Extracting Cuts From Video Streams in Real Time

Jugoslav Joković, Danilo Đorđević

Abstract – **An approach for extracting scene boundaries for streaming videos is proposed. Presented approach extends the mainstream hard cut scene detection algorithm based on the color-histogram difference by adding new criteria to increase precision and improve computational efficiency, by using additional detection criterion based on spatial distribution of luminance blocks difference during scene cut (and also by utilizing the temporal properties of video by sampling at different time resolutions). We note the speed of extraction which is crucial in streaming tasks and give results on precision and reliability.**

Keywords – **Cut detection, Video Summarization, real-time**

I. Introduction

A scene is defined as sequence of successive video frames captured with one continous operation of capture device. Isolating the original structure of the video by dividing the original sequence to separate scenes is called temporal video segmentation. It strives to detect transitions created by author and find boundary of original scenes that video consist of.

Automatic scene cut detection as a primary method in temporal video segmentation initiates starting and basic step of every video signal processing for displaying or further manipulation. Among primary application of video segmentation in creating the new content (*video editing*), automatic scene cut detection finds its application in algorithms for improving video quality (*video enhancement*), as well as in systems for detection of copyrighted videos and repeated sequnces. For the first case it is desirable to know scene boundaries for complex object tracking algorithms to stop running on the position where scene breaks. In the second case the exact times where scene change occured is used as a *fingerprint* for the video sequence, for later matching with other fingerprints in the database. Here it is also necessary to keep the precision of scene change detection to the videos that are transformed (purposefully or not) with changes in frame rate, bitrate, aspect, contrast etc.

Thanks to the advance of computer science and video technologies there are plenty of effects in video editing and, consequently, a lot of transition types. Majority of these effects can be classified into three base categories, where each transition type can be mathematicaly modelled:

1. Abrupt transitions (*cuts)*, defined as direct concatenation of two scenes without adding any kind of transitional frames, so that transition sequence is empty.

2. Blended transitions (dissolves), where the sequence I(t,x,y) of duration T results from combining two sequnces,

Authors are with the Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Nis, Serbia, E-mail: jugoslav.jokovic@elfak.ni.ac.rs, djdane81@gmail.com

where the first sequence vanishes and second is forming:

$$I(t,x,y) = f_1(t) * I_1(t,x,y) + f_2(t)*I_2(t,x,y),\ t \in [0,T]$$

3. Nonlinear transitions (without specific pattern), where the sequence I(t,x,y) is created by combining two sequences $I_1(t,x,y)$ and $I_2(t,x,y)$ without mathematicaly specified pattern.

Temporal video segmentation is widely researced in video processing field. Resulting methods concentrate mainly on first two transition types, which are in majority of all transitions that are used in real video streams, while computationaly generated effects are mostly ignored. Alternative clasification of scene transitions [1] is based on how two scenes are spatialy and temporaly divided. Their detection is based on identification of new statistical process that the video sequence is being exposed (hard cuts), or that the video sequence was scaled with mathematicaly simple and well defined function (dissolves).

Basic idea of transition detection (scene cut) detection results from assumption that some video characteristics are different during transitions than during scene itself. This is the reason why algorithms for scene cut detection select some features and analyze their change. When significant change of monitored video features occurrs, a scene cut is declared. Color histogram, edges [2] or motion are the main types of video features used for transition detection [1,3]. Usually, smallest units of video – frames - are monitored, where each frame is characterised individualy with its feature [1,2,3,4]. Another (seldom) approach is monitoring the group of frames with some common feature, such as *motion* [5]. Scene change expresses locality principle, meaning that reliable detection is based on some local frame (frame feature) group where discontinuity occurs [6]. Generally, the steps for scene cut detection typical algorithms are frame feature selection, forming local feature group and feature change detection.

This paper presents results of use of algorithms for hard cut scene detection based on on some subsistent examples. For efficiency improvement address approach with additional criterion of spatial distribution of luminance blocks during scene change and time sampling in smaller time intervals and detection on larger time sampling intervals where we used.

One of the possible applications of scene detection is in video summarization area. Typically, video is continuously being read and segmented by using transition detection techniques and than saved back on (other) storage. Once cuts are extracted, it is possible to choose representative by selecting one frame from each cut (usually first or the middle frame) but other combinatations exists as well [8]. It is crucial for such systems to have very fast, possibly real-time transitions detection. A simplified diagram for this systems is shown in Fig 1. For performing transition detection we do not assume in our work that the video stream is MPEG encoded stream [9], although this assumption can simplify some processing steps since in these cases frames are already divided into blocks.

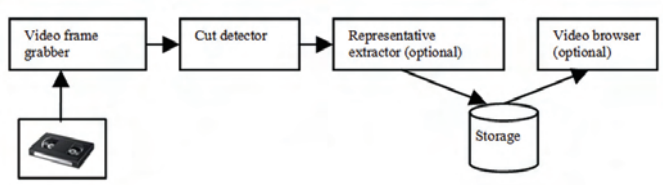


Fig. 1. Typical video summarization system diagram

II. Base Algorithm For Hard Cut Scene Detection

For all frame features used in hard cut scene detection, in literature, as well as in practice, the ones that treat spatial pixel distribution (luminance and chrominance) dominate among others. For its fingerprint compactness and amount of information in it, colour histogram highlights here. It can be crated by concatenation of component histograms to one integral fingerprint, tending to create such fingerprint representation that can be suitable for later comparison with other fingerprints with proper metrics. Metrics most used are simple (*Manhattan*) distance, Euclidean distance and histogram intersection. Regardless on the metrics being used, it is necessary to perform histogram normalization according to the frame size before comparison with other histograms. When scaling the frame to larger spatial resolution the histogram is also being scaled. Moreover, histogram in YC_bC_r domain is used much more than histogram in RGB domain, for its dominance in video format representation.

Abrupt scene changes are detected by finding the discontinuity in appropriate video frame representation and common approach is by calculating differences between frame features (color histograms) among consecutive frames and monitoring their difference - CHD (Color Histogram Difference). Scene changes (cuts) are declared when this difference surpasses some determined threshold. Typical CHD shift is depicted in Figure 2. It is obvious that scene change occurs where CHD plot peaks.

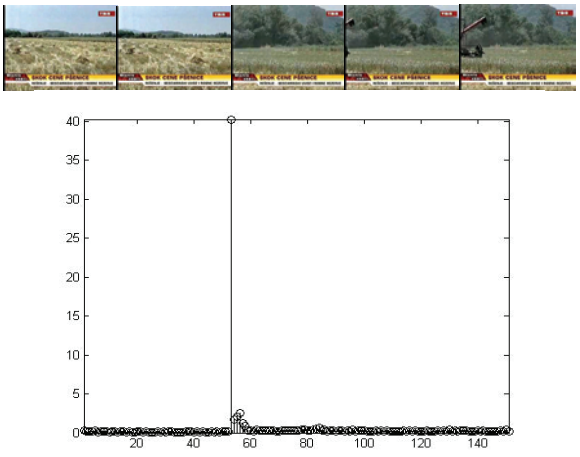


Fig. 2. Color Histogram Difference (CHD) for the video sequence »*News*«

The simplest approach for hard cut scene detection is by taking the global threshold (for example ratio between maximum peak intensities or a function between average values of CHD). Nevertheless, this approach is inefficient because sometimes small changes in CHD can seem as scene cut and vice versa – scene cut indeed occurred but change in CHD was not substantial to surpass the fixed threshold, what depends greatly on the video content being monitored. By introducing the adaptive threshold [6] that establishes itself on average CHD intensity parameter in the window of diameter w. In that case for every frame in the sequence of *2w+1* frames in width, where *w* ranges from 3 to 7 frames, and for all frames inside the window (except the central one) the average value of chosen feature (CHD) is calculated. The hard scene cut over central frame in window is declared when following conditions are fulfilled: its CHD (or $f(x)$) is maximal inside widow:

$$f(t) \geq f(x) \ \forall x \in [t-w, t+w] \qquad (1)$$

and ratio between its CHD value and average CHD values in the window (excluding central frame) crosses some empirically defined threshold ratio:

$$ratio = \frac{(f(t) + c)}{c + \sum_{x \in [t-w, t+w] \setminus \{t\}} f(x)} \qquad (2)$$

As a consequence of characteristic appearances in video relating to frequent motion, a new condition is added – CHD has to be significantly greater (ranging from 2 to three times) than the next maximum CHD in the window. Parameter *c* in expression (2) is used as a correction to avoid situations when scene changes around central frame are neglectingly small, and where a larger motion in the scene (central frame) is manifested as a peak in CHD plot leading to false detection. This method for hard cut scene detection confirmed greatest accuracy and precision [2,4,6].

Functional limits of this algorithm is that it cannot detect two scene cuts in one window, what can be a drawback in the action movies where scene changes can occur even on 3 to 5 frames, but because of great speed human audio-visual system treats these changes as a part of the one same scene.

III. Algorithm For Scene Cut Detection With Additional Spatial-Temporal Criterion

By testing the base algorithm on large video test sequences it can be noticed that previously described approach has drawbacks especially with encoded videos with large difference between I- and P- frames, what can defect the CHD between successive frames. Also, CHD can be disturbed with previously described insignificant motion on the scene which is perceptually irrelevant. Both cases exhibit very little luminance change.

Following that point, an additional criterion is introduced – *luminance difference* (YD), which is based on frame luminance block division and tracking the luminance change on *kxk* blocks for the frames where cut is suspected to occur, where YD is calculated as a sum of absolute difference between each of *kxk* corresponding luminance block average for two frames.

Algorithm for detection can be described by using the following steps:

1. Decode each video frame in video stream by using the *ffmpeg* video library [7] for frame decoding. For each frame calculate color histogram (convert the 3D color histogram to 1D representation), by performing quantization of each pixel intensity value and than form 2D matrix which will sum pixel intensities on *x* and *y* coordinates for each video component (RGB or YC_bC_r).

2. Calculate difference in color histograms between two consecutive frames (CHD) by using the sum of absolute differences of proper array elements (*Manhattan* distance).

3. For each CHD array element (for each frame feature) create a window of size *2w+1* where mean value of CHD will be calculated for elements within window, as well as ratio of this mean value with window-central CHD value.

4. If following conditions are met, declare scene cut:

- Central frame has maximum value inside window
- It is *n* times greater than the next maximum CHD value within window
- The ratio between CHD of the central frame and the mean value of CHD values within window is greater than some empirically declared threshold.
- Luminance difference (YD) on *kxk* blocks for the frames surpasses empirically declared threshold (global threshold)

This approach can help to eliminate flaws of base algorithm, to eliminate compression effects and to lower detection of motion as cuts, with lowering empirically chosen thresholds. Introducing the new criterion further improves base algorithm for cuts being missed in the cases of high motions scenes (action movies) where motion inserts high CHD values very close to the actual cuts, causing the average CHD within window to be much higher than in one at still scenes. Similar problem occurs also in videos that have smaller frame rate and high motion on the scene that cover larger portion of the frame.

This implementation of the algorithm is especially suitable for streaming performance. For both CHD and YD we only need one incoming frame from the frame grabber to extract the color histogram and average per-block luminance, and than calculate differences between those features from previous frame. After that we operate only on the array of feature vectors of the window size *2xk +1* for the cut detection phase. Thus we managed to minimize the memory operations.

Although the explained algorithm proved itself as the most reliable among other algorithms, some improvements are possible. Usage of every frame in detection can be avoided by utilizing the temporal characteristics of video. Namely, we assume that two scene changes aren't possible within some time period of t_s, where 0.5 seconds covers over 99% of possible video content for different nature. As changes are noticeable mainly in the luminance, we will use luminance difference (YD) over *kxk* blocks for two frames sampled at t_s, as described earlier in this section. For the temporal characteric we use sampling time t_s of 0.5seconds, where we are sure that double scene change won't occur. When noticeable change above fixed threshold occurs, we treat such changes either a as a cut or as a high motion. In order to reduce motion, we use higher sampling resolution and calculate YD over higher block number. When significant change is found we declare a cut.

Modified algorithm for cut detection can be described by using the following steps:

1. Decode video frames sampled at temporal resolution of t_s. Use *ffmpeg* video library for frame decoding.

2. Calculate YD between two time sampled frames, by using the sum of absolute differences over chosen block size. If this difference doesn't cross the defined «critical» threshold repeat previous steps, if it is not the case:

3. Start with frame by frame decoding from the previously sampled frame and apply the improved base algorithm, described on the beggining of this chapter.

IV. Results and Analyses

For the video sequences »*News*« luminance difference (YD) between successive frames is tested, for the whole frame as well as for blocks of size *kxk*. First, for illustrational purposes, Fig. 3 depicts results of hard cut scene detection for the video sequence »*News*«, through alongside view of YD feature with time sampling of t_s=0.5 seconds. It can be noticed how algorithm with higher time sampling enables to find periods in a video sequence (frame groups) for which we can say that they don't have cuts.

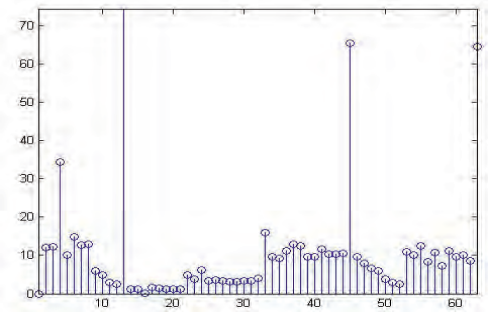
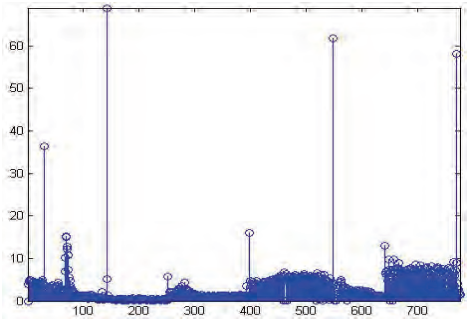
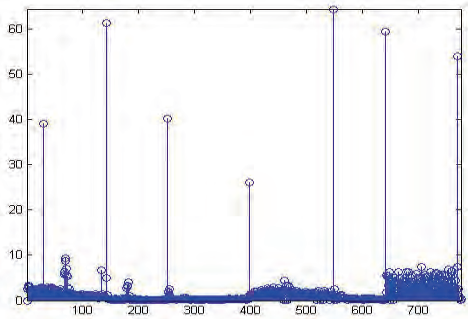


Fig. 3. Cut detection for the video sequence „News“ based on YD for the whole frame, with time sampling t_s=0.5s,

Further, Fig. 4. presents YD with frame time sampling of the video's frame rate (25*fps*), for the whole frame and for the 8x8 blocks within frame. Figure 4.a shows YD plot which is the supporting feature for reliable cut detection where this plot peaks. The result on Figure 4.b, where frame luminace difference is formed by block division, illustrates improvement of the algorithm in increasing reliability of cut detection and lowering the possibility of detecting the motion on the scene as a cut. Comparing the plots in Fig. 4, it can be seen that cuts are more noticeble in the when frame divided into blocks, while values of YD in case frames representing motion are less than YD based on whole frame. As an illustration, in Table 1 are presented results on YD versus number of blocks that frame divided into, for characteristic frames in the video sequence „News“. It can be noticed that in the case of whole frame values YD of some cut frames are less than other representing motion. However, dividing frames into blocks provides possibility of establishing of unique threshold and avoiding false cut detection.



a)



b)

Fig. 4. Cut detection for the video sequence „News“ based on YD for the whole frame, a) for the whole frame, b) for the frame blocks 8x8,

TABLE 1. YD VERSUS NUMBER OF BLOCKS FOR THE SEQUENCE „NEWS“

Frame	*Type*	*Number of blocks*			
		1	2	4	8
30	Cut	36.38	19.95	33.81	39.03
70	Motion	15.27	2.24	4.15	9.36
142	Cut	68.89	61.09	61.09	61.25
252	Cut	5.76	37.22	38.58	40.26
282	Motion	4.39	0.25	0.39	0.58
398	Cut	16.05	13.33	18.40	26.11
460	Motion	6.73	0.91	3.55	4.49
548	Cut	16.05	48.41	51.40	64.37
641	Cut	16.05	53.68	55.79	59.37
656	Motion	9.86	1.42	2.29	2.94
705	Motion	8.38	1.53	5.02	7.41
769	Cut	16.05	53.55	53.55	53.87

When complete frame is in the memory, total processing time depends on complexity of the algorithm for extracting features from each frame (CHD and YD at the same pass over frame) which is in our case O(NxM), where NxM is frame size. Cut detection in the feature space is with uncomparably smaller complexity to the feature extraction. In the terms of real time processing with frame grabbing we are limited to performance of frame grabber (in our case *ffmpeg* as software implementation) that also depends on installed hardware.

The speed of described algorithm depends mostly from the frame resolution of the video file. For the sequence of different resolution, in the Fig. 5. average execution time for clip in length of 200s at 25fps is presented. About one third of total time of algorithm execution was spent for frame decoding from *ffmpeg* library. Testing was performed on comodity PC configuration Intel P4 2 GHz with 2GB RAM.

It can be seen that for frame resolutions of double CIF we have real-time performance, while for full DVD resolution of 720x576 we are two times slower than real-time. If we can consider frame decoding as a significant time loss, our performance is near real-time.

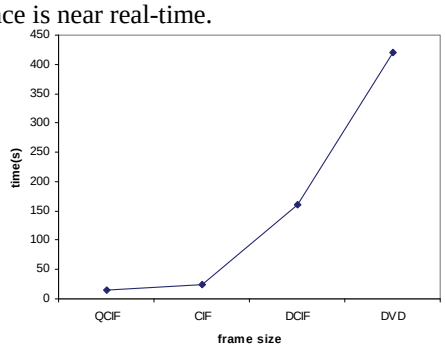


Fig. 5. Average execution time for clip in length of 200s at 25fps

V. CONCLUSIONS

Based on obtained results, the conventional algorithm for scene cut detection by using differences in color histograms of adjacent frames was strengthen with additional detection criteria oriented at spatial and temporal characteristics of a video signal, where the changes in mean luminance difference are used to distinguish cuts from motion for the given scene (to some extent). To increase the speed of algorithm, frame extraction is performed on specific fixed time intervals and afterward the mean block luminance change of adjacent frames is used as a spatial criterion. For a characteristic sequence, together with increased efficiency in processing, the results with lowered false detection rate and higher detection precision rate were attained. Also, speed of the calculation algorithm is considered in terms of real-time performance.

REFERENCES

[1] R. Lienhart, *"Reliable Transition Detection in Videos: A Survey and Practitioner's Guide"*, International Journal of Image and Graphics, vol. 1, no. 3, 2001.

[2] R. Lienhart, *"Comparison of Automatic Shot Boudary Detection Algorithms"*, Proceedings of Storage and Retrieval for Still Image and Video Databases VII, SPIE vol. 3656-29, January, 1999.

[3] I. Koprinska and S. Carrato, *"Temporal video segmentation: A survey"*, Signal Processing: Image Communication, 2001.

[4] Timothy C. Hoad, Justin Zobel: Video Similarity Detection for Digital Rights Management. ACSC 2003

[5] http://www.mathworks.com/matlabcentral/fileexchange/, Block Matching Algorithms for Motion Estimation by Aroh Barjatya

[6] Ba Tu Truong, Chitra Dorai, Svetha Venkatesh, "*New Enhancements to Cut, Fade, and Dissolve Detection Processes in Video Segmentation*", ACMM 2000.

[7] FFmpeg - http://sourceforge.net/projects/ffmpeg

[8] Borko Furht, Video Databases, book, 2003.

[9] A. Fouad, M. Bayoumi, M. Onsi, "*Shot Transition Detection with Minimal Decoding of MPEG Video Streams*", 2006.