

# Video cut detection method based on a 2D luminance histogram using an appropriate threshold and a post processing

Youssef Bendraou<sup>1,2</sup>, Fedwa Essannouni<sup>2</sup>, Driss Aboutajdine<sup>2</sup> and Ahmed Salam<sup>1</sup>

<sup>1</sup> University of Lille North of France, ULCO LMPA, BP 699, 62228 Calais, Cedex, France

<sup>2</sup> Faculty of Sciences, GSCM-LRIT Laboratory Associate Unit to CNRST (URAC 29),  
University Mohamed V-Agdal Rabat B.P. 1014, Morocco  
youssefbendraou@gmail.com, fessannouni@gmail.com  
aboutaj@ieee.org, ahmed.Salam@lmpa.univ-littoral.fr

*Abstract:* Segmenting a video is a fundamental component in video content based and automatic video analysis. Video cut detection is considered as a mainstay for video retrieval, especially for the keyframe extraction and the video summarization. In this paper, we present an algorithm for partitioning a video into shots. This algorithm abide to the two following steps. First, we calculate the difference in the luminance histogram between successive frames, which we compare against a predefined threshold to detect the cuts. A statistical study of the dissimilarity measures allows us to calculate that threshold in such a way that there are no missed shot. Second, we use a post processing to eliminate the false detections. Experimental results are presented to demonstrate the good performance of the proposed detector with high values of the evaluation criteria.

*Key-Words:* Video cut detection, Luminance Histogram, Post processing

## 1 Introduction

Algorithms of shot boundary detection have an old and rich history in the video processing. Different techniques have been developed in the literature with reliable performances. Shot change detection is the base for almost all video summarization and high-level video segmentation approaches. A video shot is defined as a continuous sequence of frames taken from a single camera and representing an action over time. Shots are the elementary units which produces a video, and as such, they are considered as the primitives for higher level video retrieval and video content analysis. According to the literature [3, 4, 8, 9, 13] the transitions between shots can be classified in two types : It may be an abrupt or a gradual transition as shwon in fig 1. An abrupt shot transition, also called hard cut or cut, is a sudden change from a video shot to another one. The second type which is the gradual transition [3, 5, 14] occurs when the transition between two shots is accomplished gradually over several frames. The difficulties in finding scene cuts are the camera and objects motion, lighting variations and special effects.

The basis of any video shot boundary detection method consists in detecting visual discontinuities between successive frames. Most of existing approaches [8, 9, 15] use a similarity metric difference between

two consecutive frames, where, if this similarity is higher than a predefined threshold, then a shot boundary is detected. Previous studies confirmed that there is two broadly approaches for video shot boundary detection which are the pixel domain processing and the compressed domain. Comparison between different methods showed that those who operate in the pixel domain are more accurate compared to methods in the compressed domain which are faster.

In the next section, related works on scene change detection are discussed. Following the analysis of advantages and disadvantages of each approach, we propose another solution to the problem in section 3. Section 4 gives an overview of the results obtained and a comparison with the methods explained in this paper. Conclusion and some future improvements are provided in the last section.

## 2 Previous work

In this section, we give an overview of the most popular existing approaches for detecting shot boundaries. According to the literature [2, 5, 8], these approaches rely on extracting features from frames, measuring the visual discontinuities between two consecutive frames, then comparing with the threshold to declare even a frame is a cut or not. Some of the video cut detection methods are discussed in the next subsections.



Figure 1: Different type of transitions in a video. First row : hard cut, others : gradual transition

## 2.1 Pixel based approaches

One of the first metrics that have been used, we can list the pixel wise comparison [4] which evaluate the differences in the intensity values of corresponding pixels in two consecutive frames. The easiest way to detect if two frames are different is to calculate the sum of absolute pixel differences using equation (1) and compare it against a threshold. The main drawback of these methods is that they are very sensitive to camera motion, object movement and noises.

$$D(k, k+1) = \sum_{i,j} |I_k(i, j) - I_{k+1}(i, j)|, \quad (1)$$

## 2.2 Edge change fraction

Zabih et al. [5] introduce another popular scene change detection schema based on edge detection using the edge change ratio (ECR). In their work, the method can detect and classify a variety of scene changes including cuts, fades and dissolves. The ECR is defined as follows :

$$ECR(k) = \max\left(\frac{X_k^{in}}{\sigma_k}, \frac{X_{k-1}^{out}}{\sigma_{k-1}}\right) \quad (2)$$

where  $X_k^{in}$  and  $X_{k-1}^{out}$  represent respectively the number of incoming and outgoing edge pixels in frame  $k$  and  $k-1$ .  $\sigma_k$  is the number of edge pixels in frame  $k$ . If the obtained ECR is greater than a Threshold  $T$ , a cut is detected. This method provides a large number of false detection when a high-speed motion occurs in the video scenes.

## 2.3 Histogram based methods

Histogram differences based methods are the most used for video cut detection, since they are fast, accurate and very effective [4, 13, 14, 10]. Several similar studies have been performed so far in this sense, with a difference in the choice of the parameters used, such as the color space, the threshold calculation which will determine the shot changes or even in the case of a pre processing. In most cases the similarity measure is calculated according to the equation (3).

$$CHD_k = \frac{1}{N} \sum_{r=0}^{2^B-1} \sum_{g=0}^{2^B-1} \sum_{b=0}^{2^B-1} |p_k(r, g, b) - p_{k-1}(r, g, b)|, \quad (3)$$

where  $p_k(r, g, b)$  is the number of pixels color  $(r, g, b)$  in the frame  $k$  of  $N$  pixels. If this distance is greater than a predefined threshold, a cut is detected.

The work in [10] was subject to a comparison of the histogram differences in different color space with different metrics. The authors noticed that the histogram intersection (see equation (4)) in the HSV color space gives the best results. They also underline that the luminance is an important feature for detecting the shots, but it didn't perform well alone.

$$INTH_k = 1 - \frac{1}{N} \sum_{i=0}^{2^B-1} \min(h_k(i), h_{k-1}(i)), \quad (4)$$

In their work, Priya et al. [2] divided each frame into  $R$  regions. The bin wise histogram difference

between each block of two successive frames is calculated using the same equation (3). The similarity between two consecutive frames is represented by the sum of similarities between all regions in those frames (equation (5)). In their work, they use a global threshold value.

$$BBHD_k = \sum_{r=1}^R CHD_{k,r}. \quad (5)$$

The drawback of this method is that it may produce missed shot if two frames have a quite similar histogram while their contents are dissimilar.

## 2.4 Other similarities

Apart from these traditional methods, there are many different approaches based on other features and similarity measures. A. Whitehead et al. [11] present a new approach that uses feature tracking as a metric dissimilarity. The authors use a corner-based feature tracking mechanism to indicate the characteristics of the video frames over time. The inter-frame difference metric is the percentage of lost features from frames  $k$  to  $k+1$ . In the case of a cut, features should not be tracked. However, there are cases where the pixel areas in the new frame coincidentally match features that are being tracked. In order to prune these coincidental matches, they examine the minimum spanning tree of the tracked and lost feature sets. In their work, they also propose a method to automatically compute a global threshold to achieve a high detection rate.

We can conclude, from this state of art, that a good video cut detection method highly depends on the feature extracted, the similarity measure used and the threshold calculated. It is true that the methods based on histogram differences give the best results although they remain limited. Their major drawback is that they are sensitive to the illumination condition of the video. A small variation of light in the same shot can be detected as a scene cut.

## 3 The Proposed Method

As we have seen in the precedent section, every type of methods have its advantages and disadvantages and we can say that there is no method which can detect all the scene cuts and gives a perfect results till now. In this paper, we propose a new video shot boundary detection technique, in two steps, based on the histogram differences and the correlation coefficient.

In our work, the histogram differences operates in a combined color space  $YV$ . We represent a frame with its luminance component  $Y$  [16] and the brightness component  $V$  from the HSV color space. This

new combination was used in order to retrieve additional information from color images, and has shown its effectiveness for video scene change detection. The advantage of this combined color space is that it is less sensitive to different illumination changes that can operate in a video sequence. In the following, we explain the different steps of our approach.

First of all, each frame is converted in its luminance and brightness components, then we calculate the histogram differences between every two consecutive frames using the equation (6) to construct the dissimilarity vector. A statistical analysis of the latter will lead us to define an appropriate global threshold. We will explain in more detail the selection and calculation of our threshold in section 3.2. The feature extraction used is explained in the section 3.1. The extracted features are compared against the threshold  $T$ , where if the distance is greater than  $T$ , a video cut is detected. Two consecutive frames belonging to the same shot will have a much smaller distance than two consecutive frames belonging to different shots. The idea here is to use the histogram differences as a first filter, so to have afterwards a selection of scene cuts  $SC$ . This first selection is made using the calculated threshold  $T$  which is adjusted in such a way that there is no missed shot. Therefore, we will have a high rate of false detections. To overcome this, we calculate the correlation coefficient between each selected frame in the first step and its previous one in the video. If this ratio is less than a predefined threshold, we keep the current frame as a scene change, otherwise, it will be considered as a false detection. Fig 2 outlines the different steps of the proposed method.

### 3.1 Feature extraction Algorithm

The following steps explain the first part of our algorithm. We calculate the histogram differences between each two consecutive frames which we compare with the threshold  $T$ , so to have a set of scene cuts  $SC$  that contains all the true scene cuts  $TSC$  and some false detections  $FD$  that we eliminate thereafter. **Step 1 :** Extract Luminance and Brightness from the RGB color image.

**Step 2 :** Calculate the histogram distance between two successive frames using the following equation

$$HD_k = \sum_{j=1}^B |Y_k(j) - Y_{k-1}(j)| + |V_k(j) - V_{k-1}(j)|, \quad (6)$$

where  $Y_k(j)$  and  $V_k(j)$  denotes respectively the luminance and brightness histogram value for the  $k$ th frame. It can be presented as the sum of the pixels belonging to the color bin  $(y, v)$  in the frame  $k$ .  $B$  represents the number of bins.

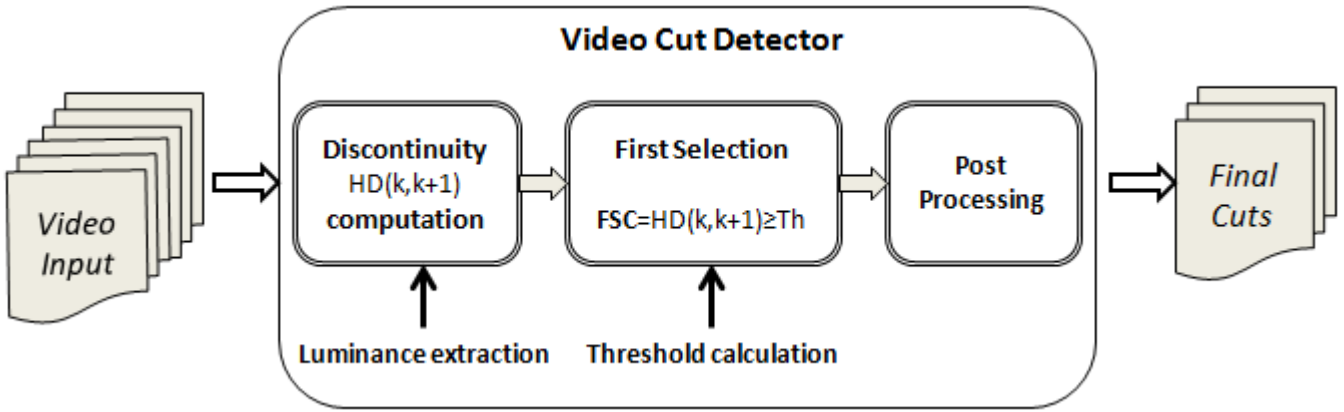


Figure 2: Different steps of our video cut detector

**Step 3 :** Repeat steps 1 and 2 until there are no more frames in the given video sequence to extract the similarity feature  $HD_k$  between all the consecutive frames.

**Step 4 :** Calculate the threshold  $T$  (section 3.2) for the distance extracted. A scene cut is detected when the similarity  $HD_k$  is larger than  $T$

$$\begin{cases} HD_k > T & frame_k \in SC \\ HD_k \leq T & frame_k \notin SC \end{cases} \quad (7)$$

where the correspondig  $k$ th frame is the cut frame. The set SC will be as follows :

$$SC = \{TSC_1, TSC_2, \dots, TSC_n, FD_1, FD_2, \dots, FD_m\}.$$

The cut detection results for the first step are shown in fig 3. After that, we will conduct a post-processing to eliminate false detections FD.

### 3.2 Threshold calculation

Many video cut detection algorithms have been proposed in the past, and in most approaches several parameters and thresholds are used to find the shot boundaries. The common challenge of all these methods is the selection of the threshold that can determine the level of variation between distances, to thereby define a shot boundary.

Our initial goal is to define a policy choice of the threshold that best characterizes the shot changes in a video sequence. Such a choice of the threshold is dependent on the HD distance vector, which represents the similarity between each two consecutive frames. If we consider the observations vector HD, we can observe from fig 4 that the distribution of its values have the allure of a log-normal distribution<sup>1</sup>. And

<sup>1</sup>If a random variable  $X$  is log-normally distributed, then  $Y = \log(X)$  has a normal distribution [1].

we know that if a random variable  $X$  is normally distributed with  $N(\mu, \sigma)$ , then the interval  $\mu \pm 2\sigma$  covers a probability of 95.5% of the observations [1], which means that only 4.5% of the observations are left in the interval  $]0; \mu - 2\sigma] \cup [\mu + 2\sigma; +\infty[$ . Since we are only interested in wide distances we can restrict the threshold selection to the interval  $I_c = [\mu + 2\sigma; +\infty[$ . The threshold should be chosen so as to detect any shot change. We assumed that the vector of similarity HD will have the same characteristics, and we have fixed the minimum value of the interval  $I_c$  as an initial threshold (equation (8)). Afterward, we have tested and varied our threshold to select one that will give the best results, ie more generally our threshold is calculated using equation (9).

#### 1. Initial Threshold

$$T = \min(I_c) = \bar{x} + 2\sigma_x, \quad (8)$$

#### 2. General Threshold

$$T = \bar{x} + \alpha\sigma_x. \quad (9)$$

Where  $\bar{x}$  and  $\sigma_x$  are the mean and the standard deviation.  $\alpha$  is a fitting parameter. According to our experiment, the number of missed shot increase linearly with the value of  $\alpha$ . Otherwise, if  $\alpha$  is small, the number of false positive will be high. We have tested and varied the values of  $\alpha$  to calculate the threshold, in order to choose the one that will give the best results. We noticed that when we choose a small value for  $\alpha$ , we have very good results for some videos while the number of false detections is higher for others, but at least in both cases, the smaller  $\alpha$  is, the lesser the number of missed shots will be. However, when we take a larger value, the number of missed shot is high, but at least there is fewer false detections.

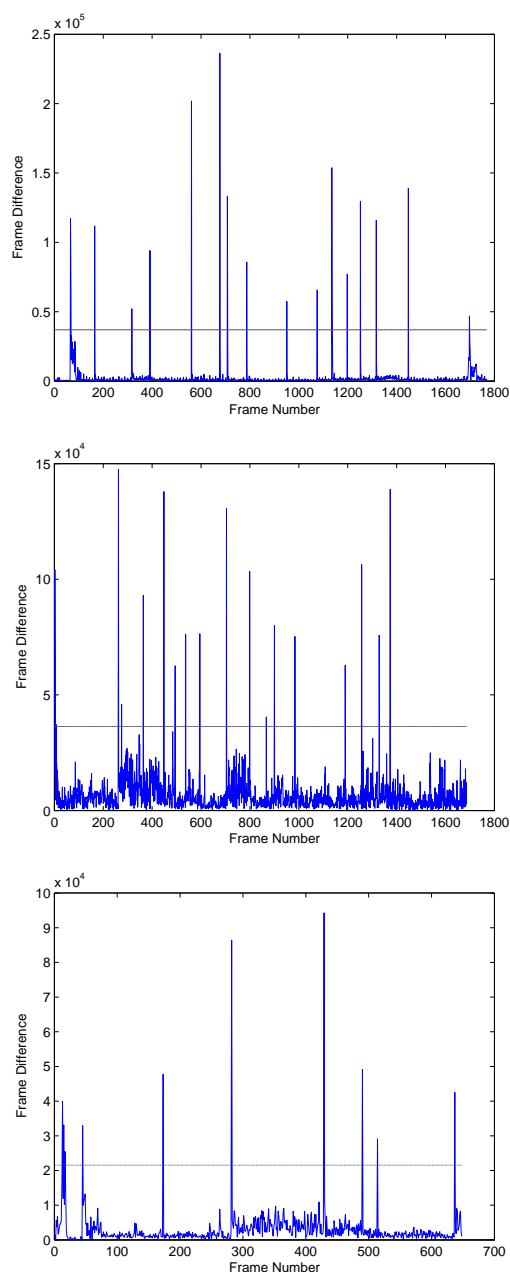


Figure 3: Dissimilarity measure for the video segment : Top: Bor03.mpg, Middle: Indi001.mpg, Bottom: Lisa.mpg

### 3.2.1 Threshold selection

Since the threshold depends strongly on the parameter  $\alpha$ , the challenge is to find out how to select the optimal value  $\alpha^*$  that will render the best result for each video. In this work we evaluate our approach basing on statistical error measures. For each video there is two classes of detection errors. The first error type is related to the quantity of false detections, which can be reduced by post-processing. While the

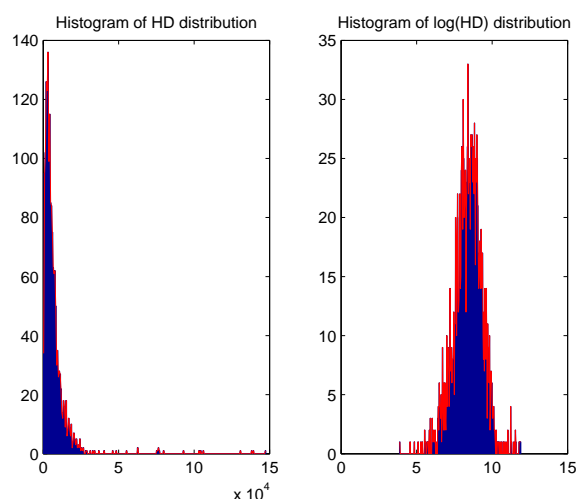


Figure 4: An example of the HD distribution for the video Indi001.mpg on the left vs its log distribution on the right.

second class of detection error is due to the amount of missed shots, and are most of the time difficult to recover. This statement is an important point to consider while attempting to select the appropriate parameter  $\alpha$  for our approach. Since the policy of parameter selection is to minimize the number of missed shots given by the recall measure, we select as optimal value  $\alpha^*$ , the one result to the minimum recalls average among a specific collection of videos (see Table 3). Such optimal parameter  $\alpha^*$ , is expected to result into a small rate of false detections, which are eliminated considering a special post processing.

### 3.3 Post processing

In our work, we adjust the parameter  $\alpha$ , used to calculate the threshold, so as to ensure that the number of missed shots will be minimal. Obviously, the number of false detections will be high. Most of them are due to illumination change. The fact that the methods based on the difference histogram are sensitive to different illumination change in a scene, it is necessary to take an interest in these illumination changes, so to be able to eliminate them. In our case, the choice of the post-processing should be invariant to these changes. A lighting changes between two images (having the same visual content) can be seen as a linear combination according to the equation 10. A special case of the mutual information is the correlation. More specifically, the correlation is a particular case in which the dependence relationship between the two variables is strictly linear, which is the case for the illumination changes in our work.

Table 1: Description of the experimental video sets

Video sequences	#frames	duration	#cuts
Commercial2.mpg	235	8	0
Bor03.mpg	1770	59	14
Indi001.mpg	1687	57	15
Sexinthecity.mpg	2375	95	41
VideoAbstract.mpg	2900	116	17
Lisa.mpg	649	21	7
<b>TOTAL</b>	<b>9616</b>	<b>356</b>	<b>94</b>

Table 2: The results of the two steps

	STEP 1		STEP 2	
	$N_F$	$N_M$	$N_F$	$N_M$
Commercial2.mpg	0	0	0	0
Bor03.mpg	7	0	0	0
Indi001.mpg	4	0	0	0
Sexinthecity.mpg	3	0	1	0
VideoAbstract.mpg	2	0	0	0
Lisa.mpg	3	0	0	0
<b>TOTAL</b>	<b>19</b>	<b>0</b>	<b>1</b>	<b>0</b>

$$I_1 = \alpha I_2 \quad (10)$$

We calculate the correlation coefficient  $C$  between each  $frame_k \in SC$  and its previous one using the equation (11). If this ratio is significantly higher than 0.5 : this means that the two frames are similar to quite a high percentage, in this case the  $frame_k$  will be considered as a false detection FD and it will be removed from the set SC. Otherwise, the  $frame_k$  will be a true scene cut and then maintained in the set SC.

$$C = \frac{\sum_{i=1}^N (X_i - \bar{X}) \cdot (Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^N (X_i - \bar{X})^2} \cdot \sqrt{\sum_{i=1}^N (Y_i - \bar{Y})^2}}, \quad (11)$$

where  $X$  represents the  $frame \in SC$  and  $Y$  its previous one in the whole video.  $\bar{X}$  is the mean value and  $N$  is the number of pixels.

## 4 Experimental Results

In this section, we present the experimental results to evaluate the success of the proposed model. We tested our method on various standard video databases, especially against a set of videos used in [11] available at



Figure 5: Sample pictures taken from video sequences

[6], and other videos from [7] as shown in Table 1. Fig 5 illustrates some frames belonging to the video sequences used. Our proposed algorithm is implemented in Matlab 2009.

To evaluate the proposed cut detection algorithm, the precision (P), the recall (R) and the combined measure (F) are calculated using equations (12), (13) and (14) [2, 3, 11, 15]. The precision measure is defined as the ratio of the number of correctly detected cuts to the sum of correctly and falsely detected cuts. The recall is defined as the ratio of the number of detected cuts to the sum of the detected and undetected ones. The higher these ratios are, the better the performance.

As previously discussed in section 3.2, we set the minimum value of the parameter  $\alpha$  to 2, which gives us a percentage of 100% for the recall criterion R, or we noticed that the number of false detections remains high. To do this, we calculated the threshold for different values of  $\alpha$  as shown in Table 3, and select the optimal value  $\alpha^*$  as the maximum value that gives a minimum of false detections (which we eliminate thereafter) and no missed shots. Also among the parameters that we adjust at the beginning, the number of bins that we set to 8. After several tests for different values, we noticed that the higher this parameter is, the more lost information we have. Once this is done, we now turn to the experimental results.

First we show in Table 2 the difference between the results of the first stage and the second one which filters the false detections; with  $N_F$  and  $N_M$  are respectively the number of false and missed detections. After that, we present the experimental results with the precision and the recall measures in Table 4. We compared our experimental results with some of the existing methods like the feature tracking method (FTrack) [11], the block based histogram method (BBHD) [2], the pixel wise differences (PWD) [4] and the histogram intersection (INT3D) [10].

Table 3: Variation of the parameter  $\alpha$ 

	Com2.mpg		Bor03.mpg		Indi001.mpg		SITC.mpg		VidAbs.mpg		Lisa.mpg	
	P	R	P	R	P	R	P	R	P	R	P	R
$\alpha = 2$	0.00	1.00	0.54	1.00	0.58	1.00	0.89	1.00	0.74	1.00	0.64	1.00
$\alpha = 2.25$	0.00	1.00	0.61	1.00	0.63	1.00	0.91	1.00	0.81	1.00	0.70	1.00
$\alpha = 2.5$	0.00	1.00	0.64	1.00	0.68	1.00	0.93	1.00	0.89	1.00	0.70	1.00
$\alpha = 2.75$	1.00	1.00	0.78	1.00	0.71	1.00	0.93	0.98	0.94	1.00	0.70	1.00
$\alpha = 3$	1.00	1.00	0.78	1.00	0.75	1.00	0.98	0.95	0.94	0.94	0.70	1.00
$\alpha = 3.5$	1.00	1.00	0.82	1.00	0.74	0.93	1.00	0.93	1.00	0.94	0.78	1.00
$\alpha = 4$	1.00	1.00	0.87	0.93	0.82	0.93	1.00	0.88	1.00	0.94	0.75	0.86



Figure 6: False detection in red

This comparison shows that the proposed method give better results than the existing ones with a percentage of 100% for the recall, which means that there are no missed shots for all the videos. This result is significant when constructing a video summary, from the moment that the number of redundant key frame will be minimal. Also we can observe that our algorithm provides only one false detection shown in fig 6. The red surrounded frame represents the false detection. At first glance, this is due to a rapid movement in the scene, but the fact that the frames 2220 and 2221 are similar, allows us to think that, in the opposite case this movement will not be as significant and will not cause false detection. Fig 7 depicts the comparison of the combined measure  $F$  with various existing methods. Maximum cuts are identified and the average value for the combined measure is 0.99 where for other methods the combined measures are 0.95, 0.96 and 0.88.

$$P = \frac{\text{No. of True shot detected}}{\text{True shot detected} + \text{Falseshot}} \quad (12)$$

$$R = \frac{\text{No. of True shot detected}}{\text{True shot detected} + \text{Missedshot}} \quad (13)$$

$$\text{CombinedMeasure}(F) = \frac{2.P.R}{(R + P)} \quad (14)$$

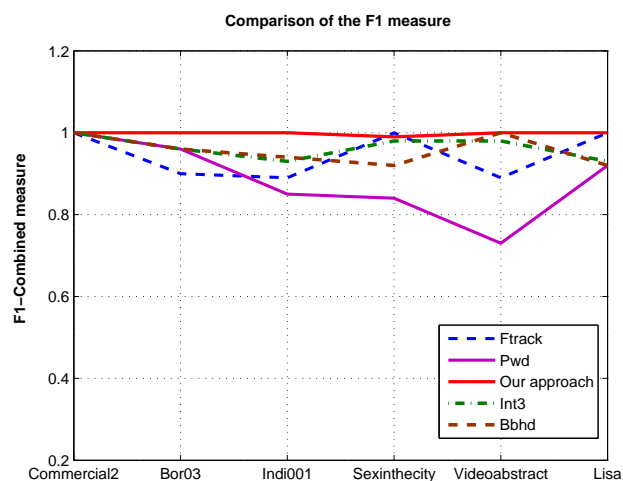


Figure 7: Comparison of the F1 combined measure of discussed methods

## 5 Conclusion

A new algorithm for video cut detection in two steps is proposed in this paper. Our work shows that the methods based on histogram differences, by themselves, are not very efficient, but when using a second filter, we can have very good results. The experimental results show that the proposed method produces much better results. Our approach lay out by a recall percentage of 100%. This result is obtained abiding the two following step, first performing a statistical analysis of the dissimilarity measures allowing us to find an accurate threshold. Second, applying an appropriate post-processing enhancing the precision rate. Thereafter, we consider to extend our approach performing the gradual shot detection allowing our model to cover a larger prospect.

Table 4: Experimental results

	Our Method		FTrack [11]		BBHD [2]		PWD [4]		INT3D [10]	
	P	R	P	R	P	R	P	R	P	R
Commercial2	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Bor03	1.00	1.00	0.82	1.00	0.93	1.00	0.93	1.00	1.00	0.93
Indi001	1.00	1.00	0.98	0.82	0.94	0.93	0.92	0.80	0.93	0.93
Sexinthecity	0.98	1.00	1.00	1.00	0.88	0.97	0.84	0.84	0.98	0.98
VideoAbstract	1.00	1.00	0.89	0.89	1.00	1.00	0.75	0.71	0.94	1.00
Lisa.mpg	1.00	1.00	1.00	1.00	1.00	0.86	1.00	0.86	0.88	1.00
<b>AVERAGE</b>	<b>0.99</b>	<b>1.00</b>	<b>0.95</b>	<b>0.95</b>	<b>0.96</b>	<b>0.96</b>	<b>0.91</b>	<b>0.87</b>	<b>0.95</b>	<b>0.97</b>

## References:

- [1] E. Limpert, W. A. Stahel, and M. Abbt. Log-normal Distributions across the Sciences: Keys and Clues. *BioScience*, 51(5): 341–352, May 2001.
- [2] Priya G.G.L. and Domnic S. Video cut detection using block based histogram differences in RGB color space. *International Conference on Signal and Image Processing (ICSIP)*, Chennai-India, December 2010.
- [3] Yoo, Hun-Woo and Ryoo, Han-Jin and Jang, Dong-Sik. Gradual shot boundary detection using localized edge blocks. *Multimedia Tools and Applications*, 28(3): 283–300, 2006.
- [4] John S. Boreczky and Lawrence A. Rowe. Comparison of Video Shot Boundary Detection Techniques, *Journal of Electronic Imaging*, 5, 122–128, April 1996.
- [5] Ramin Zabih and Justin Miller and Kevin Mai. A Feature-Based Algorithm for Detecting and Classifying Scene Breaks, *ACM Multimedia 95*, 189–200, 1995.
- [6] Video dataset (Web) : <http://www.site.uottawa.ca/~laganier/videoseg/>
- [7] Video dataset (Web) : <http://www.open-video.org/>
- [8] Jun Yu and M.D.Srinath. An efficient method for scene cut detection. *Pattern Recognition Letters*, 22: 1379–1391, 2001.
- [9] Purnima.S.Mittalkod and G.N.Srinivasan. Shot boundary detection : an improved algorithm, *International journal of enginnering Sciences Research*, 4, pp. 8504-8512 (2013)
- [10] U. Gargi, R. Kasturi, and S.H. Strayer. Performance characterization of video-shot-change detection methods, *IEEE Trans. Circuits Syst. Video Technol.*, vol. 10, pp. 1–13, Feb. 2000
- [11] Anthony Whitehead and Prosenjit Bose and Robert Laganieri. Feature Based Cut Detection with Automatic Threshold Selection, *CIVR*, 410–418, Springer 2004.
- [12] Xinghao Jiang, Tanfeng Sun, Jin Liu, Wensheng Zhang and Juan Chao. An Video Shot Segmentation Scheme Based on Adaptive Binary Searching and SIFT. *Advanced Intelligent Computing Theories and Applications. With Aspects of Artificial Intelligence*, 6839: 650–655, 2012.
- [13] G. Lupatini and C. Saraceno and R. Leonardi. Scene Break Detection : A Comparison, *International Workshop on Research Issues in Data Engineering, IEEE Computer Society*, 0, pp. 34, 1998.
- [14] Lienhart Rainer. Comparison of Automatic Shot Boundary Detection Algorithms, *Proc. IS&T/SPIE Storage and Retrieval for Image and Video Databases VII*, 3656: 290–301, 1999.
- [15] Pardo Alvaro, Simple and robust hard cut detection using interframe differences, *In Progress in Pattern Recognition Image Analysis and Applications*, pp. 409–419, Springer Berlin Heidelberg 2005.
- [16] Publication CIE N 15.2, Colorimetry, Second Edition, *Bureau central de la commission internationale de l'eclairage*, Vienne Autriche (1986).