International Journal of Signal Processing, Image Processing and Pattern Recognition
Vol. 10, No. 12 (2017), pp.103-118
http//dx.doi.org/10.14257/ijsip.2017.10.12.08

A Classification Approach to Video Shot Boundary Detection

Bose Lungisani, Edwin Thuma and Gabofetswe Malema

Department of Computer Science, University of Botswana, Gaborone, Botswana
bose.lungisani@mopipi.ub.bw, thumae@mopipi.ub.bw, malemag@mopipi.ub.bw

Abstract

Video content retrieval just like information retrieval requires some pre-processing such as indexing, key-frame selection and most importantly accurate video shot boundary detection. Accurate detection of video shots give way for video information to be stored in a manner that will allow easy access. Several algorithms have been developed in this field of study and tested even at the TRECVID 2002, 2005 and 2007 tasks evaluation conferences. Challenges on accurate detection of these different types of video transitions have always been from large object and camera motions as well as fast zooming, flashlights, and change in luminance. These attributes differ from one video sequence to another and features of one video sequence cannot always match with features of another video. Therefore, in our work we use a video specific machine learning approach that leverages information from several shot boundary detection algorithms in order to improve the detection of the shot boundaries on a video sequence. Our results suggest that a classifier built from a combination of block-based motion estimation, RGB histogram based block-based cross-correlation coefficient and RGB histogram based sum of squared difference provided better results with an average F1 score of 0.752 on shot boundary detection for the seventeen videos tested. This proves that a combination of luminance and motion based algorithms improves the detection of video shot boundaries. We also found that the detection of shot cuts and gradual transitions can be improved by using features generated by several shot boundary detection algorithms.

Keywords: *shot, shot boundary detection, trained classifier, frame, image, RGB histogram, gradual transition, shot cut, motion estimation, cross-correlation coefficient, sum of squared difference, TRECVID 2007*

1. Introduction

Video shot boundary detection can be described as an initial and fundamental step in video content indexing, browsing and retrieval. Literature has shown that several applications for the detection of video shot boundaries have been developed based on different algorithms. Existing applications in video shot boundary detection use features for the representation of video frames, shot boundary detection techniques, and shot change detection methods [1]. Features used in the representation of video include color, motion, histograms, image edges, rectangular block of a frame, whole frame, single pixel, *etc*. Shot boundary detection techniques are methods that are used for detecting boundaries and examples include pixel-based shot boundary detection, histogram based shot boundary detection, block based shot boundary detection as well as shot boundary detection using motion descriptor [1]. Shot change detection methods are used to detect the similarity or dissimilarity of features for consecutive frames in a video sequence and decide if a shot change exists between such frames or not. As explained by Rathod *et al.*, [2], applications such as multimedia information systems and distance learning use large amounts of data

Received (September 9, 2017), Review Result (December 1, 2017), Accepted (December 5, 2017)

hence leading to the demand for efficient techniques to store, retrieve, index and summarize video content accordingly.

Since a shot is the basic unit of a video, video shot boundary detection becomes the groundwork for video retrieval and management as described in a research work carried out by Huo *et al.*, [3]. Therefore, in order to effectively search, index and manage these large quantities of video information, a robust video shot boundary detection algorithm is pre-required to help with content-based access to the video library [4]. Starting in the early 1990s, a number of institutions had already begun projects such as the *Query By Image Content* (*QBIC*), *Columbia VideoQ (object-oriented search engine)*, and the *Virage* that have connected to digital video libraries in order to handle video content intelligently [5]. In addition, the National Institute of Standards and Technology (NIST) organized the TREC Video Retrieval Evaluation (TRECVID) 'track' [6] devoted to research in automatic segmentation, indexing and content-based retrieval of digital video. For the TRECVID 'track', NIST provided a large test collection to organizations interested in video content analysis and retrieval research. A journal article by Lou *et al.*, [7] elaborates that approaches to video shot boundary have evolved and most of them have circulated around pixel-to-pixel difference, pixel-to-neighborhood difference, histogram difference, object tracking, motion estimation, *etc*. Most of these approaches are weakened by failure to deal with large object and camera motions. A review on shot boundary detection by Kumar *et al.*, [1] describes all possible boundaries that can be found in a video information as well as shot boundary detection methods and shot change detection methods that can be used based on the nature of the video or shot transition as shown in Table 1.

Table 1. A Summary of Algorithms and Methods used in Video Shot Boundary Detection

Shot Boundary Detection Methods	**Shot Change Detection Methods**
• Pixel Comparison • Transform-Based Difference e.g. Discrete Cosine Transformation (DCT) coefficient • Histogram-Based Difference with similarity measures/metrics such as Euclidean distance, Chi-Square, sum of squared difference, cross-correlation coefficient. • Statistical Difference • Motion Vector	• Static Thresholding • Adaptive Thresholding • Probabilistic Thresholding • Trained classifier

In this article, we hypothesize that we can improve the detection of shot boundaries on video sequences by leveraging information from several shot boundary detection algorithms. In particular, we build a classifier using features from sum of squared difference, cross-correlation coefficient as well as motion estimation. When building our classification models, we will follow the user-specific machine learning approach for improving touch accuracy on mobile devices as proposed by Weir *et al.*, [8]. In their work, Weir *et al.*, [8] noticed performance improvement for user specific models than models trained on all subjects. In the same vein, we will build and test our classification models using a video-specific machine learning approach to improve shot boundary detection. We use this approach because features of one video sequence cannot always match with features of another video. To validate our main hypothesis, we use 17 video sequences from NIST, who provided groudtruths in terms of shot boundaries that exist per video sequence. This dataset will be used to train and test our classifiers.

2. Related Work

To date, several studies have proposed various algorithms as solutions to the problem of shot boundary detection. These algorithms are based on various modalities of frames and special events on the temporal axis of the video. The detection of shot boundaries provides a basis for video abstraction, indexing, and high-level segmentation. Our in-depth look into existing shot boundary detection algorithms will be summarized into three categories which are; Pixel-Based Comparison, Luminance-Based algorithms that involve Histogram-Based difference and Motion - Based algorithms.

2.1. Pixel-Based Comparison

In a recent review on shot boundary detection, Kumar *et al.*, [1] describe pixel based shot boundary detection as the simplest but computationally intensive method for determining shot boundaries in a video. In summary, pixel based methods involve computation of the difference between corresponding pixels of two consecutive frames. For a difference greater than a specified threshold value, a shot boundary is assumed. In an earlier study, Patel *et al.*, [9] proposed a shot boundary detection algorithm that relies on pixel wise difference with adaptive thresholding on a compressed and uncompressed video. The algorithm is illustrated in Figure 1:

i. As illustrated in Figure 1, *SD* is the standard deviation of all frames and T_f is the threshold. If the *SD* of any frame is found to be below the threshold value, then that frame will be declared a monochrome frame and therefore processing proceeds for fade boundary detection.

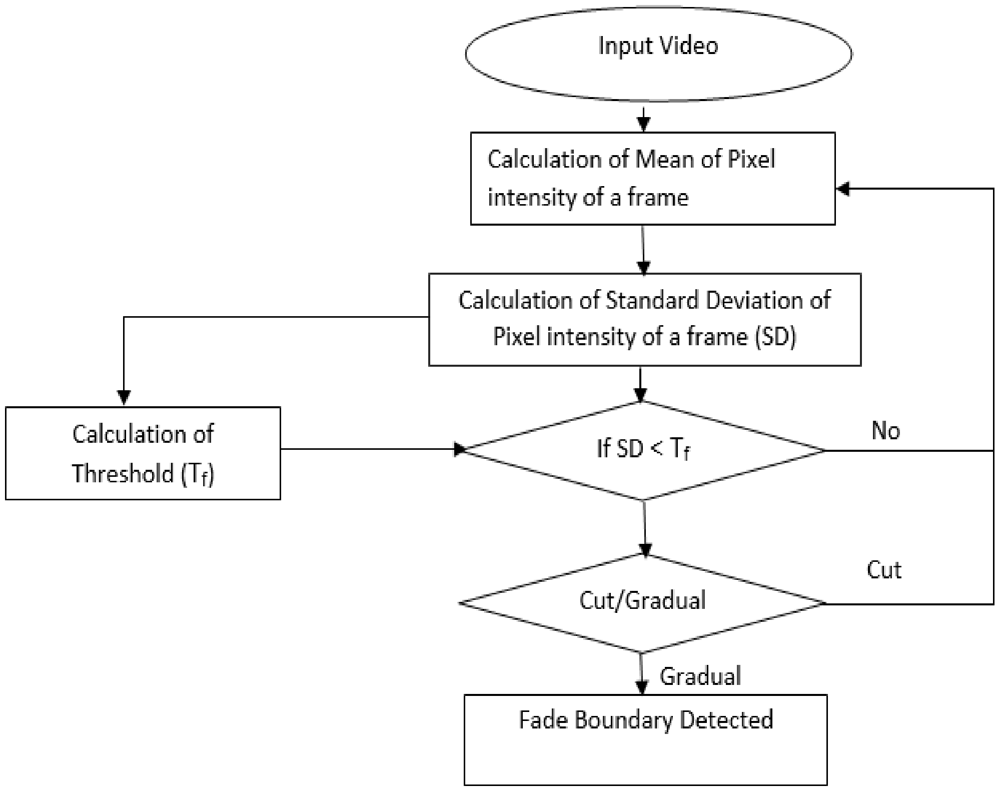

Input Video
Calculation of Mean of Pixel intensity of a frame
Calculation of Standard Deviation of Pixel intensity of a frame (SD)
Calculation of Threshold (Tf)
If SD < Tf
No
Cut/Gradual
Cut
Gradual
Fade Boundary Detected

Figure 1. An Illustration of Shot Boundary Detection Approach by [9]

One study by Han *et al.*, [10] used a pixel-based comparison approach that accentuated edit constancy (shapes of cut or fade/dissolve in low pass filtered frame differences signal) effects while suppressing motion effects using low pass filtering in histogram space in order to alleviate the problem of large camera and object motions in detecting video shot boundaries. The approach also presents a shot representation method for effectively selecting key frames based on the contents of the video for indexing. Han *et al.*, [10] concludes that for more meaningful key frame selection, more information of the frame is needed, such as motion and texture.

In contrast to Patel *et al.*, [9] and Han *et al.*, [10], Saric *et al.*, [11] deployed the twin-comparison algorithm and dominant color region on the detection of shot boundaries in soccer video. Their approach concentrated on soccer video as it is considered one of the challenging video domains for robust shot boundary detection due to a lot of camera and object motions on the same background. As opposed to the standard twin-comparison algorithm, they introduced a combination of twin-comparison and dominant colored pixel ratio to improve the detection of shot boundaries where large object and camera motions are involved. The feature that they introduced was based on the absolute difference between two frames in their ratios of dominant (grass) colored-pixels to total number of pixels. A comparison of these two algorithms done on a dataset of approximately 30 minutes of soccer video showed that a standard twin-comparison algorithm had lower a precision rate 48% due to its sensitivity to object and camera motions while the proposed algorithm had a higher precision rate of 65%. A recall rate of 42% for their proposed approach was lower due to some of the criterions they used that led to the elimination of some true gradual transitions.

2.2. Luminance-Based Algorithms

Luminance is of paramount importance in shot boundary detection and it is characterized by changes in video sequence colors, which include flashlights, contrast, background colors, *etc*. It has been reported in the literature that most algorithms that respond to luminance variations use the color histograms either in the form of primary colors of Red (R), Green (G) and Blue (B) or in shot RGB as well as other color spaces [12]. Some of the color spaces used in shot boundary detection include HSV and YUV.

Mas *et al.*, [12] provides an in-depth analysis of a video shot boundary detection algorithm based on the color histograms of frames belonging to a video sequence which involves conversion from RGB space to HSV space. Histogram difference was computed using equation (1):

$$HistDiff[i] = \sum_{j=1}^{M} |h_i(j) - h_{i-1}(j)| \qquad (1)$$

Where $h_i(j)$ and $h_{i-1}(j)$ are the histograms of two consecutive frames i and $i+1$ and $HistDiff[i]$ is the total histogram difference between two consecutive frames. In detecting shot cuts using the color histogram signal difference, the signal was first convolved with a rectangular window so that small variations due to the histogram difference computations are eliminated. Therefore, obtaining a rectangular shaped signal where the middle of the rectangle is considered the frame of an abrupt transition as shown in Figure 2.

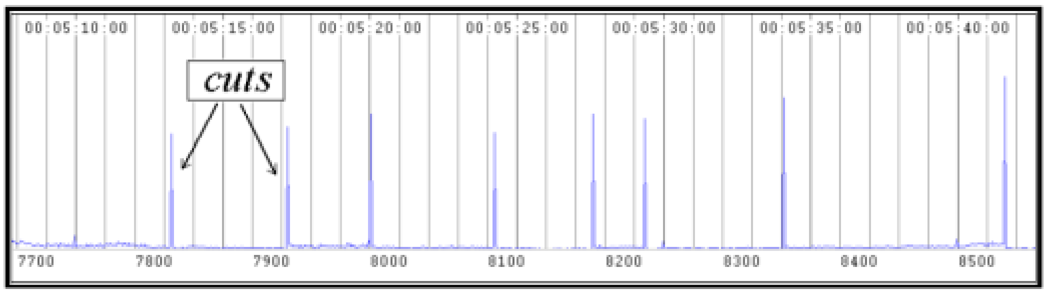


Figure 2. Colour Histogram Differences Signal of a Video Sequence with Cut Boundaries [12]

Furthermore, the convolving of signals with the rectangular window also modifies gradual transitions to appear as triangular shaped signals as shown in Figure 3. Hence the detection of gradual transitions is achieved.

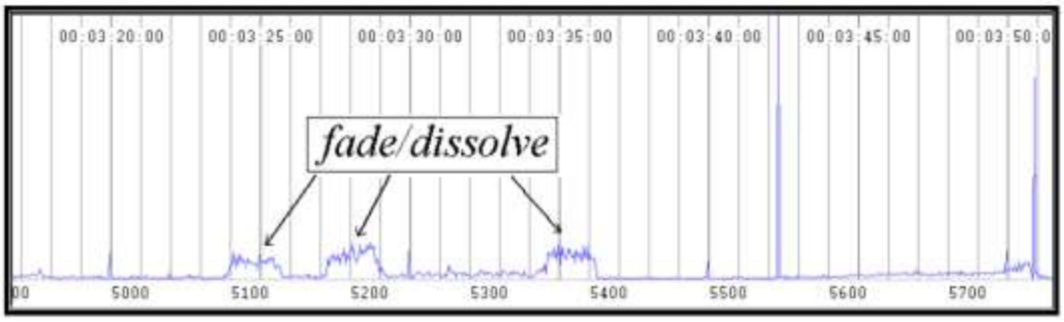


Figure 3. Color Histogram Differences Signal of Video Sequence with Gradual Transitions [12]

Another luminance based technique by Kathiriya *et al.*, [13] used χ2 (Chi-Square) based shot boundary detection and key frame extraction for a video using the histogram difference of two frames. In their approach, they computed the whole image intensity for three basic colors by first converting frames into three different color of R (Red), G (Green), and B (Blue). Furthermore, they extracted the histogram difference using equation (2):

$$D_f(k, k+1) = \sum_{i=1}^{3} \frac{[H(k,i) - H(k+1,i)]^2}{H(k,i)} \tag{2}$$

Where $D_f(k, k+1)$ – is the total difference of the combined individual histograms of two consecutive frames k and $k+1$. $H(k,i)$ and $H(k+1,i)$ stands for the histograms of Red, Green and Blue of consecutive frames. There was a mean difference calculated through another formula using the total difference and the total number of frames as well as the standard deviation. This was used for the calculation of the thresholds for both the abrupt and gradual transitions. Abrupt and gradual transitions were detected in cases where the mean difference was found to be greater than the abrupt threshold and the gradual threshold respectively. This type of method is supported mostly on uncompressed video types. This method was also used by Rathod *et al.*, [2] who developed and tested the algorithm for shot boundary detection and key frame extraction (KFE) using color difference. The Chi-Square similarity metric was used to compute the histogram difference between corresponding blocks of consecutive frames in a video information. Figure 4 summarizes the steps that are involved in the detection of shot boundaries for key frame selection.

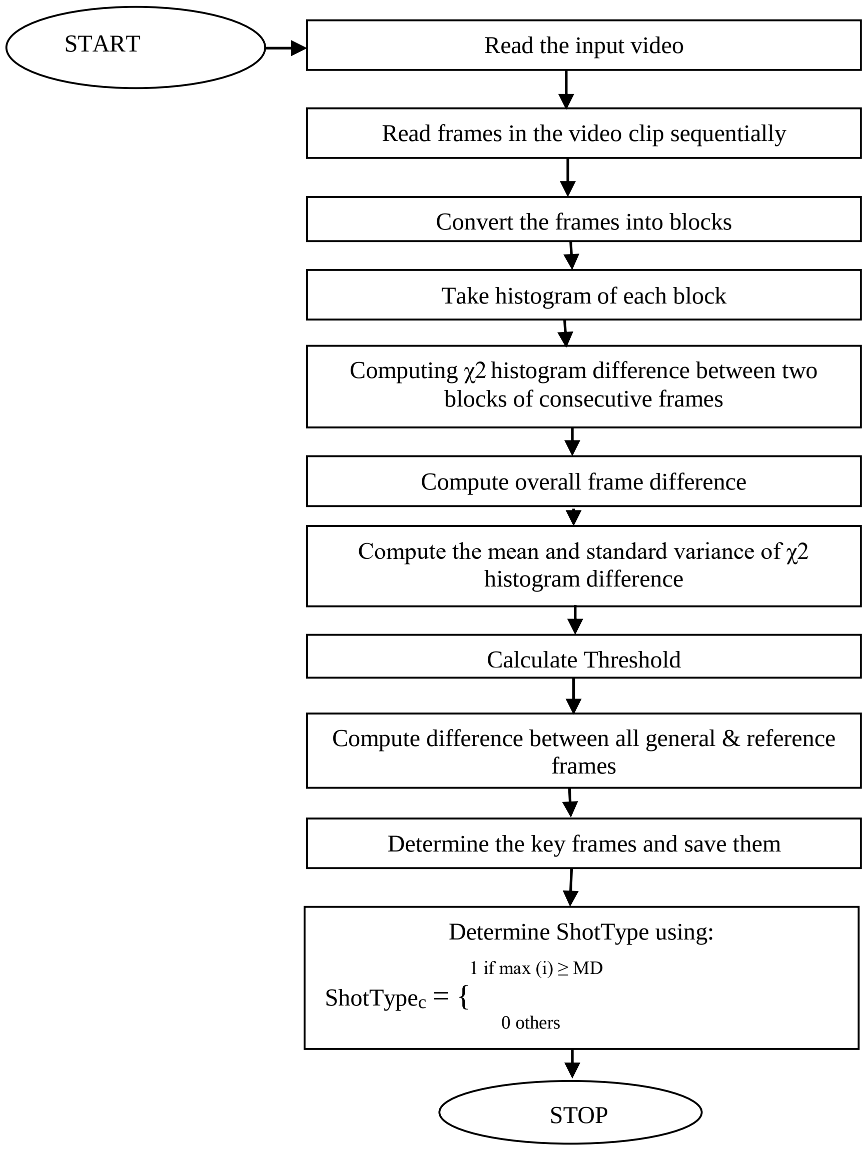

START
Read the input video
Read frames in the video clip sequentially
Convert the frames into blocks
Take histogram of each block
Computing χ2 histogram difference between two blocks of consecutive frames
Compute overall frame difference
Compute the mean and standard variance of χ2 histogram difference
Calculate Threshold
Compute difference between all general & reference frames
Determine the key frames and save them
Determine ShotType using:
ShotType_C = { 1 if max (i) ≥ MD; 0 others
STOP

Figure 4. Design Flow of the Algorithm for SBD and KFE [2]

The proposed solution was tested with videos of different types such as movies, sports and cartoons. Their algorithm accepted input video in the form of ".avi" with the video size ranging from 3MB to 4.5MB. Experimental dataset used by Rathod *et al.*, [2] was from the Open Video Project. In their empirical evaluation, they reported that their algorithm could detect all shot boundaries with an efficiency of 95% to 98%.

Sum of Squared Difference(SSD) is described as a popular similarity metric with less computational costs in determining the best match between two signals or images. This type of metric is commonly used in abrupt transitions detection and it is not computationally intensive. SSD is part of the Analysis of Variance (ANOVA) package of simple formulas used to solve complex problems [14]. Fouda [15] points out that pattern matching is an important technique in image processing, which is widely used in signal processing and machine vision applications. Therefore, computing the difference between two images in a scene or video is of vital importance, which requires fast and accurate metrics and algorithms. In their journal article, Chasanis *et al.*, [16] describe a method for detecting abrupt or cut transitions based on the differences between adjacent frames. They argued that wherever a shot cut exists, there are very high differences in terms of pixel-wise difference or even color histogram difference. Hence, the sum of squared algorithm can be used to detect shot cuts as applied in image processing. Another SSD approach by Jacobs *et al.*, [17] combined color, edge and motion features to detect both shot cuts and gradual transitions. Their approach uses RGB histogram values calculated through sum of squared difference within a five frames width window, edge change ratio between consecutive frames and also frames at a distance of 10 [17].

Cross-correlation coefficient is another luminance based algorithm which also relies on the color histogram. According to Lyon [18], cross-correlation or cross-covariance can be done in any number of dimensions and can de described simply as template matching between two signals. As described by Rani *et al.*, [19], through signal processing concepts, cross correlation is a measure of similarity between two waveforms as a form of time-lag applied to one of them. Moreover, correlation determines the degree of similarity between two signals. It also consists of the dot product used in quantifying the degree of similarity or interdependence between two signals [20]. Some of the uses of cross correlation include; X-ray diffraction data analysis, security systems in terms of pattern recognition, water traffic monitoring, and measuring pulse broadening and distortion. The method of cross correlation underlines implementations of the Fourier transform, where signals of varying frequency and phase are correlated with the input signal. Hence the degree of correlation in terms of frequency and phase represents the frequency and phase spectrums of the input signal. Therefore, this metric is widely adopted in image processing and video analysis. Cross-correlation coefficient algorithms return values between 1 and -1. A value of "1" indicates a complete match between two signals in signal processing and hence the frames or images under consideration are from the same shot in a video sequence. A value of zero represents a complete mismatch between two signals or consecutive video frames and therefore most probably frames from different video shots can be detected through that. A value of "-1" implies the two objects are identical but mirrored through a 180 degrees phase angle. In their work, Chen *et al.*, [21] proposed a robust hash scheme based algorithm, which used a 2D-DCT temporal maximum occurrence where a video sequences is firstly divided into shots. Therefore, deriving video hash in a unit of shot. In their approach, Chen *et al.*, [21] used a distance measure computed from the cross correlation of feature frame hashes in order to get the differences between consecutive frames for shot boundary detection. Their experimental results suggest that this approach is robust in video identification, authentication and verification applications [21].

2.3. Motion-Based Algorithms

In 2013, Verma *et al.*, [22] published a research paper in which they analyzed various motion estimation algorithms on shot boundary detections based on block matching technique. These block matching algorithms on motion estimation are as follows: Exhaustive Search (ES), Three Step Search (TSS), New Three Step Search (NTSS), Simple and Efficient Search (SES), Four Step Search (4SS), Diamond Search (DS), and Adaptive Rood Pattern Search (ARPS). In a nutshell, all these block matching algorithms stated above, divides the current frame into macro blocks that are then compared with the reference block and its neighbors in the previous frame hence creating a motion vector. This motion vector defines the motion in a macro block for one frame to another. In their work, they noted that the motion calculated for all the macro blocks constitutes the motion estimation of the current frame. In matching the macro blocks with each other, they used the Mean Absolute Difference (MAD) given by equation (3) and the Mean Squared Error (MSE) given by equation (4). As depicted by equation (5), the Peak-Signal-to-Noise-Ratio (PNSR) was used to characterize the motion compensated image that was created by using motion vectors and macro blocks from the reference block.

$$Mean\ Absolute\ Difference\ (MAD) = \frac{1}{N^2} \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |C_{ij} - R_{ij}| \quad (3)$$

$$Mean\,Square\,Error\,(MSE) = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(C_{ij} - R_{ij})^2 \quad (4)$$

Where N is the side of the macro block, C_{ij} and R_{ij} are in current block and reference block respectively, being compared.

$$PNSR = 10\log_{10}\frac{[(Peak\ to\ peak\ value\ of\ original\ data)^2]}{MSE} \quad (5)$$

Even though these methods work well in detecting shot boundaries, their computational costs are very high. However, results revealed that the ARPS algorithm provided better results taking into account its PNSR performance. This was due to the fact that it involved less steps. In the same vein, Volkmer [23] proposed a method for detecting gradual transition using an average frame similarity. Due to the fact that gradual changes involve several frames, this approach catered for evaluation of the average inter-frame distance in a set of frames, rather than examining only individual frames while also using adaptive thresholding to increase effectiveness across different types of video content. Furthermore, Volkmer [23] argued that frames in a gradual change should not be examined individually but rather an average distance between sets of *pre-frames* and *post-frames* to the current frame should be computed. This procedure resulted in two values, which were referred to as *PrePostRatio* – used to detect gradual changes. As described further by these authors, the history of the PrePostRatio values are maintained while calculating a moving average and standard deviation in order to come up with an adaptive threshold. In their findings, they reported that the approach performed better on large video collections with recall and precision of *83%* and *75%* respectively. However, the approach detects the start and end of gradual transitions accurately. Most motion estimation algorithms use block based matching strategy and average the pixel difference over blocks in order to make a block-based motion detection decision on video sequences. A summary of the motion representations described by Wang [24] is illustrated in Figure 5.

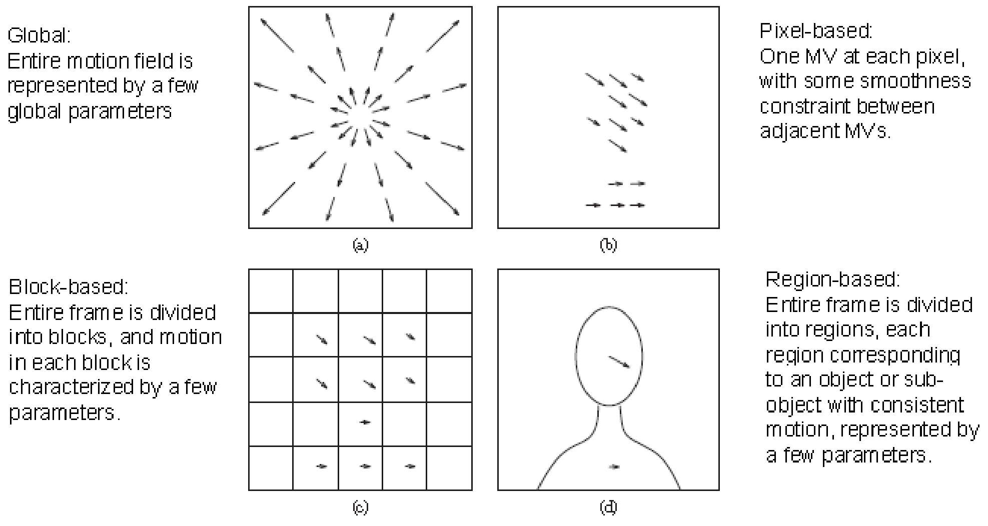

Global:
Entire motion field is represented by a few global parameters
Pixel-based:
One MV at each pixel, with some smoothness constraint between adjacent MVs.
(a)
(b)
Block-based:
Entire frame is divided into blocks, and motion in each block is characterized by a few parameters.
Region-based:
Entire frame is divided into regions, each region corresponding to an object or sub-object with consistent motion, represented by a few parameters.
(c)
(d)

Figure 5: Motion Representations [24]

3. Methodology

We begin Section 3.1 by outlining our research questions, followed by Section 3.2 where we describe the dataset for our baseline systems and our proposed solution. In Section 3.3 we describe how we analyze the baseline systems and in Section 3.4 we describe the proposed solution setup. Lastly in Section 3.5 we describe how we build our classifiers.

3.1. Research Questions

RQ1: Which shot boundary detection algorithm can better detect both shot cuts and gradual transitions using global thresholding. In this article, we will be comparing the cross-correlation coefficient, the sum of squared difference and motion estimation?

RQ2: Can we improve the detection of both shot cuts and gradual transitions by integrating either motion estimation and cross-correlation coefficient, motion estimation and sum of squared difference or motion estimation, sum of squared difference and cross-correlation coefficient by using video specific features and machine learning?

3.2. Description of the dataset for our baseline systems and our proposed solution

As show in Table 2, 17 video sequences were used as training and testing data in carrying out experiments using baseline systems and our proposed solution. In Table 2, we describe the characteristics of all the video sequences in the form of the total number of frames, number of gradual transitions, and number of shot cuts per video sequence.

Table 2. Videos used for Training, Testing and their Characteristics [25]

Video Sequence	No. of frames	No. of cuts	No. of gradual transitions	Total number of frames involved in gradual transitions	Sum of all shots
BG_2408	35892	103	18	310	121
BG_9401	50049	89	3	63	92
BG_11362	16416	104	4	141	108
BG_14213	83115	107	60	2777	167
BG_34901	34389	225	15	276	240
BG_35050	36999	100	2	51	102
BG_35187	29025	135	23	1455	158
BG_36028	44991	87	0	0	87
BG_36182	29610	96	13	224	109
BG_36506	15210	77	6	305	83
BG_36537	50004	259	30	963	289
BG_36628	56564	196	6	162	202
BG_37359	28908	165	5	93	170
BG_37417	23004	84	4	37	88
BG_37822	21960	120	9	220	129
BG_37879	29019	95	4	154	99
BG_38150	52650	215	4	98	219
Totals	637805	2257	206	7329	2463

3.3. Baseline Systems description and setup

In order to answer research question one (RQ1), our baseline systems were based on motion estimation, cross-correlation coefficient and sum of squared difference using global thresholding as the shot change detection method. Setting a global threshold value is one of the challenging tasks in accurate video shot boundary detection as shown in the literature. The challenges include the following:

- It is manual based as it involves analysing all the features where shot cuts and gradual transitions exists while also taking into consideration the features of frames not involved in shot boundary transitions.
- A high threshold value for the detection of shot boundaries can lead to missing of shot cuts for either shot cuts or gradual transitions. A lower threshold value can lead to false shot boundary detections. Setting the threshold either high or low can lead to shot cuts either being detected as gradual transitions or frames not involved in gradual transitions begin detected as frames involved in gradual transitions.

The following were considered in setting up the global threshold for all the three algorithms (motion estimation, the sum of squared difference and the cross-correlation coefficient) for the detection of shot boundaries of all the videos described in Table 2:

- The first video *(BG_2408)* as described in Table 2 was used to provide the feature parameters based on motion estimation, sum of squared difference, and cross-correlation coefficient.
- In order to set the global threshold for detecting shot cuts, and gradual transitions the minimum, maximum and average values for motion estimation, cross-correlation coefficient and sum of squared difference for video *BG_2408* were analysed and compared to the general view of all the values of features involved in either shot cuts or gradual transitions.
- For shot cut detections, there can only be one threshold value (T_{Global}). For algorithms based on motion estimation and sum of squared difference, if a value is greater than T_{Global}, then a shot cut has been detected. As for cross-correlation coefficient the opposite happens. A value less than T_{Global}, implies that a shot cut has been detected.
- The detection of gradual transitions based on global thresholding is more challenging as it involves setting two threshold values. This is so, because gradual transitions involve several frames of videos where they exist. Unlike with shot cuts, only two frames are involved. As compared to shot cuts, the differences between frames involved in gradual transitions can either be high or low, hence making it difficult to distinguish them between shot cuts or frames not involved in gradual transitions. Therefore, two thresholds are required to try to cater for a range of features involved in a gradual transition. T_{Low} as the lower threshold value boundary and T_{High} as the upper threshold boundary for gradual transitions detection. Table 3 describes the baseline systems used to carry out experiments in Section 4. These baseline systems use global thresholding as a shot change detection measure.

Table 3. Baseline Systems used to Carry Out Experiments in Section 4

Algorithm	**Description**
F	Sum of Squared Difference (SSD) and Global Thresholding
G	Cross-Correlation Coefficient and Global Thresholding
H	Motion Estimation and Global Thresholding

Table 4 shows the threshold values set based on the features of *BG_2408* video sequence as described in Table 2 for the baseline systems described in Table 3.

Table 4. Threshold Values for the Three Different Algorithms used in the Detection of Shot Boundary Transitions Set using Video 1 - BG_2408 from the Test Collection for TRECVID 2007 Activity

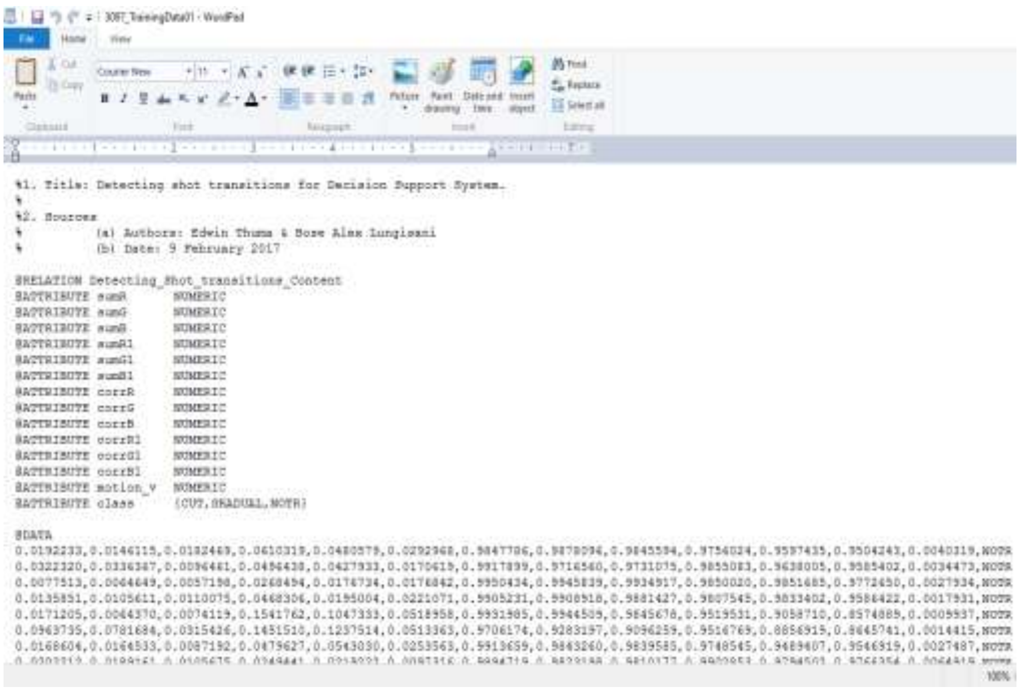
		Sum of Squared Difference (F)			**Cross-Correlation Coefficient (G)**			**Motion Estimation (H)**
Transition Type	**Threshold Type**	Red	Green	Blue	Red	Green	Blue	16X16 pixels of blocks
CUTS	T_{Global}	0.5000000	0.5000000	0.5000000	0.7000000	0.7000000	0.7000000	0.9000000
GRADUAL	T_{Low}	0.0800000	0.0800000	0.0800000	0.1000000	0.1000000	0.1000000	1.5000000
	T_{High}	0.1303597	0.1506553	0.1280129	0.9000000	0.9000000	0.9000000	4.0000000

The threshold values in Table 4 produced better results. The features were extracted based on the differences of the RGB histograms for the sum of squared difference and the cross-correlation coefficient while for motion estimation the features were extracted based on the differences of the frame pixels between blocks of consecutive frames. Therefore, each colour component in the RGB histogram have its own threshold value based on features for each colour component in a video sequence. However, in most cases, these threshold values may not be different based on the luminance distribution in the video sequence as illustrated in Table 4. The same threshold values are used across the 17 videos used as test collection as a way of applying global thresholding with all the three baseline algorithms.

3.4. Proposed Solution Setup

In order to answer research question two (RQ2), the evaluation was done based on the shot boundary task for TRECVID, a TREC-style video retreival evaluation benchmarking platform in video analysis. TRECVID 2007 test collections for shot boundary task are used. These are videos from sports, movies, *etc*. About *4.172 Gigabytes* of the TRECVID 2007 lasting about 15 hours comprising of 17 videos are used in developing the training and testing datasets. The characteristis of these videos are described in Table 2. Groud-truths used for evaluation purposes were obtained from TRECVID 2007 task for shot boundary detection. These were established manually by navigating through the 17 videos, frame by frame in order to locate where shot boundaries in terms of shot cuts and gradual transitions exist for training data purposes as well as establishing the groudtruths.

In designing the training data, we generated features from the sum of squared difference and the cross-correlation coefficient on the RGB histograms for consecutive frames (*frame(i)* and *frame(i+1))* as well as features for frames (*frame(i)* and *frame(i+2)*) in order to improve the detection of gradual transitions. For motion estimation, features for the difference (displacement) of the corresponding pixels based on blocks of *7x7* and *16x16* pixels between consecutive frames were extracted and used as part of the training data. The reason behind choosing such small blocks in size was to increase the sensitivity to motion within frames of video. Large block sizes reduces the sensitivity to motion and hence vital information in terms of motion is lost leading to error prone detections. The sizes of the blocks also depend on the size of the searching window used in motion estimation, which further depends on the resolution of the video frames. Lastly, we created training and testing instances for each video sequence as illustrated in Figure 6 in the form of an *.arff* file format. An *.arff* file is a format compatible with Weka 3.8 Data Mining Software used in this article for evaluation and testing purposes.



```
%1. Title: Detecting shot transitions for Decision Support System.
%
%2. Sources
%         (a) Authors: Edwin Thuma & Bose Alex Lungisani
%         (b) Date: 9 February 2017

@RELATION Detecting_Shot_transitions_Content
@ATTRIBUTE sumR       NUMERIC
@ATTRIBUTE sumG       NUMERIC
@ATTRIBUTE sumB       NUMERIC
@ATTRIBUTE sumR1      NUMERIC
@ATTRIBUTE sumG1      NUMERIC
@ATTRIBUTE sumB1      NUMERIC
@ATTRIBUTE corrR      NUMERIC
@ATTRIBUTE corrG      NUMERIC
@ATTRIBUTE corrB      NUMERIC
@ATTRIBUTE corrR1     NUMERIC
@ATTRIBUTE corrG1     NUMERIC
@ATTRIBUTE corrB1     NUMERIC
@ATTRIBUTE motion_v   NUMERIC
@ATTRIBUTE class      {CUT,GRADUAL,NOTR}

@DATA
0.0192233,0.0146115,0.0182469,0.0610319,0.0480579,0.0292968,0.9847706,0.9878094,0.9845594,0.9754024,0.9597435,0.9504243,0.0040319,NOTR
0.0322320,0.0336387,0.0096461,0.0494438,0.0427933,0.0170619,0.9917899,0.9716560,0.9731075,0.9855083,0.9638005,0.9585402,0.0034473,NOTR
0.0077513,0.0064649,0.0057198,0.0268494,0.0176734,0.0176842,0.9950434,0.9945839,0.9934917,0.9850020,0.9851685,0.9772650,0.0027934,NOTR
0.0135851,0.0105611,0.0110075,0.0468306,0.0195004,0.0221071,0.9905231,0.9908918,0.9881427,0.9807545,0.9833402,0.9586422,0.0017931,NOTR
0.0171205,0.0064370,0.0074119,0.1541762,0.1047333,0.0518958,0.9931985,0.9944509,0.9845678,0.9519531,0.9058710,0.8574885,0.0009937,NOTR
0.0963735,0.0781684,0.0315426,0.1451510,0.1237514,0.0513363,0.9706174,0.9283197,0.9096259,0.9516769,0.8856915,0.8645741,0.0014415,NOTR
0.0168604,0.0164533,0.0087192,0.0479627,0.0543030,0.0253563,0.9913659,0.9843260,0.9839585,0.9748545,0.9489407,0.9546919,0.0027487,NOTR
```

Figure 6. A Snapshot for Training and Testing Data for BG_3097_xvid.avi Video Sequence

A combination of algorithms for our proposed solution are described in Table 5. These combinations were built from the sum of squared difference, the cross-correlation coefficient and motion estimation as well as different sizes and features of frames of video sequences. These included blocks of frames, whole frames, as well as RGB histograms.

Table 5. Combinations of Proposed Algorithms used to Carry Out the Experiments in Section 4

Algorithm	Description
A	Sum of Squared Difference (SSD) and Motion Estimation
B	Cross-Correlation Coefficient and Motion Estimation
C	SSD, Cross-Correlation Coefficient and Motion Estimation
D	Block-Based Cross-Correlation Coefficient and Motion Estimation
E	SSD, Cross-Correlation Coefficient, Block-Based Cross-Correlation Coefficient and Motion Estimation

3.4. Training and Classification of Gradual transitions, Cuts and Non-transitions in a Video Sequence

In our empirical evaluation, we used random forest classifier in Weka 3.8 to build and test our classification models. According to the literature, random forest classifier is not sensitive to irrelevant features and it handles real and discrete data very well. The structure of the training data file has the attributes, class and the data itself as shown in Figure 6. The attributes represent the values of features that are contained on the data part of the training data structure. Class represents the classification of the data values into *CUT* (abrupt transition), *GRADUAL* (special edit effects such as fades and dissolves), *NOTR* for no transition between frames of video sequences. It has to be noted as explained earlier in Section 3.3 that video features based on the sum of squared differences and the cross-correlation coefficient were extracted for each colour channel (R,G,B) for histograms of frames compared and hence the attributes *sumR*, *sumG*, *sumB*, *sumR1*, *sumG1*, *sumB1*, *corrR, corrG, corrB, corrR1, corrG1, and corrB1* in Figure 6. As for motion estimation, features were obtained based on the comparison of blocks of sizes *7x7* and 16*x16 pixels* in order to come up with two further attributes for *motion7* and *motion16*. All these features can be seen in a combined algorithm that utilizes all the three algorithms' features. Otherwise, some features for some algorithms will be seen on different training data files depending on which algorithm is being used at that particular moment.

4. Experimental Setup

We built our classifiers using features generated from the sum of squared difference, the cross-correlation coefficient similarity metrics and motion estimation using random forest classifier in Weka 3.8 Data Mining Software for shot change detection. We normalized the input data to a range of [-1,1] in order to cover up for all our input data as some features had negative values. However, our output range was factored to 1.0 to get values in the range of [0,1]. For training and testing our classifiers, we used the 10-fold cross validation technique. Our algorithms in Table 4 were developed using *python scripting language* and *Open Source Computer Vision (OpenCV) library* under Linux platform. In order to decode a compressed input file in the form of a video clip *(.avi format)* into an uncompressed video *(images of .png file extension)* as input files for the SSD and the cross-correlation coefficient algorithms, a video decoder was developed and used. The decoder was developed using *python* and *OpenCV* on a *Linux* platform. Another package of scripts was developed under the same platforms to extract the histograms of images in order to quantify these images, which are frames of the input video in this case. Therefore, there were three main algorithms developed that use a similarity metric in the form of the sum of squared difference, the cross-correlation coefficient and motion estimation. For each algorithm, there was synchronization to a decoder, similarity metric and the RGB histogram extractor. Furthermore, extra four algorithms were developed based on the combination of the three main algorithms explained earlier in this paragraph. In total, there were five more algorithms developed and compared to our baseline systems on shot boundary detection (Table 5).

The features for the three different baseline algorithms (Table3) were labelled as shot cut *(*CUT), gradual transitions *(GRADUAL)* or non-transitions *(NOTR)*. In addition, we generated the training and testing datasets for the different algorithms summarized in Table 5 using the TRECVID 2007 'track' test collection for shot boundary detection. Furthermore, Weka 3.8 was used to build and test shot change detection classification models using these different datasets.

5. Experimental Results and Evaluation

Like other information retrieval tasks, the performance was evaluated by precison and recall as shown by equations (7) and (8), representing a fraction of releveant documents retrieved and a fraction of retreived documents that are relevant respectively. Weka 3.8 was used to calculate these evaluation measures. The F1-measure was used to rank the different algorithms. The F1-measure is a harmonic average of precison and recall and it is a combination of recall and precision with equal weight as shown by equation (6):

$$F1 - meaure(recall, precision) = \frac{2 \times recall \times precision}{recall \times precision} \tag{6}$$

$$\boldsymbol{Precision} = \frac{Total\ number\ of\ shot\ boundaries\ correctly\ identified\ by\ the\ algorithm}{Total\ number\ of\ shot\ boundaries\ identified\ by\ the\ algorithm} \tag{7}$$

$$\boldsymbol{Recall} = \frac{The\ number\ of\ shot\ boundaries\ correctly\ identified\ by\ the\ algorithm}{Total\ number\ of\ shot\ boundaries} \tag{8}$$

Symbolically:

$$Recall = \frac{N_C}{N_C+N_M} \qquad\qquad Precision = \frac{N_C}{N_C+N_F}$$

Where N_C, N_M, N_F are the number of correctly detected, the number of missed shot transitions, and the number of falsely detected shot transitions respectively. In a nutshell, recall is the percentage of transitions detected while precision is the percentage of detected transitions that are actually correct.

Table 6. Baseline Systems Summary of Averages for Evaluation Results of Algorithms [F-H] using Global Thresholding based on Precision, Recall, F1-Measure for the 17 Videos used as Test Collection

Algorithm	Overall			Cut			Gradual transitions		
	F1(Rank)	Recall	Precision	F1(Rank)	Recall	Precision	F1(Rank)	Recall	Precision
F	0.318	0.323	0.334	0.361	0.309	0.436	0.274	0.337	0.232
G	0.513	0.518	0.515	0.605	0.590	0.629	0.421	0.445	0.401
H	0.292	0.641	0.190	0.417	0.844	0.277	0.167	0.438	0.103

As we set out to investigate research question one (RQ1) *(Which shot boundary detection algorithm can better detect both shot cuts and gradual transitions using global thresholding)*, it can be seen from Table 6 that all baseline algorithms can better detect shot cuts with the cross-correlation coefficient (G) as the best among the three algorithms with an F1 score of 0.605. This is so because setting up a global threshold on the cross-correlation coefficient is much easier than for the sum of squared difference and motion estimation as the output values for the cross-correlation coefficient range are between -1 and 1 inclusive. However, the same cannot be said about motion estimation and the sum of squared difference as their output values have no defined range. Therefore, this means that the cross-correlation coefficient eliminates both false-positives and false-negatives which may occur due to luminance variances such as flashlights, contrast, etc. The weakness of motion estimation and the sum of squared difference can further be seen from Table 6 with algorithms F and H on the detection of gradual transitions due to the challenge in setting up the threshold values. Algorithm G for the cross-correlation coefficient detected gradual transitions better.

Table 7. Random Forest Classifier Summary of Averages for Evaluation Results for Algorithms [A – E] based on Precision, Recall and F1-measure for the 17 Videos Used as Test Collection

Algorithm	Overall			Cut			Gradual transitions		
	F1(Rank)	Recall	Precision	F1(Rank)	Recall	Precision	F1(Rank)	Recall	Precision
A	0.634	0.631	0.847	0.907	0.915	0.906	0.415	0.314	0.789
B	0.625	0.619	0.825	0.909	0.914	0.904	0.369	0.284	0.754
C	0.691	0.637	0.869	0.916	0.916	0.914	0.434	0.318	0.812
D	0.677	0.637	0.855	0.868	0.892	0.881	0.458	0.348	0.826
E	0.752	0.703	0.892	0.920	0.921	0.920	0.560	0.461	0.855

In order to answer research question two (RQ2) *(Can we improve the detection of both shot cuts and gradual transitions by integrating either motion estimation and the cross-correlation coefficient, motion estimation and the sum of squared difference or motion estimation, the sum of squared difference and the cross-correlation coefficient by using video specific features and machine learning?)*, we combined motion based algorithms and luminance based algorithms and built our classifiers from a combination of features from motion estimation, the sum of squared difference and the cross-correlation coefficient. As it can be seen from Table 7, a combination of features generated using the sum of squared difference, the cross correlation coefficient, the block-based cross correlation coefficient and motion estimation (algorithm E) detected both shot cuts and gradual transitions better than all other combination of features (algorithms A, B, C and D) with an average F1 score of 0.752 on all the 17 video sequences used in our test collection. Therefore, this shows that a combination of more features, which are video specific improves the detection of both shot cuts and gradual transitions. It can further be seen that the use of features from blocks of frames of video sequences improves the detection of shot boundaries. Features for blocks of frames were used in algorithm E. However, the detection of gradual transitions is still a challenge as our best algorithm E in Table 7 has an F1 score of 0.56 in gradual transition detection as compared to an F1 score of 0.920 in shot cut detection. Even though gradual transitions are difficult to detect, high precision values of above 0.75 for all algorithms show that there are less falsely detected gradual transitions but more missed gradual transitions. Overall, our results show that the use of a video specific trained classifiers improves both shot cuts and gradual transitions detections. This is supported by the marked improvement in F1 scores of our proposed solutions depicted in Table 7 as compared to the background algorithms in Table 6. We also observed that a combination of motion based, and luminance based algorithms in the form of the cross-correlation coefficient and the sum of squared difference based on the RGB histograms improves the detection of both shot cuts and gradual transitions. Motion estimation deals with turbulences from challenges of large object and camera motions such as camera tilting, fast zooming as well as fast movement of objects. Features from the cross-correlation coefficient and the sum of squared difference based on the RGB histograms deal with luminance variations from flashlights, contrast, *etc*.

6. Conclusions

This study set out to investigate whether we can improve the detection of shot boundaries on video sequences by leveraging information from several shot boundary detection algorithms using a supervised machine learning approach. To achieve this, we built several classification models using a combination features (Algorithms A, B, C, D and E) generated by the sum of squared difference, the cross-correlation coefficient and motion estimation algorithms. We built and tested our classification models using video specific features. Overall, our results show that the use of video specific trained classifiers improves the detection of both shot cuts and gradual transitions. This is supported by the marked improvement in the F1 scores of our proposed solutions depicted in Table 7 as compared to our baseline algorithms in Table 6. It also emerged from this study that the detection of shot cuts and gradual transitions can be improved by using features generated by several shot boundary detection algorithms. In particular, algorithm E (F1 score of 0.752) used a combination of four different shot boundary detection algorithms to generate features for training and testing our classification models while algorithms A, B and D (F1 score of between 0.625 to 0.677) used a combination of two different shot boundary detection algorithms to generate features for training and testing our classification models. Algorithm C (F1 score of 0.691) used a combination of three different shot boundary detection algorithms to generate features for training and testing our classification models.

References

[1] H. B. Basanth Kumar, "A Review on Shot Boundary Detection", Oriental Journal of Computer Science & Technology, **(2014)**, pp. 39-44.
[2] G. I. Rathod and D. A. Nikam, "An Algorithm for Shot Boundary Detection and Key Frame Extraction Using Histogram Difference", International Journal of Emerging Technology and Advanced Engineering, **(2013)**, pp. 155-163.
[3] Y. Huo, P. Zhang and Y Wang, "Adaptive Threshold Video Shot Boundary Detection Algorithm Based on Progressive Bisection Strategy", Journal of Information & Computational Science, **(2014)**, pp. 391-403.
[4] C. Zhang and W. Wang, "A Robust and Efficient Shot Boundary Detection Approach Based on Fisher Criterion", Nara, **(2012)**.

[5] A. Amiri and M. Fathy, "Video Shot Boundary Detection Using Generalized Eigenvalue Decomposition and Gaussian Transition Detection", Computing and Informatics, vol. 30, **(2011)**, pp. 595-619.

[6] M. Luo, D. DeMenthon and D. Doermann, "Shot Boundary Detection usinf Pixel-to-Neighbor Image Differences in Video", n.d.

[7] Y. Luo, T.-D. Wu and J.-N. Hwang, "Object-based analysis and interpretation of human motion in sports video sequences by dynamic bayesian networks", Computer Vision and Image Understanding, vol. 92, **(2003)**, pp. 196-216.

[8] D. Weir, "A User-Speific Machine Learning Approach for Improving Touch Accuracy on Mobile Devices", University of Glasgow, Cambridge, **(2012)**.

[9] U. Patel, P. Shah and P. Panchal, "Shot Detection Using Pixel Wise Difference with Adaptive Thresholding and Color Histogram Method in Compressed and Uncompressed Video", International Journal of Applications, **(2013)**, pp. 38-44.

[10] S. Han, K. Yoon and I. Kweon, "A new Technique for Shot Detection and Key Frames Selection in Histogram Space", Taejon: Robotics & Computer Vision Lab, **(2000)**.

[11] M. Saric, H. Dujmic and D. Baricevic, "Shot Boundary Detection in Soccer Video using Twin-comparison Algorithm and Dominant Color Region", JIOS 32.1, **(2008)**, pp. 67-73.

[12] J. Mas and G. Fernandez, "Video Shot Voundary Detection Based on Color Histogram", Barcelona, **(2003)**.

[13] P. V. Kathiriya, "Chi-Square Based Shot Boundary Detection and Key Frame Extraction for Video", Research Inventy: International Journal Of Engineering And Science, **(2013)**, pp. 17-21.

[14] J. T. Newsom, "634 Data Analysis I", Portland State University, **(2013)**.

[15] Y. M. Fouda, "One-Dimensional Vector based Pattern Matching", King Faisal University, Al-Ahsa, **(2014)**.

[16] V. Chasanis, A. Likas and N. Galatsanos, "Simultaneous detection of abrupt cuts and dissolves in videos using support vector machines", Pattern Recognition Letters, **(2009)**, pp. 55-65.

[17] A. Jacobs, "Automatic shot boundary detection combining color, edge, and motion features of adjacent frames", Bremen: TZI - Center for Computing Technologies, n.d.

[18] D. Lyon, "The Discrete Fourier Transform, Part 6: Cross-Correlation", Journal of Object Technology, **(2010)**.

[19] G. Usha Rani, "Comparative Study of Traditional Metric for Shot Boundary Detection during Video Retrieval", International Journal of Engineering Technology, Management and Applied Sciences, **(2015)**, pp. 6-12.

[20] S. Adrian Martinez, "Acoustic signal detection through the cross-correlation method in experiments with different signal to noise ration and reverberation conditions", Gandia, n.d.

[21] Q. Chen, J. Tian and D. Wu, "A Robust Video Hash Scheme Based on 2D-DCT Temporal Maximum Occurrence", **(2010)**.

[22] V. Verma Sr. and R. Mishra, "Various Fast Block Matching Algorithm for Video Shot Boundary Detection", Electrical and Electronics Engineering, **(2013)**, pp. 11-17.

[23] T. Volkmer, "The Moving Query Window for Shot Boundary Detection at TREC-12", Melbourne, n.d.

[24] Y. Wang, "Video Processing & Communications", Polytechnic University, Brooklyn, **(2003)**.

[25] J. Chen, J. Ren and J. Jiang, "Modeling of Content-Aware Indicators for Effective Determination of Shot Boundaries in Compressed MPEG Videos", n.d.