Comparison of Automatic Shot Boundary Detection Algorithms

Rainer Lienhart[1]
Microcomputer Research Labs, Intel Corporation, Santa Clara, CA 95052-8819
Rainer.Lienhart@intel.com

ABSTRACT

Various methods of automatic shot boundary detection have been proposed and claimed to perform reliably. Although the detection of edits is fundamental to any kind of video analysis since it segments a video into its basic components, the shots, only few comparative investigations on early shot boundary detection algorithms have been published. These investigations mainly concentrate on measuring the edit detection performance, however, do not consider the algorithms' ability to classify the types and to locate the boundaries of the edits correctly. This paper extends these comparative investigations. More recent algorithms designed explicitly to detect specific complex editing operations such as fades and dissolves are taken into account, and their ability to classify the types and locate the boundaries of such edits are examined. The algorithms' performance is measured in terms of hit rate, number of false hits, and miss rate for hard cuts, fades, and dissolves over a large and diverse set of video sequences. The experiments show that while hard cuts and fades can be detected reliably, dissolves are still an open research issue. The false hit rate for dissolves is usually unacceptably high, ranging from 50% up to over 400%. Moreover, all algorithms seem to fail under roughly the same conditions.

Keywords: video content analysis, shot boundary detection, hard cut detection, fade detection, dissolve detection

1 Introduction

The detection of edits is fundamental to any kind of video analysis and video application since it enables segmentation of a video into its basic components: the shots. Various automatic shot boundary detection algorithms have been proposed (see [2,7,11,12,13,14,15,16] and the references therein). Usually, their performance was measured only on a (very) small and limited set of test videos which commonly suggested that the proposed algorithms perform reliably. Despite the importance of reliable shot boundary detection few comparative investigations have been published [3,5]. They assess the performance of early shot boundary detection algorithms with respect to edit detection in general, but not with respect to their ability to classify correctly the type of edit and its temporal extent.

OUR CONTRIBUTION. This paper extends these comparative investigations in two different respects: On the one hand, newer algorithms designed explicitly to detect more complex editing operations such as fades and dissolves are taken into account; on the other hand, besides the algorithms' ability to detect edits as such, also their ability to classify the types of edits and locate their boundaries are examined. Both aspects distinguish this research from existing publications [3,5,6].

2 Segmentation Methods

The number of possible edits is quite large. Well-known video editing programs such as Adobe Premiere or Ulead MediaStudio provide more than 100 different and parameterized types of edits. In practice, however, 99% of all edits fall into one of the following three categories:

- hard cuts,
- fades, or
- dissolves.

Therefore, in the following, we concentrate on these three types of edits. They capture more than 99.9% of all edits in our video test set.

Four shot boundary detection algorithms will be investigated: the best and most balanced "older" algorithm based on color histogram differences [3], the recently proposed algorithm based on the edge change ratio [15], and two algorithms specialized on fades [9] and dissolves [8] exclusively. The matrix in Table 1 summarizes which type of edit is detected by what algorithm.

1. This research was mainly performed while the author was at University of Mannheim, Praktische Informatik IV, 68131 Mannheim, Germany

Feature \ Type of Edit	Hard Cuts	Fades	Dissolve
Color Histogram Differences	x		
Edge Change Ratio	x	x	x
Standard Deviation of Pixel Intensities		x	
Contrast			x

Table 1: Matrix showing which type of edit is detected by what algorithm

2.1 Color Histogram Differences

The color histogram-based shot boundary detection algorithm is one of the most reliable variants of histogram-based detection algorithms. Its basic idea is that the color content does not change rapidly within but across shots. Thus, hard cuts and other short-lasting transitions can be detected as single peaks in the time series of the differences between color histograms of contiguous frames or of frames a certain distance k apart.

Let $p_i(r, g, b)$ be the number of pixels of color (r,g,b) in frame I_i of N pixels. Each color component is discretized to 2^B different values, resulting in $r, g, b \in [0, 2^B - 1]$. Usually B is set to 2 or 3 in order to reduce sensitivity to noise and slight light, object as well as view changes. Then, the color histogram difference CHD_i between two color frame I_{i-1} and I_i is given by

$$CHD_i = \frac{1}{N} \cdot \sum_{r=0}^{2^B-1} \sum_{g=0}^{2^B-1} \sum_{b=0}^{2^B-1} |p_i(r, g, b) - p_{i-1}(r, g, b)| \qquad (1.1)$$

A hard cut is detected if within a local environment of radius l_c of frame I_i only CHD_i exceeds a certain threshold, henceforth called θ_c. Note that instead of using a global threshold, one may also use a local threshold as presented in [13]. This option, however, was not considered our work. In order to cope with a very particular type of hard cut which consists of one transitional frame, in a pre-processing stage double peaks (i.e. groups of $s_c = 2$ contiguous CHD_i exceeding θ_c) were modified into single peaks at the higher CHD_i.

Table 2 summarizes the parameters of the hard cut detection algorithm based on color histogram differences.

Parameter	Description
θ_c	Threshold for cut detection
s_c	Maximal sequence length of contiguous CHD values exceeding θ_c which are transformed into a single peak to enable hard cut detection

Table 2: Parameters of the hard cut detection algorithm based on color histogram differences

2.2 Edge Change Ratio

The edge change ratio (*ECR*) is defined as follows. Let σ_n be the number of edge pixels in frame n, X_n^{in} and X_{n-1}^{out} the number of entering and exiting edge pixels in frames n and n-1, respectively. Then

$$ECR_n = max(X_n^{in}/\sigma_n, X_{n-1}^{out}/\sigma_{n-1}), \qquad (1.2)$$

gives the edge change ratio ECR_n between frames n-1 and n. It ranges from 0 to 1. The edges are calculated by the Canny edge detector [4]. In order to make the measure robust against small object motions, edge pixels in one image which have edge pixels nearby in the other image (e.g. within 6 pixels' distance) are not regarded as entering or exiting edge pixels. Moreover, before calculation of the *ECR* a global motion compensation based on the Hausdorff distance is performed [15].

According to Zabih et. al. hard cuts, fades, dissolves and wipes exhibit a characteristic pattern in the *ECR* time series. Hard cuts are recognized as isolated peaks; during fade-ins/fade-outs the number of incoming/outgoing edges predominates [15]; and during a dissolve, initially the outgoing edges of the first shot protrude before the incoming edges of the second shot start to dominate the second half of a dissolve (see Figure 1).

In the following, many details of the detection algorithm are mentioned which were omitted in the original work [15]. They were extracted by a thorough analysis of the freely available implementation.

In a pre-processing step the *ECR* time series was smoothed by means of a gliding mean value of radius *r*, which, however, was computed only for those points in the *ECR* time series which exceeded the threshold $sumreg_{min}$. All other points were set to zero. Moreover, an $ECR > 1$ was assigned to the first and last frames in a monochrome frame sequence. This new *ECR* time series is called ECR_{sumreg}. Next, the local maxima were determined for this new time series. They were defined as the largest value within a radius s. These local maxima were taken as the centers of edits and expanded in each direction until the *ECR* dropped below threshold θ_n or the maximal duration $transrad_{max}$ of edits was reached. Each edit was classified by the following rules:

- Isolated maxima are classified as hard cuts. Isolation is tested by means of the quotient ECR/ECR_{sumreg} for which it is required to exceed threshold θ_c.
- Start and Stop points of fades are identifies by local maxima of $ECR > 1$. By evaluation of ECR^{in} and ECR^{out} a fade-in is distinguished from a fade-out.
- All other maxima are automatically recognized as dissolves or wipes. Both edits are distinguished from each other by looking at the spatial distribution of the *ECR*. If the change of edges is initially concentrated in one frame half and moves than on to the other frame half, a wipe is detected.

The parameters of this edit detection algorithm are summarized in Table 3.

Parameter	Description
r	Radius of the gliding mean value for smoothing
$sumreg_{min}$	Threshold for *ECR* values, in order to be smoothed and not set to 0
s	Radius for determination of local maximum
θ_n	Threshold for *ECR* values, which are considered to be part of an edit
$transrad_{max}$	Half of the maximal allowed duration of edits
θ_c	Threshold for hard cut detection

Table 3: Parameters of the edit detection algorithm based on the edge change ratio

We have since recognized some drawbacks to the original implementation which are listed in the following. We also present how they can be overcome:

- Abruptly entering and exiting lines of text are clearly visible within the *ECR* time series. Though these peaks are not as high as those of hard cuts, they may result in false hits. These false hits can be eliminated by looking at both ECR^{in} and ECR^{out} instead of only at *ECR*. For hard cuts both values should exhibit the peak, while for entering and existing lines of text this should only be the case for either ECR^{in} or ECR^{out}. Exceptions to this are hard cuts from and to monochrome frames.
  Therefore, the classification of hard cuts must be extended by the following rule: Either ECR^{in} and ECR^{out} exhibit an isolated maximum or if only one of them does then ECR^{in}/ECR^{out} should be 0 and the subsequent/preceding frame monochrome.
- In principle all fades are recognized. Unfortunately, all hard cuts from monochrome frames are also classified as such. This misclassification is caused by marking the border frames of monochrome frame sequences as $ECR > 1$. The contiguous *ECR* value is always 1, independently of whether a fade or a hard cut follows. From an edge-free frame to a frame with edges the ratio of in entering edges is always 1. Thus, the quotient ECR/ECR_{sumreg} of the local maximum is about 0.5, for both hard cuts and fades. Depending on θ_c, all these cases are consequently classified as hard cuts or fades.
  This misclassification can be resolved by not marking the borders to/from monochrome frame sequences as $ECR > 1$ but by applying a special processing to them: In the direction of the potential fade, it is checked whether several frames have an ECR above θ_c. If this is the case, a fade is detected, otherwise, a hard cut.

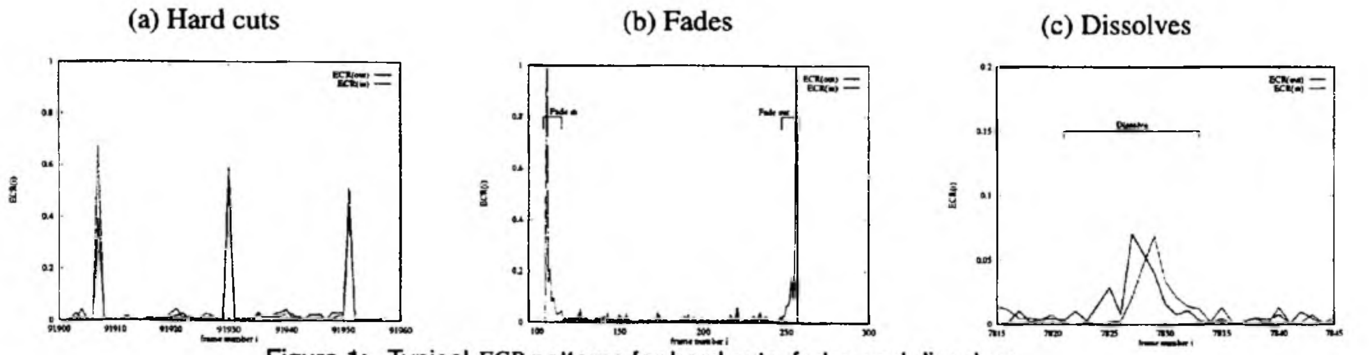


Figure 1: Typical *ECR* patterns for hard cuts, fades and dissolves

- If strong motion immediately before or after a hard cut cannot be compensated by the global motion compensation, the strength of the local maximum is usually not sufficient to be judged as a hard cut. Instead, the edit is classified as a dissolve. We have no solution to that problem.

2.3 Standard Deviation of Pixel Intensities

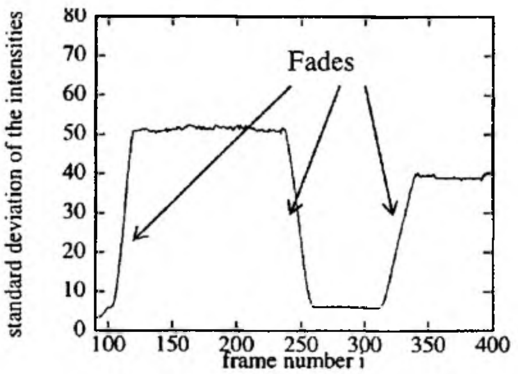


Figure 2: Characteristic pattern of the standard deviation of pixel intensities during a fade-in and fade-out.

During video production fades are produced by a monotone and usually linear scaling of the pixel intensities over time. This intensity scaling is clearly visible in the time series of the standard deviation of pixel intensities as depicted in Figure 2. Its precise pattern can be theoretically derived as follows:

Recall that a fade out $E(x, y, t)$ of length d^i from shot S_{i-1} starting at time t_e^{i-1} can be modeled by [7]

$$E(x, y, t) = S_{i-1}(x, y, t + t_e^{i-1}) \cdot (1 - t/d^i),\ t \in [0, d^i] \qquad (1.3)$$

Substituting the right side of equation (1.3) by X and denoting the expectation value operator of the pixel intensities of a frame by μ, the following conversions can be performed:

$$\sigma(E(x, y, t)) = \sigma(X) = \sqrt{\mu(X^2) - \mu^2(X)} \qquad (1.4)$$

After back-substituting X by the right side of (1.3), it follows

$$= \sqrt{\mu(S_{i-1}^2(x, y, t + t_e^{i-1}) \cdot (1 - t/d^i)^2) - \mu^2(S_{i-1}(x, y, t + t_e^{i-1}) \cdot (1 - t/d^i))} \qquad (1.5)$$

$$= \sqrt{(1 - t/d^i)^2 \cdot \mu(S_{i-1}^2(x, y, t + t_e^{i-1})) - ((1 - t/d^i)^2 \cdot \mu^2(S_{i-1}(x, y, t + t_e^{i-1})))} \qquad (1.6)$$

$$= (1 - t/d^i) \cdot \sqrt{\mu(S_{i-1}^2(x, y, t + t_e^{i-1})) - \mu^2(S_{i-1}(x, y, t + t_e^{i-1}))} \qquad (1.7)$$

Under the reasonable assumption that the average frame intensity does not change significantly from frame to frame within shots, the second multiplicand in (1.7) can be regarded as roughly constant over a short period of time. Hence, the intensity scaling is directly displayed in the standard deviation of the pixel intensities. The scaling function used during video production (here $1 - t/d^i$) and the standard deviation of the pixel intensities are identical except for a constant factor.

Based on this characteristic pattern of fades in the standard deviation of pixel intensities a simple fade detector [8,9] can be constructed as follows:

```
1. Search for all monochrome frame sequences $R = \{R_1, ..., R_r\}$ in the video. A sequence $R_i$ of monochrome
   frames is identified by a sequence of frames whose standard deviation of pixel intensities is below $\sigma_{max}$.
2. For each range $R_i = \{f_a, ..., f_e\}$ of monochrome frames do
   2.1. // Search for fade in
        2.1.1. Set $n = 2$ and calculate the line of regression over $\{\sigma(f_e), ..., \sigma(f_{e+n})\}$
        2.1.2. Increment $n$ and re-compute the line of regression.
        2.1.3. If the correlation decreases by more than 3% or the slope is more than halved
               then if the minimum fade length $l_{min}$ is not reached go to 2.2, else go to 2.1.2
               else add further points if they vary not more than 25% around the line of regression.
        2.1.4. The sequence $F_i = \{f_e, ..., f_{e+n}\}$ is finally classified as a fade-in if the line of regression
               has at least a correlation of $\rho_{min}$ and a slope $\gamma_{min}$.
   2.2. // Search for fade out
        2.2.1. Set $n = 2$ and calculate the line of regression over $\{\sigma(f_{a-n}), ..., \sigma(f_a)\}$
        2.2.2. Increment $n$ and re-compute the line of regression.
        2.2.3. If the correlation decreases by more than 3% or the slope is more than halved
               then if the minimum fade length $l_{min}$ is not reached go to 2, else go to 2.2.2
               else add further points if they vary by not more than 25% around the line of regression.
        2.2.4. The sequence $F_i = \{f_{a-n}, ..., f_a\}$ is finally classified as a fade-out if the line of regres-
               sion has at least a correlation of $\rho_{min}$ and a slope $\gamma_{min}$.
```

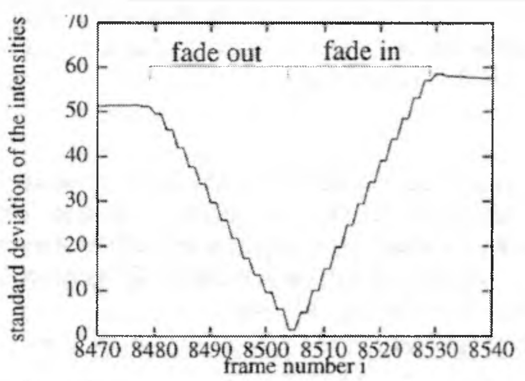


Figure 3: Staircase-like pattern of the standard deviation of pixel intensities during fades in TV spots.

This algorithm detects also a very special type of fade which we observed during TV spots in our experiments: Instead of the scaling factor is having been adjusted for every frame, it had been altered only every second frame. This resulted in a staircase-like pattern such as shown in Figure 3.

To reduce possible false hits, the actual algorithm calculates the standard deviation individually for each of the three RGB color channel. The same characteristic pattern holds for each of them since the intensity can be viewed as a linear combination of the RGB pixel values ($Y = 0,2125R + 0,7154G + 0,0721B$). Note, however, that unlike all other requirements, the slope requirement should only be applied to the color channel with the steepest slope.

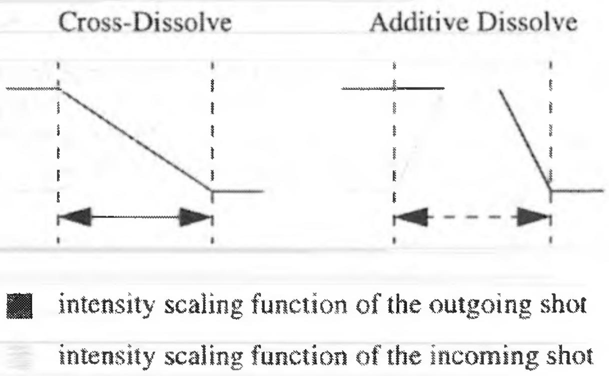
The parameters of the fade detection algorithm are summarized in Table 4.

Parameter	Description
l_{min}	minimum length of main linear segment
ρ_{min}	minimum correlation
γ_{min}	minimum slope of the steepest linear segment
σ_{max}	maximum standard deviation at the end of a fade-out or at the beginning of fade-in.

Table 4: Parameters of the fade detection algorithm based on the standard deviation of pixel intensities.

2.4 Edge-based Contrast

Dissolves are produced by fading out the outgoing and fading in the incoming shot. Two types of dissolves are common: the cross-dissolve and the additive dissolve [1]. Their respective scaling functions for incoming and outgoing shots are shown in Figure 4. Independent of the type of scaling function a spectator observes a loss of contrast and sharpness of the images during a dissolve that generally reaches its maximum in the middle of the dissolve. Hence, the basic idea of the subsequently defined edge-based contrast feature is to capture and emphasize the loss in contrast and/or sharpness to enable dissolve detection.



Cross-Dissolve
Additive Dissolve
intensity scaling function of the outgoing shot
intensity scaling function of the incoming shot

Figure 4: Typical intensity scaling function applied to produce dissolves

The edge-based contrast feature captures and amplifies the relation between stronger and weaker edges. Given the edge map $K(x, y, t)$ of frame f_t (we use the Canny edge detector [4]) and a lower threshold value θ_w for weak and a higher threshold value θ_s for strong edges, the strengths of strong and weak edge points are summed up by

$$w(K) = \sum_{x,y} W_K(x, y) \text{ and } s(K) = \sum_{x,y} S_K(x, y) \tag{1.8}$$

with

$$W_K(x, y) = \begin{cases} K(x, y) \text{ if } \theta_w \le K(x, y) < \theta_s \\ 0 \quad \text{else} \end{cases} \text{ and } S_K(x, y) = \begin{cases} K(x, y) \text{ if } \theta_s \le K(x, y) \\ 0 \quad \text{else} \end{cases} . \tag{1.9}$$

Then, the following formula defines the *edge-based contrast* (EC)

$$EC(K) = 1 + \frac{s(K) - w(K) - 1}{s(K) + w(K) + 1}, \; EC(K) \in [0, 2] \tag{1.10}$$

It possesses the following features:

- If an image lacks strong edges, the *EC* is 0. Examples are night scenes of little contrast and monochrome frames.
- If the number of weak edges clearly exceeds the number of strong edges, the *EC* lies between 0 and 1.
- If the number of weak edges is roughly equivalent to the number of strong edges, the *EC* is about 1.
- If the number of strong edges clearly exceeds the number of weak edges, the *EC* lies between 1 and 2.
- If the image contains only strong edges, the *EC* approaches 2.

Note, that the *EC* is only little affected by slow local or global motion. However, rapid motion may influence it in a manner similarly to that of a dissolve, since edges get blurred.

Figure 5 depicts some examples of how dissolves temporally influence the *EC*. It can easily be recognized that a dissolve coincidences with places of distinct local minima, surrounded by steep flanks. The boundaries of a dissolve occur in company with the abrupt end of the steep flanks. This characteristic *EC* pattern of dissolves can be qualita-

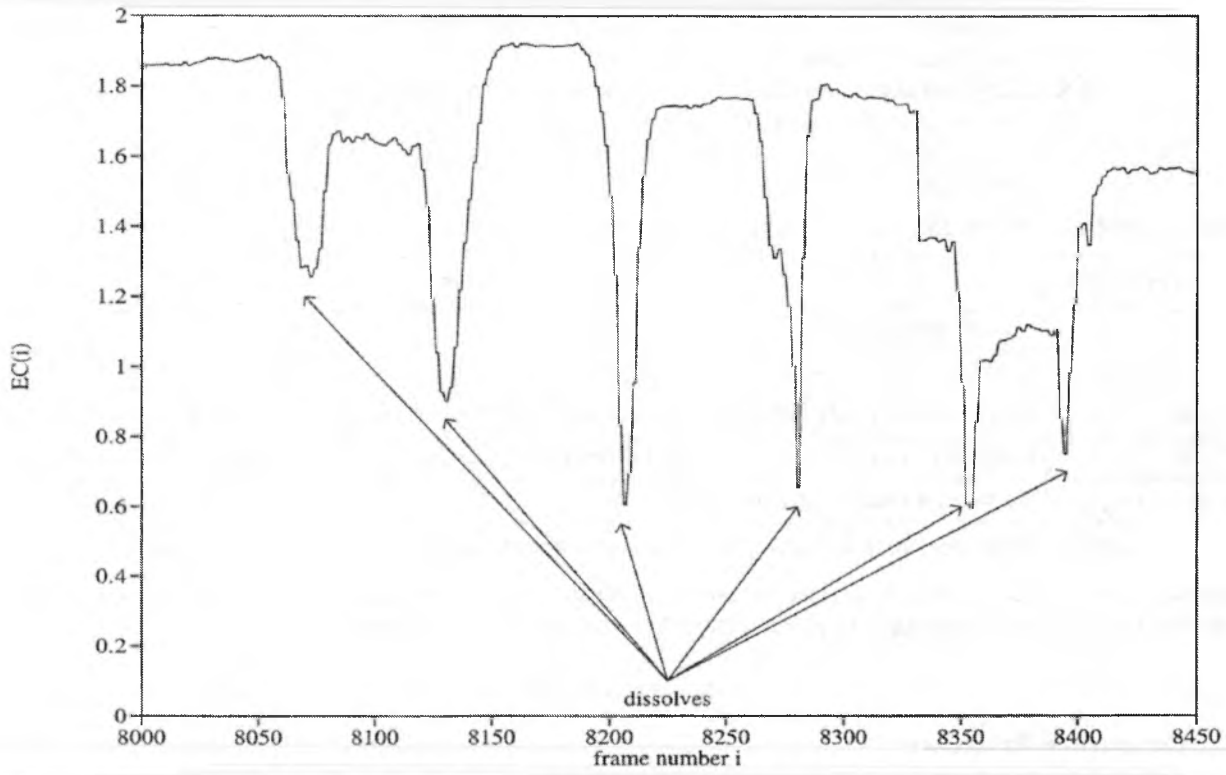


Figure 5: Some examples of how dissolves temporally influence the *EC*

tively explained for cross-dissolves as follows: Commonly the content within a shot changes only gradually from frame to frame, as does the *EC*. Consequently, the graph forms a plateau or an easy rise/descent. During a dissolve, however, the outgoing shot loses its contrast, leading to a reduction of the sum of the strength of strong edges in favor of the sum of the strength of weak edges. As a result, the *EC* decreases rapidly, reaching its minimum in the middle of a dissolve, where the strong edges of the outgoing shot are basically gone and the edges of the incoming shot are still weak. From that point on, the incoming shot gains in contrast. The sum of the strength of strong edges increases to the disadvantage of the sum of the strength of weak edges. Consequently, the *EC* increases rapidly.

The characteristic dissolve patterns in the graph of the *EC* can be identified as follows:

1. Remove all small fluctuations by means of a median filter of size m. Fluctuations may be caused by slight local and/or global motion. In order to preserve the local minima (i.e. the center of a dissolve) and steep rims (i.e. the borders of a dissolve) apply the median filter only to those EC_i values in the graph where none of the following conditions are true:

$$\left\lfloor \frac{m}{2} \right\rfloor \neq \sum_{j=1}^{\left\lfloor \frac{m}{2} \right\rfloor} sign(EC_{i-j} - EC_{i-j+1}) \text{ or } \left\lfloor \frac{m}{2} \right\rfloor \neq \sum_{j=1}^{\left\lfloor \frac{m}{2} \right\rfloor} sign(EC_{i+j} - EC_{i+j-1}) .$$

2. Calculate the relative change g_i from EC_i to EC_{i+1}, i. e. $g_i = (EC_{i+1})/(EC_i + \varepsilon)$.
3. Find all local minima. Local minima are identified as points in the *EC* time series of the properties $g_j \geq 1, j \in \{i+1, ..., i+m\}$ and $1/g_j \geq 1, j \in \{i-m, ..., i-1\}$.
4. For each local minimum at frame i do:

```
4.1.   // Determine left boundary:
    4.1.1. Start at the local minimum, i.e. set $l = i$
    4.1.2. While ( $\sum_{j=1}^{k} 1/g_{l-k} \geq \theta_k$ ) $l$--
    4.1.3. While ( $1/g_{l-k} \geq \theta_n$ ) $l$--
    4.1.4. Calculate the line of regression through $\{EC_{l-k}, \ldots, EC_i\}$. If ($correlation < \rho_{min}$) then
           discard the candidate dissolve and continue loop, i.e. select next local minimum and go
           to 4.1, otherwise decrement $l$ until $EC_{l-k}$ deviates more than ±25% from the line of
           regression or the correlation decreases.
4.2.   Determine right boundary correspondingly. Let the right boundary be at frame $r + k$
4.3.   If $max\{EC_{l-k} - EC_i, EC_{r+k} - EC_i\} \geq \delta$ and $(r + 2k - l + 1) \geq disslen_{min}$ and if the frame
       sequence $\{f_{l-k}, \ldots, f_{r+k}\}$ contains no fade, then the frame sequence $\{f_{l-k}, \ldots, f_{r+k}\}$ represents a
       candidate dissolve.
4.4.   Experiments show, that for some, especially long lasting dissolves several nearby candidate dis-
       solves may be found. Therefore, all candidate dissolves whose local distance is within a radius
       of $disslen_{min}$ are integrated into one solution by choosing the longest candidate dissolve with
       the highest correlation.
```

Table 5 summarized the parameters of the dissolve detection algorithm.

Parameter	Description
m	Size of median filter.
k, θ_k, θ_n	Thresholds for determination of dissolve boundaries
ρ_{min}	Required minimal correlation for left-hand and right-hand flank around the dissolve center
δ	Required minimal *EC* difference between the left/right borders and the center of a dissolve.
$disslen_{min}$	Required minimal length of a dissolve

Table 5: Parameters of the dissolve detection algorithm based on the edge-based contrast feature.

Note that in some video genres such as commercials or music clips of love songs dissolves may occur in rapid succession. It therefore may happen that their determined boundaries overlap slightly.

3 Quality of Detection

3.1 Comparison Procedure

Given the total number of edits, their locations and types, the performance of the different algorithms are measured by three basic numbers:

- **hit rate** h which is the ratio of correctly detected shot boundaries to its actual number
- **miss rate** m which is the ratio of missed shot boundaries to the actual number of shot boundaries, i.e. $1.0 - h$
- **false hits f** which is the ratio of falsely detected shot boundaries to the actual number of shot boundaries

The assignment of detected hard cuts to one of these three cases is simple, since a hard cut does not have any duration and thus occurs at an unambiguous time. However, this is not true of fades and dissolves. They also have an extent. Since the main concern of any shot detection algorithm is to detect either edits in general or a certain type of edit, we decided to count each detected edit as a hit if it temporally overlapped with an actual edit of that type. Multiple detections of the same edit were counted only once.

The hit and false hit rate of each algorithm is influenced by the setting of its parameters. Therefore, we will show how the performance will change with the parameters and what good values are. For each algorithm we will also note qualitatively how well the extent of the edits was determined.

3.2 Video Test Set

The shot boundary detection algorithms were applied to four videos with diverse features (see Table 6). The videos were digitized at 25 fps in M-JPEG at a resolution of 360x270 and a compression of 1:15. A human observer determined for each video the precise locations and duration of the edits.

The first video, named "Dissolves", was selected especially for the measurement of the dissolve detection performance. It therefore consists of 276 dissolves lasting from only 0.16 sec. (4 frames) up to over 5 sec. (>100 frames). The first third was digitized from a live concert called "Night of the Proms '97" showing artists on stage, lighted by headlights. This sequence is somewhat tricky given its dark background and the rapid lighting changes. The last two thirds of that sequence were captured from TV commercials. In contrast to it, "Groundhog Day" is a very calm feature film. Its average duration of shots is much longer, and it exhibits some distinct camera operations. "Heute" is representative of a typical newscast. Anchor person and reports are shown in turn. Within this video sample there exist some spatially restricted edits. They were not classified as edits in our work. The final video sample contains one episode of "Baywatch". It was recorded together with its commercials.

Video	Dissolves	Groundhog Day	Heute (Newscast)	Baywatch	Σ
duration (hh:mm)	00:17	01:34	00:11	00:51	02:53
# of cuts	140	773	78	976	1896
# of fades	12	7	1	19	39
# of dissolves	276	6	2	101	385
total # of shots	429	787	82	1097	2395
Ø shot duration	2.36	7.19	7.56	2.77	

Table 6: The test video set.

3.3 Experimental Results

3.3.1 Color Histogram Differences

This hard cut detection algorithm is controlled by 3 parameters. The most important one is θ_c. Its effects on hit, false hit and miss rates at $l_c = 5$ and $s_c = 2$ are shown in Figure 6 for the four video sequences. False hits and misses are mainly caused by action scenes and a several artistic edits. Figure 6 also shows a common problem. There is no global threshold that gives best results for all types of videos. A local threshold such as that proposed in [13] may resolve that problem.

3.3.2 Edge Change Ratio

Many parameters of this algorithm have to be chosen properly. In general, the following statements are true:

- The hit rate and the number of false hits decreases for hard cuts and increases for dissolves with increasing r.
- A high value of $sumreg_{reg}$ lowers the hit and false hit rates since it reduces the number of maxima found, i.e. the number of possible edit locations.
- A larger radius s for isolated local peaks reduces the hit and false hit rates.
- A higher threshold θ_n also reduces the hit and false hit rates since some dissolves will fall short of the required minimum length.
- An increase in θ_c results in a shift from hard cuts to dissolves.
- The parameter $transrad_{max}$ helps to suppress long-lasting and thus difficult-to-detect edits. The higher its value, the lower the false hit rate.

The following parameter setting yields the best experimental results (see Table 7)

Parameter	r	$sumreg_{min}$	s	θ_n	$transrad_{max}$	θ_c
Value	4	0.05	10	0.01	20	0.4

Video	Hard Cuts		Fades		Dissolves	
	hit rate	false hits	hit rate	false hits	hit rate	false hits
Dissolves	90.00%	17.86%	0.00%	27.27%	71.74%	48.91%
Groundhog Day	97.41%	13.71%	100.00%	657.14%	66.67%	37100.00%
Heute	91.03%	12.82%	0.00%	100.00%	0.00%	5500.00%
Baywatch	69.36%	9.32%	47.37%	526.32%	66.34%	707.92%

Table 7: Performance of the edge change ratio in detecting various edit types.

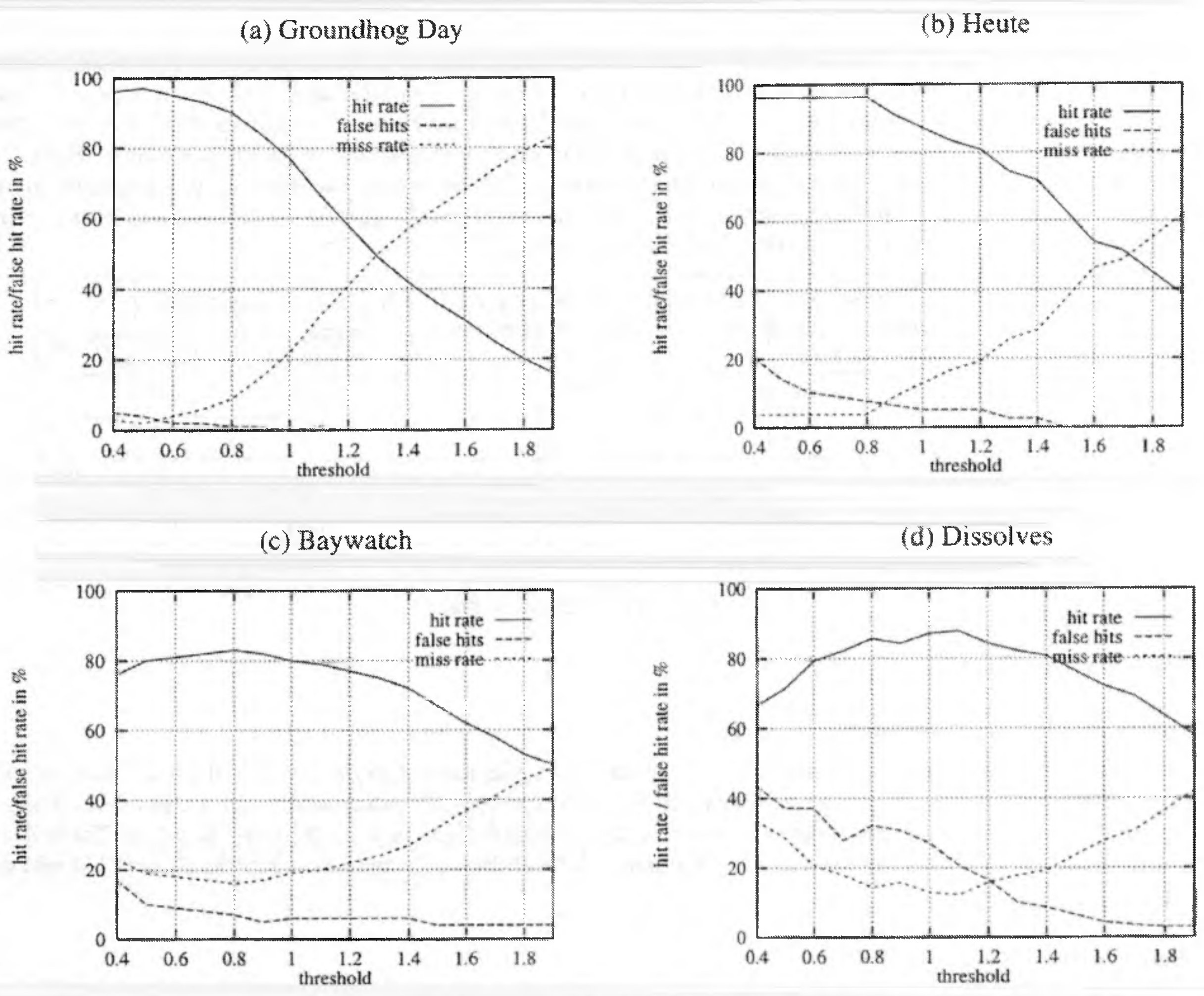


Figure 6: Performance of hard cut detection with color histogram differences in dependence of threshold θ_c at $l_c = 5$ and $s_c = 2$

Some results are striking. Firstly, hard cut detection based on the edge change ratio does not outperform that based on the color histogram differences, although the computational burden is much greater. For "Baywatch" it is even significantly lower. Also, the fade detection performs much worse than fade detection based on the standard deviation of pixel intensities. Cuts from or to black frames were often misclassified as fades. Even more disappointing are the results for dissolves. The false hit rate was so high that the algorithm can only be classified as "not useful" for this task. Many dissolves did not show the characteristic behavior described by Zabih et.al. in the *ECR* time series. This is especially true for long dissolves in which the *ECR* change is so slight that it is hidden by noise. Despite the global motion compensation it was nonetheless very sensitive to motion.

The algorithm cannot be used to determine the boundaries of fades and dissolves. At the borders of dissolves, the *ECR* virtually fails to respond at all. The same is true for the left and right borders of a fade-out and fade-in, respectively.

3.3.3 Standard Deviation of Pixel Intensities

The performance of the fade detector was always very high. On average, the parameter combination $\rho_{min} = 0.9$, $\gamma_{min} = 0.5$, $\sigma_{max} = 10$ and $l_{min} = 10$ yielded the best balanced performance (see Table 8). The hit rate varied between 83.3% and 100%, while there were 0 false hits for "Dissolves" and "Heute". The false hit rates of 85% for "Groundhog Day" and 68.42% for "Baywatch" seem to suggest that the detector has difficulties with feature films and action series, but this is not true. Instead, the false hit rate documents that various artistic edits have been used which are not fades in the strict sense, though they have the same effect. In one example, the camera zooms in rapidly to an open, but dark mouth. In another example, the camera was part of a fight between two people, and the video swiveled from the clothing to the monochrome sand of the beach.

In most cases the fade detector was also able to determine the boundaries of a fade to within about ± 1 frames towards the fade's monochrome frames and to within ± 2 frames towards the fade's other boundary.

ρ_{min} / γ_{min} / σ_{max} / l_{min}	0,85 / 1,0 / 10 / 10		0.90 / 0,5 / 10 / 10		0,90 / 0,5 / 15 / 10	
Video	**hits**	**false hits**	**hits**	**false hits**	**hits**	**false hits**
Dissolves	75.00%	8.33%	83.33%	0.00%	100.00%	0.00%
Groundhog Day	85.71%	57.14%	100.00%	85.71%	100.00%	242.86%
Heute	100.00%	0.00%	100.00%	0.00%	100.00%	0.00%
Baywatch	94.74%	36.84%	94.74%	68.42%	94.74%	231.58%

Table 8: Performance of the fade detector using the standard deviation of pixel intensities.

3.3.4 Contrast Change

There is great diversity in the duration of dissolves. Some last only a fraction of a second, others last up to 5 seconds. The required minimum duration of dissolves $disslen_{min}$ therefore orients itself to the shortest dissolves occurring in our test videos in order not to lower the hit rate from the outset. It was set to 4 frames at 25 fps. The required correlation ρ_{min} of a dissolve's flanks was determined experimentally to be between 0.85 and 0.9.

The parameters k, θ_k and θ_n have the strongest influence on the number and width of found dissolves. Setting k and θ_k too low will result in a high false hit rate; setting them too high, in particular the irregular dissolves will be lost. Good results were achieved with k = 5 and $\theta_k = 0.015$. θ_n determines the final extension of the dissolves. We used $\theta_n = 0.0075$ in the experiments.

Parameter δ has the deepest impact on the hit and false hit rates as shown in Figure 7. The false hits decrease drastically with the increase of δ.

The results of dissolve detection are shown in the second column of Table 9. The hit rate ranges from 73.3% to 100% for "Dissolves", "Heute" and "Baywatch". Only "Groundhog Day" shows a very low hit rate of 16.67%. However, the dissolves in "Groundhog Day" are not representative. For instance, the first dissolve blends from a cloudy sky into another cloudy sky. Even the author did not recognize the dissolve the first time!

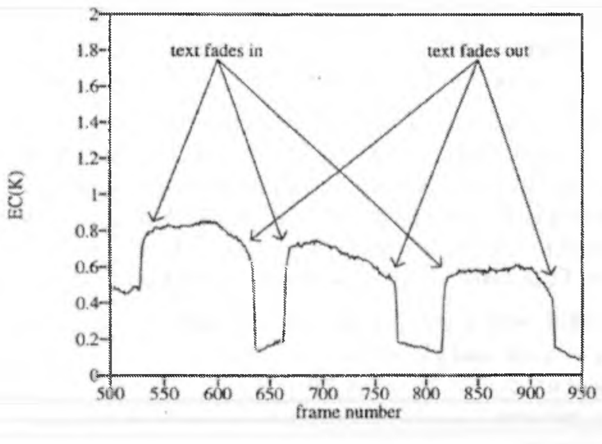


Figure 8: Text occurrences and their effects on the EC.

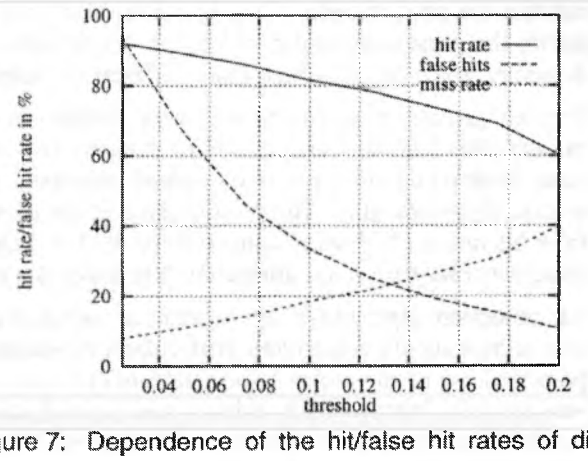


Figure 7: Dependence of the hit/false hit rates of dissolve detection from δ (video="Dissolves", $disslen_{min}=4$, $\rho_{min}=0.85$, $k=5$, $\theta_k=0.015$ and $\theta_n=0.0075$)

Not all dissolves exhibit such a characteristic pattern as in the illustration in Figure 5. There are numerous situations which degrade the characteristic pattern, such as very long dissolves. Furthermore, there are also effects which can result in a similar *EC* pattern. One example is the superimposition of text (see Figure 8). Several steep flanks caused by the fading in and out of the actors' names in the opening sequence of "Groundhog Day" are clearly visible. The higher *EC* values belong to frames with text, the lower to text-free frames. At the transition the text is smoothly faded in and out. If the fade-out of the previous actor's name is followed immediately by the fade-in of the next actor's name, the *EC* pattern is identical to that of a dissolve and can only be ruled out by the proper choice of δ

Enhancements

The main problem encountered by any dissolve detection method is that there exist many other events that may show the same pattern in the feature graph. One way to reduce the false hits is to check for every candidate dissolve whether its boundary frames still qualify for a hard cut after removal of the candidate dissolve. Table 9 summarizes the results for various θ_c. It demonstrates clearly that this scheme drastically reduces the false hits, while the hit rate decreases only slightly. With $\theta_c = 1.6$ the false hits for "Groundhog Day" are reduced from 8500% to 400%, those for "Heute" from 1150% to 150%, those for "Baywatch" from 558% to 182% and those for "Dissolves" from 35% to 10%. Note that at $\theta_c = 0.8$ the contrast feature always with the exception of "Groundhog Day" shows a higher hit rate at a much lower rate of false hits than does the ECR feature (see Table 9).

test video		$\theta_c = 0$		$\theta_c = 0.8$		$\theta_c = 1.6$		ECR	
		hits	**false hits**	**hits**	**false hits**	**hits**	**false hits**	**hits**	**false hits**
Dissolves	%	81.5	34.8	77.90	20.3	56.2	10.2	71.7	48.9
	#	255	96	215	56	155	28	198	135
Groundhog Day	%	16.7	8500	16.67	3100	16.7	400	66.7	37100
	#	1	255	1	93	1	24	4	1113
Heute	%	100	1150	100	700	100.0	150	0.0	5500
	#	2	23	2	14	2	3	0	110
Baywatch	%	73.3	558.4	71.3	313.9	54.5	182.2	66.4	707.9
	#	74	564	72	317	55	184	67	715

Table 9: Performance of dissolve detection at $disslen_{min} = 4$, $\rho_{min} = 0.85$, $k = 5$, $\theta_k = 0.015$, $\theta_n = 0.005$, $\delta = 0.1$.

4 Conclusion and Future Research Direction

The performance of various existing shot detection algorithms was tested on a diverse set of video sequences. The evaluation focused on the detection, localization and recognition of the three most important types of edits. It turned out that the performance of the universal shot detection boundary algorithm based on the edge change ratio cannot justify the great computational burden. Its performance was always inferior to that of the specialized shot boundary detectors based on color histogram differences, standard deviation of pixel intensities and edge-based contrast.

The recognition of hard cuts was very reliable in most cases. Hit rates of 95% at 5% false hits are attainable. In essence, the false hits are caused by dark or very dynamic scenes with strong object motion, blasts or fast camera pans. Fade recognition not only worked extremely reliably but also very precisely. False hits were mostly caused by artistic, fade-like edits. The performance of the dissolve detectors is more or less dissatisfying. Hit rates of 80% at a false hit rate of 20% were achieved for the test video "Dissolves", however, in real videos with only few dissolves, these percentages are not attainable. Normally, the number of false hits exceeds the number of actual dissolves by far.

All detection algorithms are influenced negatively by global and local motion in the video. Therefore, future approaches should concentrate particularly on identification of local and global motion. Several research groups have proposed use of the audio information to enhance shot boundary detection. In our experience, this will help only in very specific domains such video conferences. Doubtless, a "perfect" shot boundary detection algorithm will only be feasible once the video contents are understood better by computers.

The code for running the various shot boundary detection algorithms can be downloaded via ftp from the host ftp.informatik.uni-mannheim.de or via WWW at http://www.informatik.uni-mannheim.de/~lienhart/MoCA/.

References

[1] *Adobe Premiere 4.0 Handbuch*, Adobe Systems, San Jose, CA, USA, 1995

[2] P. Aigrain and P. Joly. The Automatic Real-Time Analysis of Film Editing and Transition Effects and Its Applications. *Computer and Graphics*. Vol. 18, No. 1, pp. 93-103, 1994.

[3] J. S. Boreczky and L. A. Rowe. Comparison of Video Shot Boundary Detection Techniques. In *Storage and Retrieval for Still Image and Video Databases IV*, Proc. SPIE 2664, pp. 170-179, Jan. 1996.

[4] J. Canny. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 8, No. 6, pp. 34-43, Nov. 1986.

[5] A. Dailianas, R. B. Allen, P. England: Comparison of Automatic Video Segmentation Algorithms. In *Integration Issues in Large Commercial Media Delivery Systems*, Proc. SPIE 2615, pp. 2-16, Oct. 1995.

[6] U. Gargi, R. Kasturi, and S. Antani. Performance Characterization and Comparison of Video Indexing Algorithms. Proc. *IEEE Conference on Computer Vision and Pattern Recognition*, Santa Barbara, CA, pp. 559-565, June 1998.

[7] A. Hampapur, R. C. Jain, and T. Weymouth. Production Model Based Digital Video Segmentation. *Multimedia Tools and Applications*, Vol.1, No. 1, pp. 9-46, Mar.1995.

[8] R. Lienhart. *Methods Towards Automatic Video Analysis, Indexing and Retrieval*. Ph.D. thesis, University of Mannheim, June 1998. in German.

[9] R. Lienhart, C. Kuhmünch, and W. Effelsberg. On the Detection and Recognition of Television Commercials. In *Proceedings of the International Conference on Multimedia Computing and Systems*, Ottawa, Ontario, Canada, pp. 509-516, June 1997.

[10] A. Nagasaka and Y. Tanaka. Automatic Video Indexing and Full-Motion Search For Object Appearances. In Proc. *Second Working Conf. on Visual Databases Systems*, pp. 113-127, Sept. 1991.

[11] O. Otsuji and Y. Tonomura. Projection Detecting Filter for Video Cut Detection. Proc. *First ACM International Conference on Multimedia*, pp. 251-257, 1993.

[12] K. Otsuji, Y. Tonomura, and Y. Ohba. Video Browsing Using Brightness Data. In *Visual Communication and Image Processing*, Vol. SPIE 1606, pp. 980-989, 1991.

[13] B.-L. Yeo and B. Liu. Rapid Scene Analysis on Compressed Video. *IEEE Transactions on Circuits and Systems for Video Technology*, Vol. 5, No. 6, December 1995.

[14] J. Yu, G. Bozdagi, and S. Harrington. In *Proc. International Conference on Image Processing*, Vol. 2, pp. 498-501, 1997

[15] R. Zabih, J. Miller, and K. Mai. A Feature-Based Algorithm for Detecting and Classifying Scene Breaks. *Proc. ACM Multimedia 95*, San Francisco, CA, pp. 189-200, Nov. 1995.

[16] H. J. Zhang, A. Kankanhalli, and S. Smoliar. Automatic Partitioning of Full-Motion Video. *Multimedia Systems*, Vol. 1, No. 1, pp.10-28, 1993.