Technologie-Zentrum Informatik

Technical Report 41

Blink - Towards A Reliable Movie Shot Boundary Detection

Christoph Brachmann and Thorsten Hermes

TZI-Bericht Nr. 41
2007

Universität Bremen

TZI-Berichte

Herausgeber:
Technologie-Zentrum Informatik
Universität Bremen
Am Fallturm 1
28359 Bremen
Telefon: +49-421-218-7272
Fax: +49-421-218-7820
E-Mail: info@tzi.de
http://www.tzi.de

ISSN 1613-3773

Blink - Towards A Reliable Movie Shot Boundary Detection

Christoph Brachmann and Thorsten Hermes

Center for Computing Technologies - University of Bremen, Am Fallturm 1,
Bremen, Germany
chris09@tzi.de, hermes@tzi.de

Abstract. Despite of intense research in shot boundary detection methods hardly any standards have been established that would be fairly reliable for most video processing applications. On the other hand, automatic analysis of full-length feature movies has experienced an increasing interest during the past few years and among others, it requires a reliable shot boundary detection. Thus, the following work aims at reviewing current shot boundary detection methods in order to determine the most suitable technique for movie footage. Next to an analysis of shot boundaries in six Hollywood action movies, several current shot boundary detection methods have been thoroughly tested on these movies.

1 Introduction

The research on automatic detection of shot boundaries in digital video, like hard cuts, fades, dissolves, wipes, etc. has been actively done for more than 15 years now, however no methodology has proved yet a 100 % (or at least 99.x %) reliability throughout various types of video. From the related work discussed in the next section it can be concluded that at the most, results achieve precision and recall values at around 95–98 % for cuts, and 90–95 % for gradual transitions[1]. However, these values are only achieved with a particular selection of thresholds of the corresponding methods applied to a particular set and type of video footage and so it is uncertain how these methods will perform in a different environment. Very often slight changes of the thresholds cause significantly different results without making clear which method or which threshold is suitable for a specific application.
Therefore, there are two possibilities of dealing with automatic shot boundary detection techniques. First, with the best combination of methods and thresholds investigated for a test set of various video types one can apply these methods to the target video footage with an expected error rate somewhere between 2 % and 20 %. The other way – which is the approach of this work – is to adjust the detection methods for a specific type of footage in order to lower the range of uncertainness concerning the expected accuracy.

[1] See Section 4 for the definition of the terms *precision* and *recall*.

In the past few years, there has been an increasing interest in information retrieval from full-length feature movies [7, 19, 38, 2, 45] wherein a shot boundary detection was one of the essential components of the particular retrieval systems. Hence, this work investigates some of the existing detection methods in order to provide further suggestions for shot boundary detection techniques when applied to full-length feature movies. Due to time constraints this work will focus on the Hollywood action movie genre but future work will analogously deal with other movie genres.
The following report is organized as following: in section 2 an overview of the very active field of research of shot boundary detection methods is given. In section 3 shot boundary statistics of six Hollywood action movies are given which help to determine the strategy for selecting a specific detection method. Following to that a selection of existing detection techniques is provided in section 4 along with detection results for the movies presented in section 3. Finally, the results are discussed in section 5 also providing perspectives for future work in this topic.

2 Related Work

There has been a lot of research on shot boundary detection techniques. The most recent detailed overview is given in [13]. The following section will just focus on presenting samples of the most commonly used techniques.
Due to their characteristic properties, abrupt shot transitions, i. e. hard cuts, are generally modeled as sudden visual discontinuities in the video signal while gradual shot transitions, like dissolves, fades or wipes, are usually detected by a continuous, regular change of a specific visual feature. The methods vary in the selection of the relevant features and metrics which will be discussed in the following.

2.1 Hard Cut Detection Methods

Pixel Intensity Differences The simplest technique to determine a visual discontinuity is to calculate average pixel intensity differences between two adjacent video frames. Hanjalic [13] and Boreczky et al. [4] refer to several older works based on this technique but even with modifications like smoothing filters or adaptive thresholds this technique has turned out to perform insufficiently due to its sensitivity to motion or illumination changes. Thus, in recent publications this technique is hardly mentioned for the detection of cuts except if the results of the shot detection are not crucial to a system's performance like in [7], or if it is part of a multi-feature detector [53, 31].

Histogram-based Statistics Until today the most popular discontinuity feature is a gray-level or color histogram difference between two adjacent frames (see fig. 1). The main reason is that despite of its simplicity the computational load of histogram-based methods is low while the performance is relatively good.

Hence, many variations of this technique exist.
Especially in older works gray-level histograms have been used for the difference calculations [48, 14, 5, 9, 50, 23, 53, 38]. Meanwhile experiments also have been done with other color spaces like RGB [22, 52, 49], HSV [35, 22, 43, 8], YUV [18, 33, 10, 28], or the so-called opponent color representation of RGB [56]. The number of bins used for histograms varies from 24 to 256 bins.
The basic idea of histogram-based cut detection is to add up all bin-wise absolute differences and normalize this sum by the number of bins, as done in [48, 14, 9, 44, 53, 22]. This corresponds to the mathematical L_1 *norm*. Niblack et al. [27] applied the L_2 *norm* to a shot boundary detection algorithm, while Stricker et al. [40] argued that both norms do not fully satisfy the notion of similarity and proposed the L_∞ *norm* for shot boundary detection. Nagasaka et al. [24] used the χ^2-*test* which was adopted by several works [29, 55]. Miene et al used a similar metric which they refer to as *Gray Histogram* X^2 [23]. Cabedo et al. [5] introduced the *cosine measure* and compared it to the above mentioned L_1, L_2, L_∞ and χ^2 metrics. Another metric is the *histogram intersection* defined by Swain et al. [42] and used by [35, 49, 52] for cut detection.
Despite of this broad range of variations there is no selection of color space, bin number or metric that would be clearly bad or good. In general, histogram-based cut detection algorithms are able to detect around 90 % (± 5 %) of hard cuts. The more influential and therefore very subtle aspect about these methods is to determine the appropriate thresholding technique in order to distinguish between cuts and non-cuts (see section 2.3).

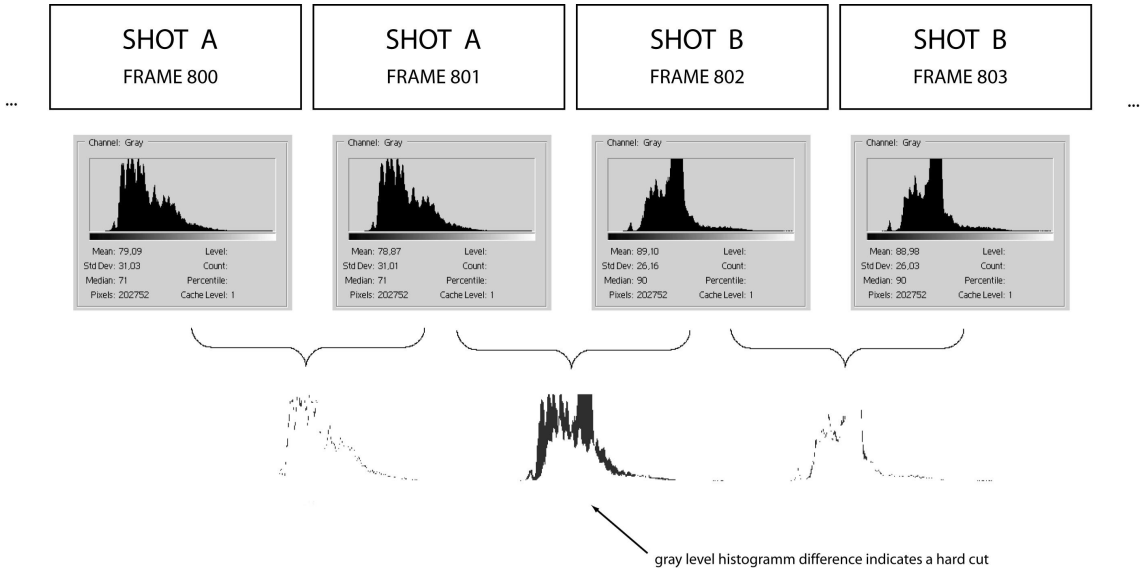


Fig. 1. Hard cut detection by histogram differences. The bigger histogram difference in the middle indicates a hard cut between the 2nd and the 3rd frame.

Edge-based Statistics After computing edges by applying a filter like Sobel or Canny, edge direction histograms can be computed as done by [8]. Petersohn [31] used *edge energy* which is the sum of edge intensities in a frame. Mainly due to the higher efficiency of histogram-based approaches edge-based methods have rarely been considered for cut detection.

Color Anglograms Zhao et al. [54] developed a shot boundary detection based on *color anglograms*. For each HSV channel of the source frame they connected pixel blocks of similar intensity to each other by a triangulation algorithm. Then they created histograms counting the angles of the resulting connections for each possible HSV color. Cuts were detected by comparing these anglograms for adjacent frames.

Motion Compensation Porter et al. [32] used block-matching motion compensation to compute inter-frame differences (block size: 32×32). A similar approach was done by Hanjalic [13]. The more blocks were not matched to each other from frame to frame the higher the probability of a hard cut in between. Petersohn [31] also included such a technique in his multi-feature cut detector. Whitehead et al. [47] checked the loss of inter-frame features detected by the Lucas and Kanade tracker [21]; a sudden high loss of features indicated a cut.

Wavelet Statistics Wavelets provide representations of images with spatial and frequency information. Miene et al. [23] proposed to apply the Fast Fourier Transform to each frame. As part of a multi-feature cut detection Ciocca et al. [8] applied a low-pass and a high-pass filter to different resolutions of the image. Tahaghoghi et al. [43] used the coefficients of a 6-tap Daubechies wavelet transform of each frame.

Spatio-Temporal Coherency The basic idea of spatio-temporal-based shot detection is to align the same sub-region of each frame, for example a pixel row, next to each other. This results in a new 2D image with specific patterns at shot boundaries, as for example a vertical discontinuity at a hard cut (see fig. 3), which are then detected by image segmentation. Ngo et al. [26, 25] used the center pixel row and column and a diagonal scan of each frame. Kim et al. [17] and Seo et al. [37] who referred to this technique as *visual rhythm*, experimented with different region forms and selected the diagonal scan as most effective. Guimarães et al. [11] aligned whole frame histograms as vertical gray-scale lines.

Information Theory-Based Statistics Černeková et al. [6] computed the *mutual information* of a video signal and searched for sudden discontinuities in order to detect cuts.

Multi-Feature Algorithms Petersohn [31] combined edge energy and pixel intensity differences with motion compensation. Ciocca et al. [8] used color and edge direction histograms combined with wavelet statistics. After computing frame

and block wise histogram differences Qi et al. [33] compared inter-frame camera motion. Ren et al. [36] set up two machine learning algorithms (k-nearest neighbor (kNN) and neural network) scanning up to 139 computed features including standards, like difference mean, histogram difference, χ^2-test or The Earth Mover's Distance (EMD), and own ones, like b-coefficient (based on image moments) or c-coefficient (based on gray-level medians). Boccignone et al. [3] developed a novel inter-frame similarity metric: *attention consistency*. Adopting the idea of human attentive vision at images, a path with so-called "focus of attention" (FOA) regions was computed by intensity, color and orientation contrast (see fig. 2). Then, this saliency map was tested for spatial, temporal and visual consistency in order to detect cuts (and dissolves).

Fig. 2. Boccignone et al. [3] computed a saliency map for each frame which adopts the idea of human attentive vision. From the 2nd to 3rd frame the significant change of the saliency map indicates a hard cut.

2.2 Gradual Transition Detection Methods

Pixel Intensity Differences Truong et al. [44] detected fade-out/fade-in pairs by searching for parabolic shapes in the luminance variance and mean curves. Additionally, they took the corresponding 1st order derivative curves into account, and they also used these in order to detect dissolves. Ciocca et al. [8] just considered the pixel intensity standard deviation but only from the central frame's region detected by a low-pass filter. Han et al. [12] determined blocks of interest (BOI), that are blocks with higher color variance relative to frame variance, and used the variance correlation of corresponding BOIs of two adjacent frames in order to detect dissolves. Joyce et al. [15] take even three subsequent frames (A, B, C) into consideration. For frame pairs (A, B) and (B, C) correlation distances are calculated based on inter-frame intensity differences. The difference of these two correlation distances is then considered as a feature for dissolve detection.

Histogram-based Statistics In contrast to cut detection, only few approaches included a histogram-based technique in order to detect gradual transitions. Mas et al. [22] filtered the color histogram difference curve by morphological operators and combined it with mean pixel block intensities. Zhai et al. [52] used

the histogram intersection but both, Mas et al and Zhai et al, did not detect the exact type of gradual transition. Cabedo et al. [5] used his own developed *cosine measure* for histogram-based detection of dissolves. Joyce et al. [15] calculated the correlations of two adjacent inter-frame histogram differences in order to detect wipes.

Edge-based Statistics Zabih et al. [51] introduced the *edge change ratio* (ECR) which indicated the ratio of appearing and disappearing edges. Lienhart [20] used a ratio of strong and weak edges while Song et al. [39] simply added up edge intensities in each frame and searched for U-shapes in this *edge energy* curve.

Motion Compensation Hanjalic [13] used a block-matching algorithm for 4×4 pixel blocks and added up the differences of the best matches in a frame pair. Specific changes of the difference curve indicated dissolves or wipes. Porter et al. [32] used double block-matching motion compensation: one for blocks covering the whole frame and the other only for blocks of interest, i. e. blocks with higher color variance. Tracking these 32×32 pixel blocks over time both calculations were compared to each other in order to indicate fades.

Spatio-Temporal Coherency The spatio-temporal slices of Ngo et al. [26, 25] and the visual rhythm segementation of Kim et al. [17] and Seo et al. [37] allowed a detection of wipes and dissolves (see fig. 3). Guimarães et al's [11] histogram-based spatio-temporal analysis detected only fades.

Information Theory-Based Statistics Černeková et al. [6] searched for parabolic shapes in the *joint entropy* curve in order to detect fade-ins/fade-outs.

Multi-Feature Algorithms Zhang et al. [53] combined histogram differences with average intensity differences in order to find gradual transitions, however without detecting the exact transition type. In their dissolve detector Su et al. [41] looked for a monotonous intensity change and checked the motion vectors. Petersohn [30] combined several features in order to detect dissolves: edge energy, image and histogram differences, dissolve linearity and evenness, and global motion. The wipe detection of Petersohn [31] comprised evenness, image differences and Hough transforms. Ren et al's huge machine learning system [36] mentioned above detected fades and dissolves. The kNN-classifier of Qi et al. [33] detected gradual transitions but not the exact type. Finally, as mentioned before Boccignone et al. [3] used their attention consistency measure for detecting dissolves.

2.3 Classification Techniques

The crucial element of all shot boundary detection methods remains the classification decision which usually means to select an appropriate parameter or threshold for a specific signal. Thus, different approaches exist for this problem. In a basic scenario a global threshold is heuristically set based on some test

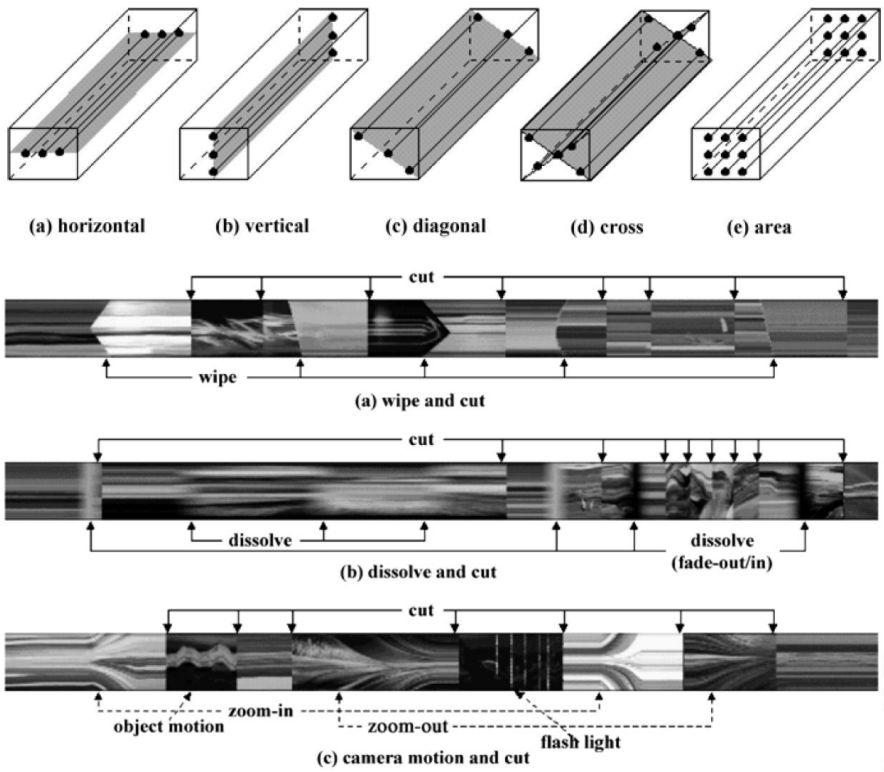


Fig. 3. For spatio-temporal analysis a specific sub-region of each frame is vertically aligned in a new image. Below, three such images have been produced with diagonal sampling of the source video. Shot boundaries cause specific patterns in the *visual rhythm*. Source: Kim et al. [17].

data. In order to adapt a threshold technique to various types of video footage Miene et al. [23] used a percentage value. The more common way is to move a sliding window over n frames and to calculate a local threshold. Then for example, an outlier is detected if a feature value is maximum in the window and if it is x times greater than the 2nd-largest value in the window [48, 44]. Often the mean value of the window (or the left and right side of the window) is taken as a threshold [14, 9, 44, 53, 18, 56]. Zhang et al. [53] updated the local threshold only if the new mean did not vary too much in order to omit false positives. Joyce et al. [15] filtered the input signal by subtracting the mean of past values. Yusoff et al. [50] proved that such locally adaptive thresholds usually perform better than global ones.

Some approaches try to integrate a higher-level statistically founded classification technique. Pardo [28] calculated a probability for a shot boundary based on the previous n frames. Vascencelos et al. [46] introduced a Bayesian frame-

work that sets the threshold according to pre-computed frequencies of usual shot length in movie trailers. Whitehead et al. [47] defined a rule system to determine an automatic thresholding level.
Especially in complex, multi-feature algorithms with several decision steps the configuration of the whole system becomes extremely difficult and subtle. Thus, several approaches include a machine learning algorithm for training, like *support vector machines (SVM), neural networks* or *k-nearest neighbor (kNN)* classifiers [49, 10, 25, 36, 33].

2.4 Feature Extraction from MPEG Streams

Several approaches deal directly with compressed video which is usually an MPEG stream. Features like global motion, YUV image or histogram differences are computed by extracting DC images from I-, P- and B-frames of an MPEG video [5, 55, 39, 13, 15]. Jun et al. [16] even used MPEG macroblocks as a feature itself in order to detect dissolves. Kim et al. [17] and Seo et al. [37] used the DC images for computing the *visual rhythm* for spatio-temporal analysis. Feature extraction from MPEG stream often allows a real-time detection of any type of shot boundaries for two reasons: first, complex decompression is omitted and second, pre-computed encoded information is used as a feature. By that the computational load of a shot detection method is heavily reduced. However, an MPEG-based shot detection is naturally restricted to that type of video footage.

2.5 Testing Full Movies

Relating to the specific domain of movies it is worth noting that only very few of the aforementioned related work tested their methods with full-length movies [41, 15]. Due to time and feasibility constraints often only excerpts of movies, for example 10-minute-sequences, are tested [39, 12, 3, 37]. However, in order to provide normalized quality results for a movie shot boundary detection it is by far more helpful to test the full movie especially because movies appear as (relatively) standardized video footage.

3 Nature of Movie Shots

As mentioned above within the scope of this work we focus on Hollywood action movies. Shot boundaries for six such movies were manually annotated through which valuable information for the automation of shot boundary detection can be derived. After a definition of the various boundary types, statistics for the annotated movies are given. Some special borderline cases are then discussed in the last subsection.

3.1 Types of Shot Boundaries

Shot boundaries are usually grouped into *abrupt* and *gradual* transitions. Abrupt transitions are made of significantly changes from frame to frame whereas gradual transitions increasingly hide one shot and introduce a new one by slight inter-frame changes.

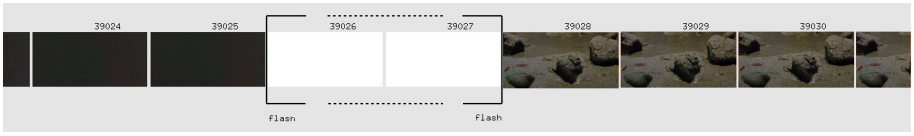


Fig. 4. Some movies use two cuts very close to each other (often with white frames in between) which causes a flashing effect. Hence, we refer to such editing effects as *flash* transitions.

Abrupt Transitions The most popular transition is a *(hard) cut* which is simply the result of concatenating two shots with each other (see fig. 1). In fact, a cut itself cannot be noticed by humans as it lasts for a fraction of a second but it is noticed because of differences between the adjacent shot contents. Furthermore, in contrast to all previous research work mentioned above we introduce another abrupt transition, the *flash* transition. During the manual annotation of Hollywood movies we observed a special transition type using two cuts very close to each other (see fig. 4). Mostly, with white frames in between this sort of transition causes a flashing effect which is not perceived by viewers as two cuts but as one special transition. Interestingly, several shot detection approaches included flash-sensitivity in order to avoid false positives [53, 31, 34, 11, 6] but in these works the flash was not explicitly considered to separate two shots.

Gradual Transitions Gradual transitions can be subdivided into *dissolves* and *wipes* (see fig. 5). Dissolves gradually *blend* a new shot into the previous one while wipes gradually *replace* the previous shot by the new one. Blending from or into one-colored frames (usually black, sometimes white) is usually treated as a special case of dissolve and so we refer to these as *fade-in* and *fade-out*. Wipes are usually further subdivided depending on their attributes (direction, geometrical shape, speed etc.) but because of their rare occurrence in our movies we do not specify these.

3.2 Shot Statistics

We manually annotated six different action movies adding up to 10h44m video footage with 14,608 shot boundaries. In table 1 the absolute frequencies of shot boundaries for all movies are given. The most striking aspect about these figures is the high frequency of cuts because hard cuts are not only the most popular

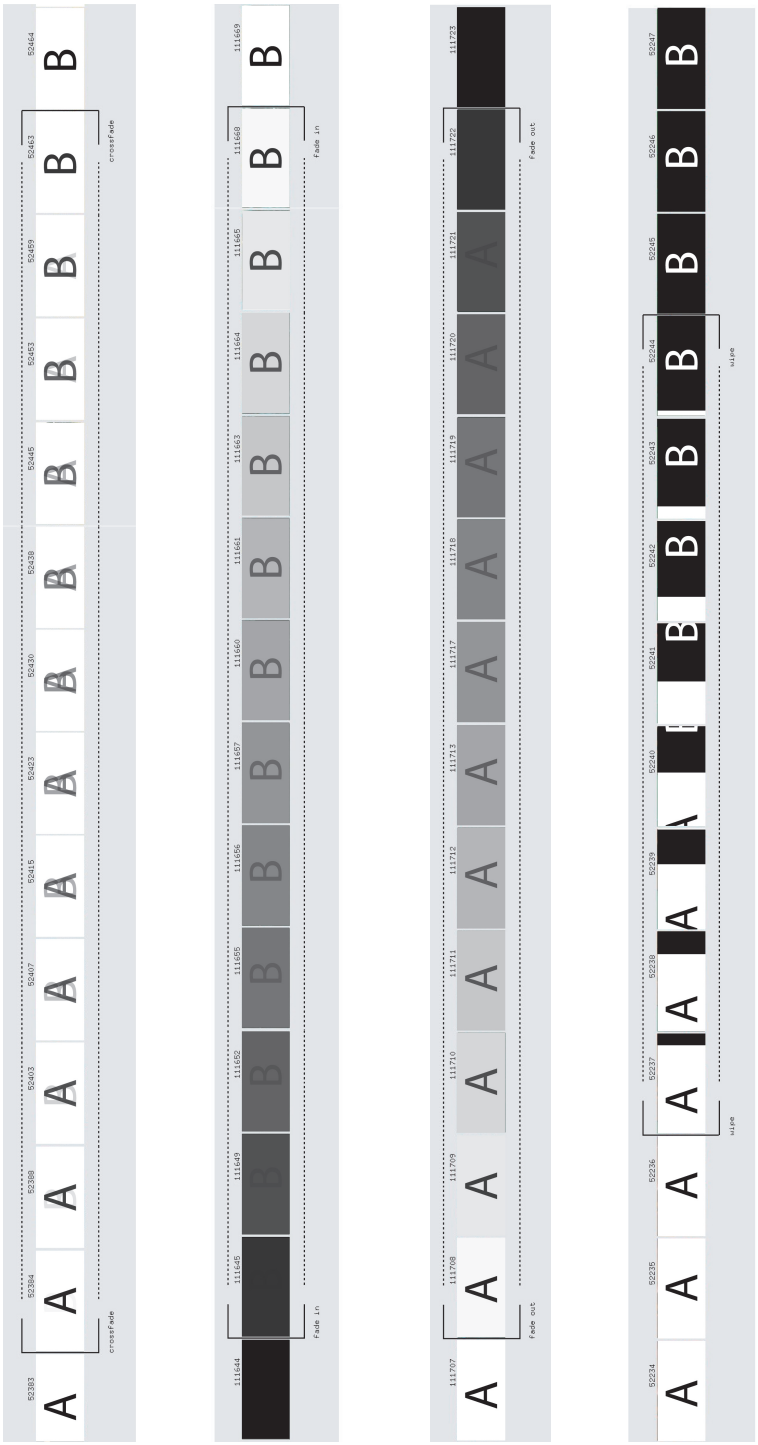


Fig. 5. (a) A *dissolve* blends two shots into each other. Blending from or into a one-colored shot is separately treated as *fade-in* (b) and *fade-out* (c), respectively. (d) shows a horizontal *wipe* transition.

Movie Title	C	FL	D	FI	FO	W	Total	Tr/sec
Bad Boys (1995)	2559	3	1	1	8	0	2572	0.38
Blade (1998)	2524	18	5	4	18	0	2569	0.37
Charlie's Angels (2000)	1798	0	18	3	3	20	1842	0.33
Terminator 2 (1991)	2654	13	8	1	5	1	2682	0.30
The Transporter (2002)	2391	0	19	3	4	0	2417	0.45
Transporter 2 (2005)	2487	2	17	8	10	2	2526	0.50
all movies	14413	36	68	20	48	23	14608	0.38

Table 1. Shot boundary type frequencies in six action movies. C: cut, FL: flash, D: dissolve, FI: fade-in, FO: fade-out, W: wipe. Tr/sec is the ratio of the total number of transitions and the movie length in seconds.

transition type but they virtually make up nearly all transitions in our annotated movies. This becomes even more obvious when looking at the relative frequencies listed in table 2. In total, 98.67 % of all shot boundaries in the tested movies are hard cuts. Evenmore, it can be concluded from the location distribution diagrams in fig. 6 that some of the rare non-cuts (non-green vertical lines) are located at the beginning and end of movies. That is because title shots are more often joint with gradual transitions, in particular fade-ins (gray), fade-outs (black) and wipes (blue). Hence, the relevance of hard cuts is even higher for movie segmentation within the actual movie plot.

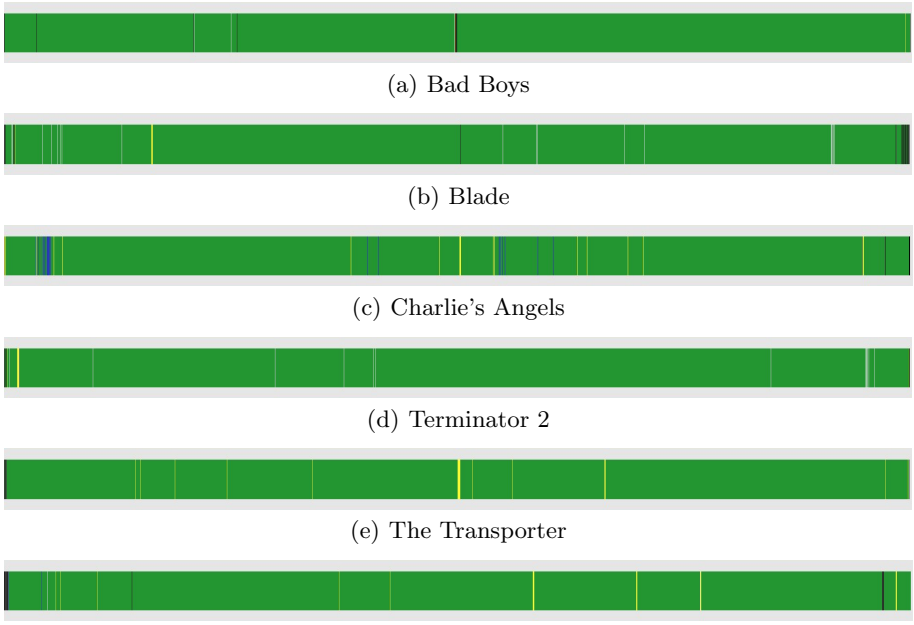
As mentioned above, Vascencelos et al. [46] introduced a framework that computes probabilities according to usual shot length in movie trailers. In fig. 7 shot length histograms for our six movies are shown and actually, there are similarities throughout all distributions, however it must be questioned to what extent this can be generalized and used for automatic shot detection. First, among our six movies there are differences big enough to cause false computations, comparing the distribution curve of *Charlie's Angels* and *The Transporter*, for example. And second, for artistic reasons a movie editor could deviate from this typical shot length histogram, as for example (very extremely) done by Alfred Hitchcock when producing *Rope (1948)*. This movie is made of only nine shots.

3.3 Borderline Cases

During the manual annotation a few problems arose when determining a certain type of shot boundary. In one example a hard cut occurs in the background while the title in the front does not change. In the strict sense, it is not a cut as only part of the frame abruptly changed. The problem stems from the difficulty to define *how big* the abrupt change has to be in order to cause a hard cut. Another

Movie Title	cut	flash	dissolve	fade-in	fade-out	wipe
Bad Boys (1995)	99.49 %	0.12 %	0.04 %	0.04 %	0.31 %	0 %
Blade (1998)	98.25 %	0.70 %	0.19 %	0.16 %	0.70 %	0 %
Charlie's Angels (2000)	97.61 %	0 %	0.98 %	0.16 %	0.16 %	1.09 %
Terminator 2 (1991)	98.96 %	0.48 %	0.30 %	0.04 %	0.19 %	0.04 %
The Transporter (2002)	98.92 %	0 %	0.79 %	0.12 %	0.17 %	0 %
Transporter 2 (2005)	98.46 %	0.08 %	0.67 %	0.32 %	0.40 %	0.08 %
all movies	98.67 %	0.25 %	0.47 %	0.14 %	0.33 %	0.16 %

Table 2. Shot boundary type frequencies in six action movies as a percentage. Cuts are by far the most frequent transition type.



(a) Bad Boys

(b) Blade

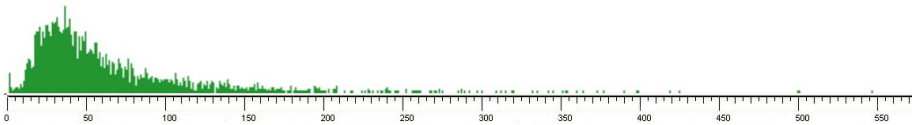
(c) Charlie's Angels

(d) Terminator 2

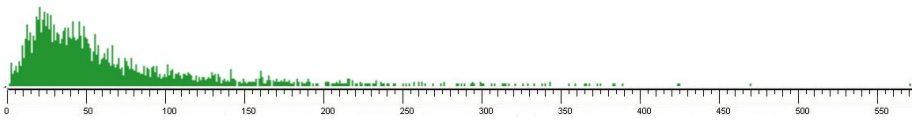
(e) The Transporter

(f) Transporter 2

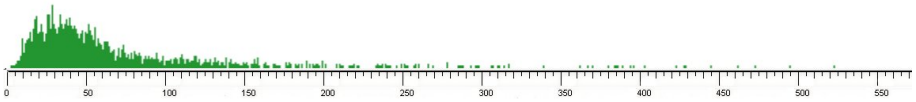
Fig. 6. Every stripe illustrates the location of shot boundaries in the corresponding movie. Each thin vertical line represents a shot boundary in order of appearance (from left to right): green = cut, white = flash, yellow = dissolve, gray = fade-in, black = fade-out, blue = wipe.



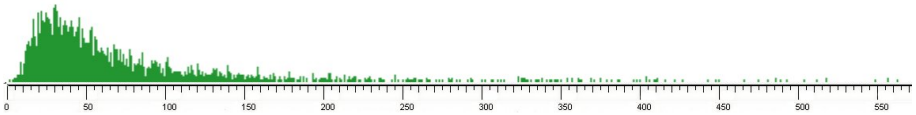
(a) Bad Boys



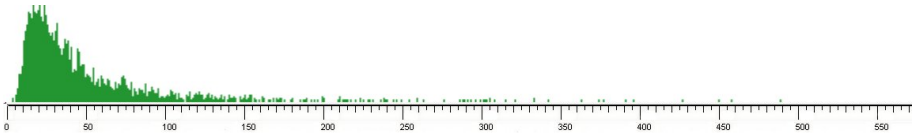
(b) Blade



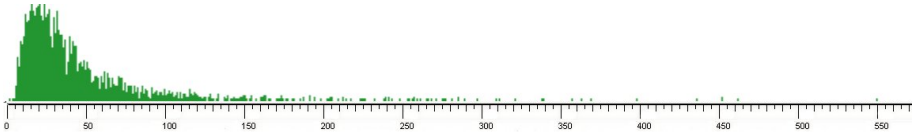
(c) Charlie's Angels



(d) Terminator 2



(e) The Transporter



(f) Transporter 2

Fig. 7. Shot length histograms of six movies. Though general tendencies are recognizable it is questionable to what extent this information can be integrated into a shot detection algorithm.

example is a close-up of a flickering computer display showing changing contents. During the manual annotation it was hard to determine the shot boundaries due to the very fast changes combined with transparency effects. Similar to that, problem arise if shots with high movement are joint by cuts and special effects. And finally, in one case a multi-shot sequence which is made of two single scaled

shots was enriched by wiping a third one into this sequence.
Due to a lack of a strictly formal description of all possible shot boundaries on the one hand, and due to a vast range of possibilities in digital video editing on the other hand, problems like the aforementioned cases will always appear in automatic shot detection. However, some of these borderline cases are not that crucial for the overall accuracy of a detection system, in particular cases like 8 (b) and (c). Hence for this field of research, it might be a good idea to distinguish between more and less relevant shot boundaries, in order to adjust the techniques primarily for the more relevant ones. Thus, we may consider a differentiated relevance for transitions in future manual annotation of video footage. However, within the scope of this work *all* manually annotated transitions were treated equally.

4 Comparison of existing methods

Based on the research review discussed in section 2 and based on the given shot statistics for six Hollywood action movies (section 3), we compared a few techniques in order to determine the one with the highest prospect of accuracy for action movies. Following to the analysis given in the previous section, it is obvious that a good movie shot boundary detection primarily depends on a reliable hard cut detection. Hence, the main focus of our comparison is put on corresponding detection methods.

4.1 Quality Measures

In order to evaluate shot detection methods, the majority of the related work discussed in section 2 makes use of *recall* and *precision* measures defined as following:

$$recall = \frac{correct\ hits}{ground\ truth} \qquad precision = \frac{correct\ hits}{all\ hits} \tag{1}$$

In most cases, shot boundaries are counted as hits, so for example, recall is the ratio of the number of correctly detected boundaries and the number of all true boundaries [4, 16, 55, 32, 39, 12, 22, 54, 31, 47, 41, 3, 28, 15]. This way recall and precision measures are sufficient for cuts but for gradual transitions an additional measure is necessary in order to check the frame range of a detected shot transition. Thus, Ngo et al. [26] classified each video frame either as a transition or non-transition frame and counted transition frames as hits. Truong et al. [44] and Zhai et al. [52] calculated both, *transition-based* and *frame-based* recall and precision values. Černeková et al. [6] computed an *overlap ratio* for gradual transitions.
In addition, recent works compute the so-called $F1$ measure in order to give one comparative figure including recall and precision [33, 47, 3, 10, 28]:

$$F1 = \frac{2 * precision * recall}{(precision + recall)} \tag{2}$$

In the following we will use transition-based *precision*, *recall*, and $F1$ for evaluation of shot detection methods tested with six action movies.

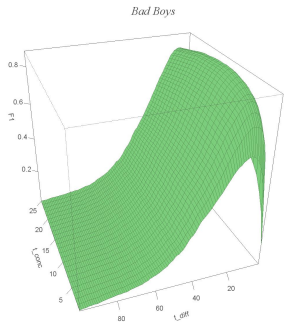
4.2 Performance of Current TZI Cut Boundary Detection

In [23] Miene et al proposed the *Gray Histogram Feature* X^2 *(GHX2)* for hard cut detection which is currently used within the video analysis software at the Center for Computing Technologies (TZI). This technique is based on gray-level histogram bin value differences combined with a percentage threshold of the maximum difference found in the video (t_{diff}). Naturally, this kind of thresholding forces the analysis of the whole video before retrieving any results which is especially disadvantageous for such video data like full-length movies. Another threshold needed for this cut detection is the minimum frame distance between two cuts. If this distance is below a threshold t_{conc} then the cut with the smaller feature difference is disregarded.
In order to determine the best threshold pair for TZI Cut Detection, we performed a batch analysis on six action movies with broad and reasonable ranges for both thresholds. As we know from fig. 7 that a vast number of shots usually lasts for less than 25 frames, we set t_{conc} (denoting the minimum shot length) within the interval of [1..25] frames. The other threshold t_{diff} has been tested with all possible integer values, i. e. [1..100], (denoting the difference threshold as a percentage). In fig. 9 the results for the particular movies are shown. The bottom axes denote the two thresholds while the height axis denotes the $F1$ value achieved by the corresponding threshold combination. Despite of similar characteristics for all six movies, the distribution varies, for example, the results for *Terminator 2* considerably differ from the other five movies. Thus, when selecting a specific threshold combination it turns out that the highest achievable accuracy for this six movies is $F1 = 0.8661$ (recall: 0.8661, precision: 0.8689). In fig. 10 all possible recall/precision pairs are plotted and in table 3 the first 14 threshold combinations are listed providing the best *F1* value.

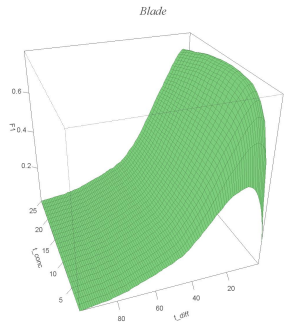
4.3 Other Cut Detection Methods

In section 2 several techniques for shot boundary detection by histogram differences were mentioned. From that, we set up a specific selection of some techniques to be tested with our six action movies. However, within the context of this report we solely focus on this shot boundary type due to the given relevance of detecting cuts. Hence, the majority of the chosen methods is based on histogram differences but we also included a motion-based method and the entropy-based method of Černeková et al. [6].

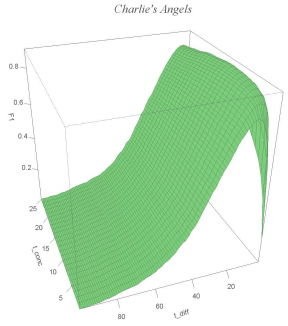
Histogram-based methods For the following formulas, let H_f denote an n-bin histogram of video frame f, so that $H_f(i)$ denotes the corresponding value of ith bin, and $Dif(f, f-1)$ denotes the resulting histogram difference of two subsequent video frames.



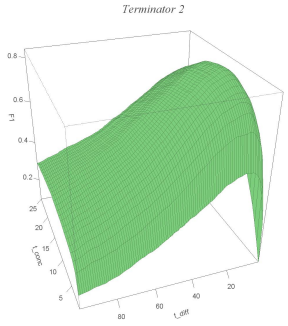
(a) Bad Boys



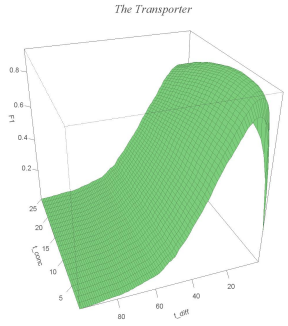
(b) Blade



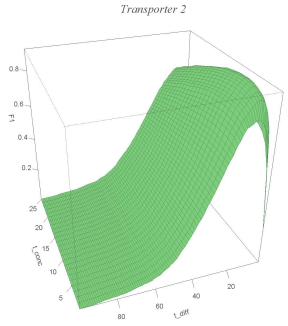
(c) Charlie's Angels



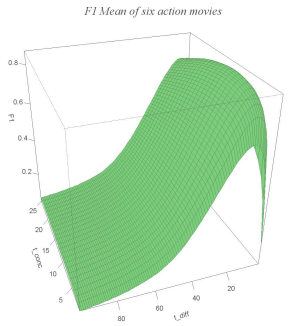
(d) Terminator 2



(e) The Transporter



(f) Transporter 2



(g) Mean of all six movies

Fig. 8. 2475 different threshold combinations of the TZI shot detection and their corresponding performance given by the *F1* measure (0.0: bad, 1.0: perfect).

t_{diff}	t_{conc}	precision	recall	f1
7	9	0.86894074	0.86610393	0.86613869
7	10	0.85858924	0.87634365	0.86597079
7	8	0.87864553	0.85295533	0.86422034
8	10	0.84878550	0.88292150	0.86420885
8	9	0.85881880	0.87186280	0.86407937
7	11	0.84602532	0.88474833	0.86357157
6	10	0.86366970	0.86642853	0.86326432
6	9	0.87444963	0.85547919	0.86313097
8	8	0.86816427	0.86035236	0.86304330
8	11	0.83649211	0.88997457	0.86114927
6	8	0.88425802	0.84223373	0.86096395
6	11	0.85019777	0.87477341	0.86059252
7	7	0.88736983	0.83777933	0.86047855
9	9	0.84526013	0.87818282	0.86035808

Table 3. A list of TZI cut detection threshold combinations providing the best *F1* value for six action movies.

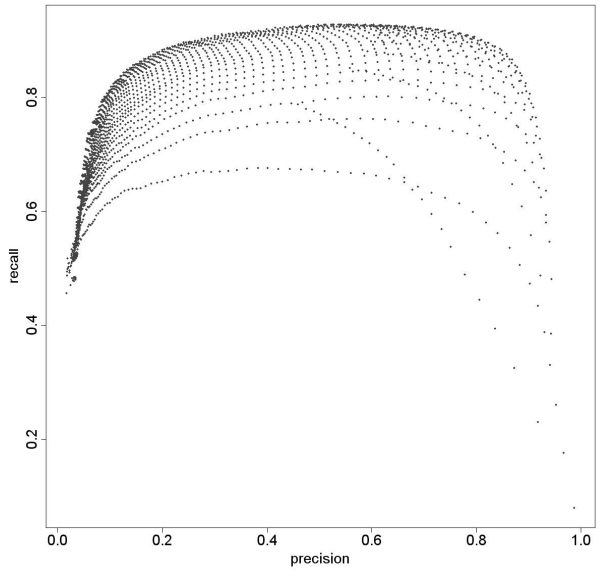


Fig. 9. Each dot stands for the accuracy achieved by TZI cut detection for six action movies. The most top right position represents the best possible result.

– Histogram Intersection (INTERS)

$$Dif_{inters}(f, f-1) = \sum_{i=1}^{n} Max(H_f(i), H_{f-1}(i)) \tag{3}$$

– Histogram Correlation (CORR)

$$Dif_{corr}(f, f-1) = \frac{\sum_{i=1}^{n} H'_f(i) * H'_{f-1}(i)}{\sqrt{\sum_{i=1}^{n} H'_f(i)^2 * \sum_{i=1}^{n} H'_{f-1}(i)^2}} \tag{4}$$

where:

$$H'_f(i) = H_f(i) - \frac{1}{n} * \sum_{i=1}^{n} H_f(i) \tag{5}$$

– χ^2 (Chi-Square) Histogram Distance (CHISQR)

$$Dif_{chisqr}(f, f-1) = \sum_{i=1}^{n} \frac{H_f(i) - H_{f-1}(i)}{H_f(i) + H_{f-1}(i)} \tag{6}$$

– Bhattacharyya Histogram Distance (BHATTA)

$$Dif_{bhatta}(f, f-1) = \sqrt{1 - \sum_{i=1}^{n} \sqrt{H_f(i) * H_{f-1}(i)}} \tag{7}$$

– Gray Histogram X^2 Distance (GHX2)

$$Dif_{ghx2}(f, f-1) = \sum_{i=1}^{n} \frac{(H_f(i) - H_{f-1}(i))^2}{Max(H_f(i), H_{f-1}(i))} \tag{8}$$

Note that the GHX2 is the same histogram distance as used for the current TZI shot detection. In contrast to that, we will use it with a different thresholding method in our experiments.

Feature Movement As part of a TZI video analysis software [2], global motion is estimated based on inter-frame movement of features detected by the Pyramidal Lucas Kanade feature tracker provided by the OpenCV library [1]. Assuming that a hard cut causes chaotic feature movement between two frames, we calculate the average *movement per feature (MPF)* for each frame pair.

$$Dif_{mpf}(f, f-1) = \frac{1}{m} * \sum_{i=1}^{m} Mov_{feat}(i, f, f-1) \tag{9}$$

where $Mov_{feat}(i, f, f-1)$ denotes the movement of ith feature from frame $f-1$ to frame f, and m denotes the number of tracked features. Dif_{mpf} then denotes the average movement per feature and is considered as a measure for inter-frame changes indicating cuts by sudden high peaks.

Information Theory According to [6], Černeková et al's entropy-based shot boundary detection performed very well, especially on movie footage. Hence, we tested their *Mutual Information (MI)* metric which aims at detecting hard cuts. We also included their *Joint Entropy (JE)* metric for cut detection which was originally intended to detect gradual transitions. Refer to the corresponding publication for detailed description of the two entropy-based metrics.

Classification Each differencing technique is tested with gray-level histograms (256 bins) and HLS color histograms (180 bins). For the detection of outliers (indicating possible cuts), we move a one-dimensional, sliding window (size: 7) along all video frames and for each frame at the window center, the local difference mean μ_{Dif} of the surrounding frames is calculated (without considering the center frame). Furthermore, we calculate an estimated standard deviation σ_{Dif} of the difference mean within the sliding window.
In order to calculate a threshold t we then add a certain value to the mean. For our experiments we use two different ways to determine the added value:

$$t_a = a * \mu_{Dif} \tag{10}$$

and

$$t_b = \mu_{Dif} + b * \sigma_{Dif} \tag{11}$$

Note that t_a and t_b can be smaller than μ_{Dif}, which happens if $a < 1$ or $b < 0$. In this case a cut is detected if

$$Dif(f, f-1) < t_{a/b}. \tag{12}$$

For $a > 1$ and $b > 0$ a cut is detected if

$$Dif(f, f-1) > t_{a/b}. \tag{13}$$

Method	a	b
Movement per Feature (Dif_{mpf})	4.0...12.0	2.0...14.0
Mutual Information (Dif_{mi})	0.2...0.8	-14.0...-2.0
Joint Entropy (Dif_{je})	1.0...1.4	2.0...14.0
Histogram Intersection (Dif_{inters})	0.75...0.95	-14.0...-2.0
Histogram Correlation (Dif_{corr})	0.75...0.95	-14.0...-2.0
χ^2 (Chi-Square) Histogram Distance (Dif_{chiqsr})	4.0...12.0	2.0...14.0
Bhattacharyya Histogram Distance (Dif_{bhatta})	2.0...5.0	2.0...14.0
Gray Histogram X^2 Distance (Dif_{ghx2})	4.0...12.0	2.0...14.0

Table 4. a and b ranges with which the corresponding cut detection methods were tested.

Results We tested the methods mentioned above with various ranges of a and b (see table 4) using the thresholding from equations 10 and 11. In fig. 11 the cut detection performance is shown for all methods applied to the six action movies with the thresholding from equation 10 (t_a). The methods GHX2, CHISQR and MI prove to perform best. Among these, the histogram-based detection methods achieve higher precision values, while the entropy-based MI method achieves higher recall values. With the exception of BHATTA all other methods achieve unsatisfactory results with the thresholding type t_a. In fig. 12 the same methods were tested with the same movie footage but with thresholding t_b from equation 11. In this case all histogram-based methods perform quite well, while the other methods do not achieve any satisfactory results. Among the histogram-based methods, GHX2 and CHISQR again prove to perform best.
Interestingly, histogram-based methods generally perform better with HLS images (lines with circles) than with gray-level images (lines with squares). Apparently, more detailed image information leads to slightly better results. As expected, the JE method does not apply to cut detection. The motion-based cut

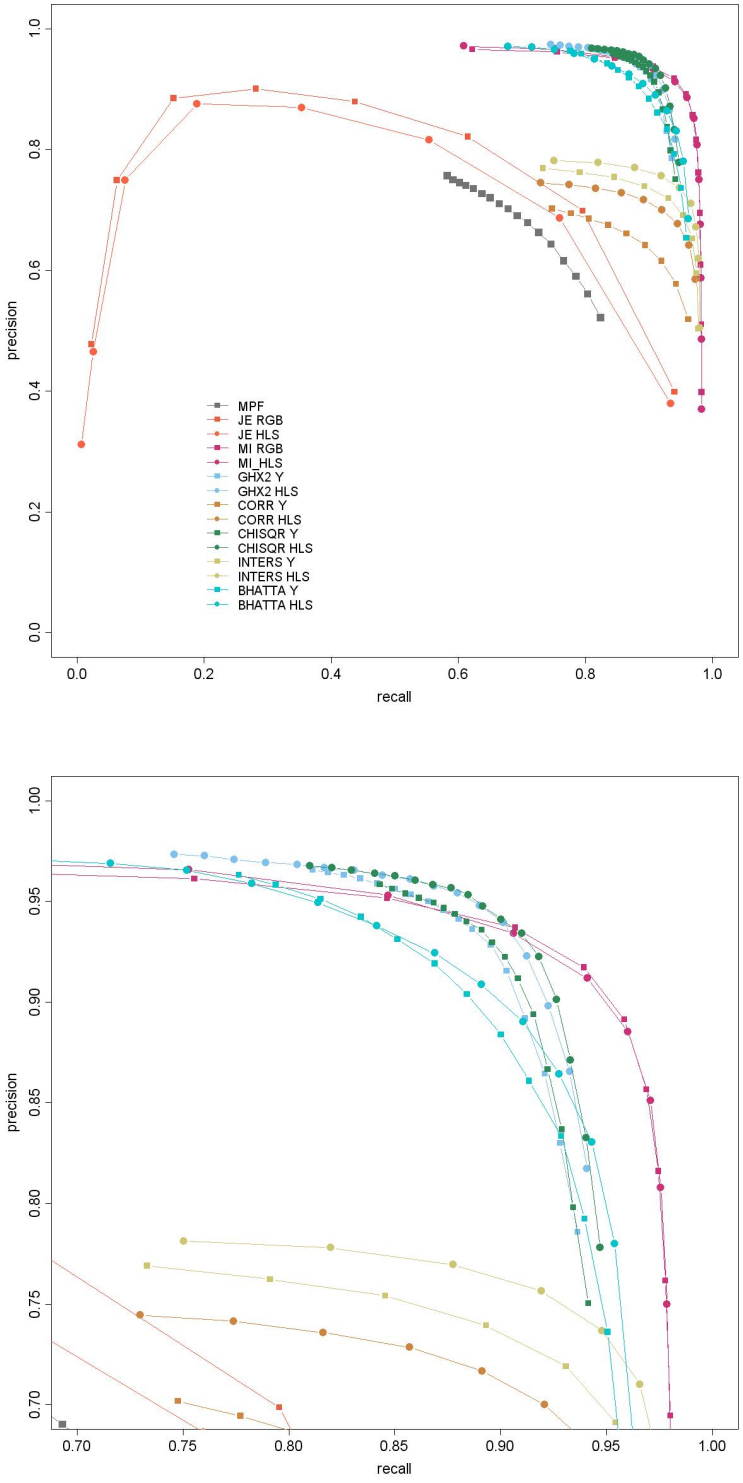


Fig. 10. Cut detection performance for the six action movies with thresholding using equation 10 (t_a). In the magnified extract below, the methods GHX2, CHISQR and MI prove to perform best, the first two with better precision values and the latter one with better recall values.

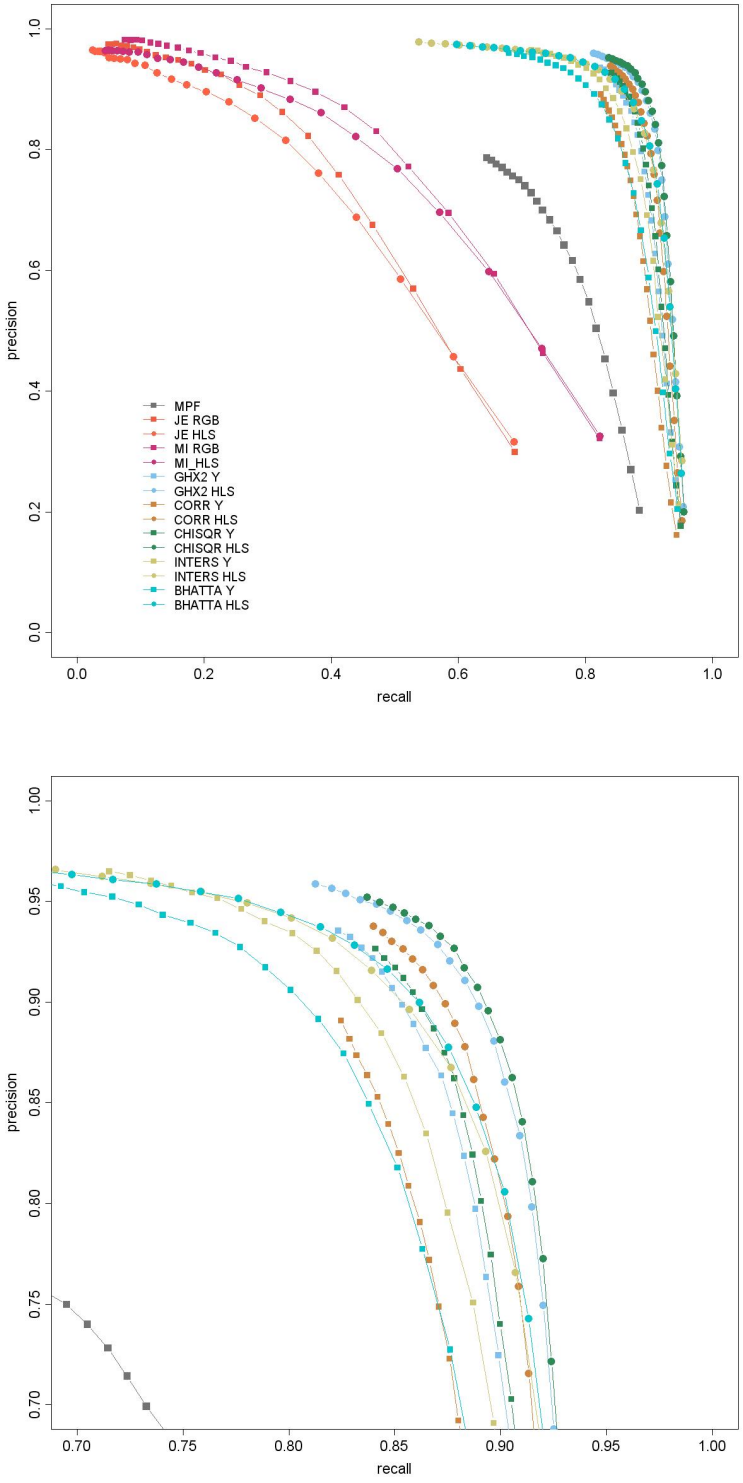


Fig. 11. Cut detection performance for the six movies with thresholding using equation 11 (t_b). All histogram-based detection methods perform well and better than the other ones. In the magnified extract below, the methods GHX2 and CHISQR prove to perform best, again.

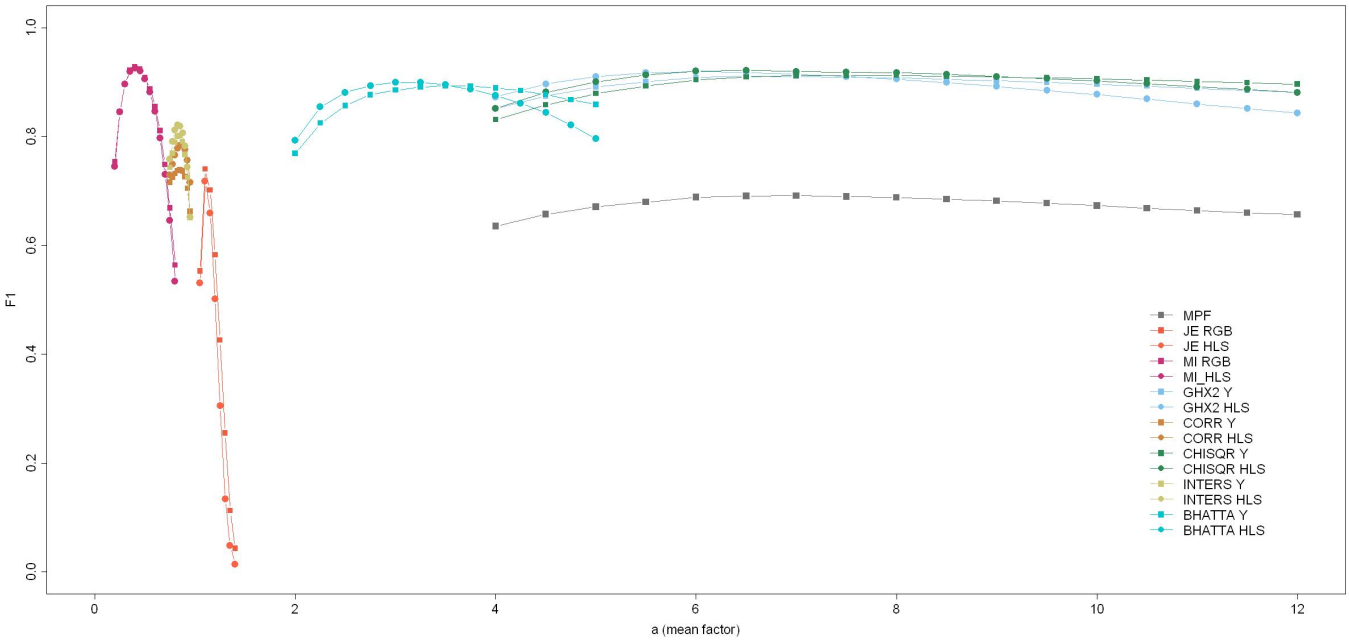


Fig. 12. Threshold a plotted against $F1$ of the corresponding cut detection method.

detection also fails to provide satisfactory results. Comparing the two thresholding methods t_a and t_b, it can be seen that the best results in fig. 11 are higher than the ones in fig. 12, but we will discuss this more detailed based on further analysis later. An important aspect about any detection method is the selection of an appropriate threshold. In fig. 13 and fig. 14 the selected parameters a and b, respectively, are plotted against the $F1$ value, indicating the performance of the corresponding cut detection method for a specific threshold selection. In table 5 the best method/threshold combinations sorted by $F1$ are listed. As already visible from the plots, the best three methods are MI, GHX2, and CHISQR, which at most achieve $F1$ values above 0.91 for the six action movies. The following methods are BHATTA, CORR and INTERS with maximum $F1$ values between 0.85 and 0.90. JE and MPF do not achieve any $F1$ values above 0.75.
Generally, all histogram-based methods perform slightly better with HLS images than with gray-level (Y) images. For the two entropy-based methods (JE, MI) there were only little differences between RGB-based and HLS-based analysis. MPF was tested with gray-level images only.
Among the two thresholding methods we used for our experiments, the σ-based thresholding (t_b) is more suitable for the methods CORR, INTERS, and MPF, while the t_a-thresholding leads to better results with the methods MI, GHX2, CHISQR, BHATTA, and JE.

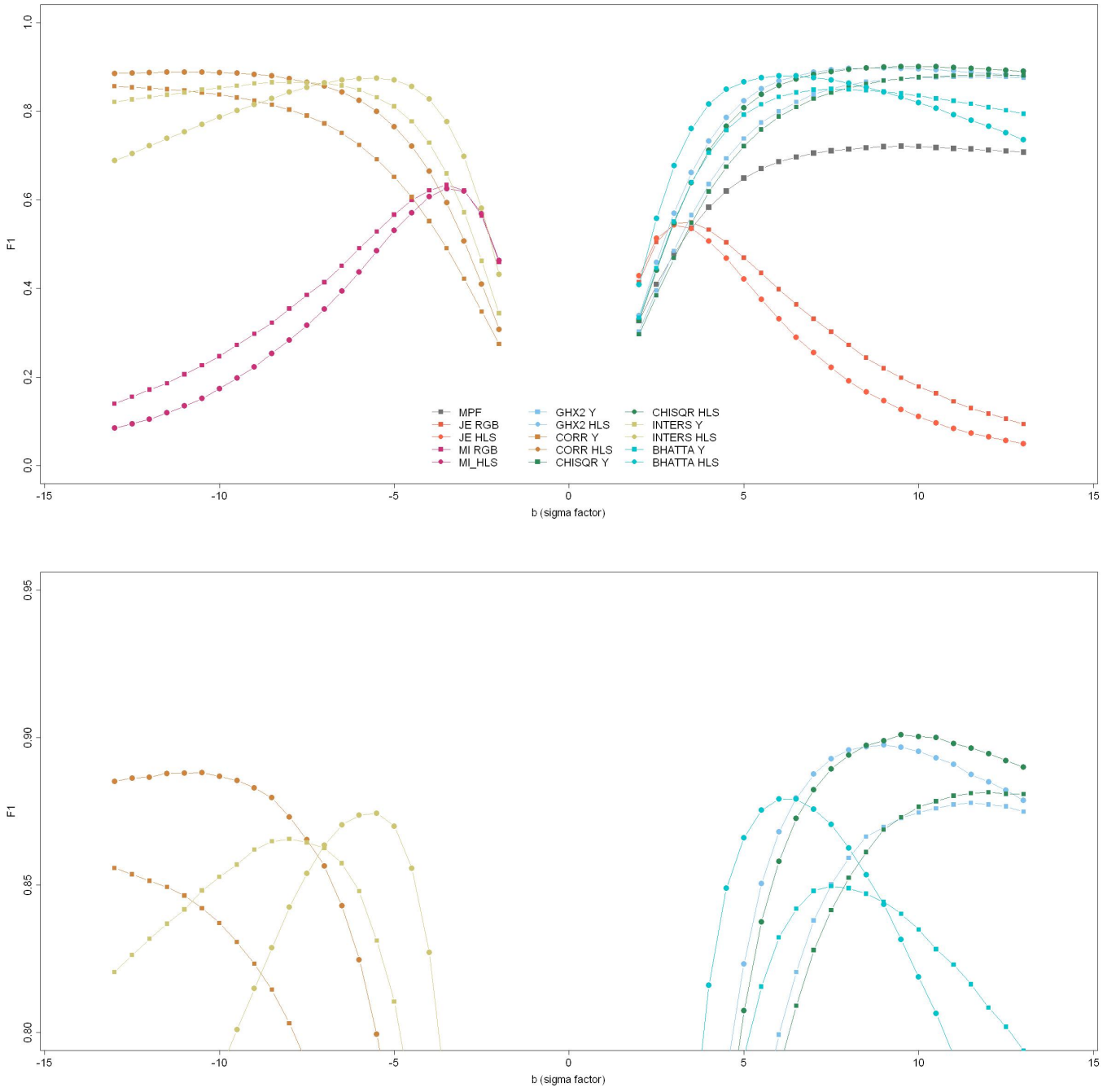


Fig. 13. Threshold b plotted against $F1$ of the corresponding cut detection method. In the image below the upper plot region is magnified.

The best TZI cut detection result from table 3 is ranked among the CORR and INTERS methods although it is based on the same histogram distance calculation as GHX2. However, in the TZI cut detection a different thresholding is used. Thus, by changing the TZI thresholding method into t_a thresholding the TZI software might improve its $F1$ performance by around 0.05 – at least for movie footage.

5 Conclusion & Future Work

This work aimed at investigation of a reliable movie shot boundary detection for movie footage. Starting from the current TZI shot boundary detection, we manually annotated six contemporary Hollywood action movies and thoroughly tested the TZI method. After that, we analyzed the characteristics of movie shot boundaries through which we realized that a reliable movie shot boundary detection needs to be a reliable cut detection, in the first place[2]. Thus, we then selected a bunch of current cut detection methods based on our research of related work and tested them with the six action movies, too. The results show that a thorough selection of a detection method can lead to quite good results however there are still possibilities for improvement.
First, through our research on cut detection we came across other cut detection methods that could be examined with our movie footage. Second, different combinations of detection methods should also be considered. And of course, more video footage – especially movies – should be tested in order to substantiate the investigation results. Furthermore, other transition types than the hard cut should be included. Their impact on precision and recall values hardly matters, however, their semantic importance for movie analysis surely is relevant.
Thus, starting from this work we will continue on further investigation of shot boundary detection methods that perform best with feature movies.

References

1. *Open Source Computer Vision Library.* http://www.intel.com/technology/ computing/opencv/, March 2006.
2. *Semantic Video Patterns.* http://www.tzi.de/svp, April 2006.
3. G. Boccignone, A. Chianese, V. Moscato, and A. Picariello. Foveated shot detection for video segmentation. *Circuits and Systems for Video Technology, IEEE Transactions on*, 15(3):365–377, 2005.
4. J. S. Boreczky and L. A. Rowe. Comparison of video shot boundary detection techniques. In I. K. Sethi and R. C. Jain, editors, *Proc. SPIE Vol. 2670, p. 170-179, Storage and Retrieval for Still Image and Video Databases IV, Ishwar K. Sethi; Ramesh C. Jain; Eds.*, pages 170–179, Mar. 1996.

[2] This is why this work has been named *Blink*, as the effect of cuts is supposed to correspond to the blink of a human eye.

Rank	Method	Color	t_a	t_b	precision	recall	f1
1.	MI	RGB	0.4	-	0.91715450	0.93936267	0.92790200
2.	MI	HLS	0.4	-	0.91176983	0.94084200	0.92566250
3.	MI	RGB	0.45	-	0.89130400	0.95828900	0.92320150
4.	MI	RGB	0.35	-	0.93686817	0.90681733	0.92140850
5.	CHISQR	HLS	6.5	-	0.93396967	0.90994467	0.92140500
6.	MI	HLS	0.45	-	0.88527700	0.96011400	0.92039767
7.	CHISQR	HLS	6.0	-	0.92249200	0.91802283	0.91987233
8.	CHISQR	HLS	7.0	-	0.94108367	0.90013583	0.91971817
9.	MI	HLS	0.35	-	0.93409867	0.90606150	0.91956100
10.	GHX2	HLS	6.0	-	0.93929150	0.90127267	0.91947667
11.	CHISQR	HLS	7.5	-	0.94753617	0.89151333	0.91818583
12.	GHX2	HLS	6.5	-	0.94795133	0.89016000	0.91771933
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
19.	CHISQR	Y	7.5	-	0.92963600	0.89600150	0.91195483
20.	CHISQR	Y	7.0	-	0.92243500	0.90217550	0.91163867
21.	GHX2	Y	6.5	-	0.92830883	0.89538900	0.91100567
44.	CHISQR	HLS	-	9.5	0.92643400	0.87813700	0.90090250
49.	BHATTA	HLS	3.0	-	0.89017583	0.91048050	0.89945983
73.	BHATTA	Y	3.5	-	0.90381617	0.88408300	0.89318833
89.	CORR	HLS	-	-10.5	0.91607683	0.86316850	0.88802833
142.	INTERS	HLS	-	-5.5	0.89627617	0.85712467	0.87430500
	TZI	*Y*	-	-	*0.86894074*	*0.86610393*	*0.86613869*
159.	INTERS	Y	-	-8.0	0.91527300	0.82260817	0.86556467
178.	CORR	Y	-	-13.0	0.89078233	0.82468950	0.85577883
308.	JE	RGB	1.1	-	0.69864333	0.79551717	0.74024466
327.	MPF	Y	-	9.5	0.73984917	0.70495600	0.72058983
331.	JE	HLS	1.1	-	0.68636533	0.75953733	0.71748367

Table 5. An overall view of the best method/threshold combinations within the scope of this work. From the top, the first 12 best combinations are shown, following by the best result achieved by each method with color or gray-level images. Furthermore, the best result of the TZI cut detection is included.

5. U. Cabedo and S. Bhattacharjee. Shot detection tools in digital video. In *Noblesse Workshop on Non-Linear Model Based Image Analysis (NMBIA'98)*, Proc. of the SPIE Conf. on Visual Communications and Image Processing, pages 231–236, Ecublens, 1998. IEEE.
6. Z. Černeková, I. Pitas, and C. Nikou. Information theory-based shot cut/fade detection and video summarization. *IEEE Trans. Circuits Syst. Video Techn.*, 16(1):82–91, 2006.
7. H.-W. Chen, J.-H. Kuo, W.-T. Chu, and J.-L. Wu. Action movies segmentation and summarization based on tempo analysis. In *MIR '04: Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, pages 251–258, New York, NY, USA, 2004. ACM Press.
8. G. Ciocca and R. Schettini. Dynamic storyboards for video content summarization. In *MIR '06: Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 259–268, New York, NY, USA, 2006. ACM Press.
9. R. Dugad, K. Ratakonda, and N. Ahuja. Robust video shot change detection. In *IEEE Second Workshop on Multimedia Signal Processing*, pages 376–381, Redondo Beach, CA, USA, 1998. ACM Press.
10. H. Feng, W. Fang, S. Liu, and Y. Fang. A new general framework for shot boundary detection and key-frame extraction. In *MIR '05: Proceedings of the 7th ACM SIGMM international workshop on Multimedia information retrieval*, pages 121–126, New York, NY, USA, 2005. ACM Press.
11. S. J. F. Guimarães, M. Couprie, A. de Albuquerque Araújo, and N. J. Leite. Video segmentation based on 2d image analysis. *Pattern Recogn. Lett.*, 24(7):947–957, 2003.
12. S.-H. Han and I.-S. Kweon. Detecting cuts and dissolves through linear regression analysis. *Electronics Letters*, 39(22):1579–1581, 2003.
13. A. Hanjalic. Shot-boundary detection: unraveled and resolved? *IEEE Trans. Circuits Syst. Video Techn.*, 12(2):90–105, 2002.
14. A. Hanjalic, M. Ceccarelli, R. L. Lagendijk, and J. Biemond. Automation of systems enabling search on stored video data. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 427–438, 1997.
15. R. A. Joyce and B. Liu. Temporal segmentation of video using frame and histogram space. *IEEE Transactions on Multimedia*, 8(1):130–140, 2006.
16. S.-B. Jun, K. Yoon, and H.-Y. Lee. Dissolve transition detection algorithm using spatio-temporal distribution of mpeg macro-block types (poster session). In *MULTIMEDIA '00: Proceedings of the eighth ACM international conference on Multimedia*, pages 391–394, New York, NY, USA, 2000. ACM Press.
17. H. Kim, J. Lee, J.-H. Yang, S. Sull, W. M. Kim, and S. M.-H. Song. Visual rhythm and shot verification. *Multimedia Tools Appl.*, 15(3):227–245, 2001.
18. Y. Li and C.-C. J. Kuo. *Video Content Analysis using Multimodal Information.* Kluwer Academic Publishers, 2003.
19. R. Lienhart, S. Pfeiffer, and W. Effelsberg. Video abstracting. *Commun. ACM*, 40(12):54–62, 1997.
20. R. W. Lienhart. Comparison of automatic shot boundary detection algorithms. In M. M. Yeung, B.-L. Yeo, and C. A. Bouman, editors, *Proc. SPIE Vol. 3656, p. 290-301, Storage and Retrieval for Image and Video Databases VII, Minerva M. Yeung; Boon-Lock Yeo; Charles A. Bouman; Eds.*, pages 290–301, Dec. 1998.
21. B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI81*, pages 674–679, 1981.
22. J. Mas and G. Fernandez. Video shot boundary detection based on color histogram. In *TREC Video Retrieval Evaluation Online Proceedings*, 2003.

23. A. Miene, A. Dammeyer, T. Hermes, and O. Herzog. Advanced and adapted shot boundary detection. In D. W. Fellner, N. Fuhr, and I. Witten, editors, *Proc. of ECDL WS Generalized Documents*, pages 39–43, 2001.
24. A. Nagasaka and Y. Tanaka. Automatic video indexing and full-video search for object appearances. In *Proceedings of the IFIP TC2/WG 2.6 Second Working Conference on Visual Database Systems II*, pages 113–127. North-Holland, 1992.
25. C.-W. Ngo. A robust dissolve detector by support vector machine. In *MULTIMEDIA '03: Proceedings of the eleventh ACM international conference on Multimedia*, pages 283–286, New York, NY, USA, 2003. ACM Press.
26. C.-W. Ngo, T.-C. Pong, and R. T. Chin. Video partitioning by temporal slice coherency. *IEEE Trans. Circuits Syst. Video Techn.*, 11(8):941–953, 2001.
27. W. Niblack, R. Barber, W. Equitz, M. D. Flickner, E. H. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin. QBIC project: querying images by content, using color, texture, and shape. In W. Niblack, editor, *Proc. SPIE Vol. 1908, p. 173-187, Storage and Retrieval for Image and Video Databases, Wayne Niblack; Ed.*, pages 173–187, Apr. 1993.
28. A. Pardo. Simple and robust hard cut detection using interframe differences. In *CIARP*, pages 409–419, 2005.
29. N. V. Patel and I. K. Sethi. Video shot detection and characterization for video databases. *Pattern Recognition*, 30(4):583–592, 1997.
30. C. Petersohn. Dissolve shot boundary determination. In *EWIMT*, 2004.
31. C. Petersohn. Shot boundary detection system. In *TREC Video Retrieval Evaluation Online Proceedings*, 2004.
32. S. Porter, M. Mirmehdi, and B. Thomas. Detection and classification of shot transitions. In T. Cootes and C. Taylor, editors, *Proceedings of the 12th British Machine Vision Conference*, pages 73–82. BMVA Press, 2001.
33. Y. Qi, A. Hauptmann, and T. Liu. Supervised classification for video shot segmentation. *ICME*, 1:689–692, 2003.
34. L. Qing, W. Wang, and W. Gao. Illumination invariant shot boundary detection. In *IDEAL*, pages 1097–1101, 2003.
35. Z. Rasheed, Y. Sheikh, and M. Shah. On the use of computable features for film classification. *IEEE Trans. Circuits Syst. Video Techn.*, 15(1):52–64, 2005.
36. W. Ren and S. Singh. Automatic video shot boundary detection using machine learning.. In *IDEAL*, pages 285–292, 2004.
37. K.-D. Seo, S. J. Park, J.-S. Kim, and S. M.-H. Song. Automatic shot-change detection algorithm based on visual rhythm extraction. In *ICIAR (1)*, pages 709–720, 2006.
38. A. F. Smeaton, B. Lehane, N. E. O'Connor, C. Brady, and G. Craig. Automatically selecting shots for action movie trailers. In *MIR '06: Proceedings of the 8th ACM international workshop on Multimedia information retrieval*, pages 231–238, New York, NY, USA, 2006. ACM Press.
39. B. C. Song and J. B. Ra. Automatic shot change detection algorithm using multistage clustering for mpeg-compressed videos. *Journal of Visual Communication and Image Representation*, 12:364–385(22), September 2001.
40. M. A. Stricker and M. Orengo. Similarity of color images. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 381–392, 1995.
41. C.-W. Su, H.-Y. M. Liao, H.-R. Tyan, K.-C. Fan, and L.-H. Chen. A motion-tolerant dissolve detection algorithm. *IEEE Transactions on Multimedia*, 7(6):1106–1113, 2005.
42. M. J. Swain and D. H. Ballard. Color indexing. *Int. J. Comput. Vision*, 7(1):11–32, 1991.

43. S. M. M. Tahaghoghi, H. E. Williams, J. A. Thom, and T. Volkmer. Video cut detection using frame windows. In *ACSC '05: Proceedings of the Twenty-eighth Australasian conference on Computer Science*, pages 193–199, Darlinghurst, Australia, Australia, 2005. Australian Computer Society, Inc.
44. B. T. Truong, C. Dorai, and S. Venkatesh. New enhancements to cut, fade, and dissolve detection processes in video segmentation. In *MULTIMEDIA '00: Proceedings of the eighth ACM international conference on Multimedia*, pages 219–227, New York, NY, USA, 2000. ACM Press.
45. B. T. Truong, S. Venkatesh, and C. Dorai. Extraction of film takes for cinematic analysis. *Multimedia Tools Appl.*, 26(3):277–298, 2005.
46. N. Vasconcelos and A. Lippman. Statistical models of video structure for content analysis and characterization. *IEEE Transactions on Image Processing*, 9(1):3–19, 2000.
47. A. Whitehead, P. Bose, and R. Laganière. Feature based cut detection with automatic threshold selection. In *CIVR*, pages 410–418, 2004.
48. B. Yeo and B. Liu. Rapid scene analysis on compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(6):533–544, December 1995.
49. J. Yuan, J. Li, F. Lin, and B. Zhang. A unified shot boundary detection framework based on graph partition model. In *MULTIMEDIA '05: Proceedings of the 13th annual ACM international conference on Multimedia*, pages 539–542, New York, NY, USA, 2005. ACM Press.
50. Y. Yusoff, W. J. Christmas, and J. Kittler. Video shot cut detection using adaptive thresholding. In *BMVC*, 2000.
51. R. Zabih, J. Miller, and K. Mai. A feature-based algorithm for detecting and classifying production effects. *Multimedia Syst.*, 7(2):119–128, 1999.
52. Y. Zhai and M. Shah. A multi-level framework for video shot structuring. In *ICIAR*, pages 167–173, 2005.
53. D. Zhang, W. Qi, and H.-J. Zhang. A new shot boundary detection algorithm. In *PCM '01: Proceedings of the Second IEEE Pacific Rim Conference on Multimedia*, pages 63–70, London, UK, 2001. Springer-Verlag.
54. R. Zhao and W. I. Grosky. A novel video shot detection technique using color anglogram and latent semantic indexing. In *ICDCSW '03: Proceedings of the 23rd International Conference on Distributed Computing Systems*, page 550, Washington, DC, USA, 2003. IEEE Computer Society.
55. D. Zhong and S.-F. Chang. Video shot detection combining multiple visual features. Technical report, Columbia University, December 2000.
56. J. Zhou and X.-P. Zhang. A web-enabled video indexing system. In *MIR '04: Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval*, pages 307–314, New York, NY, USA, 2004. ACM Press.