VIDEO SEGMENTATION USING PARTIALLY DECODED MPEG BITSTREAM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

IŞIL BURCUN KAYAALP

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF ELECTRICAL AND ELECTRONICS ENGINEERING

DECEMBER 2003

Approval of the Graduate School of Natural and Applied Sciences

_____
Prof. Dr. Canan ÖZGEN
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

_____
Prof. Dr. Mübeccel DEMİREKLER
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

_____
Assoc. Prof. Dr. Gözde Bozdağı AKAR
Supervisor

Examining Committee Members

Assist. Prof. Dr. Aydın ALATAN _____

Assoc. Prof. Dr. Gözde Bozdağı AKAR _____

Prof. Dr. İsmet ERKMEN _____

Assoc. Prof. Dr. Volkan ATALAY _____

Assist. Prof. Dr. Cüneyt BAZLAMAÇÇI _____

ABSTRACT

VIDEO SEGMENTATION USING PARTIALLY DECODED MPEG BITSTREAM

KAYAALP, Işıl Burcun

MSc, Department of Electrical and Electronic Engineering

Supervisor: Assoc. Prof. Dr. Gözde Bozdağı AKAR

December 2003, 74 pages

In this thesis, a mixed type video segmentation algorithm is implemented to find the scene cuts in MPEG compressed video data. The main aim is to have a computationally efficient algorithm for real time applications. Due to this reason partial decoding of the bitstream is used in segmentation.

As a result of partial decoding, features such as bitrate, motion vector type, and DC images are implemented to find both continuous and discontinuous scene cuts on a MPEG-2 coded general TV broadcast data. The results are also compared with techniques found in literature.

Keywords: Shot boundary detection, shot detection, scene change, MPEG, cut, gradual transition.

ÖZ

YARI ÇÖZÜLMÜŞ MPEG VİDEO DİZİLERİN BÖLÜTLENMESİ

KAYAALP, Işıl Burcun

Yüksek Lisans, Elektrik ve Elektronik Mühendisliği Bölümü

Tez Yöneticisi: Doç. Dr. Gözde Bozdağı AKAR

Aralık 2003, 74 sayfa

Bu tezde MPEG video verisindeki sahne geçişlerinin bulunması için bit akışı için karma tip bölütleme algoritması gerçekleştirilmiştir. Ana amaç gerçek zamanlı uygulamalar için hesaplama zamanı açısından verimli bir yöntem elde etmektir. Bu nedenle bölütleme için yarı çözülmüş bit dizisi kullanılmıştır.

MPEG-2 formatında kodlanmış genel bir TV yayını verisinin hem sürekli hem de ani geçişlerinin bulunması için, yarı çözülme işlemi ile elde edilen bit oranı, hareket vektorü tipi ve DC resim verileri kullanılmıştır. Sonuçlar ayrıca bu alandaki diğer yöntemler ile karşılatırılmıştır.

Anahtar Kelimeler: Geçiş algılaması, sahne değişimi, MPEG, ani geçişler, kademeli geçişler.

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Assoc. Prof. Dr. Gözde Bozdağı AKAR for her valuable supervision and insightful comments throughout this thesis. I am grateful to her for her guidance, encouragement, and support which guided me in the development of this study.

I would also like to thank to my parents for their support and understanding during my studies.

TABLE OF CONTENTS

LIST OF TABLES

TABLE

LIST OF FIGURES

FIGURE

LIST OF ABBREVIATIONS

DCT	Discrete Cosine Transform
IDCT	Inverse Discrete Cosine Transform
MB	Macroblock
MPEG	Motion Pictures Experts Group
NTSC	National Television Standards Committee
PSC	Picture Start Code
VLC	Variable Length Coding

CHAPTER 1

INTRODUCTION

1.1. The Purpose of Video Segmentation

Due to the advances in compression technology, the expansion of low cost storage media, and the growth of the internet, digitally available video quantity has grown enormously.

Some of the potential applications where digital video is used are multimedia information systems, digital libraries, movie industry, and Video-on-Demand services. In all these services digital video needs to be properly processed and inserted into a video server for future access. These processes include compressing, segmenting and indexing a video sequence. Then the users can query the videos by an example image or several chosen features, and the system compares this query with various segments of the video [3, 4].

Compression is done by removing the temporal and spatial redundancies from raw video data. This is a necessary step for digital management of the video data, which needs great amount of space uncompressed. Indexing is extracting the key information from the video for future access from the databases. These features are extracted from the key frames supplied by the segmentation process. Segmentation is parsing of the video into temporally coherent sections, which can be considered as content based sampling of the video. Sampling cannot be done on

specified constant time intervals, since content differs greatly in different videos. Therefore video must be segmented within "content based" intervals. This segmentation makes it possible to manage the information contained in the video. As an example, it has been observed that for a 30-min video, only 1000 out of 45,000 frames are required to represent the video content [5].

Another application that is possible with video segmentation is content based browsing of compressed video, which should be performed with real time efficiency. Different from direct fast forwarding, content based browsing allows the user to view the information containing parts of the video faster and more easily. This will require algorithms to allow this process automatically.

An example application is the newsfeed material, which is currently recorded to digital videotapes in broadcast quality. News editors scan copies of these tapes to select material, then selected items are assembled for newscasts. This process is labor intensive due to increasing volume of material to be scanned, and requires specialized and expensive equipment, which are in limited number. A need thus exists for alternative computer based methods, which allow for faster analysis of newsfeed content. Due to the huge data volumes created when newsfeeds are digitized; the storage, communication and processing of digital videos is not feasible without compression of the digitized material. Therefore, content based browsing in compressed domain will be useful in editing of TV broadcast material. After the selection process, the actual newscast will again be assembled from broadcast quality digital tapes [6].

Video segmentation can also be used for military surveillance applications. In airborne applications, a large amount of data is collected by both manned and unmanned airborne surveillance platforms. The U.S. Department of Defense has made a commitment to move the collection of this data into the digital domain to support more effective and efficient exploitation, targeting MPEG-2 decompression algorithm for compression. Additionally, this video is being digitized and archived for post-mission purposes [7]. Ground based analysts are required to continually view the video being transmitted by the sensor, and the mission can last for several hours. Detection of the transition phases of video automatically can greatly help the analysts, since the sensor is constantly transmitting data; therefore most of it is likely to contain little information of interest.

From the definition of compressing, parsing and indexing, we assume that the video is already compressed, and concentrate on the parsing problem in this study

1.2. Types of Scene Change

Video can be represented hierarchically through shots, scenes, and episodes. A *shot* is a sequence of frames which represents a continuous action in time and space. The video content is similar in shot regions. The regions where the content change occurs are therefore called shot boundaries.

A group of semantically related shots can be connected together to form *scenes*. For example, a telephone dialogue scene can be composed of several successive shots in which the first person and the second person can be viewed

consecutively. Finally the related scenes of a story can be connected together to form a bigger class named as *episodes*, sometimes referred to as *logical story units*.

Structural grouping of shots and scenes automatically requires extraction of several features and statistical behavior, and evaluating these features analytically to represent human perception. In this thesis we concentrate on the shot detection part, which is the main and basic structural unit in a video, and a key step in many applications.

There are two types of shot transitions: Abrupt and gradual. Abrupt transitions are called discontinuous cuts, where the transition boundary is between two consecutive frames with different context. A part of a video with a scene cut can be seen in Figure-1.2.1.

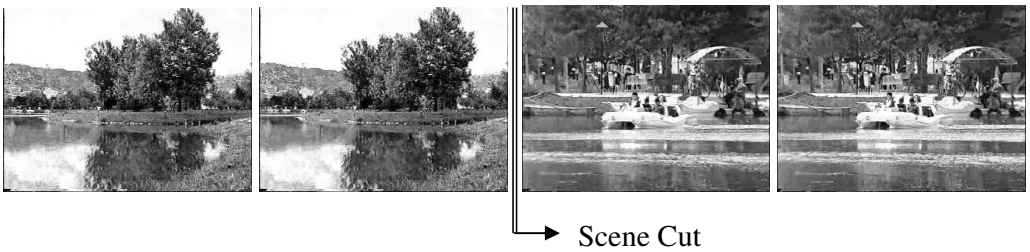

Figure-1.2.1. Successive Frames of a Scene Cut

Gradual transitions occur in a longer period, i.e. more than one frame. The number of possible transitions due to editing effects is quite high, but most of them fall into the main categories which are dissolve, fade, and wipe effects.

Dissolve is the most frequently used editing effect used to connect two scenes. A dissolve-transition is formed by linearly decreasing the intensity of the first scene and linearly increasing the intensity of the second scene in the mixed region. An example of a dissolve can be seen in Figure-1.2.2.

Figure-1.2.2. Successive Frames of a Dissolve

A dissolve sequence $D(\mathbf{x},t)$ is defined as the mixture of two video sequences $S_1(\mathbf{x},t)$ and $S_2(\mathbf{x},t)$, where the first sequence is fading out while the second one is fading in:

$$D(\mathbf{x},t) = f_1(t)\cdot S_1(\mathbf{x},t) + f_1(t)\cdot S_1(\mathbf{x},t),\ t \in [0,T]$$

The most common types of dissolves can be classified into two categories: Cross dissolves (Figure-1.2.3) and additive dissolves (Figure-1.2.4). In cross dissolves, which are the most common types, intensity scaling functions of the incoming and outgoing functions, f_1 and f_2 respectively, are defined as:

$$f_1(t) = (T-t)/T, \quad \text{and} \quad f_2(t) = t/T$$

In additive dissolves fading functions are defined as:

$$f_1(t) = \begin{cases} 1, & if\ (t \le c_1) \\ (T-t)/(t-c_1), & else \end{cases} \qquad \text{and} \qquad f_2(t) = \begin{cases} t/c_2, & if\ (t \le c_2) \\ 1, & else \end{cases}$$

where $c_1 = (0, T)$, $c_2 = (0, T)$

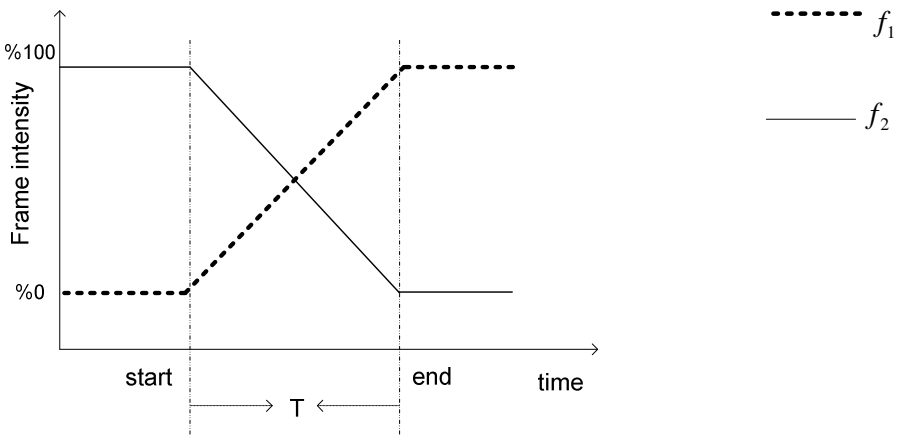

Figure-1.2.3. Cross Dissolve

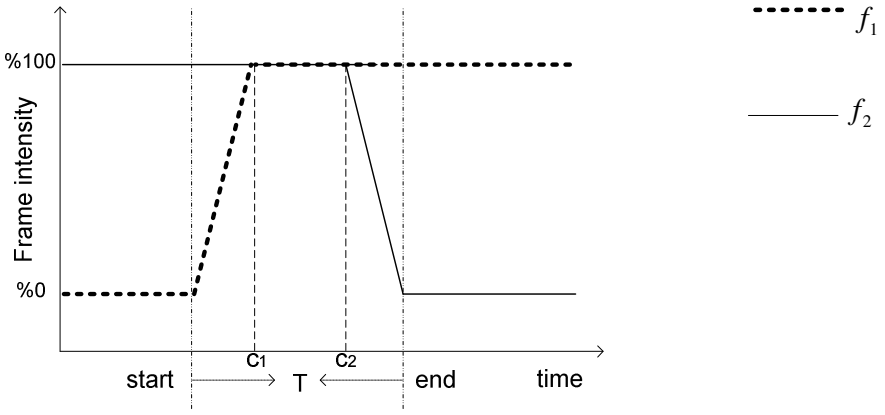

Figure-1.2.4. Additive Dissolve

A fade occurs when either of the frames of a dissolve is a solid color, such as a black scene. When the first frame is a solid color, it is called as fade-in, and when the second frame is a solid color, it is called a fade-out.

A wipe is a transition from one scene to another where the new scene is revealed by a moving boundary in the form of a line or pattern. This moving boundary can be of any geometric shape [8]. Wipe is generally used with sports and replay scenes. More artistic effects can also be generated with several editing techniques. Figure 1.2.5 shows wipe transition examples.

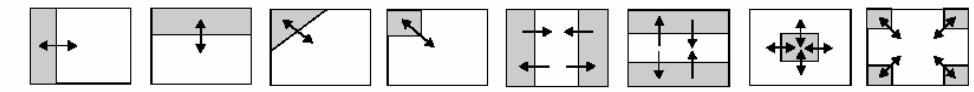

Figure-1.2.5. Primary Patterns of Wipe Transitions

In this study our aim is to find these scene changes scenes in order to parse the video. Existence of camera and object motion makes this video segmentation more difficult, since the metrics used in detection are also sensitive to these effects. Another point is the detection of gradual scene changes. These are more difficult to detect compared to abrupt cuts, since the two successive frames are generally uncorrelated in abrupt transitions, whereas in gradual transitions there is a transition-type dependent correspondence.

Methods for detecting these shot boundaries can be classified into two groups: Algorithms that work on the uncompressed domain and algorithms that work on the MPEG-compressed domain.

1.3. Compressed vs. Uncompressed Domain Algorithms

Uncompressed domain algorithms work directly on the video data. However, most video data is stored in a compressed format since uncompressed

video needs a great amount of storage. For example, television quality video requires approximately 100 GBytes of digital storage for each hour.

Therefore this compressed data must be decompressed before it can be processed. This increases the cost of processing enormously, such that MPEG requires 2.7 billion instructions for each second of NTSC quality video for decompression. Furthermore, the data must often be compressed after processing, which adds one to fifty times more overhead [9]. Another consequence is that data size is much higher in uncompressed video, and it is more difficult to manage and process this data rapidly.

The compressed domain processing techniques use the information extracted directly from compressed bitstream, and therefore are more advantageous than the uncompressed domain in computational means. To achieve this, the information already inherent in the video stream, which was included during the compression stage, is utilized. A partial decompression must still be done to extract the information necessary for detection; however this overhead is small compared to full decompression of the video stream. It is shown that in MPEG video decompression; approximately 40% of the CPU time is spent in Inverse Discrete Cosine Transform (IDCT) calculations, even when using fast DCT algorithms [10]. Methods utilizing DCT coefficients for shot detection avoid IDCT calculations are therefore computationally more efficient than uncompressed domain methods. Other methods which require lesser amount of decompression are even more efficient.

There is a duality between compressed domain and uncompressed domain methods. Pixelwise difference, i.e. frame intensity based, and histogram based metrics are used in both domains, but with different inputs. Uncompressed domain algorithms uses image obtained directly from the frame, whereas the compressed domain algorithms apply similar algorithms to the reduced DC images, or sometimes DC and AC coefficients of the encoded frame. The techniques previously proposed for finding gradual transitions according to frame intensity and histograms are also adapted later to be used with DC images. Similarly, edge based methods are used in compressed domain, utilizing the DCT coefficients of the frame instead of the full frame. There is also work on using motion information on uncompressed domain which requires costly calculations. In MPEG-domain this information is obtained from the readily available motion vectors.

The focus is shifted towards the compressed domain techniques, due to the reasons mentioned above, and the increased amount of compressed video available. This work will compare the various compressed domain techniques, and combine the efficient methods for video segmentation.

1.4. Outline of the Thesis

This thesis is organized as follows: In Chapter 2, various compressed domain algorithms and previous comparison among them are discussed. In Chapter 3, implementation of the algorithms and their comparison results are presented. Concluding remarks and results are given in Chapter 4.

CHAPTER 2

LITERATURE SURVEY

Various methods have been proposed for scene change detection, which can be classified according to their source of data for detection. In this section we will concentrate on the methods in compressed domain. These are, (i) Pixel Intensity Based Methods, (ii) Histogram Based Methods, (iii) Edge Based Methods, (iv) Motion Vector Based Methods, (v) Macroblock Type Based Methods, (vi) Bitrate Based Methods, and (vii) Mixed Data Based Methods.

2.1.Pixel Intensity Based Methods

Intensity difference based methods were of the first to be found in literature. The frame by frame difference is calculated for successive pair of frames. The reasoning is that each shot has a different intensity distribution. By taking the difference shot boundaries are detected.

Pixel based methods are quite sensitive to camera and object motion, and illumination changes in uncompressed domain. The intensity distribution is greatly changed in frames where there is a rapid change in motion. Some methods have been proposed to overcome this effect, such as filtering, or partially selecting regions of the frame in uncompressed domain [32].

In compressed video, pixel difference is calculated with DC coefficient of the Discrete Cosine Transform (DCT), i.e. the reduced DC image. A DC image is composed of DC coefficients of the DCT coded frame, and it is 64 times smaller

than the original frame. The reduction in the image size is advantageous that it reduces the signal noise, and the smoothing effect previously obtained by filtering is already available with the smaller DC image.

The extraction of DC images from I frames are straightforward, since I frames are intra coded. However, P and B frames are coded with reference to I frames, and DC images must be reconstructed from motion vectors and error data. A first order approximation can be used which approximates the second block as a weighted sum of the reference blocks, where the weight is the fraction of the area occupied by the subblock. The motion vector relation is shown in Figure 2.1.1.

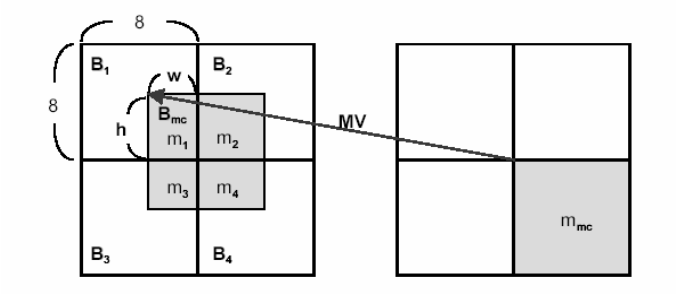

Figure-2.1.1. Motion Vector Relation between Consecutive Blocks

Exact reconstruction of the blocks is costly; on the other hand error may be propagated when approximations are made [14].

The DC term of the 2-D Discrete Cosine Transform of an NxN sized block is related to the pixel values $f(i, j)$ of the frame via $S(0,0) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x, y)$. For N=8, the DC term is 8 times the average intensity of the block. In Figure-2.1.2, the picture on the right is an enlarged view of the DC image of the frame on the left.

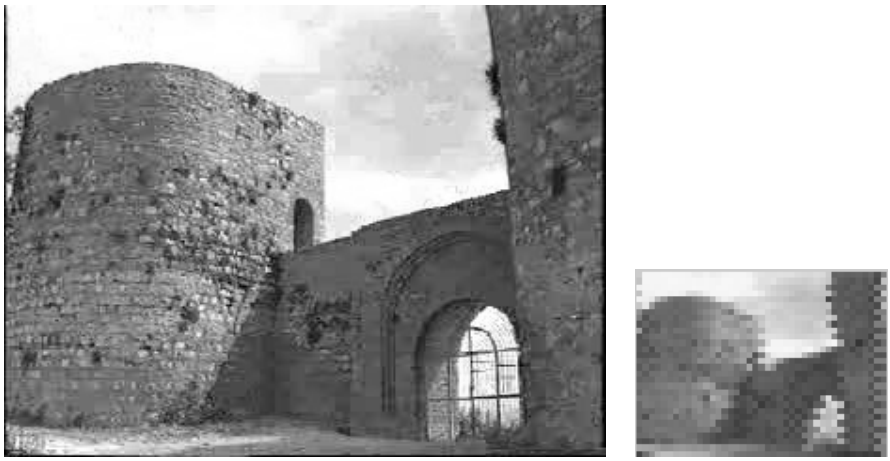

Figure-2.1.2. DC Image of a Frame

The intensity difference between DC images is one of the methods to compute frame changes. The difference calculated with:

$d(X,Y) = \sum_{i,j} \left| x_{i,j} - y_{i,j} \right|^{\alpha}$, $X = \{x_{i,j}\}$and $Y = \{y_{i,j}\}$ being two consecutive DC images, and α=1 or 2, based on the difference metric [11]. This method emphasizes the differences better.

Fernando et al. [12] uses statistical features to calculate frame differences. The mean and standard deviation of intensity values of DC images are calculated, and compared with Mean Square Error metric: $MSE_n = (\mu_n - \mu_{n-1})^2 + \left| \sigma_n^2 - \sigma_{n-1}^2 \right|$. However, using global mean for the image is a coarse figure for robust detection.

Arman et. al. [13] use selected DCT coefficients; such that first a subset of 8x8 blocks is chosen, then a subset of coefficients from each block is chosen a priori. A vector is formed with these coefficients and compared by using their inner product. DCT coefficients already represent a subset of the image; therefore a subset of these images may lead to a loss of information.

2.2.Histogram Based Methods

In this method, the histogram values of the frames are pairwise compared with several comparison methods, obtained from Discrete Cosine Transform (DCT) coefficients of the MPEG coded video. In MPEG, the native color space is YCrCb. Some algorithms use only luminance histograms, while the other uses both luminance and color histograms.

One disadvantage of the histogram method is that it can fail to detect a scene change if the frames have similar color or luminance distribution. An improvement to this effect on uncompressed domain was to take local histograms in a frame to reduce the similarity probability.

In [15], three dimensional histogram is combined to a single dimension with 8 bits by using 4 bits for Y (luminance) and 2 bits for each Cr and Cb color components. The absolute difference is used for comparing histograms.

Shin et. al. [16] uses only I frames for histogram calculation. This gives a coarser location of the shot, but avoids frame reconstruction. If more than one shot is present within one GOP, it will not be detected; however this probability is relatively low. Yakimovski likelihood, Chi-square test, Kolmogorov-Smirnov test are compared, and the Chi-square (x^2) test is found to be the best comparison metric for histogram. x^2 test is given as: $x^2 = \sum_j \frac{(HP_j - HC_j)^2}{(HP_j + HC_j)^2}$, where HP_j and HC_j represent previous and current histograms for the j[th] bin.

2.3.Edge Based Methods

Edge type methods are based on the observation that, during a cut or a dissolve, new intensity edges appear far from the locations of old edges, and vice versa. Edge information is calculated from DC images. Edge detection algorithms are the computationally most costly ones compared to other metrics.

Zabih et. al [17] use an edge based method for video segmentation. First edge detection is performed resulting in binary images using magnitude of the intensity gradient. Then dilation operation is applied to these edge images. Then the fraction of edge pixels which are more than a fixed distance *r* from the closest edge pixel in previous frame is measured, giving the proportion of entering edge pixels. It has a high value during a fade-in, or a cut, or at the end of a dissolve. Similarly, the fraction of edge pixels which are farther than *r* away from the closest edge pixel in previous frame, gives the proportion of exiting edge pixels. Maximum of these two measures gives the edge change ratio.

Shen [18] improved this method by stating that dilation in a small area might be insufficient for large object and camera movements. First, the images obtained from I frames are divided into subregions, and then Hausdorff Difference Metric is used as a comparison for similarity. The differences are thresholded and merged to eliminate noise. The peaks in the resulting graph are used for finding cuts.

Another method uses AC coefficients to derive binary edge maps, depending on the fact that AC values essentially depend on intensity differences in various orientations. For example, AC_{01} and AC_{10} values depend on intensity

differences in horizontal and vertical directions, respectively. The edge orientation, strength and offset are measured using the correlation between the AC coefficients [10].

2.4.Motion Vector Based Methods

In uncompressed domain, there are studies on the motion vector calculations with methods such as block matching [19]. However, this process was computationally very costly, since each block in each frame has to be searched within a region and motion information has to be extracted. MPEG standard already implements this motion prediction, and encodes the inter frames (P and B frames) according to the predicted values, and each macroblock in the inter frames has a motion vector type. Therefore, several methods make use of this information. Computation in compressed domain eliminates the need for calculation from raw data; however it is still a costly solution.

Stepwise reconstruction of motion vectors is used in [25] both for cut detection and camera motion characterization. Motion distance should be calculated considering that motion vectors have a two level of indirection, i.e. B frames can refer to P frames, which then refer to I frames.

2.5.Macroblock Type Based Methods

Macroblock is the 16x16 pixel motion compensation unit within a frame. Macroblocks have the frame to frame motion compensation information in MPEG. I pictures can only have Intra blocks. P and B pictures can have different modes

according to motion content. Macroblock type modes in P and B pictures are given in Figure 2.5.1 and Figure 2.5.2 respectively.

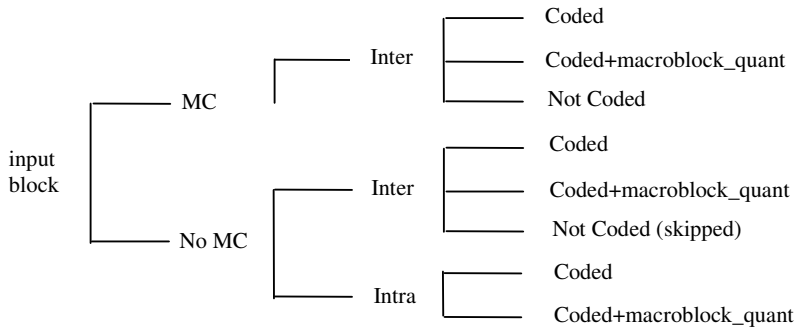

Figure-2.5.1. Macroblock Type Modes in P Pictures

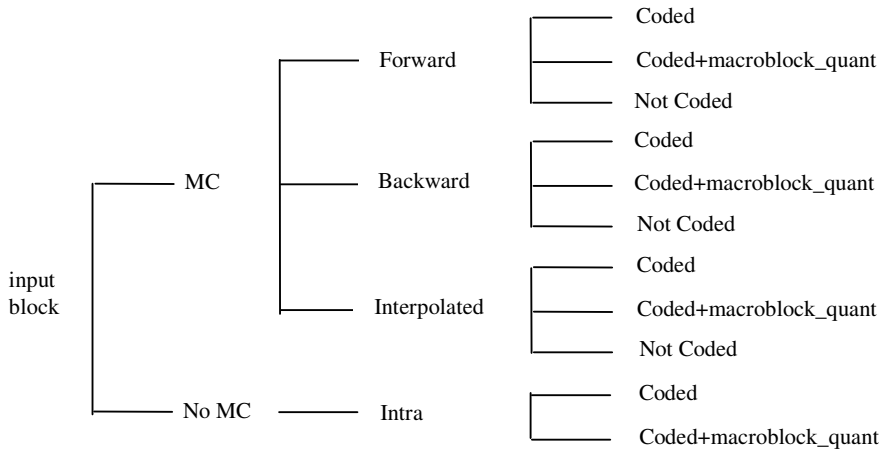

Figure-2.5.2. Macroblock Type Modes in B Pictures

The main idea for P and B frames' motion vector behavior is; if a frame is inside of a shot, then the macroblocks should be predicted well from previous or next frames. However, when the frames are on the shot boundary, the frames cannot be predicted from the related macroblocks, and a high prediction error occurs. This causes most of the macroblocks of the P frames to be intra coded instead of motion compensated. In B frames bi-directional prediction is mainly used to enhance

coding efficiency. When a shot occurs, most of the blocks are forward or backward coded according to the place of the shot.

In [20], macroblock types in B pictures are used for detection. Three possible cases are shown in Figure-2.5.3, big arrows indicating the dominant MB direction.

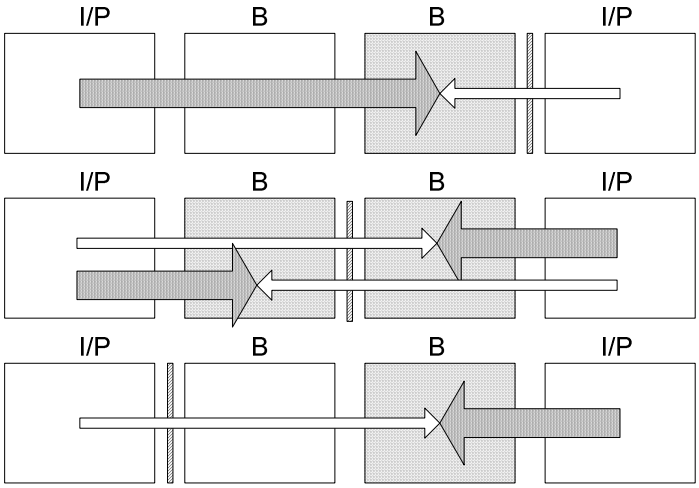

Figure-2.5.3. Motion vectors for B frames

Kuo and Chen [21] use the same approach, defining a mask for each different case. It can be extended to frames with more than two consecutive B's.

[7], [16], [22], [23] and [24] use the same idea, with similar metrics for comparison.

2.6.Bitrate Based Methods

Bitrate based methods utilize the encoding cost data for finding shot boundaries. Bitrate calculation is the computationally most efficient method together with the macroblock based approaches.

For I pictures, a scene change will lead to a significant change for both the DC and AC information of the picture. This change is reflected by the number of bits required for the picture. For P and B frames bitrate is primarily motion dependent. Also the scene cuts may lead to intra-coded blocks instead of the motion compensated ones, which in turn increases the bitrate of the frames.

In [26] number of bits used to decode frames is calculated to make a pre-segmentation. This method has a very low computation time, since the detection process requires no decoding at all. The algorithm computes the bitrate for I, P and B frames. This data is combined for all frame types, and lowpass filtered to find the gradual transitions. Highpass component is used to find abrupt cuts. This information is combined with skipped data information. Finally, the obtained data is clustered to find scene changes.

Bitrate variation is considered at the macroblock level in [27]. A *rate matrix* is setup for each frame by collecting the bitrate information of all macroblocks. By comparing bitrate information at the macroblock level, the possibility of error due to global similarity is eliminated. The difference in rate matrix is used as a detection criterion for I and P pictures. The rate difference is defined as:

$d(R,S) = \sum_{i,j} \left| r(i,j) - s(i,j) \right|$, where $R = \{r(i,j)\}$ and $S = \{s(i,j)\}$ are two rate matrices. The normalized rate difference $RI_n = \frac{d(R_n, R_{n+1})}{T_n}$, where T_i is the total number of bits required. In B pictures the bitrate change is stated not to be obvious, therefore the percentage of backward motion prediction is used for B pictures. In a scene cut, most of the motion vectors will come from the anchor frame in future

display order. Backward prediction ratio for B frames is defined as $BP_i = \frac{N_b(i)}{N_m(i)}$, where $N_b(i)$ and $N_m(i)$ represents the number of backward motion-predicted blocks and the total number of motion-predicted blocks for the i[th] frame respectively. Detection is done two stages. First I frames are checked, if there are no suspected changes, next I frames are compared. Otherwise, P and then B frames are checked to locate the shot. The algorithm uses fixed thresholds, and no method for gradual detection is proposed.

Another algorithm utilizing bitrate information is [8], which references the previous algorithm. I and B frames are detected by the same method using the bitrate metric. For P frames, the ratio of intra-coded macroblocks to the total number of macroblocks, $(\frac{MB_{\text{int}ra}}{MB_{total}})$, is calculated instead of the bitrate metric. Detection is done with stepwise refinement similarly.

For intra coded macroblocks, the size of the macroblocks depend on the number of AC coefficients, and the Variable Length Decoding (VLC). In [33] it has been shown that there is a linear relationship between the number of AC coefficients and the macroblock size. It is also shown that macroblock size is more a function of the number of items in the MB itself, and VLC does not have very much influence. Therefore the scene cut detection using MB size calculation will approximately give the same result as the number of AC coefficients, which determine the level of detail in the macroblock

In [33], macroblock size for inter frames are reconstructed using motion vectors to allow comparison. However this approach hinders the benefit of computation efficiency obtained by using MB bitrate.

2.7.Mixed Data Based Methods

Information obtained from DCT coefficients give good results, however motion compensated P and B frames has to be reconstructed from I or P frames. On the other side, motion vector information gives good results with P and B frames with smaller processing time. Therefore, in many studies these properties are unified for shot detection.

Falkemeier et. al. [28] compares I frames at the first stage by looking at differences of the DC components. If a cut is detected at this stage, the number of intra coded macroblocks is checked to further locate the cut. At the third stage forward and backward motion vectors is calculated. This approach increases the speed of computation, since only changed frames are checked instead of all frames.

Kobla et. al [29] use motion vector information form MB's to detect cuts, but also uses DC coefficients of I frames for cut validation. In the case of large quantization scales, or relatively still frames; skipped MB's dominate, giving no information about the correspondence of the next frame. DCT validation is helpful in these cases.

In [30], both motion vector information and luminance values of P and B frames are used. The metrics are combined to finally decide on the suspected scene changes.

Liu and Zick [31] use sum of square of all DCT components in a block, defined as error power, together with the motion vectors.

2.8.Uncompressed Domain Algorithms

In this section we briefly mention about uncompressed domain methods for the purpose of comparison. Histogram difference, pixelwise difference, and edge based algorithms are the most frequently used methods in the uncompressed domain. Histogram based algorithms are found to have very good performance for shot detection purposes. [39]

In [40], histogram based method is combined with pixelwise difference to increase robustness. For pixelwise comparison intensity levels is quantized to eliminate noise. Clustering is applied on these two metric to obtain shot boundary, and non shot boundary regions. Finally elimination of false positives is applied on the clustered data. We are going to use [40] for comparison, whose results are given in Section 3.6.

2.9.Summary

Pixel difference based methods are sensitive to camera and object motion in uncompressed domain. However, by using DC image versions, the average of blocks is obtained, which gives better results. Using metrics such as square of differences rather than absolute differences further improves the results. Using only I frames for comparison is sufficient for many cases, since the change within one GOP lasts about half a second. Another approach is to use all frame types for obtaining frame differences. This method is more costly since motion compensation

is necessary for inter coded frames, however, for scenes containing fast object and camera motion this extra information may be useful.

Histograms based methods, on the other hand, are comparable in performance with pixel difference based methods, and can be more robust to object motion. The biggest disadvantage of this method is; it fails to detect a scene if the two different scenes have similar histograms.

Motion vector based methods are computationally easy to obtain. They depend on encoder calculated motion estimation values, which give accurate results for calculations. The disadvantage is that gradual changes can be missed with this method.

Edge based methods are quite costly in calculations. They are more resistant to illumination changes, but shots can be missed for dark scenes, and big object motion.

Bitrate based methods are very efficient computationally, but for similar bitrate distributions, missed detections can occur.

The results for recall and precision given by authors are sometimes obtained with few data, and sometimes the results are tuned to the specific set of data under examination, due to threshold and other parameter selection. Therefore, different results are obtained in comparison papers. Another point is that, each algorithm's performance may change with the set of data observed, i.e. for some data an algorithm performs better, but for another set of data other algorithm's performance can be better.

In our comparisons which will be given in the next section, we used the algorithms which are both effective and computationally efficient. These are frame and macroblock bitrate based methods, macroblock type based methods, DC image difference and DC image variance calculations.

CHAPTER 3

SCENE CHANGE DETECTION in COMPRESSED DOMAIN

As stated in the previous section, bitrate based metrics are very efficient computationally, MB type based metrics are both efficient and less dependent on intensity variations. DC image intensity gives a filtered version of the spatial information is obtained, also more successful in gradual changes.

In this section we utilize all of these metrics to obtain a both computationally efficient and robust algorithm. MPEG encoded sequences encoded from broadcast TV is used in detection.

Three video sequences of is used for analysis. News sequence contains 66 cuts and 11 gradual transitions (dissolves). Mixed sequence has short sections of news, documentary and video clips. This sequence contains 11 gradual transitions and 47 cuts. Sequence lottery has 45 cuts and 3 dissolves.

3.1.Evaluation Criteria

The algorithms are tested on three MPEG encoded general TV broadcast. Ground truth data is manually extracted for comparison.

Bitrate metrics, DC value of I frames, and macroblock type data based metrics are tested. Also one uncompressed domain algorithm is analyzed for comparison.

Performances the algorithms are evaluated through *Recall* and *Precision* measures. The recall measure evaluates the percentage of actual shot cuts that the method in question has detected, while the precision measure is a percentage showing how accurate the method is at detecting shot boundary. They are calculated using the following formulae:

$$\text{Recall} = \left(\frac{Number\ of\ Correctly\ Detected\ Shots}{Actual\ Number\ of\ Shots} \right)$$

$$= \left(\frac{Detections}{Detections + Missed\ Detections} \right)$$

$$\text{Precision} = \left(\frac{Number\ of\ Correctly\ Detected\ Shots}{Total\ Number\ of\ Shots\ Detected} \right)$$

$$= \left(\frac{Detections}{Detections + False\ Alarms} \right)$$

A false alarm is declaring a shot, when there is no shot boundary between the frames. A missed detection is not identifying a shot, when there is a transition. Missed detections are considered to be more important than the false alarms, since they lead to loss of information. Ideally, both recall and precision should equal 1. This would indicate that all existing shot boundaries are detected correctly, without identifying any false boundaries.

3.2.Total Bitrate Based Result

Total bitrate of the I, P and B frames are calculated for several test videos. I frames represent the level of activity for the video sequence, since they are intra coded. In case of a scene change there is a change from low activity to high activity region, or vice versa. To account for this change, difference in the I frame bitrate is calculated. P and B frames are inter coded, therefore scene activity is not directly measured, however intra coded blocks will lead to increase in bitrate.

Calculation of bitrate is done through *Picture Start Code*. It indicates the start of a frame in an MPEG sequence, which is uniquely defined as 0x0010. It can be directly extracted from the bitstream and does not need Huffman decoding. By marking the Picture Start Code (PSC) locations, and counting the number of bits until the next start code, the bitrate used in encoding the video can be found. Frame type information is also obtained from this picture header.

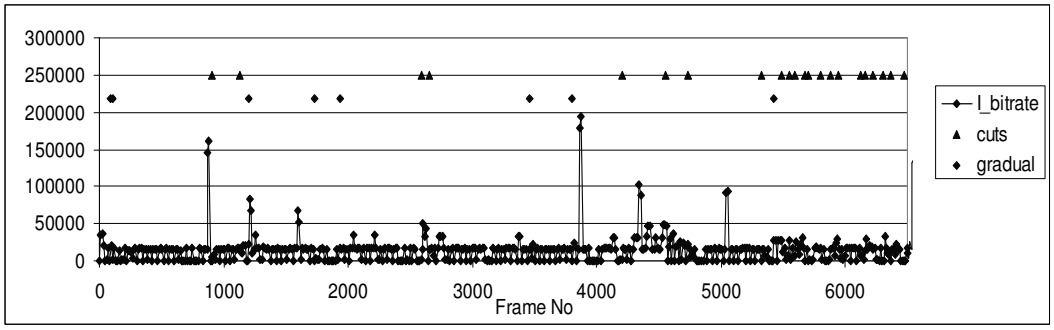

Figure-3.2.1. Difference of I Frames' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_news

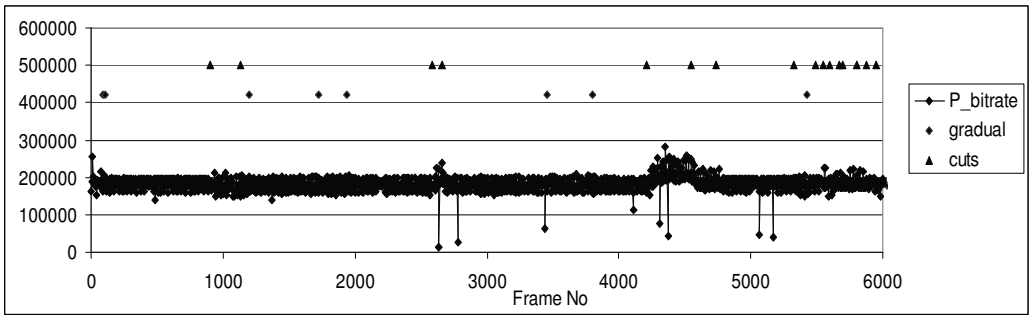

Figure-3.2.2. Difference of P Frames' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_news

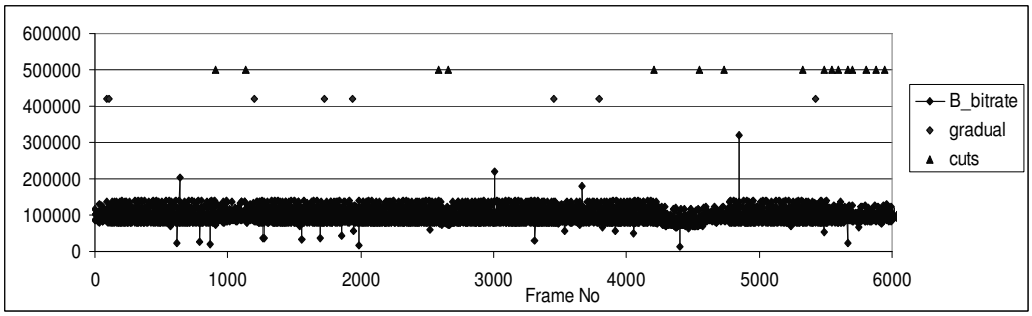

Figure-3.2.3. Difference of B Frames' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_news

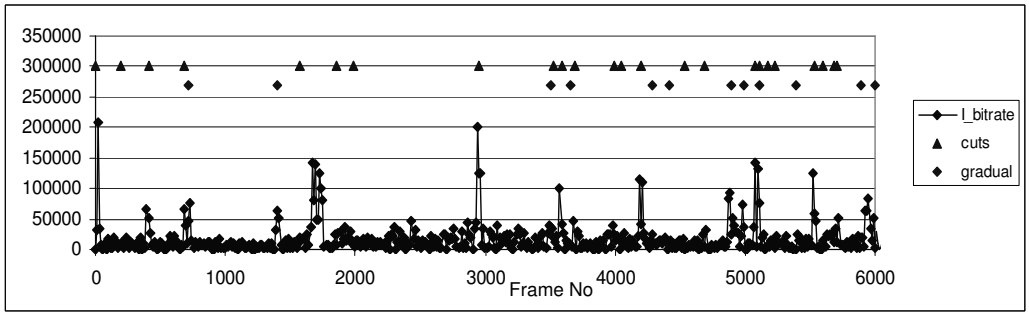

Figure-3.2.4. Difference of I Frames' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_mixed.

However, bitrate based algorithm is observed to be noisy. The reasons are, first this is an encoder-dependent algorithm, and variations such as encoding frame bitrate can affect the results. Second, when two scenes are in the same shot, it is expected to have a similar bitrate value for two frames; however, factors such as camera and object motion adversely affects the results. Also when the whole frame is considered some level of detail is lost. A region in frame can have high activity, which can cause fluctuations in the frame based comparison.

3.3.MB Bitrate Based Results

Frame based bitrate gives a rather coarse information for analysis. To improve this effect bitrate based algorithm is implemented at macroblock level. Bitrate differences of frames at the macroblock level are calculated as follows: $BR(R,S) = \sum_{i=1,j=1}^{N} \left| r(i,j) - s(i,j) \right|$, where $r(i,j)$ and $s(i,j)$ are the bitrate, i.e. data sizes of two macroblocks in two consecutive frames, and N=Number of macroblocks in a frame. Video is parsed at the bitrate level to obtain the macroblock data size. In MPEG headers are directly available up to slice level. Therefore; different from frame level headers, a VLC decoding has to be done to access macroblock headers. Then the number of bits necessary to encode the data is counted until the next macroblock header.

The results are improved compared to bitrate of frames statistic, and sharper peaks are observed in intra frames. The metric is also able to detect gradual transitions, which are plotted with diamond shape in the Figure 3.3.1.

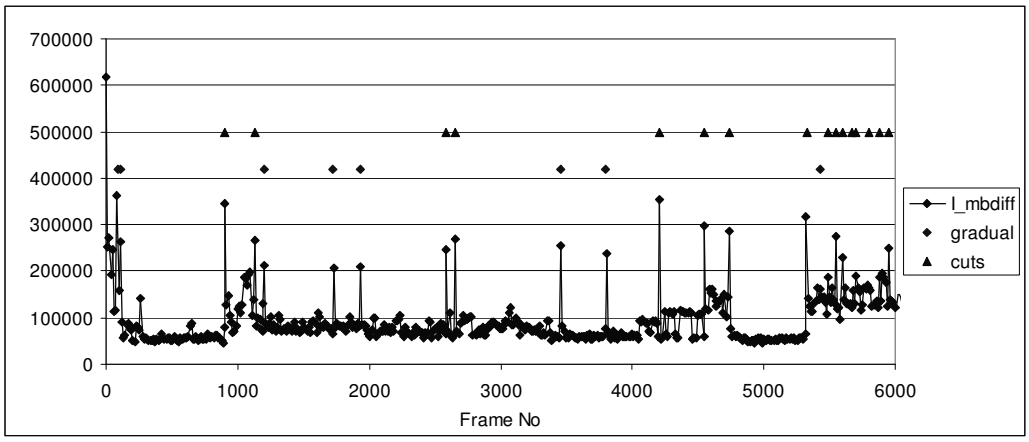

Figure-3.3.1. Difference of I Macroblocks' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_news

For scenes where there are multiple close shots, or with high activity, it is more difficult to detect shot boundaries.

P and B frames have also peaks at some cut locations, but their overall contribution is observed to be too noisy to obtain reliable information. There are spurious peaks which do not correspond to any transition. Also for some regions they follow a zigzag pattern, and their value is not consistent during the non-shot regions.

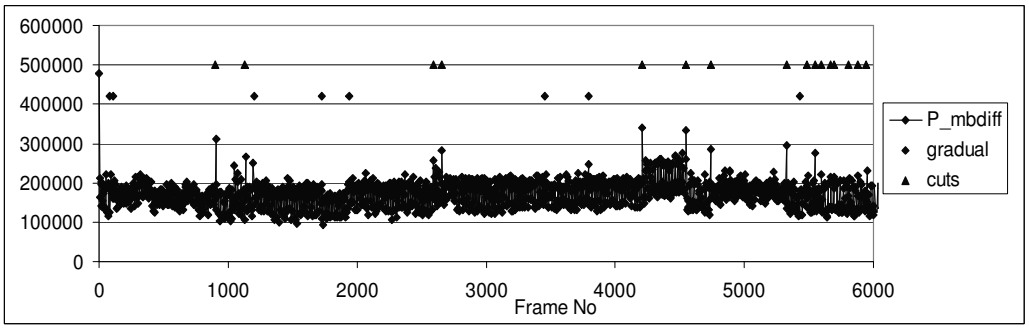

Figure-3.3.3. Difference of P Macroblocks' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_news

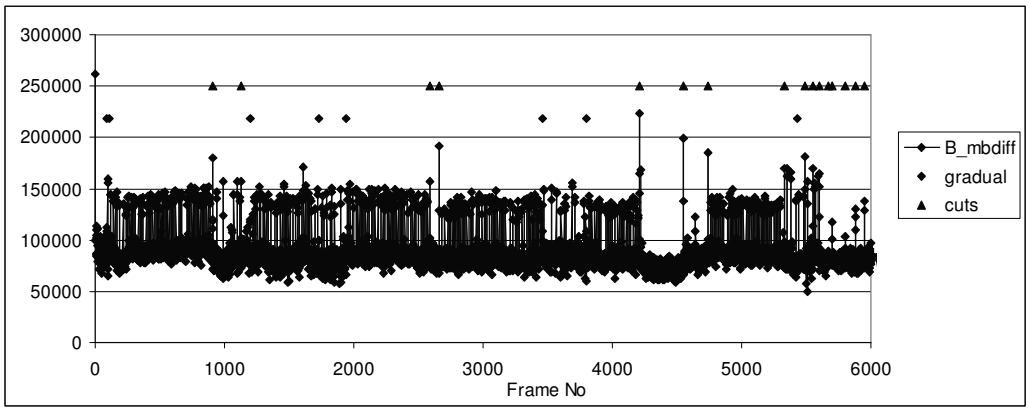

Figure-3.3.4. Difference of B Macroblocks' Bitrate Compared to Cut and Gradual Transition Locations for Sequence_news

3.3.1 Peak Detection with Sliding Window

For automatic peak detection, sliding window approach is used. Data is normalized to total MB size during calculations. Algorithm checks three conditions:

(i) Data is above a certain threshold; which is set at a low value. Since data is independent of the total macroblock size, this value can be used in general. This condition ensures that small spurious peaks are ignored.

(ii) Local maximum condition is satisfied, such that $D_n > D_{n+1}$ and $D_n > D_{n-1}$, where D_n is the analyzed data at point n.

(iii) Sliding window of size 2*N+1, centered on the data point analyzed is used. N=4 is taken for calculations. Since data resolution is GOP, smaller window size is considered to be more convenient. Mean of this window is calculated after subtracting the maximum value within the window, to obtain a reliable figure. If the

data point is above mean*2 value, and also satisfies conditions (i) and (ii), it is declared as a cut.

Sliding window calculations can be applied in real time, and memory requirement is equal to window size only. Other approaches such as “n times larger than the second largest peak” or “maximum valued sample within the window” are not preferred, because they do not allow a second valid peak in the same window. Since in our case resolution is at GOP, close peaks occur for fast content change. Threshold for condition (i) is chosen as 250 for this method.

Performance of the algorithms is given in Table 3.4.2.

	Detected	*Missed*	*False Alarm*
Cuts	44	22	2
Gradual	10	1	N/A
Recall		*Precision*	
66.7 %		95.6 %	

Table 3.4.2 MB Bitrate Algorithm’s performance with news sequence

This algorithm performs poorly where similar content change occurs. 15 of these missed detections are from a group of related shots in black and white, where encoding cost is similar. Its short dissolve detection is performance, however, is quite good.

3.4.MB Type Based Results

To obtain MB Type information, which is included in the macroblock header, VLC decoding should be done. For the first MB of each slice, the horizontal position with respect to the left edge of the picture is coded using MB address

increment. Additional MB's are coded differentially with respect to the most recently transmitted MB. Each MB Type pattern has a specific VLC codeword.

Only VLC decoding and *N*=Number of macroblocks times addition operation is necessary to obtain the MB data. Therefore it is computationally very efficient.

I pictures can only have Intra blocks. Macroblocks in P pictures usually have Fwd (forward) motion compensation, but also can be Intra coded. B pictures usually have Interp. (Interpolated, i.e. bidirectional) motion compensation, but also have Fwd, Bwd (backward) or Intra blocks. Also in the macroblock header there is "quantizer scale code and "coded block pattern" indications. We grouped them as P_Fwd, P_NoMC, B_Intra, B_Fwd, B_Bwd, and B_Interp.

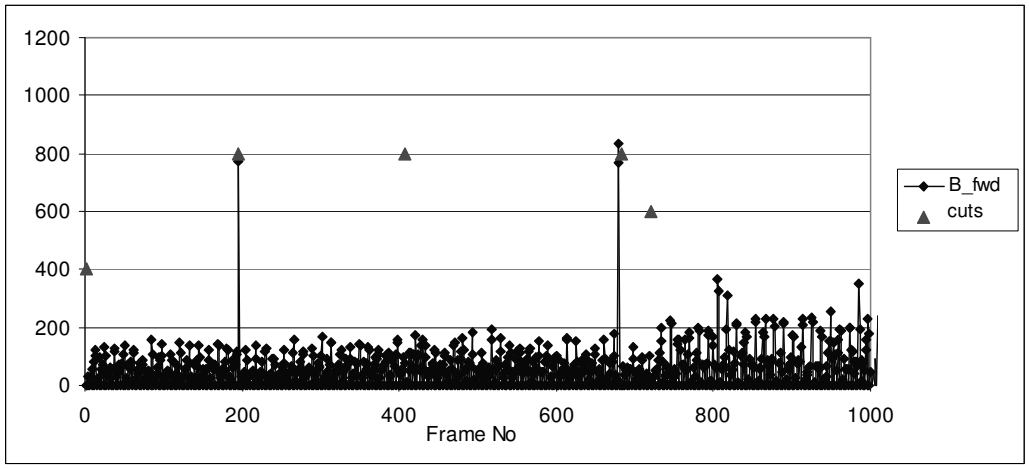

Figure-3.4.1. Number of Forward Macroblocks for B frames Compared to Cut Locations for Sequence_ mixed.

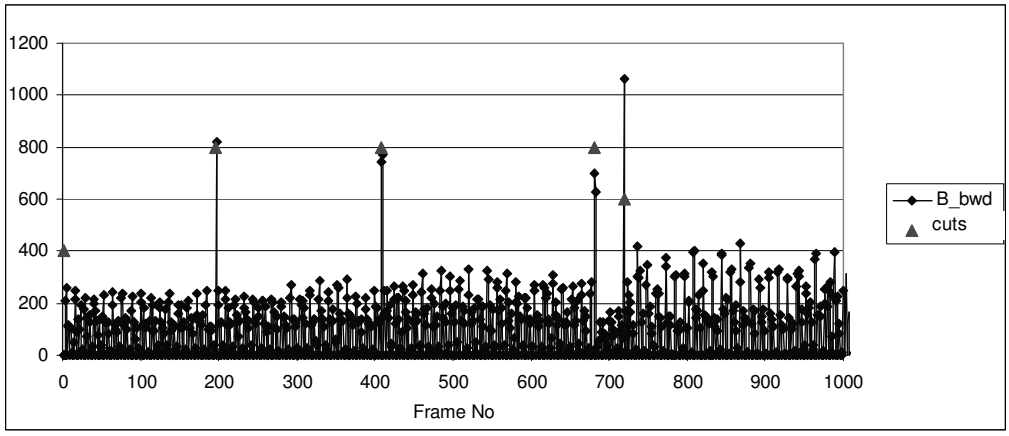

Figure-3.4.2. Number of Backward Macroblocks for B frames Compared to Cut Locations for Sequence_ mixed

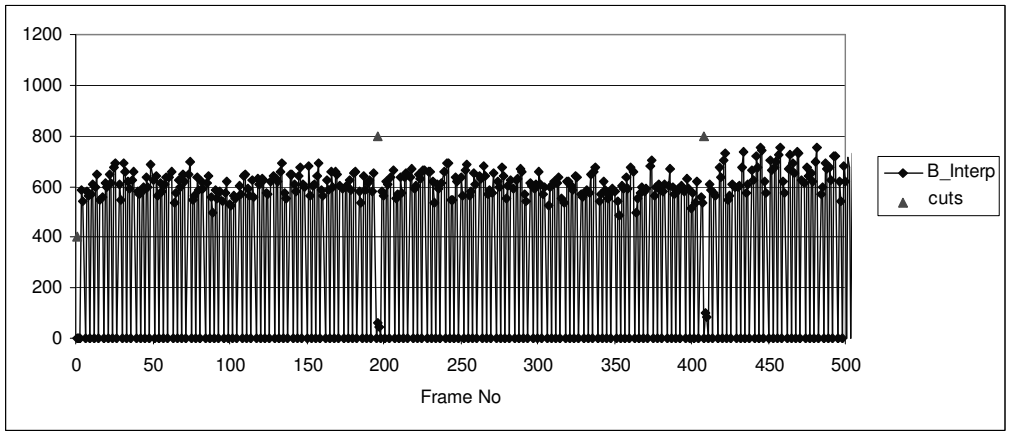

Figure-3.4.3. Number of Interpolated Macroblocks for B frames Compared to Cut Locations for Sequence_mixed

Generally MPEG video is coded with two successive B frames, which is an efficient method. Our entire test data is coded with this pattern, and analysis is done based on this. However, this method can be easily extended to other patterns of MPEG where there are one or three B consecutive B frames. [35]

For normal scenes, bidirectional prediction is preferred in B frames. When there is a shot change, this prediction drops significantly. Based on the shot type either forward or backward motion compensation is used. The number of MB's for each frame is summed to obtain a measure. When a change occurs, the prediction direction of B frames is favored towards the anchor frames belonging to the same shot. As a second indication, if the third element of the sub-GOP is a P frames, there is an increase in the intra coded macroblocks for that frame. This can be seen as increase in No MC frames for P frames, where the only possible motion is in fwd direction.

Sub-GOP based analysis is done on the videos, where the sub-GOP consists of two successive B frames and the following I or P frame (B1-B2-I/P3). The decision table is given below. A minimum of %60 percentage of blocks is accepted as dominant block type.

Dominant Prediction Direction		*Cut Location*	
B1	*B2*		
Bwd	Bwd	Before B1	…(cut) B1 B2 P…
Fwd	Bwd	Between B1&B2	…B1 (cut) B2 P…
Fwd	Fwd	Before I or P	…B1 B2 (cut) P…

Table 3.4.1 Sub GOP based decision

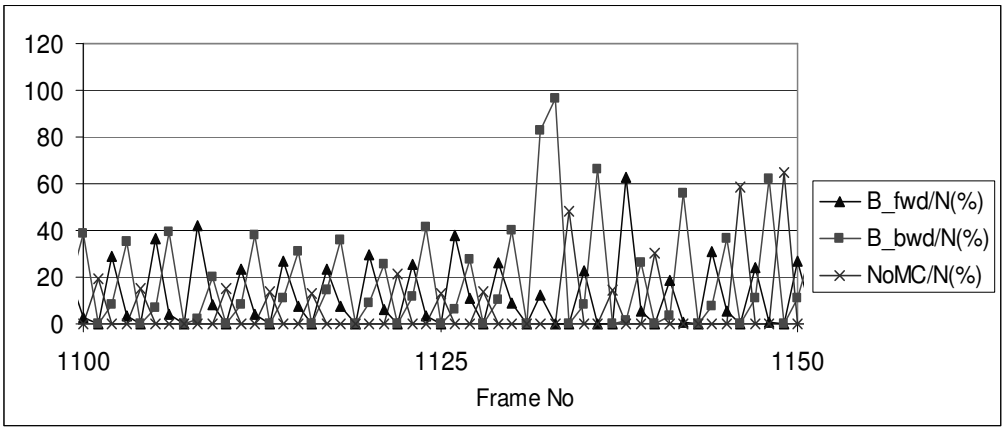

Figure-3.4.4 Percentage of Macroblocks for B and P frames of Sequence_news for the Bwd-Bwd Case

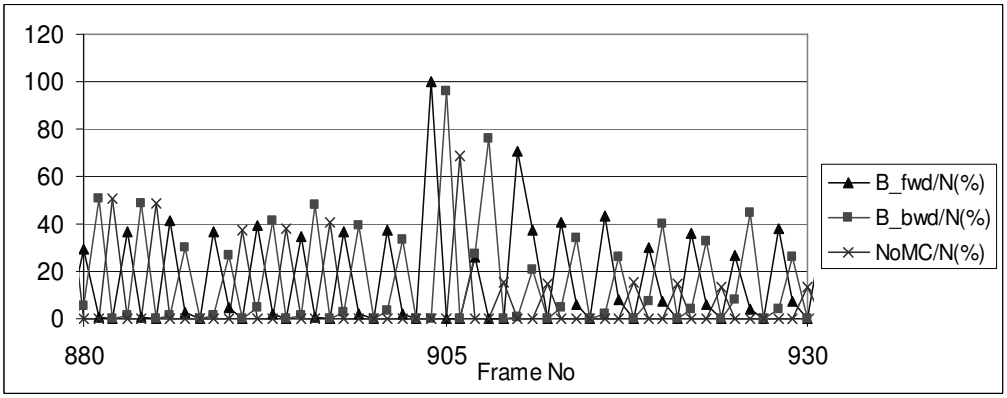

Figure-3.4.5. Percentage of Macroblocks for B and P frames of Sequence_news for the Fwd-Bwd Case

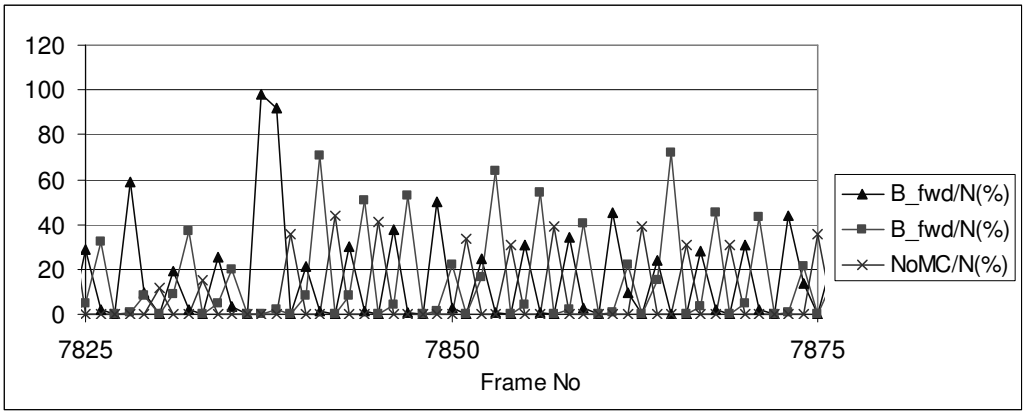

Figure-3.4.6. Percentage of Macroblocks for B and P frames of Sequence_news for the Fwd-Fwd Case

Counted macroblocks are divided into *N*= Total number of macroblocks for normalization. There can be cases where fwd or bwd blocks are greater with respect to each other, but negligible when compared to MB number, because majority of the blocks are bidirectional coded. This normalization also takes into account these cases and eliminates them.

The detection results are given for the news sequence:

	Detected	*Missed*	*False Alarm*
Cuts	64	2	8
Recall		*Precision*	
97.0 %		88.8 %	

Table 3.4.2 MB Type Algorithm's performance with news sequence

	Detected	*Missed*	*False Alarm*
Cuts	45	-	1
Recall		*Precision*	
100.0 %		97.8 %	

Table 3.4.3 MB Type Algorithm's performance with lottery sequence

	Detected	*Missed*	*False Alarm*
Cuts	39	10	2
Recall (Cuts)		*Precision (Cuts)*	
79.6 %		95.1 %	

Table 3.4.4 MB Type Algorithm's performance with mixed sequence

News sequence has a lot of flashlights which is a factor that decreases the precision rate. Some of the flashlights were eliminated because of the macroblock patterns. It is observed that a sequence with a flashlight resembles a cut in B motion compensation patterns. However, for a flashlight occurring in a B type; the other B type frame next to it has a large amount of interpolated type blocks. This number of interpolated blocks is even higher for the flashlights occurring in the first B frame. Since the algorithm evaluates macroblock counts with respect to the total macroblock size, they are eliminated. Therefore it is more resistant to false alarms than Bwd/Fwd and Fwd/Bwd ratio metrics. However, four of the false alarms due to flashlight effect are still detected false. Three of them occur at the second B frame, for which the interpolated block ratio is lower.

One of the missed scenes was a hard to detect cut, where almost 70-80% of the scene was in similar and uniform color for both shots. Therefore motion prediction at the encoder was done misleadingly in both directions for that frame.

The algorithm performs well with lottery sequence, detecting all of the cuts. For the mixed sequence, it has missed detections. Most of these come from black-white scenes from a video clip, which have very close shots which misleads the motion estimation algorithm.

For some dissolves this method is observed to detect multiple cuts during the dissolve, however this is not a reliable metric and it general it is not able to detect gradual transitions.

The macroblock based measurement is very effective and efficient for the cut transitions. It can detect the location of the cuts within frame based resolution. However as it cannot detect gradual transitions, it should be merged with other measurements for a complete detection.

3.5.DC Image Difference Based Results

DC image is the reduced version of the original image, which is composed of DC coefficients of the DCT. For I frames, DC coefficients are available after VLC decoding. However, obtaining DC coefficients of the P and B frames is computationally costly. Motion vectors have to be decoded first, and after that DC images should be reconstructed with motion compensation. Approximations can be done to obtain the DC images, but still the cost is high. In our implementation, DC images of I frames are considered only.

First DC coefficient in a frame is coded as it is, and the rest of the coefficients are coded differentially with respect to previous macroblock.

For each macroblock coded, there is 4 luminance blocks, and depending on the coding pattern 2, 4 or 8 chrominance macroblocks, which are coded in YUV color space. Luminance blocks for comparison contain the necessary intensity information for the scene; therefore luminance (Y) component of DC coefficient is used for comparison.

Sum of squared difference is used as a comparison metric. $d(X,Y)=\sum_{i,j}\left|x_{i,j}-y_{i,j}\right|^2$ is used, where $X=\{x_{i,j}\}$ and $Y=\{y_{i,j}\}$ are the luminance blocks of two consecutive DC images. This metric is found to give the best results, and have maximum divergence for DC image based comparisons [34]. $d(X,Y)$ is further divided to (N*4), where N is the total number of macroblocks, to normalize the difference.

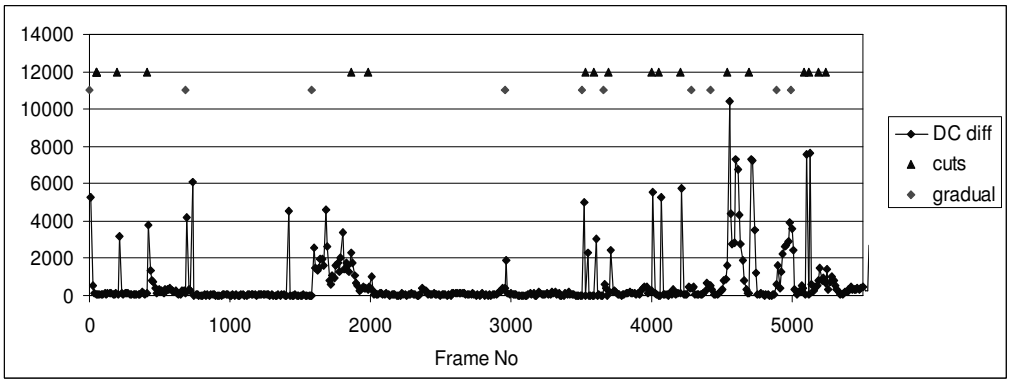

Figure-3.5.1. Squared DC Difference of I Frames Compared to Cut and Gradual Transition Locations for Sequence_mixed

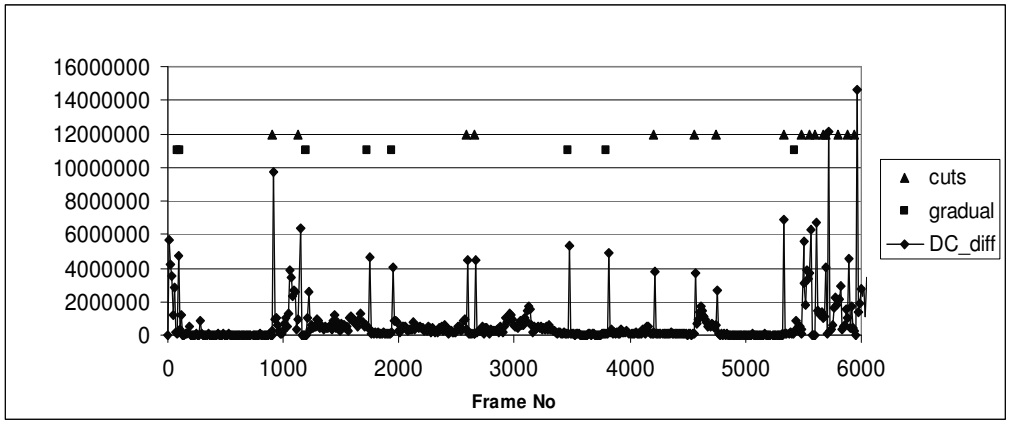

Figure-3.5.2. Squared DC Difference of I Frames Compared to Cut and Gradual Transition Locations for Sequence_news

	Detected	*Missed*	*False Alarm*
Cuts	65	1	11
Dissolve	10	1	N/A
Recall (Cut)		*Precision (Cut)*	
98.4 %		85.5 %	

Table 3.5.1 DC Difference Algorithm's performance with news sequence

	Detected	*Missed*	*False Alarm*
Cuts	35	5	10
Dissolve	2	1	N/A
Recall (Cut)		*Precision (Cut)*	
87.5 %		77.7 %	

Table 3.5.2 DC Difference Algorithm's performance with lottery sequence

	Detected	*Missed*	*False Alarm*
Cuts	49	8	19
Dissolve	6	3	N/A
Recall (Cut)		*Precision (Cut)*	
90.7 %		72.5 %	

Table 3.5.3 DC Difference Algorithm's performance with mixed sequence

Detection resolution is GOP, since only I frames are used for detection. Therefore cuts within one GOP cannot be detected. For a GOP size of 12, if the

video is 30 frames/sec, it will correspond to 0.4 seconds of resolution, which is acceptable for most of the cases. For the 24 frames long GOP, the detection resolution will be still less than one second. The shot distribution is analyzed for the test sequences and a few other videos, and it is observed that shot lengths within 10 frames constitute only %3 of the test videos.

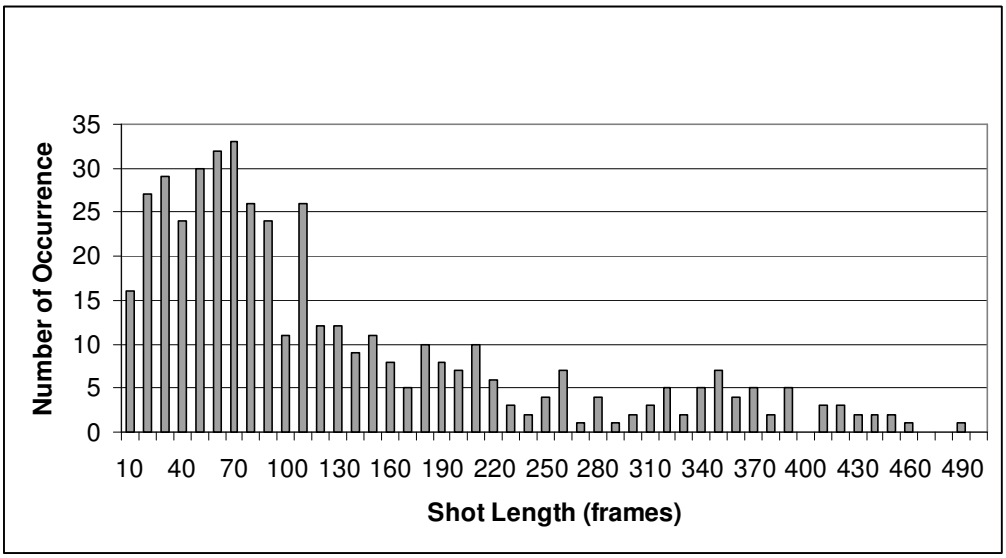

Figure-3.5.3. Shot Length Distribution for Several Videos

Also, a GOP resolution detection means the I frame of the next GOP's frame number is displayed instead of the actual GOP. However, this can be acceptable for most of the cases. GOP size will also affect the results. To detect cuts at a higher resolution, this algorithm should be combined with macroblock type based algorithm.

To detect the peaks in the DC difference, sliding window for automatic peak detection is used as explained in Section 3.3.1. Considering that luminance values

have 255 levels, a difference below %10, i.e. 25 is ignored. Therefore for the squared difference method, $25^2 = 625$ is used as the lower threshold.

Fast changing scenes lead to fluctuations in the graph. This is due to the fact that the content of the video may have changed considerably during one GOP.

Another remedy is to use the DC images of I frames as a coarse resolution, and use MB type to detect the exact location of cuts.

3.6.Gradual Detection Results

In previous sections it can be seen that most of the short dissolves can be detected with MB bitrate or DC difference based methods. The detection for longer transition time is analyzed in this section. Figure 3.6.1 shows the dissolve transition length obtained from several videos.

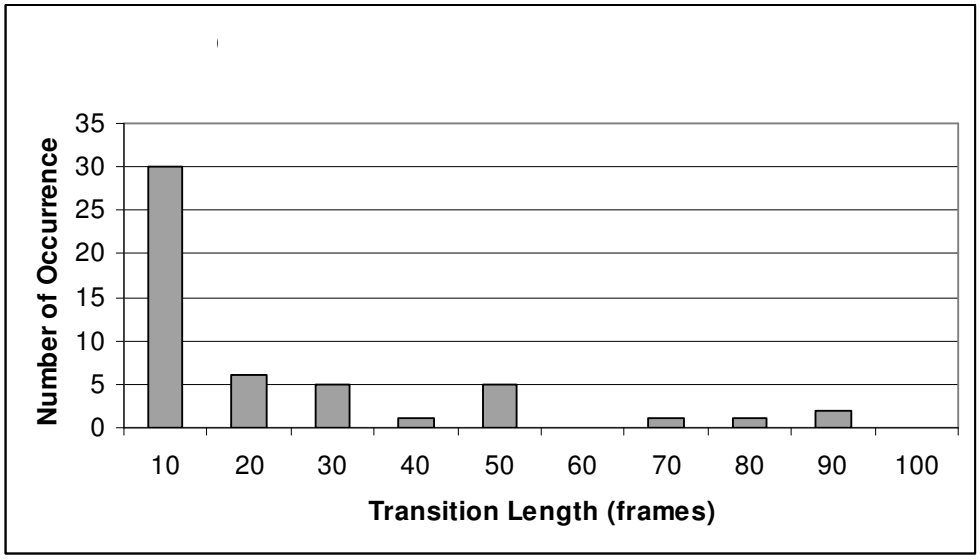

Figure-3.6.1. Gradual Transition Length Distribution for Several Videos

Although MB Type algorithms give the best results for cuts they fail to detect gradual transitions. This is due to the fact that the direction of MB's change slowly for the case of a gradual transition. Dissolve is the most commonly used editing effect, and fade-in and fade-out are considered as a subset of the dissolve.

In [36] it is proposed that the ratio FMBR = Fwd/(Fwd+ Bwd) follows a zigzag based pattern in dissolve regions, and a rather random pattern in non-dissolve regions. However for our test data a zigzag similar pattern is observed through the whole video, therefore didn't give informative results.

Intra blocks of P frames are also used for dissolve detection.[37] Dissolve regions cause an increase and a curve shaped region in the intra coded blocks. However, this behavior is also observed for scenes with high activity for our data. Besides, this curve shape was not very significant and characteristic in the test data; therefore it is not used for detection.

One useful outcome of using a rather coarse resolution for DC difference is that short dissolves can be detected this way. Dissolves around 10 frames lead to sharp peaks in the test data, which has GOP sizes of 9, 12, and 15. (Figure-3.6.1) These are detected same way the cuts are detected. This way only they cannot be classified as gradual transitions. However, since we assume that transition effect characterization as a different study, and focus on the video segmentation part only, this is accepted. Transition length for the test data is analyzed, and it is observed that most of the gradual transitions are within 10 frames in duration. However detection ratio is still low for this case, and it needs to be improved

Variance of I frames are also calculated for dissolve detection according to $d(X,Y)=\left(\sum_{i,j}\left|x_{i,j}-y_{i,j}\right|-\mu\right)^2$, where $X=\{x_{i,j}\}$ and $Y=\{y_{i,j}\}$ are the luminance blocks of two consecutive DC images and μ is the mean value of the DC image $X=\{x_{i,j}\}$.

There are two cases for dissolves, short dissolves within 1-2 GOP, or longer dissolves within, for example 5-6 GOP. To investigate ideal dissolve characteristics, artificial test data is formed, and two still images are combined with dissolve effect lengths 1 and 3 seconds respectively, and encoded.

For the short dissolve, a peak in the DC difference is observed similar to an abrupt cut. A linear rise or fall in the variance curve is observed for the variance.

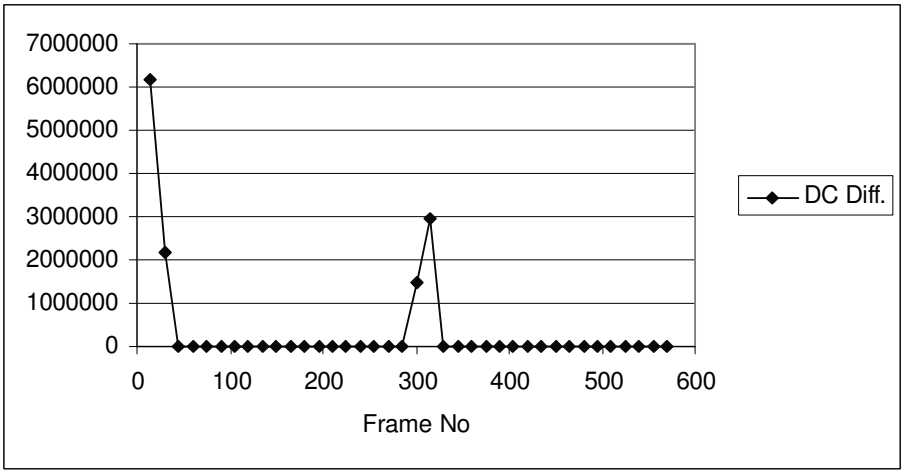

Figure-3.6.1. DC Difference of Short Dissolve Effect for the Artificial Sequence

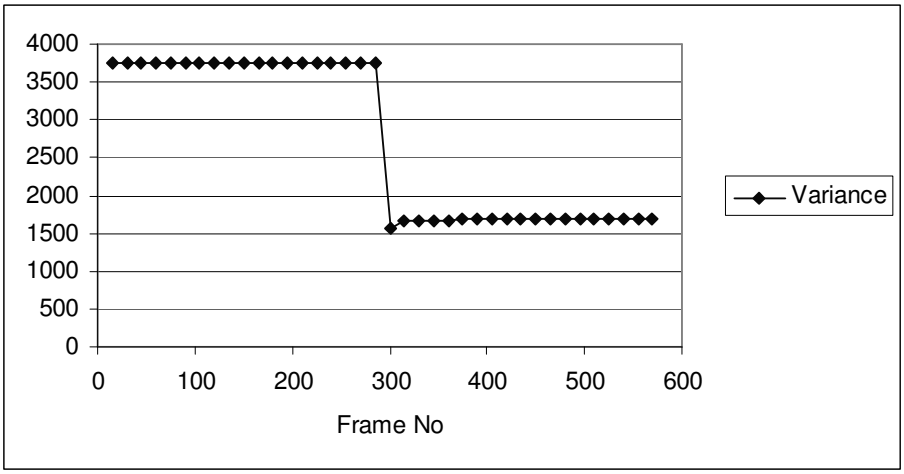

Figure-3.6.2. Variance of Short Dissolve Effect for the Artificial Sequence

For long dissolve sequences a curve is observed in the variance data. Second derivative of this curve has a zero crossing; therefore this information can be used for detecting the peaks. The DC difference during the dissolve has also a flat plateau.

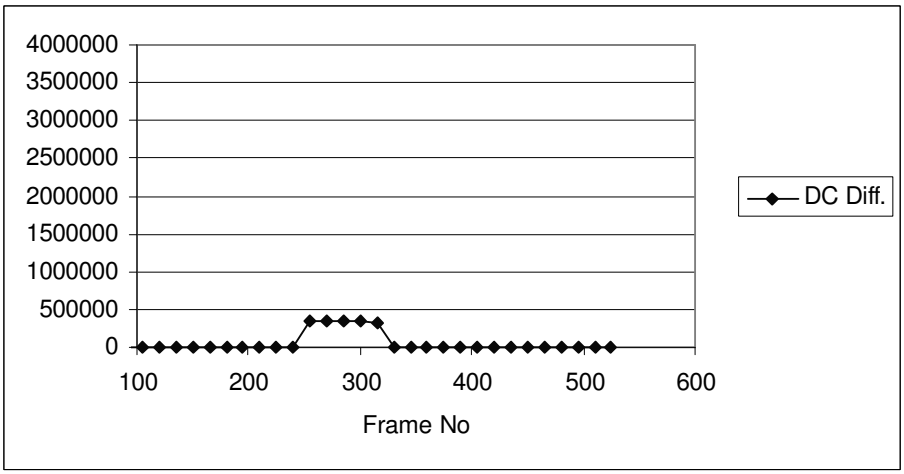

Figure-3.6.3. Variance of Long Dissolve Effect for the Artificial Sequence

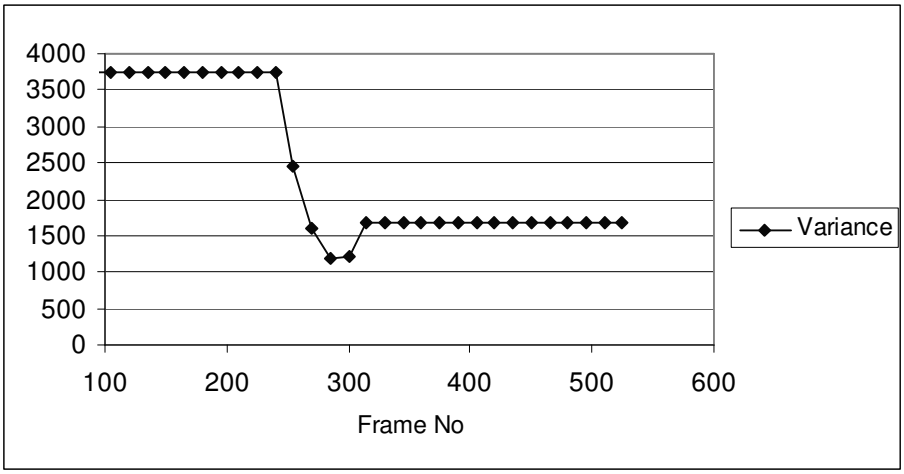

Figure-3.6.4. Variance of Long Dissolve Effect for the Artificial Sequence

For the Fade-out case, similar behavior is observed, except that one of the frames analyzed has a solid color, therefore the variance is zero in that region. Since this is the only way for a variance to become zero, this property can accurately be used in locating fades.

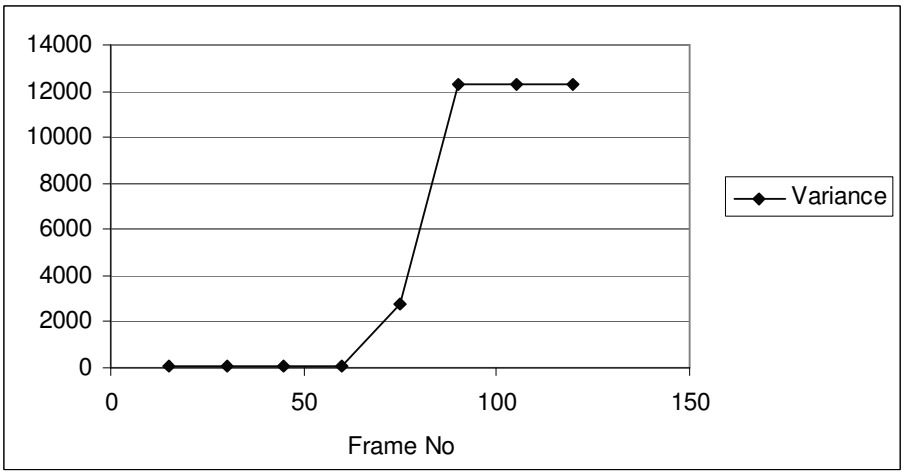

Figure-3.6.5. Variance of Fade-Out Effect for the Artificial Sequence

For the frames within one shot, variance level stays almost the same. For static scenes the difference is almost equal to zero. For faster scenes, some

variations are observed. In Figure 3.6.6, curves around 4300, 4400 and 4900 belong to a dissolve scene, whereas the others are the result of motion.

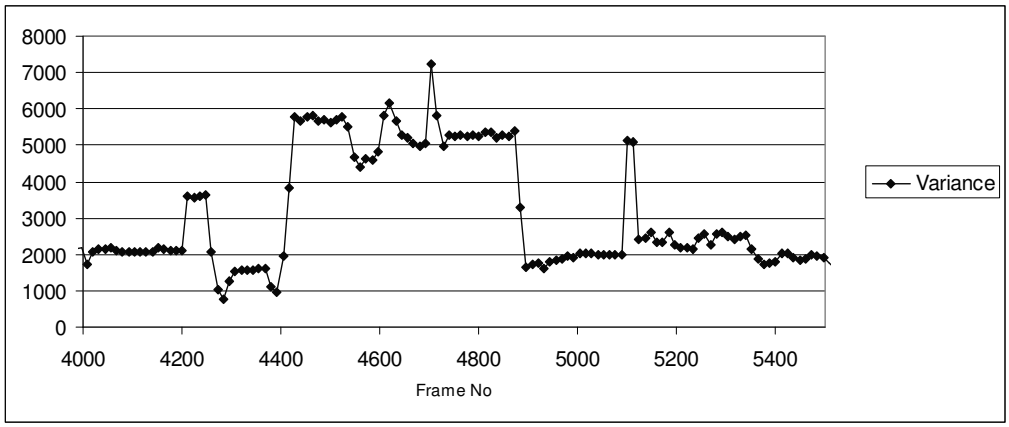

Figure-3.6.6. Variance of the Sequence_mixed

Since variance has a curve shape in gradual transitions, 2nd order derivative, which is approximated by a second order difference can be used. Zero crossings of the curve reveal the dissolve regions. However this metric is very sensitive to object and camera motion and it inserts many false positives. To reduce this effect, DC difference must be analyzed, which stays constant during the dissolve. A lower bound must also be set to reduce noise form zero crossing calculation.

3.7.Uncompressed Algorithm Results

Software which uses the Tekalp's histogram based algorithm [40] is obtained for comparison. This method combines histogram differencing on uncompressed frames, and pixel information uncompressed domain. Quantized versions of the frames are used to eliminate noise. The 2-D results are clustered to obtain the cut points. Finally false positives are eliminated through setting the

criterion that all the cuts must attain a local maximum in the histogram curve. Results are given in the table below.

	Detected	*Missed*	*False Alarm*
Cuts	61	5	9
Gradual	5	6	N/A
Recall (Cuts)		*Precision (Cuts)*	
92.4 %		87.1 %	

Table 3.7.1 Uncompressed Domain Algorithm's performance with news sequence

	Detected	*Missed*	*False Alarm*
Cuts	45	-	-
Gradual	-	-	N/A
Recall (Cuts)		*Precision (Cuts)*	
100.0 %		100.0 %	

Table 3.7.2 Uncompressed Domain Algorithm's performance with lottery sequence

False alarms are mainly observed due to flashlights in the news video, which are frequent.

3.8.General Comparison of Methods

In general, bitrate based methods are less reliable compared to DC and MB type based methods. Although they are very efficient, their robustness is low compared to other methods.

DC difference takes spatial features into account, and MB based difference takes motion based features into account. Therefore these features can be thought of complementing each other. Moreover DC difference method detects short dissolves, and MB type method gives a good accuracy for cuts. Their combined results should give higher recall and precision values.

	Detected	*Missed*	*False Alarm*	*Recall*	*Precision*
MB Difference	44	22	2	66.7 %	95.6 %
MB Type	64	2	8	97.0 %	88.8 %
DC Difference	60	6	20	90.9 %	75.0 %
Uncompressed Algorithm	61	5	9	92.4 %	87.1 %

Table 3.8.1 Comparison of Methods for Cut Detection with News Sequence

To investigate the relevance of algorithms for with MB Type and DC value methods, detected cuts are examined to determine which frames are common or different in detection. The results are given in Figure-3.8.1 and Figure-3.8.2.

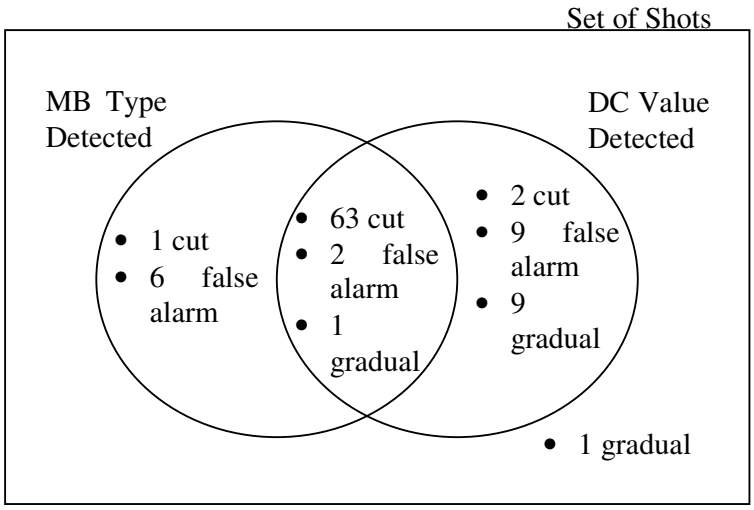

Figure-3.8.1. Relevance of Algorithms for the Sequence_news

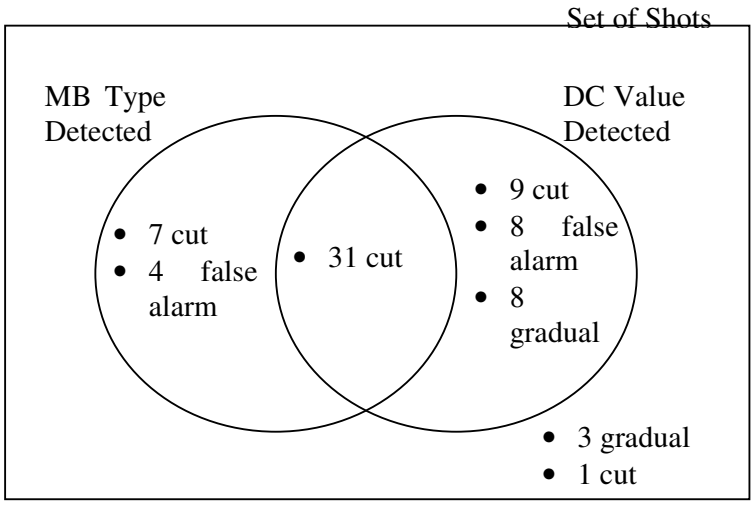

Figure-3.8.2. Relevance of Algorithms for the Sequence_mixed

It can be observed that most of the cuts can be detected both for MB Type and DC Value methods. However, false alarms for each method occur in different frames for different algorithms.

As a preliminary measurement, union of both methods is taken for comparison, whose results are given in Table 3.8.2 and Table 3.8.3. Recall value is increased more than the decrease in the precision. This is mainly due to detection of gradual changes within DC Value method.

	Recall % (Cut&Gradual)	*Recall % (Cut&Gradual)*	*Recall Change %*	*Precision Change %*
Uncompressed	85.7	87.1	+14.3	-5.4
MB Type	84.4	91.5	+15.6	-4.4
DC Value	97.4	87.2	+2.6	-0.1
MB+DC	100.0	81.7	–	–

Table 3.8.2 Union of Methods (OR logic) Compared to Other Methods for News Sequence

	Recall % (Cut&Gradual)	*Precision % (Cut&Gradual)*	*Recall Change %*	*Precision Change %*
MB Type	63.8	90.2	+29.3	-10.9
DC Value	81.8	83.0	+12.1	-3.7
MB+DC	93.1	79.3	–	–

Table 3.8.3 Union of Methods (OR logic) Compared to Other Methods for Mixed Sequence

CHAPTER 4

CONCLUSIONS

Efficient compressed domain video segmentation algorithms are implemented and their performances are evaluated. Results are compared to analyze advantages and disadvantages of the algorithms.

Frame bitrate method is the most efficient method. However, it is observed to be noisy. If two different scenes have similar bitrate, in other words, encoding cost, missed detections occur. This method is also the most sensitive method to motion. Only I frames are observed to contain reliable information.

Macroblock bitrate based method, which is proportional to AC coefficients of the image, gives improved results as compared to frame based bitrate methods. Different from MB type based method, it also detects short dissolves, since I frames are used in detection. However it has a high number of false positives and should be combined with other methods to obtain good precision.

Macroblock type based method is found to show good performance for cut detection. It has the same computational cost with MB bitrate based methods, since both of them necessitate MB header decompression. And since it depends on the motion compensation values, it is more resistant to illumination changes such as flashlights. It identifies the exact cut locations on a frame-based resolution. The disadvantage of this method is that it fails to detect gradual transitions.

DC Image difference methods also have good performance in segmentation. Effects of using only I frames has two sides. First, it can detect cuts at GOP-based resolution. Therefore shots within one GOP, which is generally unlikely, can be detected as a single cut. Also for scenes with high motion content, it may lead to false alarms if the scene content has changed considerably during one GOP. Second, it has the advantage of locating short dissolves, which are frequently used, without any further calculation. This method is more sensitive to illumination changes than the MB type based method.

Examining the detection set for the algorithms show that false detections occur in different frames for different methods. The recall ratio is increased more than the decrease in precision value when the union of the MB Type and DC Value algorithms is considered.

Comparison with uncompressed algorithms has shown that while being the most computationally effective, even in the group of uncompressed methods, these methods give similar results with uncompressed algorithms. However, compared to uncompressed domain algorithms these methods' have great computational efficiency, as they avoid decompression phase.

As a future work, P frames can be included in the DC image detection process, and computation requirements versus improvements should be analyzed. Also more effective results for gradual transitions should be investigated.

REFERENCES

[1] B. G. Haskell, A. Puri and A. N. Netravali, *Digital Video: An Introduction to MPEG-2,* Chapman& Hall, 1997

[2] A. M. Tekalp, *Digital Video Processing*, Prentice Hall, 1995

[3] C. Taşkıran, J. Chen, C. A. Bouman and E. J. Delp, *A Compressed Video Database Structured for Active Browsing and Search*, ICIP 1998

[4] B. Furht and P. Saksobhavivat, *A Fast Content-Based Multimedia Retrieval Technique Using Compressed Data,* www.cse.fau.edu/~borko/paper_SPIE-1.pdf

[5] Y. S. Avrithis, A. D. Doulamis, N. D. Doulamis, and S. D. Kollias, *A Stochastic Framework for Optimal Key Frame Extraction from MPEG Video Databases,* Computer Vision and Image Understanding, Vol. 75, No. 1/2, July/August, pp. 3–24, 1999

[6] O. Kao and G. R. Joubert, *A Content Based Internet Search Engine For Analysis And Archival of MPEG-1 Compressed Newsfeeds*, IEEE International Conference on Multimedia and Expo,2000

[7] T. Heath, T. Howlett, and J. Keller, *Automatic Video Segmentation in the Compressed Domain,* IEEE Aerospace Conference, 2002

[8] G. Boccignone, M. De Santo, G. Percannella, *An algorithm for video cut detection in MPEG sequences*, Proc. SPIE Conference on Storage and Retrieval of Media Databases 2000, pp. 523-530, Jan. 2000, San Jose, CA.

[9] B. C. Smith*, A Survey of Compressed Domain Processing Techniques*, NSF Workshop on Reconnecting Science and Humanities in Digital Libraries, University of Kentucky, October 1995.

[10] S.-W. Lee; Y.-M. Kim; S. W. Choi, *Fast Scene Change Detection Using Direct Feature Extraction From MPEG Compressed Videos,* International Conference on Pattern Recognition (ICPR'00)-Volume 3 September 03 - 08, 2000

[11] B.-L. Yeo and B. Liu*., Rapid Scene Analysis on Compressed Video,* IEEE Transactions on Circuits and Systems for Video Technology, Vol. 5, No. 6, pp. 533-544, December 1995.

[12] Fernando, W.A.C.; Canagarajah, C.N.; Bull, D.R., *A Unified Approach To Scene Change Detection In Uncompressed And Compressed Video,* IEEE Transactions on Consumer Electronics, Vol. 46, No. 3, pp. 769-779, Aug. 2000.

[13] F. Arman, A. Hsu, and M. Y. Chiu, *Image Processing On Compressed Data For Large Video Databases, Proc. of ACM Multimedia*, pp. 267-272, 1993.

[14] X. U. Cabedo and S. K. Bhattacharjee, *Shot Detection Tools in Digital Video*, In Proc. of Non-linear Model Based Image Analysis 1998, Springer Verlag, Glasgow, pp. 121–126, July 1998

[15] K. Shen, E.J. Delp, *A Fast Algorithm for Video Parsing Using MPEG Compressed Sequences,* Proc. of the IEEE Int. Conf. On Image Processing, 252-255, Oct.,1995

[16] T. Shin, J.-G. Kim, J. Kim, B.-H. Ahn, *A Statistical Approach To Shot Boundary Detection In An MPEG-2 Compressed Video Sequence*, Visual Communications and Image Processing 2000

[17] R.Zabih, J.Miller, K.Mai, *A Feature-Based Algorithm for Detecting and Classifying Production Effects*, ACM Journal of Multimedia Systems, Vol. 7, pp. 119–128, 1999.

[18] B. Shen, D. Li, and I. K. Sethi, *HDH Based Compressed Video Cut Detection*, Proc. of Visual 97, pp. 149-156, San Diego, CA, December 1997.

[19] B. Shahraray, *Scene Change Detection and Content-Based Sampling of Video Sequences*, Proc. IS&T/SPIE, In Digital Video Compression: Algorithms and Technologies, Vol. SPIE 2419, pp. 2-13, 1995.

[20] M. Sugano, Y. Nakajima, H. Yanagihara, A. Yoneyama, *A Fast Scene Change Detection On Mpeg Coding Parameter Domain*, IEEE Proceedings of International Conference on Image Processing, 1998, pp.888-892

[21] TCT. Kuo, A. Chen, *A Mask Matching Approach For Video Segmentation On Compressed Data,* Information Sciences, 2002, 141(1-2): 169-191

[22] J. Nang, S. Hong, Y. Ihm, *An Efficient Video Segmentation Scheme for MPEG Video Stream Using Macroblock Information*, 7. ACM Multimedia 1999, Vol. 1, 23-26

[23] S.-C. Pei, Y.-Z. Chou, *Efficient MPEG Compressed Video Analysis Using Macroblock Type Information*, IEEE Transactions On Multimedia, Vol. 1, No. 4, December 1999

[24] J. Calic and E. Izquierdo, *Towards Real-Time Shot Detection In The Mpeg-Compressed Domain,* Proc. Workshop on Image Analysis for Multimedia Interactive Services, WIAMIS'2001, pp 95-100

[25] R. Milanese, F. Degulliaume, A. Descombes, *Video Segmentation and Camera Motion Characterization Using Compressed Data,* SPIE Proceedings on Multimedia Storage and Archiving Systems II, Vol. 3229, November 1997.

[26] G. Bozdagi, T. Sencar, *Preprocessing Tool for Compressed Video Editing*, IEEE Int. Workshop on Multimedia Signal Processing, Sept. 1999, Denmark.

[27] J. Feng, K. T. Lo, and H. Mehrpour, *Scene Change Detection Algorithm For MPEG Video Sequence,* Proc. IEEE International Conference on Image Processing, 1996

[28] G. Falkemeier, G.R. Joubert, O. Kao, *A System for Analysis and Presentation of MPEG Compressed Newsfeeds*, in J-Y. Roger, B. Stanford-Smith, P. T. Kidd: Business and Work in the Information Society: New Technologies and Applications, 1999, pp 454-460, IOS Press

[29] V. Kobla, D. Doerman, K. Lin, *Archiving, Indexing, and Retrieval of Video in the Compressed Domain,* Proceeding of SPIE Conference on Storage and

Retrieval for Still Image and Video Databases, SPIE Vol. 2916, pp. 78-89, 1996

[30] J. Meng, Y. Juan and S. F. Chang, *Scene Change Detection in a MPEG Compressed Video Sequence*, Proceeding of SPIE Conference on Digital Video Compression: Algorithms and Technologies, SPIE Vol. 2419, pp. 14-25, 1995.

[31] H.C. Liu and G.L. Zick, *Scene Decomposition of MPEG Compressed Video*, Proceeding of SPIE, Vol. 2419, 1995.

[32] W. Xiong, J. C. M. Lee, and R. H. Ma, *Automatic Video Data Structuring Through Shot Partitioning And Key Frame Selection*, Machine Vision and Application, 10 (1997), pp. 51-65.

[33] W.J. Heng, K.N. Ngan, and M.H. Lee, *Comparison of MPEG Domain Elements for Low-Level Shot Boundary Detection,* Journal of Real-Time Imaging, Acad. Press, USA, 99 pp341-358

[34] J. Bescos, A. Movilla, J.M. Mendez, and G. Cisneros, *Real Time Temporal Segmentation of MPEG Video*, IEEE International Conference on Image Processing, ICIP'2002, vol II, Rochester, pp 401-404, Sep. 2002.

[35] A. M. Dawood, M. Ghanbari, *Clear Scene Cut Detection Directly From MPEG Bitstream*, IEEE Seventh International Conference on Image Processing and Its Applications, 1999., Conf. Publ. No. 465 , Vl: 1 , pp: 285 -289 vol.1

[36] S.-B. Jun, K. Yoon and H.-Y. Lee, *Dissolve Transition Detection Algorithm Using Spatio-Temporal Distribution of MPEG Macro-Block Types*, Proceedings of the eighth ACM international conference on Multimedia, 2000, California, United States pp: 391 - 394,

[37] W.A.C. Fernando, C.N. Canagarajah, D.R.Bull, *Sudden Scene Change Detection In MPEG-2 Video Sequence*, IEEE Signal Processing Society, 1999 Workshop on Multimedia Signal Processing September 13-15, 1999, Copenhagen, Denmark

[38] S.M. Bhandarkar and A.A. Khombadia, *Motion-Based Parsing of Compressed Video* Proc IEEE Intl. Wkshp. Multimedia Database Mgmt. Sys., Dayton, Ohio, August 5-7, 1998, pp 80-87.

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

[40] M.R. Naphade, R. Mehrotra, A.M. Ferman, J. Warnick, T.S. Huang, T.S, A. M. Tekalp, A High-Performance Shot Boundary Detection Algorithm Using Multiple Cues, IEEE Proceedings, International Conference Image Processing, Volume: 1 , 4-7 Oct. 1998 Page(s): 884 -887

APPENDIX A

GROUND TRUTH DATA for TEST VIDEOS

A. News Sequence

Transition Start	Transition End
77	87
96	106
-	905
-	1132
1196	1201
1723	1727
1933	1935
-	2587
-	2657
3456	3458
3796	3797
-	4208
-	4550
-	4738
-	5325
5406	5427
-	5488
-	5548
-	5596
-	5671
-	5699
-	5803
-	5880
-	5947
-	6120
-	6163
-	6219
-	6302
-	6363
-	6471
-	6615
-	6674
-	6775
-	6809
-	6893
7013	6919
-	7019
7663	7421
-	7669
-	7839
-	8723
-	8813
-	10210
-	10650
-	11571
-	11622
-	12337
-	13113
-	15815
-	16301
-	16555
-	17293
-	17369
-	17662
-	17844
-	17936
-	18025
-	18075
-	18501
-	18576
-	18628
-	18649
-	19195
-	19216
-	19304
19693	19483
-	19700
-	19814
-	20053
-	20074
-	20250
-	20595
-	20616
-	20717
-	20774
-	21155
-	21528

B. Mixed Sequence

Transition Start	Transition End
-	196
-	408
680	682
1576	1577
-	1859
-	1983
3495	3502
-	3530
-	3586
3645	3655
-	3693
-	3997
-	4048
-	4201
4244	4282
4366	4412
-	4534
-	4689
4868	4888
-	5080
-	5113
-	5180
-	5231
5337	5395
-	5534
-	5602
-	5685
-	5711
5883	5892
5990	6000
-	7322
7994	8022
-	8185
-	8251
-	8339
-	8403
-	8474
-	8550
-	8618
-	8719
-	8758
-	8810
-	8912
-	9015
-	9073
-	9119
-	9209
-	9254
-	9376
-	9458
-	9810
-	9858
-	9911
-	9933
-	9991
-	10035
-	10190
-	10221

C. Lottery Sequence

Transition Start	Transition End
129	136
197	207
-	455
-	542
-	901
-	993
-	2056
-	2280
-	2321
-	2529
-	2634
-	3019
-	4222
-	4528
-	4610
-	4741
-	4948
-	5053
-	5680
-	5993
-	6048
-	6248
-	6396
-	6741
-	6897
-	7286
-	7601
-	7853
-	7976
-	8414
-	8738
-	9324
-	9407
-	9469
-	9828
-	10192
-	10298
-	10379
-	10474
-	11770
-	12019
-	12073
-	12393
-	12583
-	12585
-	12655
14815	14903
-	15123