) using Kalman filters is used. The rate of feature points that are lost or initiated is used as a criterion for shot boundary detection. Boccignone et al. [22] presented an approach able to detect cuts in the compressed domain. Different features are extracted such as the normalized bit-rate difference, the number of intra-coded macro-blocks, and the number of motion vectors referring a specified frame. The algorithm uses different thresholds and analyzes I, P and B frames in cascade: if analyzing the features extracted from I frames a cut is suspected, the features extracted from the P frames are taken into account for further analysis. If a clear cut cannot be identified, the features extracted from the B frames are finally analyzed. In Han et al. [23] probability distribution functions are estimated for each transition type. To decide if a shot boundary is present or not, a Bayesian formulation is defined based on both the probability density functions and the pattern of the transitions. Recently, Fu and Zeng [24] propose a video shot boundary detection algorithm based on the computation of local color features around interest points. The frames are processed in order to identify points of interest using the Harris corner detection, and then a color histogram is computed on the regions around each point of interest. In this way, only significant areas are considered in the computation of the visual feature thus making it more robust. Another recent approach [25], determine the cut s by using a Gabor filtering approach. Each frame is filtered by six Gabor filters that differ for orientation. The feature vector is thus composed by six filtered frames. Cuts are determined by analyzing the sum of absolute differences between two feature vectors of consecutive frames. In [26], cuts and dissolves detection is carried out using a support vector machine classifier trained both to locate shot boundaries and characterize transition types. The feature vector is composed of sequences of RGB histogram dissimilarities computed between frames at different distances. Other shot boundary detection algorithms can also be found in the surveys [4], [17], [27], [28] and in the TRECVid shot boundary detection reports [29].

From this analysis emerges that the definition of a robust frame difference measure is the most crucial issue. Most of the methods employ only one visual feature (usually histogram-based) to describe the visual content of the frames and require the measure used to compute the visual discontinuities to be reliable and robust to cope with the variability that can be found in a video sequence. However, single features cannot capture enough information to model different cut effects. For example, only the color histogram is used, a highly dynamic sequence (e.g. one containing fast moving or panning effects) with frames of the same color contents would result in a series of similar frame difference values and the motion effects would be lost. Similarly, frames with the same color contents but different from the point of view of other visual attributes, are considered similar. We argue that a more robust cut detection approach can be obtained if we use several visual descriptors and, consequently, a composite frame difference measure.

In this paper we propose an algorithm for cut detection exploiting an innovative and robust frame difference measure. The measure is based on a combination of different visual features. Our cut detection approach is compared against several algorithms in the state of the art representative of different cut detection strategies. The algorithms are tested on a rather heterogeneous set of videos. First we will show that the new measure, based on a combination of different visual features, exhibits better performances while obtaining higher precision detection compared to several other methods present in the literature even using a very simple approach based on a single decision threshold. Next we present a new cut detection algorithm based on this measure. Two common video contents that greatly reduce the cut detection precision are the presence of high dynamic scenes and of camera flashes. Both these contents may generate high frame difference values than can mislead the detection algorithm to identify cuts where there are none. In order to improve the precision of our cut detection algorithm, we have equipped it with a novel temporal pattern analysis algorithm which incorporates a flashes detector. The algorithm has been tested on two very heterogeneous and complex video sets.

2 The Cut Detection Algorithm

The frame difference measure that we proposed for the cut detection employs a composition of three simple visual features: a color histogram, an edge direction histogram and wavelet statistics. These features have been chosen among the others used in the content-based image retrieval system in [30] and [31]. They are efficient to compute and effectively describe the frame content from different perspectives. A larger set of features can also be used, but as a tradeoff between feature extraction speed and descriptive power, we have limited out choice to one feature in the color, structure, and texture pictorial categories.

In [31] the color histogram is composed of 64 bins determined by manual sampling groups of meaningful colors in the HSV color space. The edge direction histogram is composed of 72 bins corresponding to intervals of 2.5 degrees. Two Sobel filters [32] are applied to obtain the gradient of the horizontal and the vertical edges of the luminance frame image. These values are used to compute the gradient of each pixel and those pixels that exhibit a gradient above a predefined threshold are considered in computing the gradient angle and then the histogram. Multiresolution wavelet analysis provides representations of image data in which both spatial and frequency information is present. In multiresolution wavelet analysis we have four bands for each level of resolution resulting from the application of two filters, a low-pass filter and a high-pass filter. For an efficient representation a three-step Daubechies (16 coefficients) multiresolution wavelet decomposition is used. A frame is first decomposed into four sub-bands LL1, LH1, HL1, and HH1 (total of 4 sub-bands). The LL1 sub-band is substituted with its decomposition into the four sub-bands LL2, LH2, HL2, and HH2 (total of 4-1+4= 7 sub-bands). The same procedure is then applied to LL2 obtaining the four sub-bands LL3, HL3, LH3 and HH3 (total of 7-1+4=10 sub-bands). To represent the energy distribution of the transformation coefficients, the mean and standard deviation are computed for each of the 10 sub-band obtained [33], resulting in a 20-valued descriptor. High order moments can also be used but we choose to limit the feature dimensionality. To compare a frame at time *t* with one at time *t-1*, a new difference measure is used to evaluate their color histograms, wavelet statistics and edge histograms visual descriptors. The difference between color histograms is computed using the histogram intersection introduce by Swain and Ballard [34], while the difference between edge direction histograms is computed using the Euclidean distance as in the case of the wavelet statistics [31]. We denote the three distances as $d_H(t,t\text{-}1)$, $d_D(t,t\text{-}1)$, and $d_W(t,t\text{-}1)$ respectively:

$$d_H(t,t-1) = \sum_{j=1}^{64} \min\left(H_t(j), H_{t-1}(j)\right) \tag{1}$$

$$d_W(t,t-1) = \sqrt{\sum_{j=1}^{20} \left(W_t(j) - W_{t-1}(j)\right)^2} \tag{2}$$

$$d_D(t,t-1) = \sqrt{\sum_{j=1}^{72} \left(D_t(j) - D_{t-1}(j)\right)^2} \tag{3}$$

where $\boldsymbol{H_t}$, $\boldsymbol{W_t}$ and $\boldsymbol{D_t}$ represent the color histogram, wavelet statistics and edge direction histogram feature vectors computed on frame *t*.

The use of multiple features poses the problem of their combination. The choice of the optimal aggregation operator is very difficult and greatly depends on the application task, consequently in the literature no conclusive solution can be found for choosing it. For example, in [35] is concluded that the sum aggregation operator is best suited in combining the output of different classifiers in an identity verification task, but this choice is contradicted in the framework of a content based image retrieval task [36]. As another example, in [35] it is also stated that due to its conservative nature, the product aggregation operator produces the worst results. Since the detection of the frame differences is related to the problem of image retrieval, for the choice of our aggregation operator, we started with the operators used in the image retrieval systems (e.g. [29]) where the features are usually combined by weighing them with suitable factors, which are usually task-dependent. Here we propose a different approach: the explicit selection of weight factors is removed by weighing each difference against the other. To achieve this, the three difference values are firstly mapped into the range [0, 1] and then combined to form the final frame difference measure (denoted $d_{HWD}(t,t\text{-}1)$) as follow:

$$d_{HWD}(t,t-1) = d_H(t,t-1)\cdot d_W(t,t-1) + d_W(t,t-1)\cdot d_D(t,t-1) + d_D(t,t-1)\cdot d_H(t,t-1) \tag{4}$$

By weighting each frame difference against each other, the measure penalizes low frame difference values while emphasizing the higher ones. The $d_{HWD}(t,t\text{-}1)$ measure produces values toward the higher range when most of the frame differences have high values. Otherwise the resulting value is squeezed towards the lower range. Figure 1 shows the $d_{HWD}(t,t\text{-}1)$ values computed on a video sequence named “multilng” (see Table 4): the highest isolated peaks correspond to all the cuts in the sequence. By comparing these values with a suitable threshold *T* we can identify the cut positions. However, this does not guarantee that all the candidate cuts identified are true cuts: strong camera motion, high dynamic scenes and flashes can produce high difference values. Strong camera motion and high dynamic scenes can be easily identified as they will correspond to several, consecutive high frame difference values. To discriminate between true cuts and flashes (they both correspond to isolated high peaks) and to discard camera motion and high dynamic scene effects we have designed a temporal pattern analysis (TPA) algorithm to be applied to the sequence of $d_{HWD}(t,t\text{-}1)$ values.

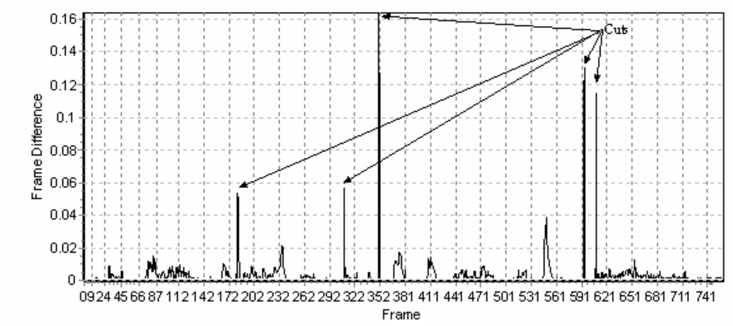


Fig. 1. Example of how the d_{HWD} measure emphasize the locations of the cuts. The video sequence is the "multilng" sequence (see Table 4 for details).

Temporal Pattern Analysis (TPA)

Given a sequence of frame descriptors, the cuts detection algorithm can be summarized as follows.

```
Input:    N, the length of the sequence,
          T and T_L, decision thresholds with  T_L < T
          w, the size of the temporal analysis window
Algorithm:
 1:  t=1;
 2:  while  t<N  do
 3:      if d_HWD(t, t-1)>T then
 4:          if  d_HWD(τ, τ-1)<T_L for all τ in {t-w, t-w+1, …, t-1, t+1, …, t+w-1, t+w} then
 5:              if d_HWD(t+1, t-1)<T then a flash is detected.
 6:              else output that a cut has been detected at position t.
             end if
         end if
 7:      t = t+1
     end while
```

In line 1 the current position is initialized. Line 2 processes all the frames in the sequence. In line 3 we check if the current frame distance is above the reference threshold and if so, the current position is a candidate cut. To discard camera motion and high dynamic scene, the candidate cut must be an isolated peak and this is checked in line 4 by verifying that the current distance is higher than the neighbor distances. If the current distance passes this test, it remains to verify that it is not an isolated peak caused by a flash. This is done by considering the frame distance computed between the previous and next frames. In the case of a flash (that usually influences only a single frame), this distance is low (line 5), while in the case of a true cut, this distance is high (line 6). Figure 2 synthesizes the underlying idea of the simple flash identification approach used in the TPA algorithm. Upward (downward) arrows indicate differences above (below) the threshold T. Top arrows represent the differences computed between consecutive frames. Bottom arrows represent differences computed on frames which are two positions apart. In line 7 the current frame index is incremented in order to process the following frame.

The TPA can be performed on-the fly implementing a data buffer of size $n=2w+1$ with FIFO policy. Every time a new frame difference is computed, it can be inserted in the buffer and then the values within it analyzed.

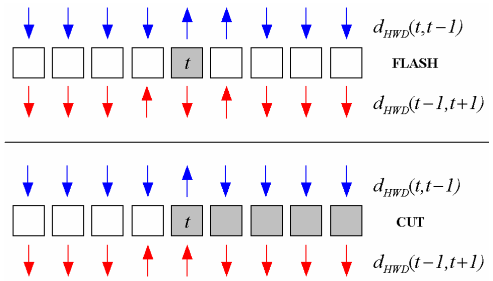


Fig. 2. Underlying idea of the flash detector. The arrows indicate whether the frame difference measure (indicated to the right) is above or below the threshold T at that position

3 Experimental results

To evaluate the proposed cut detection algorithm, we have performed two experiments. In the first experiment we evaluated the effectiveness of the d_{HWD} measure to reliably detect cuts on a complex data set. Thus the behavior of the d_{HWD} measure is evaluated by using the precision and recall curves created by varying the detection threshold. Several other cut detection algorithms found in the literature were also evaluated for comparison purpose. In the second experiment, after identifying the optimal threshold to be used the detection of candidate cuts, our cut detection algorithm was tested on a different data set. Details of the two experiments are reported in the next two sections.

3.1. Testing the HWD Measure

The aim of the experiment is to evaluate the effectiveness of the new frame difference measure in order to identify a robust global threshold which clearly separates cut and non-cut differences. To quantify the performance of the detection algorithms we use recall and precision scores. The cut detection results are classified as true positive (TP, a cut is correctly located), false positive (FP, a cut is declared in a non-cut location) and false negative (FN, a non cut is declared in a cut location). Recall is defined as the ratio between the number of cuts correctly classified and the total number of actual cuts (TP/(TP+FN)), while precision is defined as the ratio between the number of cuts correctly classified and the total number of cuts found by the detection algorithm (TP/(TP+FP)).

Availability of data sets to be used for cut detection is scarce, since very few of the video used in the evaluation processes have been made public. This makes the comparison between different algorithms difficult. In recent years the NIST Institute tried to create standard data sets with the TRECVID collection [37]. However, even these collections have some limitations. For example, the 2003 test collection is composed of 13 videos only, mostly of newscast and documentaries. More recent collections are made available to researchers only under certain conditions, while older ones are difficult to retrieve. Since the definition of a video data set is crucial for a correct evaluation of a cut detection algorithm, therefore we have thus chosen to create our video data set based on several properties. The set has been created to be statistically representative of the kinds of videos that a general-purpose video segmentation algorithm should expect to process. This means taking into account many video genres such as cartoons documentaries, movies, trailers, sport videos, music clips, non-professional videos, etc…. Next, the video sources such as Internet, digital camera, and analog conversion have been considered. Video properties such as the video format, frame resolution, frame rate, were also taken into account. The video chosen present a wide range of situations where the detection of cuts is problematical. In particular, trailers and cartoon video exhibits short shots with strong camera motions, many visual effects such as explosions, and high (unreal) dynamics in the events depicted. Many other effects can be found in commercial sequences and in news videos due to the appearance of captions or texts on the frame. Videos characterized by low fps may present relatively high frame differences within the shots. Finally, different video compressors may introduce artifacts that influence the cut detection. The data set used in this experiment[1] is listed in Table 1. The video are not very long, we favored the complexity and variability of video contents instead of having very long (and not very complex) videos.

Video Name	Format-W×H@FPS	Frames	mm:ss	Cuts
4videos	AVI-240×180@15	1,575	01:45	18
Basketball	MOV-320×240@15	448	00:30	0
Bugsbunny	MEPG-352×240@29.97	13,492	07:30	74
EEopen	MPEG-352×240@29.97	1,289	00:42	22
Football	MEPG-172×116@29.97	6,697	03:43	29
ForTheBirds	AVI-320×168@25	4,898	03:15	46
LVEB	MPEG4-320×240@12	1,518	02:06	41
News	MPEG-176×112@29.97	4,757	02:39	11
Nwanw1	MPEG-176×112@29.97	6,556	03:39	32
Restauri	MPEG-320×240@29.97	5,393	02:59	6
Voyager	MPEG-352×240@29.97	5,346	02:58	98
Weezer	MPEG-352×240@29.97	7,333	04:04	79
Total		59,302	33:50	456

Table 1. List of the videos used in the experiment.

"4videos" is a collection of four documentary videos of different subjects merged together. "Basketball" is a short sequence of a basketball game showing several flashes and very strong camera motion. "Bugsbunny" is another cartoon sequence that exhibits high motion and many editing effects. "Eeopen" is a TV series title sequence with many captions appearing. "Football", "News" and "Nwnaw1" are three video sequences captured from the TV. "ForTheBirds" is a 3D cartoon sequence taken from DVD. "LVEB" is a movie trailer that exhibits many effects by using very short shots. "Restauri" is a documentary video presenting long shots without showing particularly difficult situations. In "Voyager" a science fiction TV series is presented. Scenes showing a speaker talking are mixed with excerpts of the main fiction program. "Weezer" is a music video sequence containing many editing effects and very similar shots. A ground-truth was created from the data set considered as a whole.

Several reference cut detection strategies are used for comparison (see Table 2). The algorithms declare a cut if the frame difference measure is above a given threshold. They were chosen considering the features extracted from the frames and the measures used in the computation of the frame differences. The first algorithm (C1) refers to the d_{HWD} measure. The other algorithms can be broadly categorized into four groups: pixel-based (C2 to C6), histogram-based (C7 to C13), block-based (C14 to C16), and feature-based (C17 to C20). The last group refers to strategies that use descriptors which are more sophisticated than histograms. The table also reports the computational complexities of the algorithms: P represents the number of pixels, N the number of pixel levels and B the number of image blocks. Complexities of the algorithms C2-C19 are taken and adapted from [4]. The complexities of the C1 and C20 algorithms have been computed following the same procedure. Note that the C1 complexity takes into account also the operations required for color space transforms. Most of the operations required by C1 are due to the wavelet transform (about 3/5 of the overall computation cost). The complexity can be greatly reduced by using wavelet decomposition with few coefficients.

The results are graphically displayed in Figures 3 to 5 using the precision-recall curves obtained varying the threshold in the algorithms. In each Figure the performance of the proposed measure (C1, top curve) is reported for direct comparison. The low precision reached by all the cut detection methods is due to the very

[1] The annotated video used in the experiments can be requested, for research purposes, by contacting the authors.

difficult video data set devised. The combination of the three features in the d_{HWD} measure exhibits better performances while obtaining higher precision values compared to the other methods. Even at higher recall values, the drop in precision is mitigated. The pixel-based algorithms exhibit a similar behavior; the low initial precision remains approximately the same for recall values below 0.93, then the precision drops quickly for higher recall values. Also in Figure 4 the histogram-based algorithms exhibit similar behavior (with the exception of the 2-bits RGB histogram which shows very poor performance). These algorithms maintain a nearly constant precision level for most of the recall values. The improvement in precision of our algorithm is clearly visible. As for the feature-based and block-based algorithms in Figure 5, it can be seen that d_{HWD} shows better performance that the other methods. Among the tested methods, only the C18 algorithm has comparable results to the d_{HWD} measure for recalls below 0.90. At higher recall values the $\mathrm{d}_{\mathrm{HWD}}$ curve is well above the C18 curve (from 7% to 65% higher).

Name	Measure / Description	Complexity (as per [4])	Ref.
C1	$\mathrm{d}_{\mathrm{HWD}}$ measure (proposed measure)	O(71P+6N)	Eq. 1
C2	Interframe Difference	O(P)	[5]
C3	L1 Gray Pixel Difference	O(3P)	[5]
C4	L1 Color Pixel Difference	O(6P)	[5]
C5	Boolean Difference	O(4P)	[6]
C6	Normalized difference energy	O(5P)	[8]
C7	2-bits RGB histogram	O(3N)	[5]
C8	256 bins RGB histogram	O(9N)	[14]
C9	Weighted RGB histogram	O(9N)	[27]
C10	Kolmogorov-Smirnov statistic	O(3N)	[8]
C11	χ^2 test on intensity histogram	O(P+5N)	[27]
C12	Cross-entropy	O(P+4N)	[15]
C13	Bhattacharya distance	O(P+3N)	[8]
C14	Block-based Freund statistic	O(3P+11B)	[8]
C15	Block-based 2-bits RGB histogram	O(4NB+2B)	[5]
C16	Block-based HSV histogram	O(6NB+2B+4P)	[13]
C17	Invariant moments	O(23P)	[39]
C18	Edges Change Ratio	O(26P)	[20]
C19	Joint Probability Images	$O(2P+N^2+N)$	[18]
C20	Information Theory	$O(3P+12N^2)$	[17]

Table 2. Cut detection algorithms tested and their computational complexity.

We also tested the C1 algorithm against the product and sum combination rules:

$$d_{SUM}(t,t-1) = d_H(t,t-1) + d_W(t,t-1) + d_D(t,t-1) \tag{5}$$

$$d_{PROD}(t,t-1) = d_H(t,t-1) \times d_W(t,t-1) \times d_D(t,t-1) \tag{6}$$

The results are shown in Figure 6. It can be seen that with respect to C1, the product rule shows a slightly better while the sum rule is slightly worse. These results are expected since the product rule tends to be more precision-oriented than the sum rule, while the d_{HWD} measure can be considered as a trade off between them. At very high recall values, the precision of d_{HWD} drops faster than the other two but the overall precision, for the tested dataset, is meaningless in all the three cases.

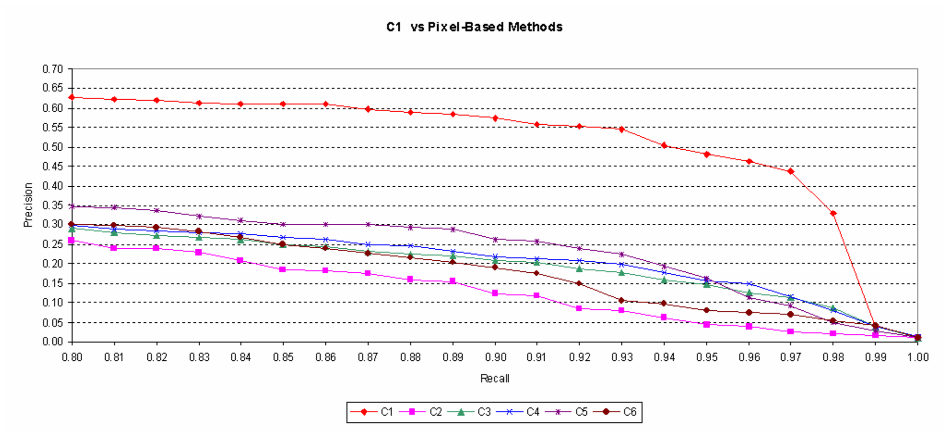
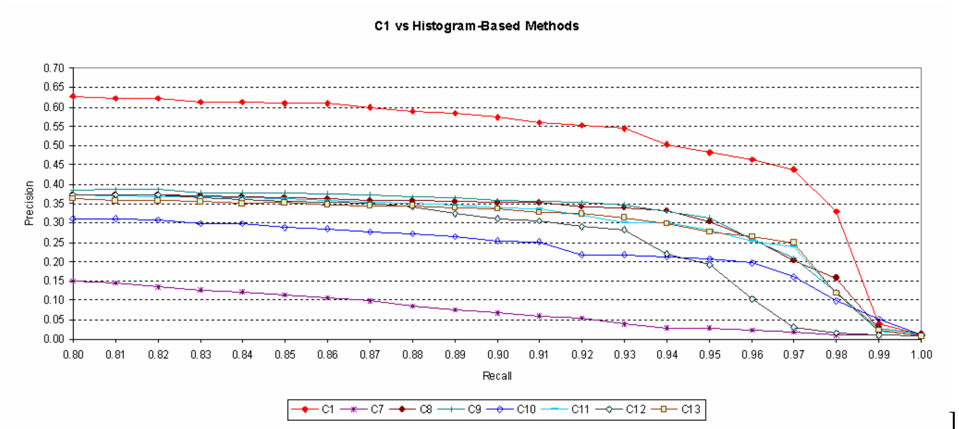


Fig. 3. Recall-Precision graphs for the pixel-based strategies by varying the detection threshold.



]

Fig. 4. Recall-Precision graphs for the block-based strategies by varying the detection threshold.

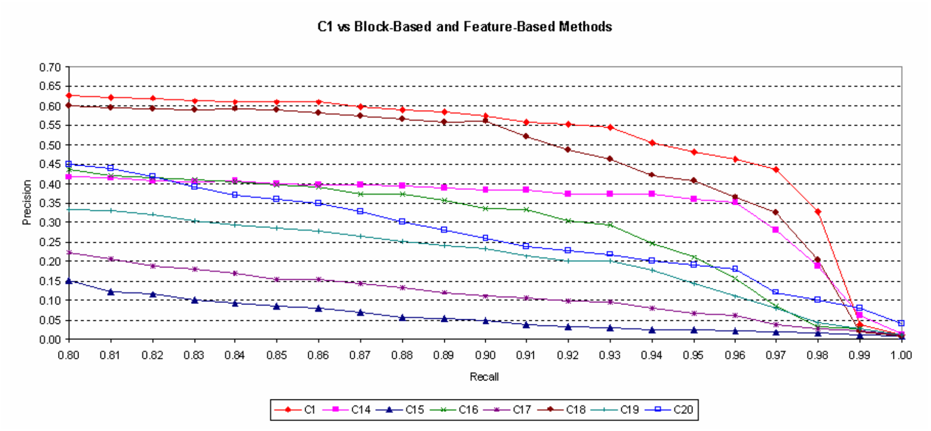


Fig. 5. The Recall-Precision graphs for the block-based and feature-based algorithms by varying the detection threshold.

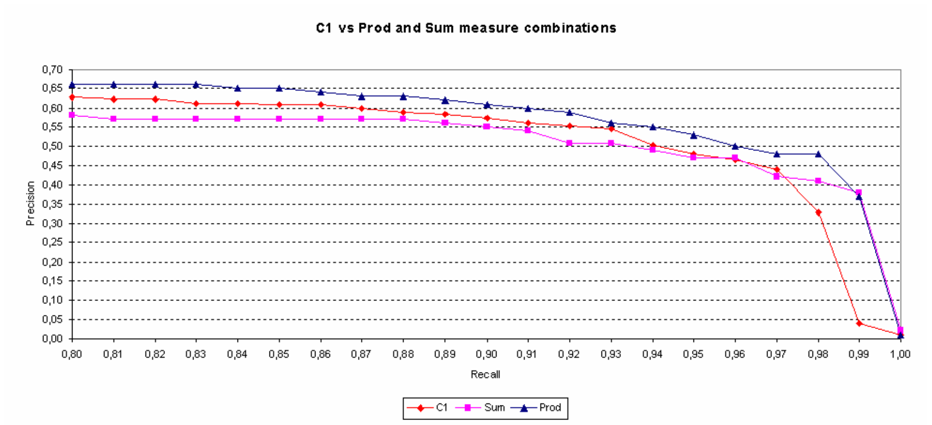


Fig. 6. The Recall-Precision graphs for the d_{HWD} measure (C1), d_{HWD} (Sum) and d_{PROD} (Prod) measures by varying the detection threshold.

3.2. Testing the Temporal Pattern Analysis (TPA)

Ideally, a good cut detection algorithm should have high recall and high precision. As it can be seen from Figures 3 to 6, as recall increases, precision decreases. Thus the choice of the optimal threshold *T* to be used with the TPA algorithm can be made by analyzing the C1 precision/recall curve, and selecting the point where precision starts to drop significantly. Based on the results of the first experiment, with a detection threshold of 0.018, this point is located at the recall value of 97%, corresponding to a precision value of about 44%. By using the TPA algorithm, recall is expected to decrease while precision is expected to increase. After experimenting several parameters combination for the buffer size *n* and the threshold T_L, they have been set to 9 elements, and 20% of *d** respectively. The lower threshold is thus dynamically adapted based on the *d** value. Table 3 shows the results of the cut detection algorithm exploiting the d_{HWD} measure on the test set using the optimal threshold and the TPA (i.e. d_{HWD}+TPA).

	d_{HWD} + TPA		d_{SUM} + TPA		d_{PROD} + TPA	
Video Name	Recall (FN)	Precision (FP)	Recall (FN)	Precision (FP)	Recall (FN)	Precision (FP)
4videos	0.94 (1)	1.00 (0)	0.89 (2)	1.00 (0)	0,94 (1)	1.00 (0)
Basketball	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)
Bugsbunny	0.87 (10)	0.83 (13)	0.88 (9)	0.95 (4)	0,99 (1)	0,91 (7)
Eeopen	1.00 (0)	1.00 (0)	0.91 (2)	1.00 (0)	1.00 (0)	1.00 (0)
Football	0.90 (3)	1.00 (0)	0.90 (3)	1.00 (0)	0,93 (2)	1.00 (0)
Forthebirds	0.96 (2)	0.96 (2)	0.80 (9)	0.97 (2)	0,98 (1)	0,94 (3)
LVEB	0.95 (2)	0.98 (1)	0.78 (9)	1.00 (0)	0,98 (1)	0,93 (3)
News	0.91 (1)	1.00 (0)	0.91 (1)	1.00 (0)	0,91 (1)	0,91 (1)
Nwanw1	0.94 (2)	1.00 (0)	0.81 (6)	1.00 (0)	1.00 (0)	0,94 (2)
Restauri	1.00 (0)	1.00 (0)	0.83 (1)	1.00 (0)	1.00 (0)	1.00 (0)
Voyager	0.83 (17)	0.94 (5)	0.69 (30)	0.98 (3)	0,89 (11)	0,94 (6)
Weezer	1.00 (0)	1.00 (0)	0.90 (8)	1.00 (0)	1.00 (0)	1.00 (0)
Average (Total)	0.94 (38)	0.98 (21)	0.86 (80)	0.99 (9)	0.97 (22)	0.96 (18)
Variance	0.0031	0.0025	0.0062	0.0005	0.0016	0.0017
Std. Deviation	0.0559	0.0500	0.0789	0.0231	0.0398	0.0389

Table 3. Precision and Recall results of the different detection strategies on the first set of videos. *T* was set to 0.018 for the d_{HWD} measure, 0.4 for the d_{SUM} measure and 0.0008 for the d_{PROD} measure. The TPA parameters were set to n=9 and T_L=0.2×d*. FN and FP indicate the false negatives and false positives respectively.

As expected, the use of the TPA algorithm dramatically increases the precision. The average recall for the d_{HWD}+TPA algorithm is 94% (±5.6%) from the previous 97%, and the precision is about 98% (±5%) from the previous 44%. Table 3 shows also the results of the cut detection algorithm exploiting the sum and product combination rules (d_{SUM}+TPA and d_{PROD}+TPA respectively). Following the same rationale as for the d_{HWD} measure, the threshold T for these two strategies has been set to 0.4 and 0.0008 respectively, while the TPA parameters remained the same as in the d_{HWD}+TPA. It can be seen that the sum combination measure shows the worst average recall with a value of 86% (±7.9%) while at the same time it exhibits the best precision with a value of 99% (±2.3%). This is due to the fact that the sum rule is more sensitive to the motion within the frame sequences: the isolated peak check in the TPA algorithm (controlled by the T_L threshold and the window analysis) often fails and thus most of the true cuts surrounded by motion are discarded. The recall can be increased by reducing the T_L threshold: experimentally we have observed that with T_L=0.5, the recall increase to 97% but the precision drops to 92%. At lower values, the precision drops rapidly. With 97% (±4%) recall and 96% (±3.9) precision, the behavior of the product measure can be considered on the overall comparable to the d_{HWD} measure. To validate the choices of the cut detection parameters, we have tested the algorithms on a completely different set of videos. Table 4 lists the videos of this second set.

Video Name	Format-W×H@FPS	Frames	hh:mm:ss	Cuts
3bears	MPEG-352×240@29.97	12,862	07:09	6
Bugslife	MPEG4-320×174@12	1,792	02:29	87
Daffyduck	MPEG-352×240@29.97	12,686	07:03	43
DonaldDuck	MPEG-352×240@29.97	11,064	06:09	107
Ds9end	MPEG-352×240@29.97	3,672	02:02	50
Generation	AVI-640×320@24	1,242	00:51	5
Ggame	MPEG-352×288@25	6,358	04:14	86
Groove	MPEG4-320×240@12	1,861	02:35	115
Making	MOV-600×240@15	1,247	01:23	40
Menace	MPEG4-480×216@12	1,784	02:28	85
MTVWhitney	MPEG4-240×176@14.96	4,339	04:49	113
Multilng	AVI-320×240@14.98	760	00:50	5
Paddle	AVI-240×180@14.87	203	00:13	3
RaymanTV	AVI-320×240@12,50	1,419	01:53	38
RoadRunner	MPEG-352×240@29.97	12,578	05:59	60
Soxmas	MPEG-240×160@30	9,374	05:12	79
Shrekaraoke	MPEG-720×480@29.97	5,086	02:49	41
Stgen	AVI-640×320@24	5,365	03:43	27
Tweety	MPEG-352×240@29.97	12,149	06:45	87
Total	--	105,841	01:08:06	1076

Table 4. List of videos used to validate the proposed cut detection algorithm.

The rationale that inspired the choice of the videos in the first data set also inspired the choice of the videos in this second data set. “3bears” is a short cartoon containing few editing effects. The video is characterized by being almost a single, uninterrupted sequence with many panning effects. “Bugslife” is a movie trailer showing very short shots and a fast paced montage. “DaffyDuck”, “RoadRunner” and “Tweety” belonging to the same production house show a similar behavior. “DonaldDuck” is an excerpt of a cartoon captured from a TV show. Several subtitles appear along the sequence. “DS9End” is a preview of a science fiction television episode. The video presents very dark scenes with many explosions and transition effects. “Generation” is an excerpt of a movie taken from a video game CD, showing dark scenes and very long shots. “GGame” and “Shrekaraoke” are two 3D cartoon sequences taken from DVDs. These videos do not present particular effects or dynamics. “Groove” is a cartoon trailer characterized by very short shots with many effects such as lightning, explosions, and strong actions. “Making” is another movie excerpts used in a video game. It shows very dark locations, and scenes with very low definition. The movie trailer “Menace”, contains many production effects as well as explosions and very highly dynamic scenes. “MTVWhitney” is a music video-clip acquired from television which exhibits very evident compression artifacts. “Multilng” is a documentary video presenting long shots without showing particularly difficult situations. “Paddle” is a very shot sequence showing some heavy chromatic changes, due to errors in the video coding. “RaymantTV” is a TV show preview. It shows a fast paced montage, with short shots, camera zoom, and matte effects. “Soxmas” is an episode of the “South Park” cartoon series. The main characteristic of this video is that due to the particular style of the cartoon, the frames exhibit very few details with large uniformly colored areas. “Stgen” is a trailer of a science fiction movie. Like many movies of this genres, it shows very dark scenes, explosions, and fast camera motion. Table 5 reports the detailed cut detection results on these videos for the d_{HWD}+TPA, d_{SUM}+TPA and d_{PROD}+TPA algorithms.

	d_{HWD} + TPA		d_{SUM} + TPA		d_{PROD} + TPA	
Video Name	Recall (FN)	Precision (FP)	Recall (FN)	Precision (FP)	Recall (FN)	Precision (FP)
3bears	1.00 (0)	0.88 (1)	0.67 (2)	1.00 (0)	1.00 (0)	0.86 (1)
bugslife	0.83 (15)	0.99 (1)	0.68 (28)	0.98 (1)	0.86 (13)	0.96 (3)
Daffyduck	0.95 (2)	0.84 (8)	0.88 (5)	0.86 (6)	1.00 (0)	0.84 (8)
DonalDuck	0.93 (8)	0.98 (2)	0.80 (21)	0.99 (1)	0.95 (5)	0.94 (6)
Ds9end	0.68 (16)	0.77 (10)	0.52 (24)	0.87 (4)	0.72 (14)	0.73 (13)
Generation	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)
GGame	0.99 (1)	1.00 (0)	0.87 (11)	1.00 (0)	0.91 (8)	0.99 (1)
Groove	0.72 (32)	0.97 (3)	0.56 (51)	1.00 (0)	0.77(26)	0.96 (4)
Making	0.90 (4)	1.00 (0)	0.83 (7)	1.00 (0)	0.93 (3)	1.00 (0)
Menace	0.79 (18)	0.99 (1)	0.55 (38)	1.00 (0)	0.82 (15)	0.97 (2)
MTVWhitney	0.82 (20)	0.96 (4)	0.71 (33)	0.99 (1)	0.86 (16)	0.96 (4)
Multilng	1.00 (0)	1.00 (0)	0.80 (1)	1.00 (0)	1.00 (0)	1.00 (0)
Padule	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)	1.00 (0)
RaymanTV	0.92 (3)	1.00 (0)	0.53 (18)	1.00 (0)	0.89 (4)	0.92 (3)
RoadRunner	0.92 (5)	0.98 (1)	0.88 (7)	1.00 (0)	0.92 (5)	1.00 (0)
Soxmas	0.96 (3)	0.92 (7)	0.92 (6)	0.95 (4)	0.99 (1)	0.94 (5)
Shrekaraoke	0.88 (5)	0.97 (1)	0.79 (9)	1.00 (0)	0.90 (4)	1.00 (0)
Stgen	0.96 (1)	0.87 (4)	0.67 (0)	1.00(0)	1.00 (0)	0.82 (6)
Tweety	0.97 (3)	0.90 (9)	0.81 (16)	0.92 (6)	0.98 (2)	0.88 (12)
Average (Total)	0.91 (136)	0.95 (52)	0.76 (286)	0.98 (23)	0.92 (116)	0.94 (68)
Variance	0.0093	0.0044	0.0231	0.0020	0.0070	0.0058
Std. Deviation	0.0967	0.0667	0.1519	0.0445	0.0839	0.0759

Table 5. Precision and Recall results of the proposed detection algorithm on the first set of videos. T was set to 0.018 for the d_{HWD} measure, 0.4 for the d_{SUM} measure and 0.0008 for the d_{PROD} measure. The TPA parameters were set to n=9 and T_L=0.2×d*. FN and FP indicate the false negatives and false positives respectively.

On this set of videos, the d_{HWD}+TPA algorithm is able to detect 91% (±9.7%) of the cuts with precision at 95% (±6.7%). Only three videos show recall results below 80% (“Groove”, “Menace” and “Ds9end”). These videos contain many repetitive lightning and explosion effects near the actual cuts that make it difficult to detect them. The flash removal algorithm is unable to remove these effects since they are not isolated, but are repeated several times and very quickly. This can be seen also from the results of the other two algorithms. In 14 out of the 19 videos the precision of the d_{HWD}+TPA algorithm is well above 95%, four of the remaining video have precision above 80% and only one has a lower precision (“Ds9end”). The behaviour of the other two algorithms is also confirmed on this dataset. Again, the d_{SUM}+TPA algorithm shows the worst recall result (76%) with the highest precision value (98%): with T_L=0.5, the recall increase to 90% but the precision drops to 92% and several videos have very low precision or recall (e.g. about 70% for the “ds9end” sequence). The results of the d_{PROD}+TPA algorithm are comparable to those of the d_{HWD}+TPA algorithm (92% against 91% for the recall and 94% against 95% for the precision). This seems to suggest that the d_{HWD} measure and the product measure are similarly effective in discriminating the frame differences. However the product combination rule is very sensitive to errors [38]: if one of the three frame difference measures is unable to distinguish two frames belonging to two different shots, the overall measure is wrong. On the contrary, the d_{HWD} measure is less sensitive to errors on a single feature since it requires that at least two differences to be wrong (i.e. the majority of the measures are wrong).

4 Conclusions

In this paper we have presented a new algorithm for cut detection, exploiting an innovative and robust frame difference measure. First of all, we have shown that the new measure, based on a combination of different visual features, exhibits better performances while obtaining higher precision detection compared to several other methods present in the literature even using a very simple approach based on a single decision threshold. This demonstrates that the combination of several visual clues is best compared to the methods that use only one single visual clue. Next we have presented the new cut detection algorithm, which implements a temporal pattern analysis and flashes removal. The algorithm has been tested on two heterogeneous and complex video sets. Results show that the d_{HWD}+TPA algorithm is able to detect most of the cuts with a very high average precision (98% and 95% respectively) and high average recall (94% and 91% respectively). Instead of performing multi-modal cut detection with both video and audio data, only visual content is considered due to the synchronization problem that audio data pose. Usually, audio and visual changes will not occur exactly at the same time, but the audio of a given sequence continues over the next one by a few seconds [40]. Moreover, it was also Sundaram et al. [41] who observed that video sequences exhibit audio changes (variations not related to shot boundaries) within a video shot more often than visual changes (21% against 10%). Also a small amount of video sequences exhibit both visual and audio changes within a shot (4%). Nonetheless, in a future research we plan to extend our algorithm to deal with multimodal video data.

Acknowledgements

The author would like to thank Prof. Raimondo Schettini, the Imaging & Vision Laboratory at DISCo for supporting this work; the reviewers and editors for their insightful feedbacks that make it possible to greatly improve the paper.

References

[1] N. Dimitrova, H. Zhang, B. Shahraray, M. Sezan, T. Huang and A. Zakhor, “Applications of video-content analysis and retrieval”, *IEEE MultiMedia*, 9(3):44-55, 2002.

[2] A. Hanjalic, “Shot-Boundary Detection: Unraveled and Resolved”, *IEEE Transaction on Circuits and Systems for Video Technology*, 12(2):90-105, 2002.

[3] R.W. Lienhart, “Comparison of automatic shot boundary detection algorithms”, *Proc. SPIE Storage and retrieval for Image and Video Databases VII*, 3656:290-301, 1998.

[4] S. Lefevre, J. Holler & N. Vincent, “A review of real time segmentation of uncompressed video sequences for content-based search and retrieval”, *Real Time Imaging*, 9:73-98, 2003.

[5] Y. Tanaka & A. Nagasaka, “Automatic Video indexing and full-video search for object appearances”, *Int. Conf. on Visual Database Systems*, 113-127, 1991.

[6] H.J. Zhang, A. Kankanhalli and W.S. Smoliar, “Automatic partitioning of full-motion video”, *Multimedia Systems*, 1(1):10-28, 1993.

[7] U. Gargi, R. Kasturi & S.H. Strayer, “Performance Characterization of Video-Shot-Change Detection Methods”, *IEEE Transaction on Circuits and Systems for Video Technology*, 10(1):1-13, 2000.

[8] W. Ren, M. Sharma & S. Singh, “Automated Video Segmentation”, *Proc. 3rd Int. Conf. on Information, Communications & Signal Processing*, 2001.

[9] I.K. Sethi & N. Patel, “A Statistical Approach to Scene Change Detection”, *Proc. of IS&T/SPIE Conf. on Storage and Retrieval for Image and Video Databases III*, 2420:329-339, 1995.

[10] A. Hanjalic, R.L. Lagendijk & J. Biemond, “A novel video parsing method with improved thresholding”, *Proc. 3rd Conf. of the Advanced School for Computing and Imaging*, 3-7, 1997.

[11] D. Zhang, W. Qi & H.J. Zhang, “A New Shot Boundary Detection Algorithm”, *Lecture Notes in Computer Science*, 2195:63-70, 2001.

[12] G. Fernandez & J. Mas, “Video shot boundary detection based on color histogram”, Notebook Papers TRECVID2003, Gaithersburg, Maryland, NIST, 2003.

[13] S.M. Lee & M.C. Ip, “A robust approach for camera break detection in color video sequences”, *IAPR Int. Work. on Machine Vision Applications*, 502-505, 1994.

[14] D. Pye, N.J. Hollinghurst, T.J. Mills & K.R. Wood, “Audio-visual segmentation for content-based retrieval”, *Proc. 5th Int. Conf. on Spoken Language Processing*, 1998.

[15] S. Kim & R.H. Park, “A novel approach to scene change detection using a cross entropy”, *IEEE Int. Conf. on Image Processing*, 3:937-940, 2000.

[16] Z. Cernekova, C. Nikou & I. Pitas, “Shot detection in video sequences using entropy-based metrics”, *Proc. IEEE Int. Conf on Image Processing*, 421-424, 2002.

[17] Z. Cernekova, I. Pitas, C. Nikou, “Information Theory-Based Shot Cut/Fade detection and Video Summarization”, *IEEE Trans. on Circuits and Systems for Video Technology*, 16(1):82-91, 2006.

[18] Z.N. Li & J. Wei, “Spatio-temporal joint probability images for video segmentation”, *IEEE Int. Conf. on Image Processing*, 2:295-298, 2000.

[19] X. Wen, T.D. Huffmire, H.H. Hu & A. Finkelstein, “Wavelet-Based Video Indexing and Querying for a Smart VCR”, *Multimedia Systems*, 7 (5):350-358, 1999.

[20] R. Zabih, J. Miller & K. Mai, “A feature-based algorithm for detecting and classifying scene breaks”, *Proc. ACM Multimedia '95*, 189-200, 1995.

[21] Y. Abdeljaoued, T. Ebrahimi, C. Christopoulos & I. M. Ivars, “A new algorithm for shot boundary detection”, *Proc. of the 10th European Signal Processing Conference*, 151-154, 2000.

[22] G. Boccignone, M. De Santo & G. Per cannella, “An algorithm for video cut detection in MPEG sequences”, *Proc. IS&T SPIE, Storage and Retrieval for Media Databases*, 523-530, 2000.

[23] S.H. Han & I.S. Kweon, “Shot detection combining Bayesian and structural information”, Proc. *SPIE Storage and Retrieval for Media Databases*, 4315:509-516, 2001.

[24] X. Fu, J. Zeng, “An efficient shot boundary detection method based on the local color features of interest points”, *Proc. IEEE Second Int. Symp. on Electronic Commerce and Security*, 25-28, 2009.

[25] T. Barbu, “Novel automatic video cut detection techniques using Gabor filtering”, *Computer and Electrical Engineering*, 35:712-721, 2009.

[26] V. Chasanis, A. Likas, N. Galatsanos, “Simultaneous detection of abrupt cuts and dissolves in videos using support vector machines”, *Pattern Recognition letters*, 30:55-65, 2009.

[27] A. Dailianas, R.B. Allen and P. England, “Comparison of automatic video segmentation algorithms”, *Proc SPIE Photonics West*, 2615:2-16, 1995.

[28] R.M. Ford, C. Robson, D. Temple & M. Gerlach, “Metrics for shot boundary detection in digital video sequences”, *Multimedia Systems*, 8:37-46, 2000.

[29] A. F. Smeaton, P. Over, A.R. Doherty, “Video shot boundary detection: Seven years of TRECVid activity”, *Computer Vision and Image Understanding,* doi:10.1016/j.cviu.2009.03.011, 2009.

[30] G. Ciocca. & R. Schettini, “A relevance feedback mechanism for content-based image retrieval”, *Information Processing and Management*, 35:605-632, 1999.

[31] G. Ciocca, I. Gagliardi & R. Schettini, “Quicklook^2: An Integrated Multimedia System”, *Int. Journal of Visual Languages and Computing,* Special issue, 12:81-103, 2001.

[32] R.C. Gonzales, R.E. Woods, “Digital Image Processing”, Prantice Hall, 135-136, 2007.

[33] W. JY. Ma, B. S. Manjunath, “Texture features and learning similarity”, *IEEE Computer Vision and Pattern Recognition Conference*, 425-430, 1996.

[34] M.J. Swain, D.H. Ballard, “Color indexing”, *International Journal of Computer Vision*, 7(1):11–32, 1991.

[35] J. Kittler, M. Hatef, R. P.W. Duin, J. Matas, “On combining classifiers*”, IEEE Transaction on Pattern Analysis and Machine Intelligence*, 20(3):228-239, 1998.

[36] Z. Stejić, Y. Takama, K. Hirota, “Mathematical aggregation operators in image retrieval: effect of retrieval performances and role in relevance feedback”, *Signal Processing*, 85: 297-324, 2005.

[37] TREC-Video collection: http://www-nlpir.nist.gov/projects/trecvid/

[38] D. Tax, R. Duin, M. Breukelen, “Comparison Between Product and Mean Classifier Combination Rules”, Proc. Workshop on Statistical Pattern Recognition, 165—170, 1997.

[39] F. Arman, R. Depommier, A. Hsu & M.Y. Chiu, “Content-based browsing of video sequences”, *ACM Int. Conf. on Multimedia,* 97-103, 1994.

[40] S. Pfeiffer, Fisher S. & W. Effelberg, “Automatic Audio Content Analysis”, *Proc. ACM Multimedia ’96*, 21-30, 1996.

[41] H. Sundaram & Chang Shih-Fu, “Video Analysis and Summarization at Structural and Semantic Levels”, *Multimedia Information Retrieval and Management*, 75-94, 2003.