A NOVEL FRAMEWORK OF SHOT DETECTION FOR UNCOMPRESSED SPORTS VIDEOS

By

Abdul Hameed

A thesis submitted to the faculty of Computer Engineering Department College of E&ME, National University of Sciences and Technology, Pakistan in partial fulfillment of the requirements for the degree of MS in Computer Software Engineering

October 08

STARTING WITH THE NAME OF ALMIGTHY ALLAH WHO IS MOST GRACIOUS AND MERCIFUL

Say I seek refuge in the Lord of mankind,

The king of mankind, The God of mankind,
From the evil of the sneaking whisperer,
Who whispereth in the hearts of mankind,
Of the jinn and of mankind
(Surah-Alnnas)

And say my Lord have mercy on

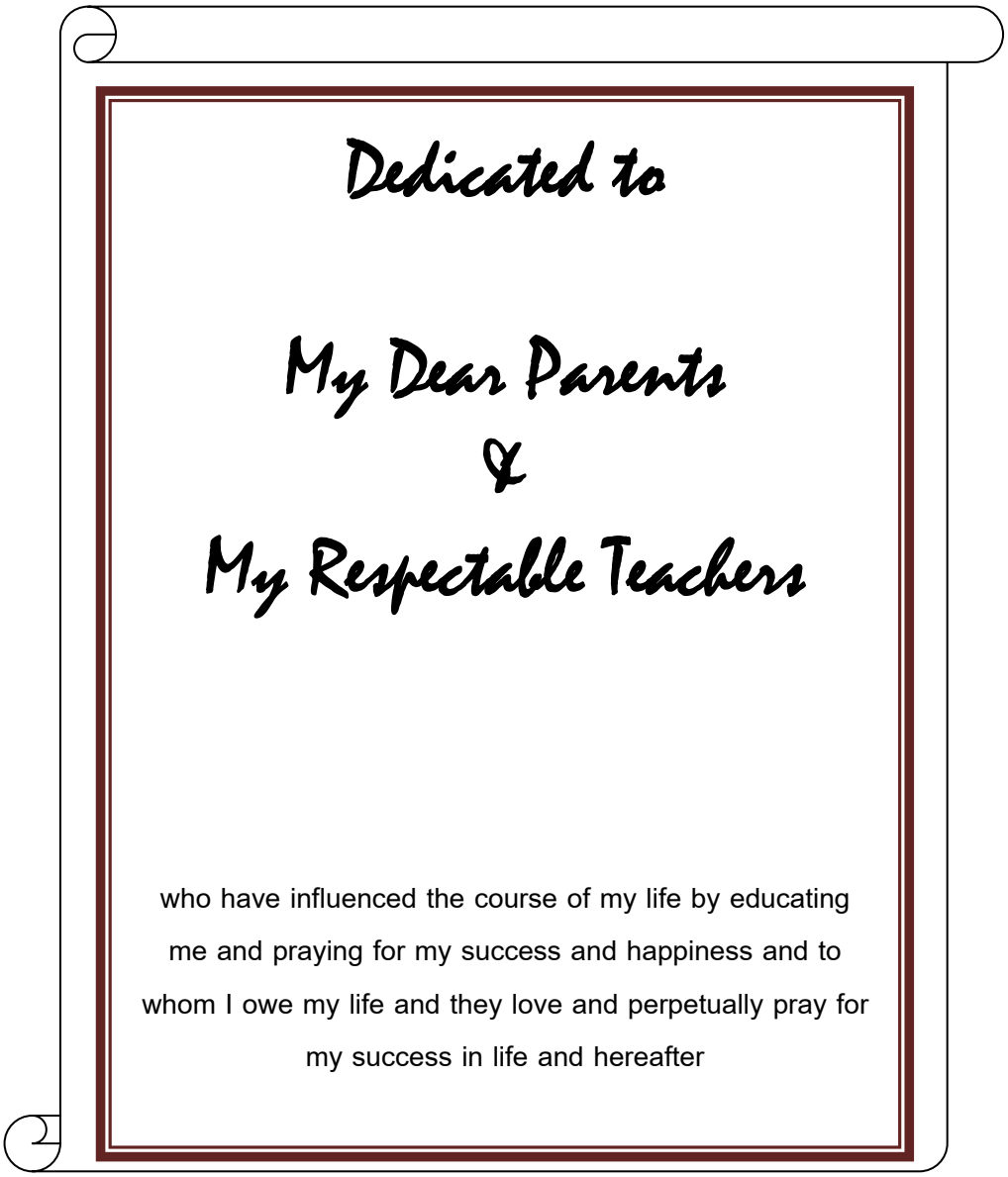
them (Parents) both, as

they did care for me when I was little.

(Surah-Israel; Ayat 24)

TABLE OF CONTENTS

DEDICATION


Dedicated to
My Dear Parents
&
My Respectable Teachers
who have influenced the course of my life by educating me and praying for my success and happiness and to whom I owe my life and they love and perpetually pray for my success in life and hereafter

ACKNOWLEDGEMENT

Up and above everything, we are grateful to almighty ALLAH, the beneficent, the merciful, and His Holy Prophet Hazrat MUHAMMAD (peace be upon him) who is forever a torch of guidance for whole humanity. All praise and thanks to Almighty Allah who has showered me with His invaluable blessings throughout my life, given me strength and spirit to complete this research work.

I was truly blessed by being surrounded by extremely intelligent and supportive people. It would never have done it without them. I am especially indebted to my project supervisor Dr. Shoab Khan, College of E&ME, NUST for giving me remarkable suggestions and constant encouragement to complete this project efficiently.

I would deeply like to thank my affectionate parents for their overwhelming support and prayers throughout the project. No words to describe my feelings of respect about my respectable teachers and loving brothers and sister whose constant prayers, love, encouragement, praise and assistance invariably buoyed me up.

My acknowledgement would be incomplete if I don't offer my special thanks to my friends for their constant cooperation and encouragement during this project. I also express my deepest appreciation to my colleagues whose unfeigned help and encouragement made the present work a reality.

I would once again like to thank all faculty members, who have always been a source of inspiration, for their cooperation and healthy academic environment thought my career at College of E&ME, Rawalpindi. My special thanks to college staff for their kind cooperation and encouragement.

Abdul Hameed

ABSTRACT

In outsized multimedia databases video segmentation is a fundamental constituent necessary to assist proficient content based retrieval and browsing of visual information. Shot boundary detection is the key temporal video segmentation task, as it is inherently linked to the production of video therefore it becomes a natural choice for segmenting a video into more manageable parts. Since scene changes almost always happen on a shot change, shot boundary detection is indispensable as a first step for scene boundary detection that accomplish other video analysis tasks.

This research presents work towards an integrated framework for computerized video shot detection. Generally, the automatic segmentation of an uncompressed video into story line ranges from very difficult to inflexible. At the other end, automatic video segmentation into constituent shots is both precisely and exactly defined and differentiated by characteristic features of the video stream by itself. In produced videos such as television and films depicting different events, shots are separated by different types of transitions, or boundaries due to different types of camera movements. Therefore identifying a shot is preferred for subdividing a video into more manageable segments. The detection of a shot change between two adjacent frames of a video requires the computation of appropriate statistical measures with their thresholds between adjacent frames of the video.

A test program in MATLAB has been developed. The overall results show that the proposed solution gives the optimal performance when it is compared with the other competitor techniques that lie in the same spatial and temporal domains. Experiment justification of the purposed framework shows the versatility and significance of the results obtained in the work.

Chapter 1

Introduction

1.1 Introduction to video shot detection

The growth of video databases is considerable in recent years due to remarkable development in recent technology, network mechanisms, and ease of use of large storage systems. The press on in digital video technology and ever mounting availability of computing outcomes have resulted in the last few years in an outburst of digital video data, especially on the internet and for the storage devices. Besides, with the arrival of digital TV broadcast with plentiful channels, it is required to organize and catalog the video data automatically, so that the end users can pick and choose the desired data with no complexity.

However, the increasing accessibility of digital video has not been escorted by an increase in its ease of access. This is due to the characteristics of video data, which is inapt for conventional form of data access. Therefore procedures have been hunted to systematize video data into more compact types or extract semantically important information [1]. These procedures can serve as a first step for a number of different data access tasks at a later stage.

Now a day, automatic classification of video content is receiving increased impact in the multimedia information processing. Detecting shots and kind of transitions present between them is really useful in investigating intra and inter shot association for high level video understanding. A shot is defined to be a sequence of frames that was (appears to be) continuously captured by the same camera and represents continuous action in time and space [2]. An important aspect of video indexing is the ability to divide video into these meaningful segments called shots.

Shot boundary detection is the most vital temporal video segmentation task, as it is inherently linked to the method that video is produced. In produced videos such as television and movies, shots are separated by different types of transitions, or boundaries. Therefore it is a likely choice for segmentation a video into more manageable divisions. Since a scene changes in a movie is directly proportional to shot change, therefore shot boundary detection is necessary as a first step for scene boundary detection. It is often the first step in algorithms that accomplish other video analysis tasks as well.

The detection of a shot change between two adjacent frames requires the computation of appropriate measures and is typically found by computing an image based distance between adjacent frames of the video, and make a note of when this distance exceeds a certain threshold. The distance between adjacent frames can be based on statistical properties of pixels [3], compression algorithms [4], or edge differences [5]. The histogram difference is the most widely used method [6]. In order to improve shot boundary detection audio and motion features have been used [7-9]. Some techniques also combine motion features with pixel differences [10].

Most of the shot detection algorithms reduce the large dimensionality of video domain by extracting a number of features in each video frame. Such features include averages or other statistical measures on luminance or color components of video frames. Histogram measures either for luminance or chrominance is also being used due to its ease and its insensitivity towards different types of camera motions [6], [11]. Beside histogram measures, edge information is also an obvious choice for characterization a video frame [12-13]. Transform coefficients such as Discrete Fourier Transform, Discrete Cosine Transform and wavelets are also used to extract image information. Motion information coupled with other measures described earlier is also used as a feature to detect shot transitions in a video sequence.

1.2 Motivation

With the enormous press on in digital technologies, especially in telecommunications and networks, a lot of audiovisual information are created and accessed by many end users through different types of media such as digital television, storage devices and internet. This advent of digital video and multimedia technologies has exploded raw video collection and most of them are oddly tidy up and never touched or watched again. Therefore the availability of such information has not been followed by a pull in its accessibility.

As a result a dilemma particularly in indexing and annotating large quantities video material is becoming critical as far as archived material is concerned. The labor-intensive indexing is currently the most accurate method but it is a very arduous and time consuming. To index material an individual has to view many hours of raw video sequentially to locate shot boundaries and textually interpret each individual shot. This amplifying popularity of multimedia applications entitle for the development of image and video processing methods for valuable distribution and representation of the visual information to provide novel video services such as interactivity, effective indexing, browsing, searching, manipulation, annotating, editing, retrieval, management, content-based access and scalability of visual information at many different levels.

The key development of the last few years is the increasing accessibility of on-line catalogs. However the predominant approach to computerize the video indexing process is to construct a video abstract. Therefore a system which is capable of abstracting raw video into shorter meaningful video automatically would thus be very beneficial for end users. These video abstracts will able to satisfy the individual time-constraints. A plethora of literature on video abstracting addresses the image tracking alone and extracting different key frames in panorama images only. Therefore a system is to be

needed which demand that sports video abstracts from raw video available to spectators. An effort has been made towards developing of such a system which would be able to segment the video into its constituent shots.

1.3 Problem statement

Efficient and effective shot boundary detection for uncompressed sports videos using different statistical metrics, expressing the similarity or dissimilarity of features computed on adjacent frames against a fixed threshold. This is for the video content which exhibits similar characteristics over time and only if the threshold is not manually adjusted for each type of video sequence.

A study is to be made which will explore these statistical measures such as correlation between the adjacent frames, computation of histogram difference followed by running average difference as the classifiers. A framework for extracting shot detection by using the threshold values of diverse statistical features for raw video frames will be structured. Two different types of uncompressed sports videos viz. Soccer and basketball are to be used for assessment on this framework. Experimental results on divergent set of test videos will reveal the effectiveness of this shot detection approach.

1.4 Aims and objectives

The primary objective of this thesis is the exploration, study of literature and analysis of prior image processing tools and techniques available which is suitable for detecting shots using different features for sports videos both in spatial as well as in temporal domain. The secondary objectives include analysis and design of the new algorithms for shot detection in sports videos and development of a test program for testing the performance of the designed algorithm and finally detailed analysis of the obtained results in terms of accuracy and success rate. This work aims to detect hard cuts of shot transitions. Another objective for this thesis is to explore different uncompressed and

compressed file formats for video and how the distribution of pixels is present in different proportions in different chromo format of uncompressed YUV file formats and how it is encoded in compressed MPEG file format for the storage on DVDs.

1.5 Thesis organization

Rest of the thesis is organized as follows. Chapter 2 gives a little background followed by an in depth review of the different shot detection techniques used in video segmentation process. It also focuses on different categories of shot change and types of camera change followed by mechanisms and types of shot detection algorithms mainly consulted and addressed by different research papers. Chapter 3 gives the formal definition of the problem in hand; solutions to the task specific problems, conditions and assumptions. This section also includes major design goals and requirements for the projects. These requirements are elicited in different categories to further emphasize the intricacies of the problem at hand. Chapter 4 illustrates the detailed design of framework being purposed and the problem decompositions into different modules of the framework. It also further explains how these modules are finally integrated to form a complete test program. Chapter 5 gives an in depth analysis of the results obtained during the experimentation and verifies the algorithm design and robustness. It also spotlights on the details of the experiments conducted using test data. Lastly, chapter 6 concludes the research and highlights the future work, which can be done to carry forward this effort.

Chapter 2

Background and Literature Review

2.1 Introduction

The concept of video segmentation is not a new one as it was intrinsically used in the production of videos well before computers was even invented. Video specialists distribute their works into a hierarchy of partitions for their ease. A video is a combination of complete and disjoint pieces put together into a sequence of scenes or story units which are subsequently segmented into more manageable parts at a later stage.

The purpose of this chapter is to present a concise review of previous work relating to the problems of video segmentation, shot cut detection and gradual transition detection in a video sequence. A number of algorithms have been devised and proposed to process uncompressed video content. This chapter at first introduces the concepts relating to video and its production such as video shot followed by different types of transitions and camera movements. This chapter also take account of the mechanism of shot boundary detection algorithms and includes the summary of existing techniques present in literature for video shot detection that operate on uncompressed video stream. The performance, traits, characteristics and limitations of each of them are briefly discussed and contrasted. This review also stimulates and presents a platform for the work currently in progress.

2.2 Video abstract

The key development of the last few years is the increasing accessibility of on-line catalogs. However the predominant approach to computerize the video process is to construct a video abstract. A video abstract is defined as a shorter sequence of frames extracted from a larger video sequence, yet conserving its important message [14]. A

video abstract can be used to catalog unindexed video, as browsing a shorter segment of video is quicker than watching a whole episode. Moreover, once a suitable video shot has been identified the frames in the video abstract can be used to retrieve similar shots [15-18].

The potential problem is to decide which frames best represent the video contents in composing such an abstract. The answer to this problem often depends on the background in which such an abstract is being employed. An approach in which the production of trailers for movies which results in video abstracts without exposing and describing too much of the story was presented in [14]. In contrast, another approach in which an abstract for a documentary or a digital video library attempt to represent all the video content is presented in [19]. For a video abstract to be a competent index, each key frame should symbolize a video segment where there is little or no considerable change in the scene content. This enables a video abstract to confine and summarize the content of the sequence at the same time removing the visual content spatial and temporal redundancy amongst video frames [20].

2.3 Video shot

The automatic video shot detection is receiving a great impact with the advances in the digital video technology and ever increasing accessibility of computing results. Traditionally, a video is entirely composed of a sequence of scenes, which are subsequently composed of a sequence of shots. There exists a hierarchical composition within a video sequence, as illustrated in fig. 2.1. The lowest level constitutes of a set of individual frames. At the next level frames are grouped together to form *video shots*, which is defined as a sequence of frames that are captured continuously from the same camera operation.

In literature video shots are also defined as the longest continuous sequence that originates from a single camera at a time. It is what the camera images in a continuous uninterrupted run. Shots combined by a common environment, surrounding or event are then

grouped together into *scenes*. A story unit or a scene is a continuous sequence that is spatially as well as temporally interconnected and thus conveying some meaning. These scenes then complete the video sequence. Once the video sequence has been broken down into a set of meaningful and manageable subdivisions (shots), work can be done to distinguish the individual components for indexing and annotation. Therefore, segmentation of a video sequence is typically the first step towards automatic annotation of digital video sequences [21-22].

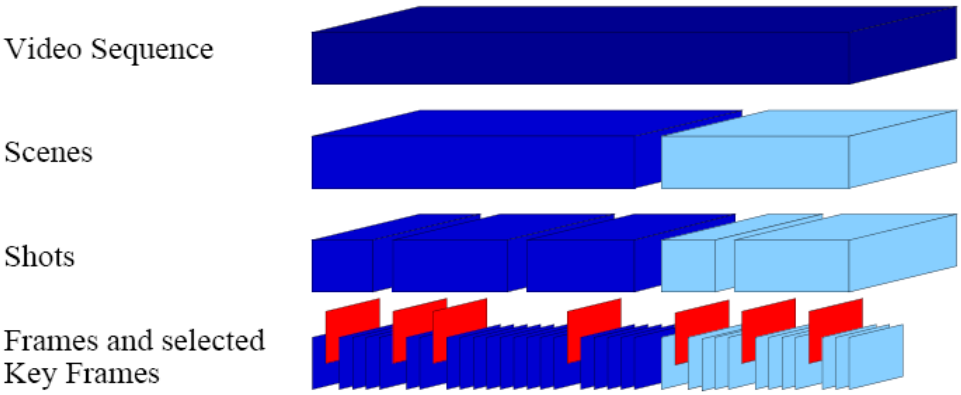


Figure 2.1: Hierarchical structure within the video sequence

The detection of boundaries between different video shots is the initial step which realizes video analysis tasks and provides a basis for all the existing video segmentation methods [23]. Once the video sequence has been broken down into a set of important, meaningful and convenient video shots, further work can be done to distinguish the individual components for indexing and annotation. Therefore, temporal sequential segmentation is typically the first step towards automatic annotation of digital video sequences [24], [25]. Since scene changes almost occur on a shot change, shot boundary detection is crucial as a primary step for scene boundary detection. Shot boundary detection is the most basic temporal video segmentation task, as it is inherently and intrinsically linked to the way to the production of video. Therefore it is a natural choice for segmenting a video into more manageable parts. The concept of video scene and video shot during the video production process is shown in figure 2.2.

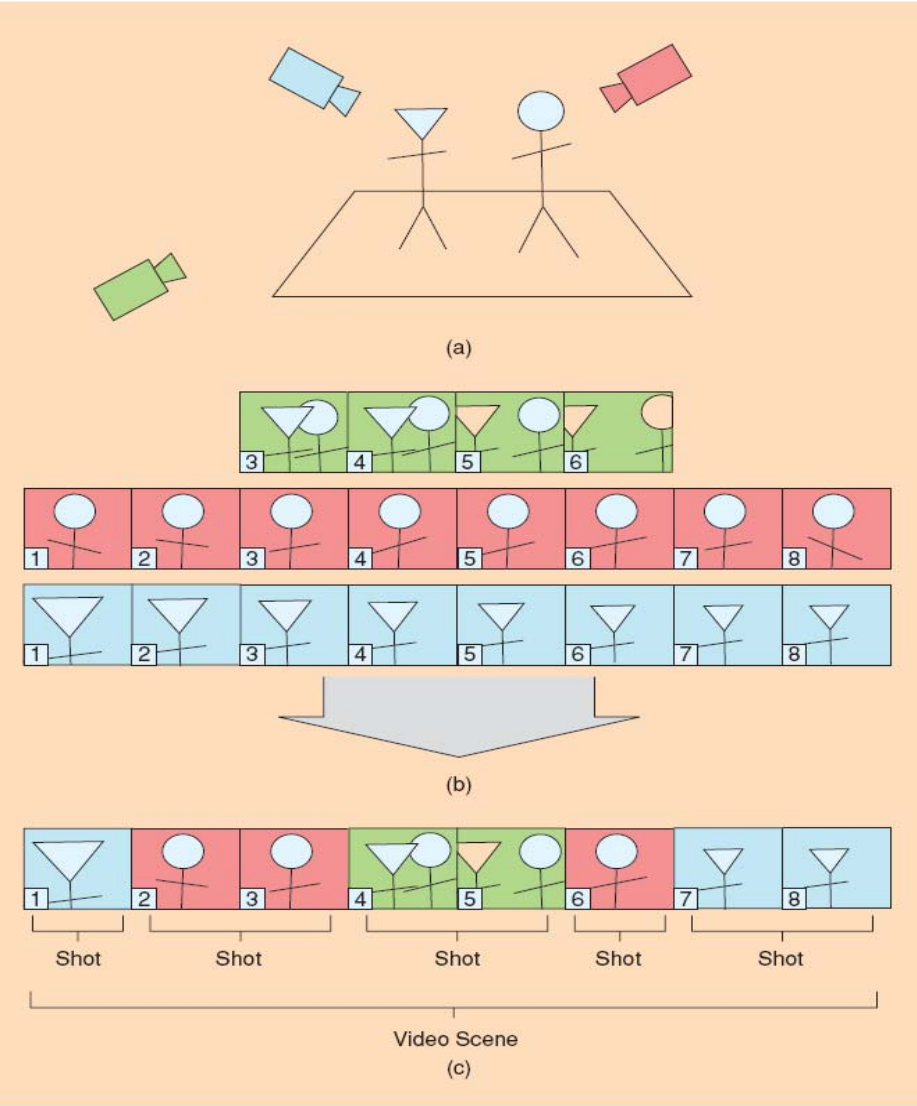


Figure 2.2: Video Production Process. (a) scene (b) different camera takes (c) shots.

2.3.1 Categories of shot change

Shot transitions are instances between camera shots that guide the viewer from one shot to another shot and are included during post-production process. The associations and connections between different shots are as important as the occurrence of shots themselves with all transitions depicting a slight change in time or space.

The video shot transitions can be categorized into two types which occur between different video shots, abrupt or discontinuous shot transitions and gradual or continuous shot transitions. During the analysis of video content majority of the time, a relationship measure between successive video frames is identified and defined. When two video

frames are effectively deviating, there may be a cut, which is the classic abrupt change. In such a case the previous frame belongs to disappearing shot and the next frame belongs to appearing shot.

Gradual transitions such as wipe, dissolve, fade and other transition types are found by using difference measures and more complicated threshold selection schemes. A wipe is actually a set of shot change techniques, where the appearing next shot and disappearing previous shot coexist in different spatial regions of the intermediate video frames. The area in use by the former shot grows until it entirely replaces the latter shot.

A dissolve occurs when the last few frames of the previous disappearing shot temporally overlap with the first few frames of the next appearing shot. During the overlap, the intensity and concentration of the former disappearing shot decreases from normal towards zero, while that of the latter appearing shot increases from zero towards normal.

A fade occurs when the previous disappearing shot loses color into a blank frame, and then the blank frame loses color into the appearing shot. Other transition types constitutes massive amount of creative special effects techniques used in video segments. Generally they are very uncommon and very difficult to detect. Table 2.1 summarizes of different shot transitions.

Shot transitions	***Description***
Cut	An immediate change from one shot to another
Fade-in	A shot increasingly appears from a steady image
Fade-out	A shot increasingly disappears to a steady image
Dissolve	The predecessor shot fades out while the successor shot fades in
Wipe	The next shot is exposed by a moving edge in form of a line or any other pattern

Table 2.1: Summary of different types of shot transitions.

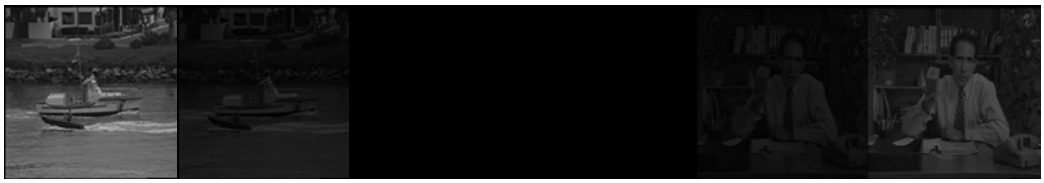
In comparison the gradual transitions are very difficult and hard to detect than cuts. These gradual transitions might comprise of special effects due to different camera panning and zooming. They must be notable from object movements and camera operations that demonstrate temporal variances within the sequence of video frames. It is particularly very difficult to detect dissolves between sequences of frames involving severe motion [13], [26-27].

Most people have seen immeasurable hours of television during their life span and share a hidden video grammar, particularly when we talk about different shot transitions. For example, a dissolve between two camera shots means a comparatively a little amount of time has spent. Different producers of video use this inherent grammar to help spectators understand the video and violating this implicit grammar will irritate the spectators' expectations.

The cut is the simple, straightforward and most common way of moving from previous shot to the next coming shot. There will be a smooth cut if some continuity and connection exists between the two video frames. For example, in depicting an interview scene, cut moves the viewer between the interviewer and interviewee. The dissolve persuades the viewer's observation of screen time and the pace of events. For example, a dissolve is used to cut down a long action for example an air plane flight and to transfer from the location of take-off to the destination of the flight. A fade indicates the beginning or end of any scene, idea or any episode being watched. In comparison of dissolve, fades often involve a more significant change of place or time. Since this film grammar is used consistently, the most frequent changes in video sequences are, cuts dissolves and fades. Figure 2.3 illustrates these three different types of shot transitions.

(a)

(b)



(c)

Figure 2.3: Most common types of transitions. (a) cut (b)dissolve (c) fade.

2.4 Types of camera operations

The basic camera operations include fixed, zooming, panning, tilting, tracking, booming and dollying. Zoom is the focal length change of a stationary camera. This brings you closer to the subject without moving the camera. For example, from a wide shot to a close shot or medium shot. If you are looking at the crowd of people at a road and you want to see individual people walking across it, you might zoom in. Besides bringing closer the zoom also moves you farther away from the subject. For example, from a close shot to wide shot or a medium shot. This might happen in such as case if you want to take a close up video shot of any flower in a field, and wish for seeing the entire field in which the flower is in, you will zoom out.

A pan is a shot taken moving on a horizontal plane either from left to right or from right to left. If you want to show a ball flying across a playing ground, you might use this shot to follow the ball from one person to another. A tilt is a camera movement in a vertical

plane, up or down. If the intended goal is to show a tall building but it is impossible to put it all in a shot, you perhaps would start from the bottom of the building and will try to go up to the top.

Tracking/ booming is horizontal/vertical transverse movement. It is also called travelling in the film language. In tracking the camera is moved towards left or right where as in booming the camera is moved towards up and down. Dollying is the horizontal lateral movement. In dolly the size of the objects in a scene is magnified or reduced by moving the camera towards or away from the object. Figure 2.4 illustrates different types of camera operations.

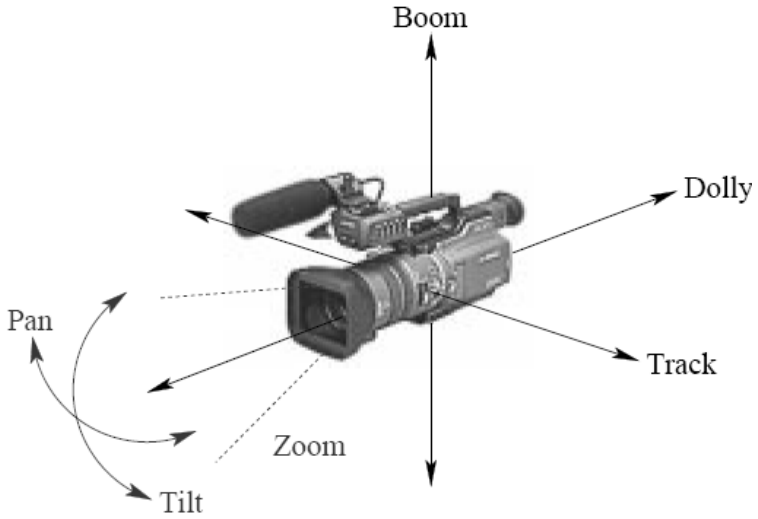


Figure 2.4: Different types of camera operation.

2.5 Mechanism of shot boundary detection algorithms

Different shot boundary detection algorithms work by exploring and exploiting one or more features from a video frame. An algorithm can then use a number of different choices to detect shot changes from these extracted features. The different approaches that can be made for each feature will result in some advantages and disadvantages. These can then be combined to help us in designing a new shot detection algorithm. Most of the shot change detection algorithms try to reduce the large facets of the video

data by considering a small number of features in each video frame. Such features can be further subdivided into 4 different categories which are as under.

2.5.1 Luminance or chrominance component

The simplest characteristic that can be used is the average grayscale luminance value of a video frame however this feature is inclined to illumination transformations. A method with additional step of using 3x3 averaging filter before the comparison to reduce camera motion and noise effects is presented in [6]. Another strong approach is being adopted by different authors to use one or more useful statistics e.g., average measures of the values in a suitable color space [28–30], like hue saturation value (HSV) for video frames.

2.5.2 Luminance or chrominance histogram measure

The most widely used feature is the grayscale or color histogram of consecutive frames in a video sequence [6], [31]. The gray level histogram differences in regions, weighted by how likely the region is to change in video sequence is also considered. Its advantage is that it is relatively discriminate, easy to calculate, and mostly insensitive to different types of camera motions such as translate, rotate, pan and zoom.

2.5.3 Edge tracking

It is related to how the human visual perceive a scene and thus it is an obvious choice for characterizing the video [13], [30]. An approach which compared color histogram, chromatic scaling and algorithm based on edge detection is present in [5]. The advantage of this feature is that it is adequately invariant to several types of motion and illumination changes present within the video frames in a sequence. However, the main disadvantage of such approach is its cost of computation, sensitivity towards noise and high dimensionality.

2.5.4 Motion vectors

It is usually used in combination with other features discussed above, since by itself it can be very irregular within a shot. This will be the case when motion changes abruptly and is obviously useless when there is no motion in the video. An approach which used motion vectors determined from block matching to detect whether or not a shot was a zoom or a pan is present in [6]. The use of motion vectors extracted as a part of the region based pixel difference if there is a large amount of camera or object motion in a shot is described in [10].

2.6 Types of shot detection algorithms

Based on the measures used to detect the difference between successive frames, the algorithms can be divided broadly into three different categories: template matching, block-based and histogram comparisons.

2.6.1 Template Matching

The template matching technique is also known as pair-wise pixel comparison and this approach assumes that there is some change between the two video frames. It calculates the differences in luminance or color intensity values of corresponding pixels in two consecutive frames for a video sequence. The simplest way is to calculate the absolute sum of differences in pixel values and compare it against a certain given threshold [32]. Equation 2.1 gives the measure of difference between two consecutive frames.

$$Difference(i, i+1) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y} \left| P_i(x,y) - P_{i+1}(x,y) \right|}{X.Y} \qquad (2.1)$$

Where i and $i+1$ are two successive video frames, $p_i(x,y)$ is the intensity value of the pixel at the coordinates (x,y) in frame i. X and Y are the total number of pixels of video frame in x and y direction respectively.

These algorithms count the number of the pixels changed, and the camera break is declared if the percentage of the total number of pixels changed exceeds a certain threshold [3], [6], [23]. The main disadvantage of this approach is that it is unable to distinguish between a small change in a large area region and large change in a small area region. In such a case cuts are misinterpreted when a small part of the frame have a large rapid and quick change.

Therefore, such techniques based on simple pixel comparison are susceptible to camera and object movements. An improved technique is to count those numbers of pixels that have a change in their values more than some threshold and to compare these totals against another threshold [6], [33]. In such approaches a cut is detected if the percentage of changed pixels within two frames of video sequence is greater than another threshold.

In this process some frame differences which are inappropriate are filtered out, but these approaches are still susceptible to object and camera movements between different frames in a shot. When a camera pans or zooms over a scene, the entire video sequence belongs to the same shot but the content of the scene changes substantially. Also, when the camera zooms, the images at the beginning and end of the zoom may be considered as representative of the entire shot. In such a case if a camera pans or zooms, a large number of pixels can be assumed and thus judged to be changed, even though there is actually a move with a few pixels with in a frame of a shot. Some approaches have been taken to reduce this effect to a certain extend by the use of a filter such as a smoothing filter, before the comparison of each pixel value and replacing it by the mean value of its neighboring pixels.

2.6.2 Block-based comparison

In comparison to template matching which is based on global image characteristics such as pixel by pixel differences, block-based comparison uses local characteristics. The

result is an increase in the strength to camera and object movement. Each video frame *i* is subdivided into *b* blocks that are compared with their corresponding blocks in next frame *i+1*. Typically, the difference between *i* and *i+1* is measured by

$$Difference(i,i+1)=\sum_{k=1}^{b} c_k D_p(i,i+1,k) \tag{2.2}$$

Here c_k is a predetermined coefficient for the block *k* and $d_p(i,i+1,k)$ is a function which describes the partial match value between the k^{th} blocks in *i* and *i+1* video frames of a sequence.

It is not a very strong approach if we use pixels to detect changes. Another approach uses matching blocks of video sequence which are compared using a likelihood ratio is present in [3]. A likelihood ratio approach is recommended based on the assumption of uniform second-order statistics measure over a region of interest [34]. A cut is declared if the number of changed blocks within two video frames is great enough, i.e. *Difference(i,i+1)* is greater than a given threshold t_1 and here $c_k=1$ instead of being a predefined coefficient in previous approach for all *k*.

This method is tolerant to slow and small object motion from frame to frame within a video sequence in comparison to template matching. The disadvantage of this approach is that there will be no change detected in such a case where two corresponding blocks that are different but have the same density function. Such situations, however, are very unlikely within a given video sequence. Another disadvantage of such approach is that it is slower due to the computations and complexity of the statistical formulas.

There is another block-based technique in which the frame is divided into 12 nonoverlapping blocks [10]. For each of these blocks the best match is found in their neighborhoods for the previous frame based upon frame intensity values. A non-linear order statistics filter is also used to merge the match values being computed. Thus by using such an approach the effect of camera and object movements is much concealed.

Cuts are detected using thresholds while gradual transitions are detected by identifying constant low level increasing matching values.

2.6.3 Histogram Comparison

By comparing the histograms of successive frames of video sequence, the sensitivity to camera and object movements can be reduced. The idea and assumption behind histogram-based approaches is that for any video sequence two frames with moving objects with unchanging background will have little differences in their histograms. Furthermore histograms are invariant to rotation of the contents within the frames and they also change very slowly as soon as scale and angle view is changed [35].

The disadvantage of such an approach is that there might be two video frames with similar histograms but have completely different content. However, the probability for such an incident is near to the ground and the technique has been purposed for dealing with such occurrence of events [23].

2.6.3.1 Types of histogram comparison techniques

Two types of histogram comparison techniques are being purposed in literature viz. comparison of global histogram and comparison of local histogram.

2.6.3.1.1 Global histogram comparison

The simplest approach uses a variation of the template matching approach. Instead of intensity values in equation (2.1), gray level histograms are compared [6], [33] and [36]. A cut is declared if the absolute sum of histogram differences between two successive video frames *difference(i,i+1)* is greater than a certain threshold *t*.

$$Difference(i, i+1) = \sum_{j=1}^{n} \left| H_i(j) - H_{i+1}(j) \right| \tag{2.3}$$

Here $h_i(j)$ is the histogram value and j is the gray value in frame i. The n is the total number of gray levels.

The comparison of color histograms is another simple and very effective approach [6]. To enhance the difference between two frames across a cut, several authors propose the use of the χ^2 test to compare the color histograms of the two successive video frames [32].

The overall performance of such an approach is not necessarily better than the simple histogram comparison. In addition, computation of χ^2 data is very slow and requires more computational time. The performance of three histogram based approaches using six diverse color coordinate systems: HSV, RGB, YIQ, l*a*b*, l*u*v* and Munsell is presented in [37].

The histogram comparison techniques are based on the fact that there is a huge difference between the video frames across a cut which results in a high peak in the histogram comparison. It can easily be detected by selection of any single threshold. However, such single threshold based approaches are not appropriate to detect gradual transitions in video sequences. Although during a gradual transition the differences between video frames are usually higher than those within a shot, they are much smaller than the differences in the case of cut and cannot be detected with the same threshold.

On the other hand, object and camera motions might cause greater differences than the gradual transition. Hence, lessening the threshold will increase the false prediction of results. A simple and effective two thresholds based technique for gradual transition recognition which accounts the cumulative differences between frames of the gradual transition is presented in [6]. The comparison of several temporal video segmentation techniques on real video sequences based upon twin comparison is present in [38].

2.6.3.1.2 Local histogram comparison

The histogram based approaches are simple and stronger to camera and object movements but fail in such case when two different images have similar histograms.

Such an incident rarely occurs for any video sequence. This is because they ignore the spatial information present in the video frame. In comparison to histogram based approach, block based comparison methods typically perform better than threshold selection mechanism but are still sensitive to camera and object motion present between the video frames. They also make use of spatial information present in the frames and these results in a computationally expensive process. By integrating these two approaches, positive points of both will be gained. Besides camera and object movement can be reduced the spatial information is also kept to produce more accurate results.

A comparison of several statistics measures based on gray-level and color pixel differences and histogram comparisons is presented in [32]. The best results were obtained by dividing the image into 16 equal-sized blocks. The χ^2 test on color histograms is used for these regions. The largest differences are discarded to reduce the effects of noise, camera and object movements within the video frames. There is another technique which is based on local histogram comparison and is proposed in [39]. A selective HSV histogram comparison algorithm is present in [40] in which blocks of video frames are compared in HSV (hue, saturation, value) color space. It is hue that makes the algorithm insensitive to changes since hue is independent of saturation and intensity value.

2.6.4 Feature-based detection

The work on feature-based shot detection which involves analyzing intensity edges between two consecutive frames is present in [5]. Since the change in video content is obvious in any video sequence, new intensity edges come into sight distant from the old edges at the stage of cut and dissolve operations. By accumulating the new and old edge pixels, different types of transitions such as cuts, fades, and dissolves are detected and classified by this approach. A canny's edge detector is also used along with the

smoothing of a gaussian matrix. After the smoothing stage a gradient threshold of a certain value is applied. Due to the presence of camera and object motion, motion compensation is used to give better results. This approach is much strong in detecting dissolves in the presence of motion. The drawback of it is that it does not handle sudden changes in brightness and multiple object motions.

Another very fascinating approach in which the intensity edges between two consecutive video frames of a video sequence is present in [13]. By counting the number of entering and exiting edge pixels different type of transitions such as cuts, fades and dissolves are identified. An algorithm for motion compensation is also incorporated in the case of little object and camera movements in a video sequence. However, the disadvantage of this technique is that it is unable to handle more than one object moving rapidly across the scene, which results in false detection of results due to the limitations of the edge detection method. The abrupt change in the shot intensity, i.e. Very dark followed by very light video frames or vice versa, may also result in poor detection of gradual transitions.

2.6.5 Clustering-based detection

The approaches discussed so far depend on comparison and suitable selection of thresholds of two successive video frames. However, these selected thresholds are typically sensitive to the type of content present in video frames of a video sequence. To overcome this drawback the unsupervised clustering algorithm is used [41]. Furthermore, this process can be seen as a 2-class clustering problem either it can be a scene change or it can be a no scene change in any given circumstances. To group such frame dissimilarities the well-known k-means algorithm is used [42].

Those video frames which are temporary adjacent on the basis of cluster, the scene change of such video frames are labeled as belonging to a gradual transition and on the

other hand those frames which are not temporary adjacent from on the basis of this cluster are considered as cuts. The two measures i.e. Color histogram difference and χ^2statistics were used for RGB and YUV color spaces. The experiments show the color histogram difference for YUV color space is the best choice in terms of overall performance on the contrary; χ^2 YUV detects the larger amount of correct transitions.

The disadvantage of such an approach is that it is unable to recognize the type of the gradual transitions. On the other hand the advantage of the clustering-based segmentation is that it eliminates the need for threshold selection and it permits different features to be used at the same time to perk up the performance. One example of the two diverse features i.e. Histogram difference and pair-wise pixel comparison were used in the clustering method is present in [43]. It was found that the use of such techniques results in both high recall and precision measures being computed.

2.6.6 Model-based detection

All the video segmentation techniques presented above identify and tackle the problem from data analysis point of view. That's why they are referred to as data driven, bottom-up approaches in literature. A number of techniques have been purposed which represent video data as a mathematical model and apply algorithms on the basis of such models of video data. A shot boundaries identification approach based on such a mathematical model based upon the video production process is presented in [14]. This model was used as a basis for the classification of the video edit types such as cuts, fades and dissolves.

Another model-based technique called differential model based on the probabilistic distribution of differences in pixel values between two successive frames is proposed in [44]. The combination of the factors such as additive zero-centered gaussian noise, a shot change model and a shot transition model are used for pixel change probability

distribution and for the different types of abrupt and gradual transitions in a video sequence. The absolute values of pixel differences are calculated and their histogram is computed and the number of pixels that change in value within a certain predetermined range computed by the models is calculated resulting in the detection of shot transitions. The results show 94-100% accuracy for cuts and 80% for gradual transitions detection.

Another approach for detection of gradual transitions based on a mathematical model of changes in intensity values during dissolves, fade ins and fade outs is presented in [45]. During the first pass, cuts are identified and detected using comparison based upon histograms of two successive frames. A step further the gradual transitions are detected by examining those video frames which are identified on the basis of their attributes between the cuts using the proposed model. This approach assumes that there is a small movement in the objects of video frames and it has very low computational requirements.

The model based shot detection using the statistical modeling techniques like a hidden markov model (HMM) is present in [46]. Separate states are used to model different transition types such as cut, shot, dissolve, fade, pan and zoom. The arcs assigned with the probability present between different states in the hmm model describe the allowable sequence of states along with their chance of occurrence. The arcs from a state to itself depict the length of time interval the video under consideration is in that particular state.

Three different types of features such as gray-level histogram distance between two adjacent frames, an audio distance based on the acoustic difference in intervals and an estimate of motion of different objects between the two successive frames are used as classifiers. The Baum-Welch algorithm is used for training the probability distributions and transition probabilities associated with them. The training data consists of features vectors marked as one of the following categories viz. shot, cut, fade, dissolve, pan and

zoom. After the training, segmentation of video is performed using the standard technique for recognition by using the Viterbi algorithm in HMM.

The advantage of the approach is that HMM framework allows any count of features to be included in a feature vector being computed. The use of HMM also eliminates the selection of threshold. This approach was tested on diverse sets of video databases and surpassed the standard threshold-based approaches in the accuracy of the temporal video segmentation existing in literature.

In the previous HMM based approach, the temporal distinctiveness of camera effect and video segment are not considered since each camera effect and video segment are modeled by only one state. The recognition of each effect mainly depends on the output likelihood of the state and this state belongs to the effect and state change probabilities which represent different arithmetical and statistical happenings of the camera effects. Other HMM based techniques spotlighted on the modeling of semantic scenes as in [47], where every shot have its own semantics and meanings. The drawback of this approach is that it can be applied after video segmentation stage to detect or merge shot by semantics.

A new HMM based framework is purposed in [48]. In this technique, each camera effect and corresponding video segment has its own individual HMM. The probabilistic characteristics and attributes of the effects can be represented in temporal direction. Furthermore, the method is used to emphasize the temporal characteristics of different shot transitions such as abrupt and gradual shot change. The audio features were also used to detect shots and merge them by semantics.

The machine learning based shot detection utilizing color and shape using HMM is present in [49]. In the purposed shot detection algorithm, multi HMMs with different states are constructed for shot transitions and non shot transitions. The framework

purposed includes four modules namely decoding, feature extraction, HMM recognition and boundary detection. The color and shape information of these frames are extracted and their difference between two features is calculated. The HSV color histogram feature belonging to color information summarizes the appearance of the embedded objects whereas the statistical corner change ration belonging to shape information summarizes the distribution of such objects.

The advantage of such approach is the exploitation of temporal characteristics of video frames and construction of hmm for each type of shot transitions. Another advantage of this approach is the elimination of the selection of threshold problem and selection of two complementary features which summarizes the shape and appearance of objects. However the disadvantage of this approach is that it is unable to detect all the gradual transitions and does not help in high level semantic analysis of sports videos.

Another approach utilizing color, shape and motion features for detecting MTV video shot using hmm is proposed in [50]. MTV videos are different from other types of videos as it has a large number of cuts and different types of gradual transitions followed by large inter frame motion present in a video sequence. Three types of features namely standard histogram distance, SCCR and motion vector information is used to depict the important information in video shot detection. Motion vectors which are computed using optical flow algorithm in a 30x30 pixel blocks help to increase the recognition accuracy of trained HMM as it alone can cause the false shot detection.

The advantage of this approach is that for MTV shot detection only one MTV video is used for the training of one hmm and after training, observation vectors are obtained and used to training each of the other HMMs. After the training MTV shot detection model is created via a standard Viterbi algorithm and highest likely hood model followed by state division is selected. However the disadvantage of this approach is the construction of a

single dimension HMM. Another disadvantage of such approach is that it is slower due to the computations and complexity of the statistical formulas used in the computations needed by Forward Backward algorithm and Baum-Welch algorithm for the re estimation of HMM parameters.

2.7 Summary

The field of video shot detection has developed and gets matured. It is now capable of detecting not only simple cuts but also different types of gradual transitions. The field has also advanced from simple different feature pixel comparison methods to accurate complex probabilistic mathematical methods of the shot transition process. This chapter gives an overview of existing techniques in literature for video shot detection that operate on uncompressed video stream. The performance, relative merits and restrictions of each of the approaches are discussed and compared. The development of the new techniques and approaches and how they apply on uncompressed domain methods are considered. In addition to the algorithms for shot boundaries detection, the related topic of types of camera operation is also reviewed. It is clear that improved solutions in this domain will involve inclusion of the a priori information about the process of shot formation as well as theoretical analysis of different editing effects.

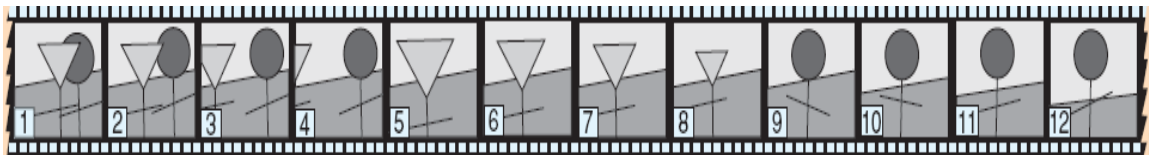
Chapter 3

System Methodology

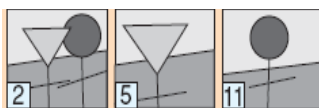
3.1 Introduction

This chapter gives an in-depth view and analysis of major characteristics of video shot segmentation and features used for spatial and temporal segmentation. At first, problem definition is given which is followed by different methods and techniques used for shot change detection to successfully identify key frames for video highlighting. A detailed analysis of problem by its decomposition into various modules along with theoretical background of adopted approach for the solution of each module is also provided.

Highlighting is the field that has attracted by far the most attention, since it is the simplest and most intuitive task especially used in sports videos. In the literature, the term highlighting is used as a type of video representation which constitutes different and divergent sets of video frames and thus results in summarization of a video. Its purpose is the selection of those video frames that contain the most relevant information for a given video under consideration. Sometimes highlighting is also referred as a key frame extraction task. Figure 3.1 represents different video frames and those which are chosen for highlighting.



(a)



(b)

Figure 3.1: Illustration of video and its highlights (a) original video (b) highlighting

3.2 Problem definition

Video segmentation is the process of decomposing a video into its component units. For the video highlighting, the input segmented data is in the form of either uncompressed video streams which need to be scrutinized so that appropriate representation of different events can be identified and thus summarized. Efficient and effective tools for segmenting of digital video are essential to allow easy access to video information. Thus the main goal of this research work is to come up with a novel and specialized technique that can be used to segment the video sequence into constitute parts by identifying the key frames present in it. Shot detection is the first phase to identify this objective.

Video segmentation is the process of decomposing a video into its component units. Consider a video *v* which is composed of *n* video shots taken from the set *s*. Then the video *v* can be represented as follows.

$$V = S_1 + T_1(S_1, S_2) + S_2 + T_2(S_2, S_3) + ... + S_{n-1} + T_{n-1}(S_{n-1}, S_n) + S_n \quad (3.1)$$

And the shot boundary detection problem can be defined as

$$\forall S_i \in V \quad \text{locate} \quad [t_b, t_e] \quad (3.2)$$

Where $s_i \in s$, the subscript *i* denotes the temporal position of the shot in the sequence (i.e. If $i < j$ shot s_i appears before shot s_j in the final representation) and t_i denotes the different types of transition between shots s_i and s_j. The + denotes the concatenation operation between shots and their transitions and t_b is the time when the video shot begins and t_e is the time at which the video shot ends thus we classify t_b and t_e as the extreme points of the video shot in any video segment. Thus the main problem is to identify these extreme points.

The system is designed by keeping in mind the Intel microprocessor on which Microsoft Windows® XP is installed. The program is written in Matlab® thus requires the Matlab software to be installed on the desired system.

3.3 Conditions and Assumptions

In order to advance without complete information, we need to make some assumptions and apply conditions. In a project, or in fact in life, we rarely have complete information. In any project, there is always a large scale of anonymity. If we start waiting until all information is readily available, we would most likely never start working on the project. The assumptions are probable failure points during the course of a project. They need to be properly scrutinized and supervised. At the start of any project these conditions should be jotted down. If new conditions and assumptions grow, they should be handled and appropriately treated. The priority of the assumptions and conditions of a project should be the same. The topic of this thesis is generally very vast and encompasses a very large problem domain. In order to confine the scope of work certain conditions and assumptions are made. The conditions and assumptions are summarized in table 3.1.

Conditions and Assumptions
Un-compressed videos such as 24bit raw YUVs are used
All checks will be performed, to make sure a proper output file is generated
The application should be able to handle the *.yuv file, that is used as input
Different chromo format support such as yuv444, yuv422, yuv420
Accurate resolution of the input file is to be provided along with input file
Two different types of sports videos viz. Soccer and basketball are used
Different statistical features used must identify shot boundaries
Threshold of statistical features are selected by training of videos
Experiments must be accompanied to recognize results on new test videos
The training videos used are having 1000 or 2000 frames
The testing video must be decomposed in the steps of 1000 frames

Table 3.1: List of conditions and assumptions

3.4 Major Design Goals

This section depicts and explains the design goals for this project. These goals must be taken into consideration before the start of the project. The list of major design goals is long and thus summarized in table 3.2.

Major Design Goals
There should be a simple and easy to understand design and coding followed by documentation.
The user interface must be friendly, easy for understanding and flexible for beginners using this project.
The end product made during this work must be robust, reliable, familiar, powerful and bug-free application for use.
The project must have comprehensive documentation to completely understand the intricacies of project.
Every time the project is run there should be a successful execution of application.
The project must address and inform all kinds of errors which occurred during execution of the project in the form of error messages to the user.

Table 3.2: List of major design goals

3.5 Requirements Analysis

The most productive artifacts are the ones in which the developers have completely understood what the product is intended to achieve for its end users and how it must complete that function. To recognize these things, it is necessary to understand what kind of job target users want to do and how the end product will shape and fit that work. It is better to identify what the end product being made does for its users. It must also satisfy the product's requirements which are the limitations and constraints in making a successful product. No product has ever got successful without prior complete

understanding of its all requirements. It does not matter what sort of work the user wishes to do, either it be a scientific or commercial one, an e-commerce, or word processing. It does not matter which programming language is used or which development tools are put to build the product.

The design goals described above in the earlier section lead to a large number of requirements for the project. These requirements, summarized in this section, can be further classified according to the major features of the framework to be described. This section discovers these requirements and concludes their precise nature. It also explains the requirements for this project and how to communicate them. By thorough analysis of the requirements for the project we can subdivide these requirements into four different categories, which are explained below.

3.5.1 Functional Requirements

A functional requirement defines a role of complete software or its constituent components. A function is expressed as a set of inputs, outputs and their behavior. Functional requirements describe the processing of the information or materials as inputs or outputs and thus support the user requirements. Functional requirements might be describing the calculations or technical details or it can be describing data manipulation or data processing or some other specific functionalities.

The functional requirements state what the end product must do. They narrate about the actions that the resulting product must perform in an order to persuade the primary reasons for its survival. The functional requirements also elaborate what is a particular behavior of a final system. This project has a number of functional requirements not limiting to the ones enlisted in table 3.3.

Functional Requirements
The user must view the useful information available on the application.
The user must be able to open any YUV file as input to the application.
The system should generate error messages in case of wrong stream or wrong file format being input by user.
The application made must calculate and show total number of frames and run the program till the number of frames.
The application must identify the different types of the chroma format either yuv444, yuv422 or yuv420.
The application must show an error message if video file being input has not the same resolution as specified by the user.
Different graphs will be displayed depending upon the statistical properties of video data.
The video shot will be identified by the application in a form of a graph.

Table 3.3: List of functional requirements

3.5.2 Non-Functional Requirements

Non-functional requirements are also known as qualities of any system. Non-functional requirements are typically used to limit the exploration of the design space, since they compel at early stages particularly in design and implementation solutions. Nonfunctional requirements are those characteristics and properties that end product must have included. Think of these properties as the characteristics or qualities that make the product fast, attractive, functional and consistent. These properties are not explicitly required because they constitute the fundamental activities of the end result.

Besides that of functional requirements there are many non functional requirements as well. The non functional requirements are also crucial but not as functional requirements. Therefore they are further divided into four different categories such as performance

requirements, interface requirements, system requirements and security requirements. Each of them is discussed and thus described in their respective sections.

3.5.2.1 Performance Requirements

Performance requirements describe how well the system executes certain functions under certain circumstances. Some examples of performance requirements might be speed of response, execution time, storage capability and throughput. Performance requirements are based on supporting tasks of end-user. Performance requirements are fundamentals in product designing and its testing like most other quality attributes. Along with other types of quality characteristics like reliability, robustness, security and usability performance requirements are considered.

The performance requirements for the project are crucial because the system is a real-time system hence performance factor is an important issue to be discussed. All the conversions and those algorithms used in the project must be efficient, although a measure of efficiency is not known yet.

A number of performance requirements are identified and are summarized in table 3.4

Performance Requirements
Video quality is an issue as it depends on the value of quantization that the user will enter. Lesser quantization values will result in poor quality video.
Another issue is size of input file. Larger the size of file it will require more bandwidth and more processing power. Moreover larger files require more time to be decoded.
This project is standalone and neither has it required a network nor it uses any external data sources to access its data. The processing time shall be proportional to the size of the input YUV file.

Table 3.4: List of performance requirements

3.5.2.2 Interface Requirements

The user interface is essential to project usability. There are many requirements that must be elicited, examined and confirmed, a part from user experience which is necessary to successful completion of project. A good user interface can predict the divergence between acceptance of a project and its failure. If the end-users find the project to be awkward or too difficult to understand, then an otherwise excellent project could be predestined to failure. The developer's goal is to make the project easy to use as possible. Beside that of performance requirements there are some interface requirements as well. Table 3.5 summarizes the list of some interface requirements.

Interface Requirements
The target user can view different data and information available on the application.
The user can open any YUV file from open submenu.
The user interface is simple and self explanatory so that the user can easily perform the required operation by following simple steps.

Table 3.5: List of interface requirements

3.5.2.3 System Requirements

Beside that of performance requirements there are some interface requirements as well. Table 3.6 summarizes the list of some system requirements.

System Requirements
The coding was being done in Matlab, which implies that software can only be used if Matlab is installed on the system.
Interface is developed in guide by using Matlab.
High processing power and sufficient disk availability is required to ensure fast processing.

Table 3.6: List of system requirements

3.5.2.4 Security Requirements

This project will only be accessible to legal users like those who can use it for educational purposes or those who have properly registered for the product. Any illegal production of the source code will not be allowed. If corrupted video files are tried to be taken as input, this may cause unpredictable results. Input of unknown file formats must be prevented. No other security or safety requirements have been identified.

3.6 Quality Attributes

The system must not crash on non-fatal or user input errors. Following table describes the severity of different quality attributes and their level attained during the project. Table 3.7 illustrates some quality attributes. In column c 'H' stands for high, 'M' stands for medium and 'L' stands for low.

Id	*Characteristic*	*H/M/L*
1	Correctness	H
2	Efficiency	M
3	Flexibility	M
4	Integrity/security	M
5	Interoperability	M
6	Maintainability	M
7	Portability	H
8	Reliability	M
9	Reusability	H
10	Testability	M
11	Usability	M
12	Availability	H

Table 3.7: Project Quality Attributes

Chapter 4

System Design and Implementation

4.1 Introduction

For effective use of multimedia databases, the significant amount of video data requires complicated processing. Currently the most efficient and successful method to do this is manual processing, but it is very slow and expensive. For this reason automated methods have been designed and purposed to explain and annotate video sequences. It is also responsible to take user's need into consideration and provide a content description, by allowing proficient access to the video information without viewing the whole content. The range of video characteristics can be very broad and viewers can be interested in any aspect of it. It can range from decomposition of video into constituent shots and scenes, or it can be the most representative key and index frames or subsequences. A part from that spectator can be interested in the camera work or the requirement of his/her can be to identity of the people that appear in it, and their movements and dialogues [22]. For this reason a framework is purposed which address these divergent set of requirements.

4.2 Framework

The proposed framework is applied in the uncompressed domain of video and consists of four modules, namely frame extraction, feature measurement, threshold selection and winner take-all selection. Since the solution is tackled in a modular manner therefore in such approach the complete problem is identified and thus decomposed into several modules. The modules are interrelated and they are formed in such a way that the output of the previous module becomes the input for the subsequent module.

However, the primary input of the system is in the form of uncompressed sports video either soccer or basketball having multiples of 1000 frames. The input to the framework is a raw video file in the form of YUV frames in any chroma format from which

individual frame can be extracted and the output of the framework is a shot list for the input video file in the form of frame numbers. The framework and process flow chart between various modules is shown in figure 4.1

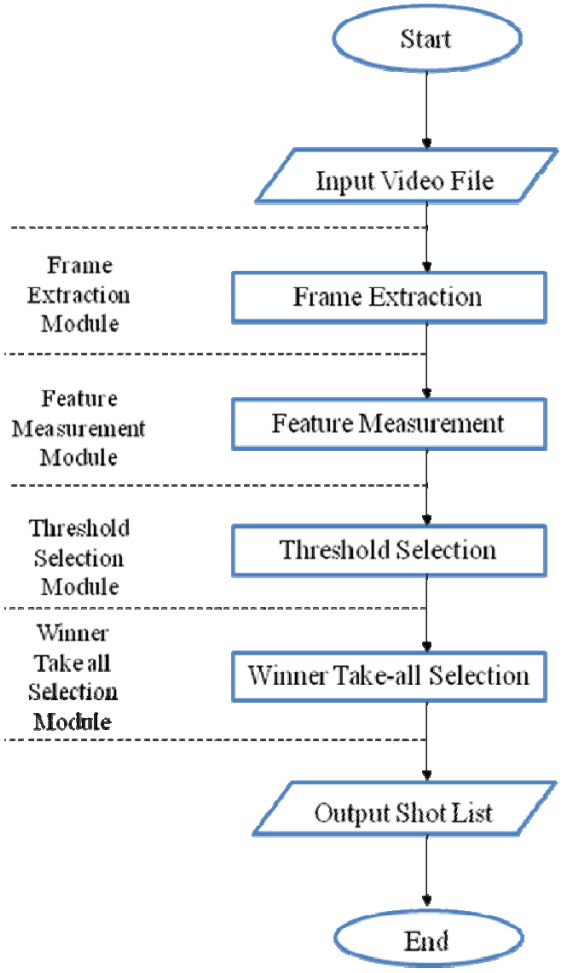


Figure 4.1: The framework and the problem decomposition in modules

In the frame extraction module individual YUV frames either in yuv444, yuv422 or yuv420 color spaces is extracted or decoded for analysis of video. In the feature measurement module different measures such as correlation, running averages such as arithmetic mean and YUV histogram difference between consecutive video frames are calculated as features which depict and portray the color and shape information of these test video frames. In threshold selection module different threshold values are identified from the training video and thus selected for those features which were extracted in the previous module. Finally in the winner take-all selection module different video shots are

identified on the basis of the voting between the outputs of previous threshold selection module. Once these key frames are identified they are represented as output in the form of video shot list.

4.3 Input to the framework

Video systems transmit video data in the form of one component that represents light and two components which represent color. YUV color space is a bit strange. It takes human perception into account and encodes video by allowing reduced bandwidth for chrominance components in the video. The Y component determines the brightness of the color (referred to as luminance or luma), while the U and V components determine the color itself (the chroma).

Since the Y space includes significant information than that of U and V space, proficient use of this component alone can be used in comparison with RGB color space. As long as luma is conveyed with full detail, subsampling process such as averaging or filtering can be used in lessening the details of the chroma components. There are many different schemes in use; this chapter provides the brief summary of some of them because the input to the purposed framework can be one of these three different schemes existing in literature.

The input raw uncompressed video file contains these different schemes and formats contained as .yuv file. The multiples of 1000 frames either 1000 or 2000 are selected for training of the framework. Ten different YUV raw videos are used for training of different features in this framework and the results are verified on a new test video sequence which was not involved in training. The test video is also segmented in the multiples of 1000 frames before the evaluation of video shot. The best results were gained when both training and test video files comprised of 1000 raw data frames.

4.3.1 Different subsampling schemes of YUV

Several different subsampling schemes have been commercially present in literature. This thesis will be addressing and focusing on those schemes which are used as an input to the framework.

4.3.1.1 YUV 4:4:4

This scheme is sometimes used in high-end film scanners and cinematic postproduction. In this scheme prior to subsampling, RGB video is denoted RGB4:4:4. Each RGB pixel can be transformed into YUV and the end result is YUV 4:4:4. The process of subsampling in true terms has not been taken place however each pixel has y, u and v values. The term 4:4:4 denotes that for every four samples of the luminance (y), there are four samples each of u and v. Thus each pixel has a luminance value (y), a u value (blue difference sample or cb) and a v value (red difference sample or cr) as shown in figure 4.2. The notation yuv4:4:4 subsampling notation, is commonly used for such scheme.

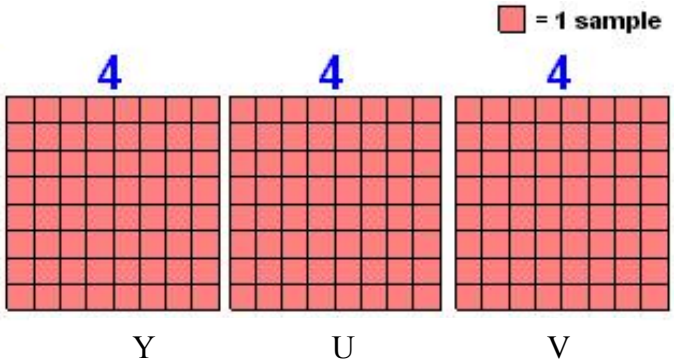


Figure 4.2: Illustration of YUV 4:4:4

4.3.1.2 YUV 4:2:2

Many high-end digital video formats and interfaces such as digital Betacam, CCIR 601 including professional DV50 systems and the 422 profile of MPEG-2 use this sampling. The two chroma components are sampled at half the sample rate of luma. The term 4:2:2 denotes that for every four samples of the luminance (y), there are two samples each of u and v, giving less chrominance (color) bandwidth in relation to luminance. So for each

pixel, it is horizontally sharing UV (chroma) with a neighboring pixel as shown in figure 4.3. This results in the reduction of bandwidth of a video signal by one-third with little or no visual difference. The notation yuv4:2:2 subsampling notation, is commonly used for such scheme.

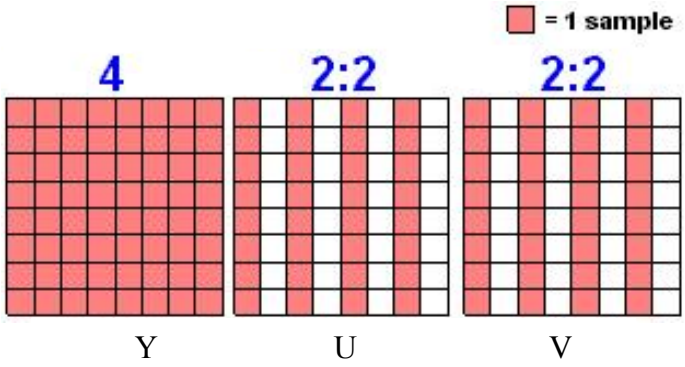


Figure 4.3: YUV 4:2:2

4.3.1.3 YUV 4:2:0

This scheme is found in all versions of MPEG, DVD, PAL, HDV, most common JPEG/JFIF, H.261, and MJPEG implementations. The term 4:2:0 denotes that for every four samples (two horizontal and two vertical) of the luminance (y), there is one sample each of u and v, giving less chrominance (color) bandwidth in relation to luminance as shown in figure 4.4. In this scheme chrominance components u and v each are subsampled by a factor of 2 horizontally and a factor of 2 vertically. The notation YUV 4:2:0 subsampling notation, is commonly used for such scheme.

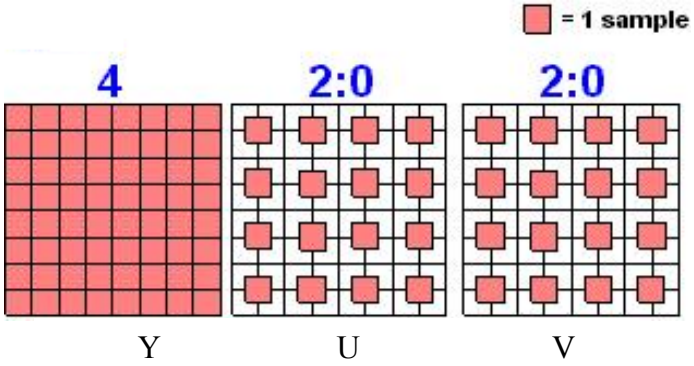


Figure 4.4: YUV4:2:0

These three different types of schemes are used for the implementation. The users are asked for which scheme they want to train the framework and test the video.

4.4 Frame extraction module

The individual frames of video file which was being input are extracted in this module. Based upon different chroma format as previously discussed, individual frames can contain variable size of data. This module is responsible to identify which frame format is used as input. It also collects and combines the corresponding y, u and v components of the video to display at the graphical output. Since y component is followed by u and v components respectively therefore the separate access to a frame with y, u and v components are made for identifying and extracting a single frame of a video sequence. This process is repeated a number of times depending upon the number of frames in any video. Since the number of frames can be a multiple of 1000 frames therefore a lot of time this module is being called for the access of individual components of luminance and chrominance incurring cost $O(n)$ where n is the number of frames in the file being input. This process helps us in later stages when individual thresholds are being assigned to each component of video frames. Figure 4.5 represents the yuv4:2:2 chroma format and their representation for display in a frame.

Y1	Y2	Y3	Y4	Y5	Y6
Y7	Y8	Y9	Y10	Y11	Y12
Y13	Y14	Y15	Y16	Y17	Y18
Y19	Y20	Y21	Y22	Y23	Y24
U1	U2	U3	U4	U5	U6
U7	U8	U9	U10	U11	U12
V1	V2	V3	V4	V5	V6
V7	V8	V9	V10	V11	V12

Figure 4.5: Representation of YUV 4:2:2 video frame in a file

The convention of different color combinations is used describing which component of luminance is used with which component of chrominance like u9, u10 and v9, v10 are to be combined y15, y16, y21, y22 block which is given blue color to correctly display the desired video frame to the user. Similarly the same convention is used with other blocks of luminance y components with chrominance u and v components. Figure also depicts the fact that two chroma components are sampled at half the sample rate of luma for yuv4:2:2 subsampling scheme. Similarly is the case with yuv4:4:4 and yuv4:2:0 subsampling scheme as these three schemes are being used in the project.

4.5 Feature measurement module

The output of previous module i.e. the individual frames being extracted act as an input to this module. The three different types of attributes are used in the feature measurement module to identify the characteristics of video frames. The first one is the standard correlation coefficient which measures the degree of associations and relationships between two different video frames. The second one is the YUV histogram difference of two consecutive frames which models the color information and the third is the running average difference, between the luminance components, which measures the change in the average intensity of the actual content between the subsequent frames.

4.5.1 Correlation coefficient

The correlation is one of the most common and most useful statistics. Correlation is a method for finding the degree of probability which describes strength and direction of a relationship between two measured quantities. It determines the extent to which values of the two variables are proportional to each other and describes the degree of relationship between two variables. The value of correlation (i.e., correlation coefficient) does not depend on the specific measurement units used for example, the correlation between

weight and height will be identical regardless of whether inches and pounds, or centimeters and kilograms are used as measurement units.

The correlation coefficient between the two consecutive frames x and y can be defined as

$$\rho_{XY} = \frac{\mathrm{cov}(X,Y)}{\sigma_X \sigma_Y} \tag{4.1}$$

Where

$$\mathrm{cov}(X,Y) = n\sum_{i=1}^{n} X_i Y_i - \sum_{i=1}^{n} X_i \sum_{j=1}^{n} Y_j$$

$$\sigma_X = \sqrt{n\sum_{i=1}^{n} X_i^{\,2} - \left(\sum_{i=1}^{n} X_i\right)^2}$$

$$\sigma_Y = \sqrt{n\sum_{j=1}^{n} Y_j^{\,2} - \left(\sum_{j=1}^{n} Y_j\right)^2}$$

σ_x and σ_y are the standard deviations of frame *x* and frame *y* respectively and *n* is the total number of pixels, after the frame extraction module, in the either frame of the video under process. The value of ρ_{xy} is between -1 and 1. If the value of ρ_{xy} is 1 or -1, it means strong correlation and if the value of ρ_{xy} is 0 it means no correlation. The closer the correlation is to 1, the stronger the relationship between the two frames, and the closer it is to 0, the weaker it is.

$$-1 \le \rho_{XY} \le 1 \tag{4.2}$$

4.5.2 YUV histogram difference

Luminance histogram considers only the luminance distribution within the video sequence. However, as we are dealing with color video sequence, we should define and utilize a color histogram. If we compute the overall number of possible colors we realize that we rise to high levels of possible colors. A simple solution consists in considering

only the most significant bits of each component YUV. As we will see later, the shot boundary detection method uses a color histogram in YUV space, where quantization is done by eliminating the least significant bits for each component.

The YUV histogram feature calculates the distance between neighbouring video frames on the basis of the distribution of chrominance and luminance pixels in the frames. As the y, u and v components of yuv frame can be present in any form; we calculate the histogram differences of consecutive frames on each component respectively and extracted the maximum YUV histogram difference as the second feature. A similar approach for HSV color space can be found in [36].

The presented method of shot boundary detection is based on the computation of differences of color histograms between frames as a measure of discontinuity. The difference can be computed as the sum of absolute difference between the bin values.

The YUV histogram difference $\mathrm{YUVHD_i}$ can be calculated as follows.

$$\mathrm{YUVHD_i} = \max(\mathrm{Y_i}, \mathrm{U_i}, \mathrm{V_i}) \tag{4.3}$$

where

$$\mathrm{Y_i} = \sum_{b=1}^{B} \left| h_i^{\mathrm{Y}}(b) - h_{i+1}^{\mathrm{Y}}(b) \right|$$

$$\mathrm{U_i} = \sum_{b=1}^{B} \left| h_i^{U}(b) - h_{i+1}^{U}(b) \right|$$

$$\mathrm{V_i} = \sum_{b=1}^{B} \left| h_i^{V}(b) - h_{i+1}^{V}(b) \right|$$

Y_i, u_i and v_i are the histogram differences of luminance y and chrominance u and v components respectively. The value of $h_i^{\mathrm{Y}}(b)$, $h_i^{U}(b)$ and $h_i^{V}(b)$ are the normalized histogram of y, u and v components respectively.

4.5.3 Running average difference

One characteristic of the running average is that if the data have a periodic variation, then applying a moving average of that period will eliminate that fluctuation. This makes the difference from the moving average useful for highlighting where the data is breaking away from the trend like a cut in a video shot. A simple moving average is the unweighted mean of the previous *n* data values. It is calculated by finding a number of consecutive means provided that each means having the same number of observations. Each successive average value will drop first value of the mean interval and add the next value in the dataset for the next mean interval. The running average is calculated by averaging together the previous values over the given period, including the current value. The running average at the beginning of a data series is not defined until there are enough values to fill the given period.

The running average of the difference of the mean values such as for arithmetic mean is another parameter which helps in the measure of shot detection. The parameter can belong to one of the two classes depending upon whether there is a single peak or two or more opposite peaks cancelling the effect of each in the sample window. The maxima and minima of mean difference are calculated and are utilized to differentiate two different types of classes depending upon this parameter. The running average difference RAD_i can be calculated as follows.

$$\mathrm{RAD_i} = \frac{1}{N}\sum_{j=i}^{i+N-1} \mathrm{X}_j \tag{4.4}$$

N is the total number of frames in the window, *j* varies from 1st frame of the window to the last frame in the window and x_j is the mean difference of the two consecutive frames.

4.6 Threshold selection module

The output of previous feature extraction module such as correlation coefficient, YUV histogram difference and running average difference are the inputs of this module. A

number of videos are tested during training phase for the algorithm and results are gathered for different values of threshold in each case. For the correlation coefficient the lower threshold is decided to be 0.40. This is confirmed on number of test videos and covers a significant amount of shot detection in results. To improve the precision and accuracy in results the value of upper threshold is decided to be 0.60. The values lying in the lower and upper threshold and other measures calculated are further processed by the selection of winner take-all selection module.

For the YUV histogram difference a single threshold is being selected in comparison to two thresholds of correlation coefficient. The threshold for this parameter has been decided to be 0.8 after testing the different number of bins for different formats of training videos. For yuv chroma format 4:2:0, the 64 bin normalized histogram each for y, u and v components respectively and 0.8 threshold value provides the desired outcome. More information is present in results section.

For running average difference the threshold is not fixed for all the videos and it changes depending upon the content in the video frames. Since the threshold is not present therefore a new model which should be adaptive and help us determine the threshold was indispensible. A model in which separate states are used to model different transition types such as shot, cut, fade, dissolve, pan and zoom is presented in [35]. A somewhat different approach in which it constructs the HMM with different hidden states for each shot transition type is presented in [37]. However such an approach will cause the high computational complexity and slow down the shot detection approach. Another model in which shot, fade and cut states are modeled by different states in one dimensional HMM is present in [38]. These three models convince and drive to model the problem with two states and thus a two state model was adopted for threshold selection of running average difference.

4.6.1 Two state model

The measures such as maximum amplitude and minimum amplitudes in a sample window facilitate us in finding the difference between the magnitudes of intensity values within a frame. If the magnitude is near to the average of intensity values present in the frame, a cut is declared at that frame. On the other hand if the magnitude is near to zero there is no cut and the video frame is within the same current shot.

The benefit of using such a two state model is twofold. Firstly it is adaptive depending upon the contents of video data and thus maximum amplitude is changing depending upon the selection of sample frames in the window. Secondly the selection of minimum amplitude within a sample window results in appropriate representation of same shot selection as this amplitude difference is unable to carry it to new shot state. Figure 4.6 shows the two state models used for shot detection of video with this approach.

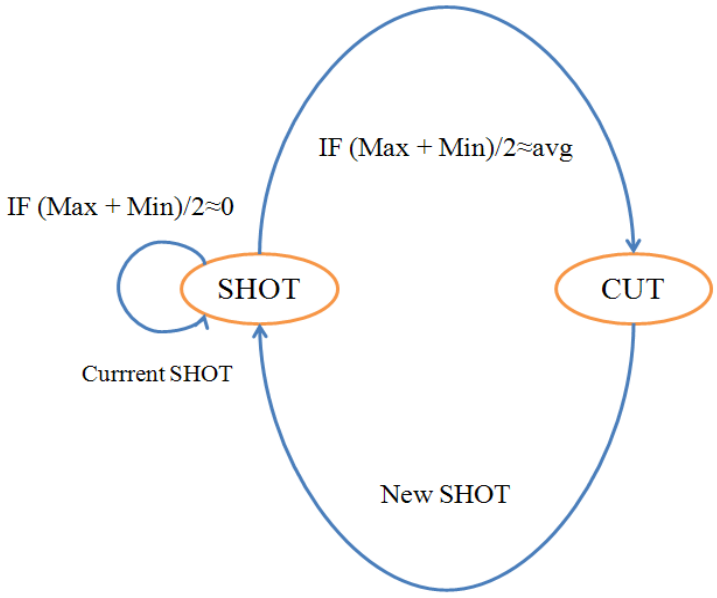


Figure 4.6: Two state model for shot detection

By using such a two state model we can model the hard cuts of the test video completely. Most of the time, the model will be in shot state. Current shot means the shot in which we are already present. New shot means that the cut has occurred and the next frame of video will come under the category of new shot. Since most of the time in a video

sequence the video frames belong to one shot during a scene therefore this model naturally model such a situation and clearly depends upon such probabilistic occurrence of shot in a scene.

The arc from a shot state to itself depict the length of time interval the video is in that particular state. It is also notable that there is no arc for cut state. This is because the previous frame will belong to the same shot state and new frame at which cut has occurred will follow the new shot arc leading to shot state. This model strictly depends upon the value of maxima and minima of the current window and thus correctly model the shot state. Thus the threshold for the three features which was input to threshold selection module are identified and being used in later stages to help us identifying the video shot boundary.

4.7 Winner take-all selection module

The output of feature values from the threshold selecting module are being put to this module as input. The winner take-all selection scheme is the last module in the framework which utilizes the upper threshold of correlation coefficient. It doesnot use its lower threshold because majority of the shot are already identified with the lower threshold. Only those video shots which lie in the range of 0.4 to 0.6 for correlation coefficient are being used by this module.

Beside this range of correlation coefficient, this module also takes in consideration the other two parameters such as YUV histogram difference and running average difference with their respective thresholds. This module depends upon the voting mechanism of these inputs and selects the appropriate frames for shot boundaries. The flowchart depecting the process of winner take-all selection followed by the working mechanism of this module is presented as under.

4.7.1 Flow chart of winner take-all selection

The flowchart for winner take-all selection is represent in figure 4.7. Here *cc* is correlation coefficient, *YUVHD* is the YUV histogram difference and *RAD* is running average difference. The *cc* and *YUVHD* are depending upon the comparison of two consecutive video frames whereas *RAD* measure is for the sample window. The size of window varies from 5 to 15. More information is present in results section. These three were the outputs of previous threshold selection module. Different checks are being made on the input in order to correctly identify the key frame at which the cut has occurred.

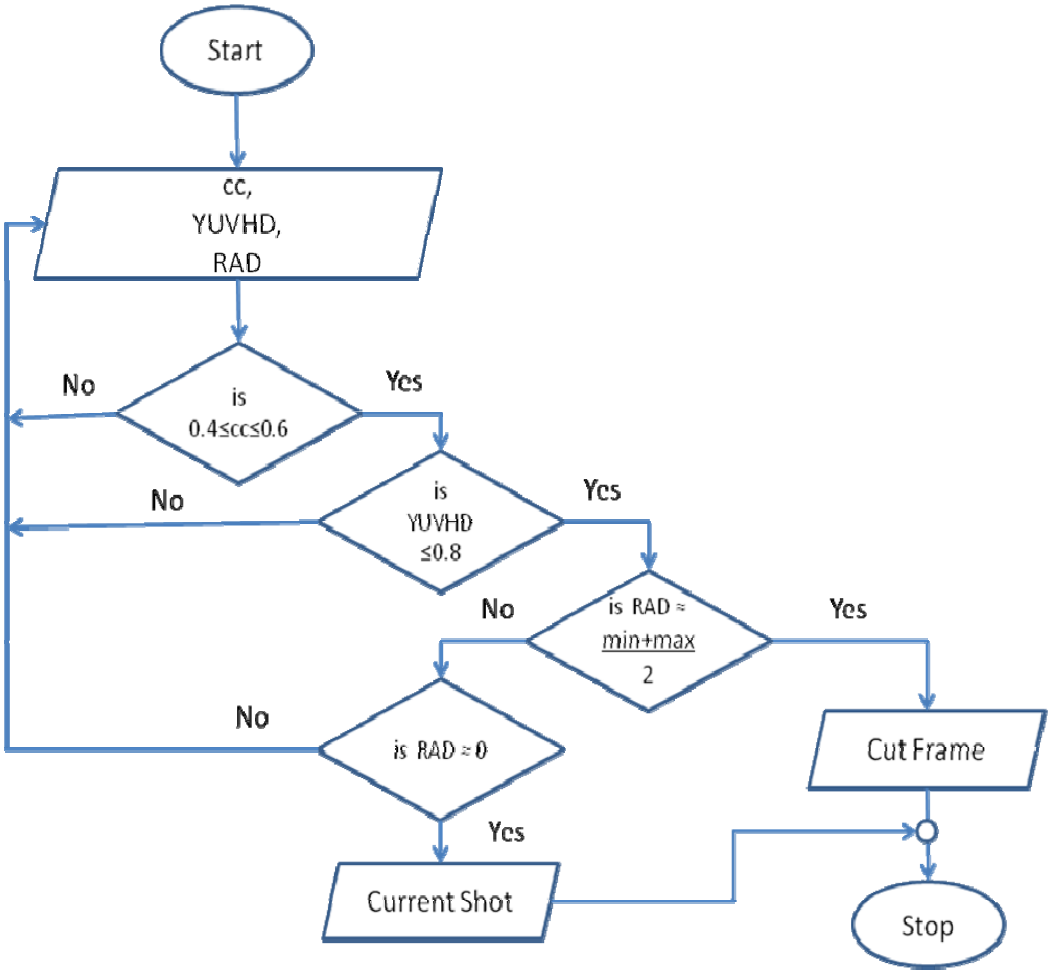


Figure 4.7: Flow chart of winner take-all selection

4.7.2 Working of winner take-all selection

1) The frames of video for which the correlation coefficient lies in the range of lower threshold and upper threshold are identified.

2) The frames of video for which the YUV histogram difference exceeds its threshold are also taken in to consideration.

3) The frames of video for which the running average difference exceeds the threshold with in a window are also gathered.

4) Those frame numbers which are common in step 2 and 3 and pass the criteria are considered for evaluation

    a) If the frames numbers from step 1 and step 4 are common identify cut at that frame. The newshot state will immediately be followed by cut state.

    b) If the frames numbers from step 1 and step 4 are not common identify current shot at that frame.

5) Those frames which are not common in step2 and step3 and do not pass the criteria also belong to current shot state.

6) All others which are common in step1 and step2 or step1 and step 3 are identified to be in shot state of the model.

7) Those frames which do not fulfil the criteria of step 1 or step2 or step 3 also belong to the same shot state.

4.8 Output of the framework

The output of the framework is a list of video shots which were correctly identified. This list also includes those frame numbers of video sequence which were identified as key frames during all stages of framework. One of the elementary assumptions made here

was that the change of content in the features of camera motion was the significant factor when determining whether to select a key frame or not. This means that if a shot contains little or no camera motion, typically a key frame will be selected. Some key frames are directly identified after the threshold selection module and some pass the tests of winner take-all selection module to be identified as shot boundary frames. The key frames are selected to minimize representational redundancy whilst still portraying the content in each shot. The list in the form of frame numbers of both stages are gathered and collectively being assigned as video shot list of this framework.

Chapter 5

System Analysis and Results

5.1 Introduction

A novel, unified approach which categorizes video shot boundaries is presented with a better decree into cuts. In this chapter experiments and results of the purposed framework for shot detection of uncompressed sports videos are presented. The problem of detecting shots in a video involving multiple motions within different blocks was identified. These were identified and learnt from training videos having deviations in the content and the sequential order of events. In order to enumerate the significance of the proposed framework it is tested on a number of sports videos of basketball and soccer games.

Reliable shot detection is very challenging in these games due to a variety of reasons, not limiting to.

- Excessive use of camera movements such as zoom ins, zoom outs, pans, tilts and booms are extensively used in these games to spotlight and track certain events in the game.
- Motion of different objects in the frames at high and low speed at different instants in time can cause false detection of threshold selection parameters.
- The various scenes often consist of different objects that are smaller in size and are fast moving as compared to the background. In such videos there are usually frequent objects in any scene and they have common occlusions with each other.
- Gradual shot transitions such as cross dissolves, fade ins and fade outs, wipes and other transition and editing effects such as mates and abrupt shot transitions such as hard and soft cuts occur at very short periods in time often following one another which results in false estimation of shot boundaries.

- Most sport videos are captured in relatively same environment such as green fields in case of soccer and the same courts and relatively same content for spectators in the basketball game; hence only correlation coefficient is unable to identify these facts. Figure 5.1 illustrates this fact in two consecutive frames of a video shot in a training video sequence.

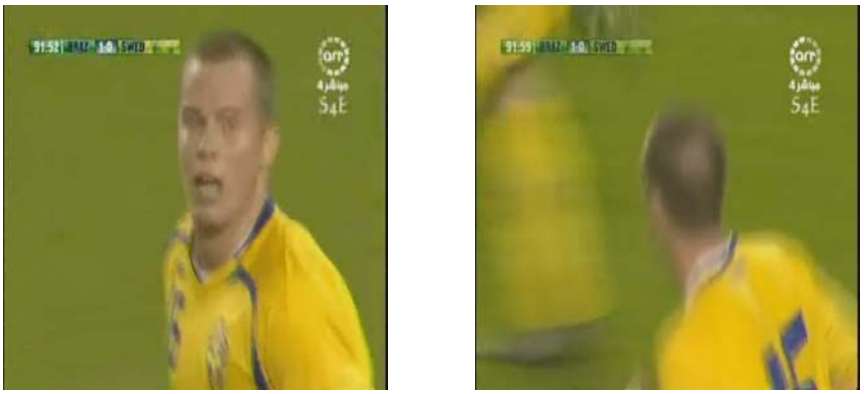


Figure 5.1: An illustration of relatively same environment of football game

5.2 Experiments

For conducting experiments, the video data is gathered from the different sources over the internet. Since most of the sources have raw uncompressed data therefore it was decided that all data should be decoded to a unified uncompressed domain which is YUV. Three different chroma formats yuv444, yuv422 and yuv420 are used. The proposed framework was tested on several sports videos of basketball and soccer games.

5.2.1 Test set

The test set consists of ten clips each having 1000 and 2000 frames contributing more than 100,000 frames of raw data and utilizing more than 10GB of hard disk space. The gathered content for basketball videos used during training have dimensions of 480x440 where as for soccer videos the size of 480x480. The original video was in an mp4 file format and Xilisoft video converter software was used to convert these files into mpg file format.

Xilisoft video converter is powerful audio/video conversion software that supports conversion of most popular audio/video formats. Besides the above powerful function, it provides versatile settings and features as well. You can convert audio/video clips with customized starting point and length, choose the destination of the output file, set the zooming mode and splitting mode for the output file, and rename the output file, etc. For example, you can easily convert WMV into AVI or MPEG format with it. It also supports the conversion of the latest H.264 video file and to output mp4 formats for Ipod and PSP etc. Following table summarizes different parameters being used for conversion of files and their corresponding values during experimentation.

Item	Value
Standard	NTSC
Video codec	Mpeg2 video codec
Bit rate	2040 kbits/sec (255kbytes/sec)
Minimum bitrate	0kbits/sec (0kbytes/sec)
Maximum bitrate	2516kbits/sec (314.5kbytes/sec)
Buffer size	1792kbits (224kbytes)
Framerate	29.97 fps
GOP	18
Zoom	Full aspect ratio

Table 5.1: Summary of different attributes and their corresponding values

5.2.2 Conversion from mp4 to YUV

The mpg file format was decoded into YUV file format by using standard mpeg-2 decoder. The output of YUV decoder is in a form of individual frames of yuv420. All other formats of YUV file is being generated by using the formulas and calculating the duplicate pixels by understanding the sub-sampling process at hand. Once all the chroma

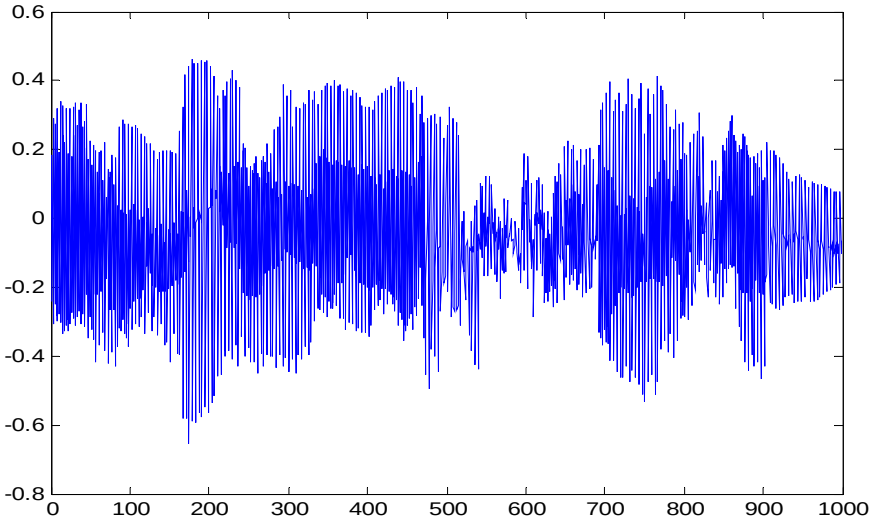
formats have been gathered appropriate gathering of them in the multiples of 1000 frames is done programmatically. Individual files being made during this process either contain 1000 or 2000 frames respectively. YUV player deluxe is used to see the converted stream.

YUV player deluxe is a full-featured tool for playback of uncompressed planar yuv video files. It is intended for researchers in the area of video compression, developers of video codec and video chips and for all specialists involved in video processing. A number of unique features and a thoroughly designed interface make this program the helpful tool necessary when the playback of uncompressed YUV video files is required. The sample output from the YUV player is as follows

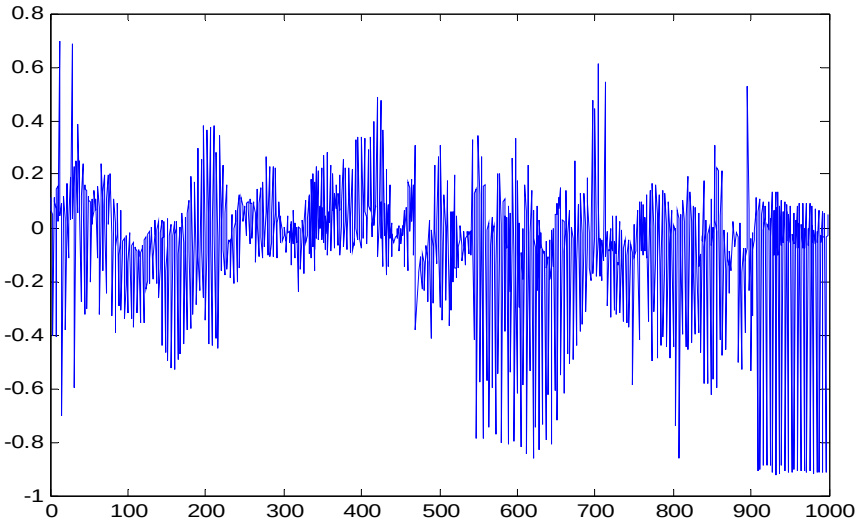
Figure 5.2: Sample output from YUVplayer deluxe of training data

5.2.3 Correlation coefficient

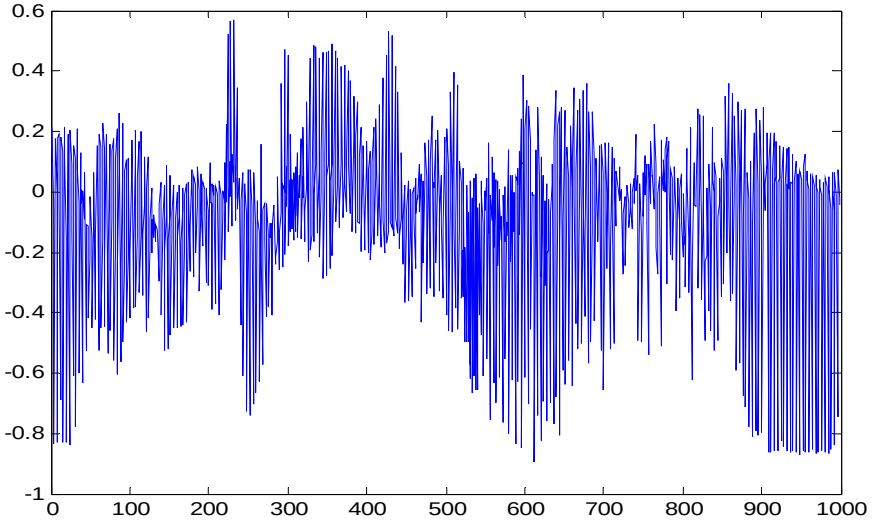
The correlation coefficient feature is first used for the training of purposed framework. The training of the framework is done on different test videos of soccer and basketball for different chroma format such as yuv420, yuv422 and yuv444 of training videos. The correlation coefficient for each y, u and v components is being computed and compared. One such measure of correlation coefficient of training video is given below.



(a)



(b)



(c)

Figure 5.3: Computation of correlation coefficient (a) luminance correlation

(b) chrominance u component (c) chrominance v component

The range of values of correlation coefficient ≥ 0.4 and 0.4 ≤ correlation coefficient ≤ 0.6 are being identified and thus used to measure the shot boundary detection between the two successive frames *x* and *y* after training a number of videos. The results depict the selection of this threshold as it identified majority of the shots in a video sequence. Figure 5.4 shows the value of correlation coefficient for test video. Note that the thresholds have been highlighted.

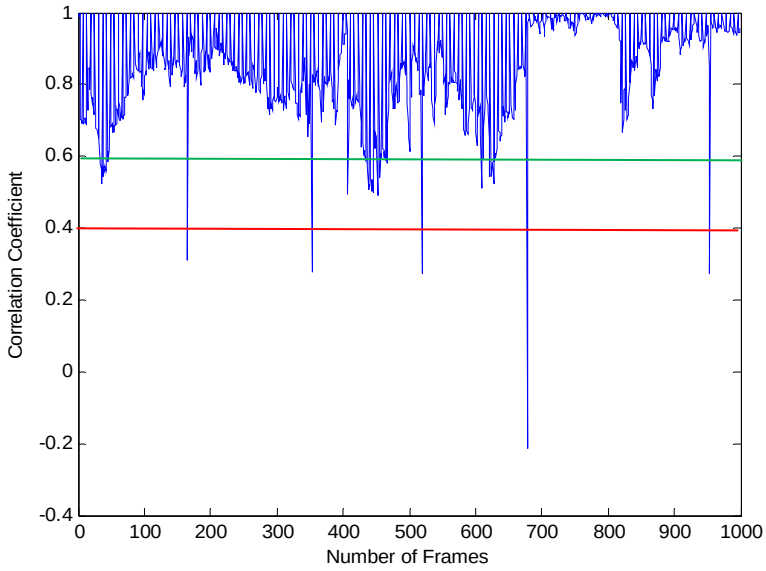


Figure 5.4: Correlation coefficient with the threshold

5.2.4 YUV histogram difference

The histogram feature is used to represent the color attribute, which determines the distance between adjacent video frames. It is based on the distribution of luminance as well as chrominance components. After finding the correlation coefficient we must have yuv histogram measure and since this histogram measure is to be computed with different bins each for y, u and v components. Different bins for {y, u, v} components in combinations of {16,4,4}, {16,16,16}, {32,8,8}, {32,16,16}, {32,32,32}, {64,16,16}, {64,32,32} and {64,64,64} are tested. The results gathered for one of training video with the combination of {64, 64, and 64} for each y, u and v components is given below.

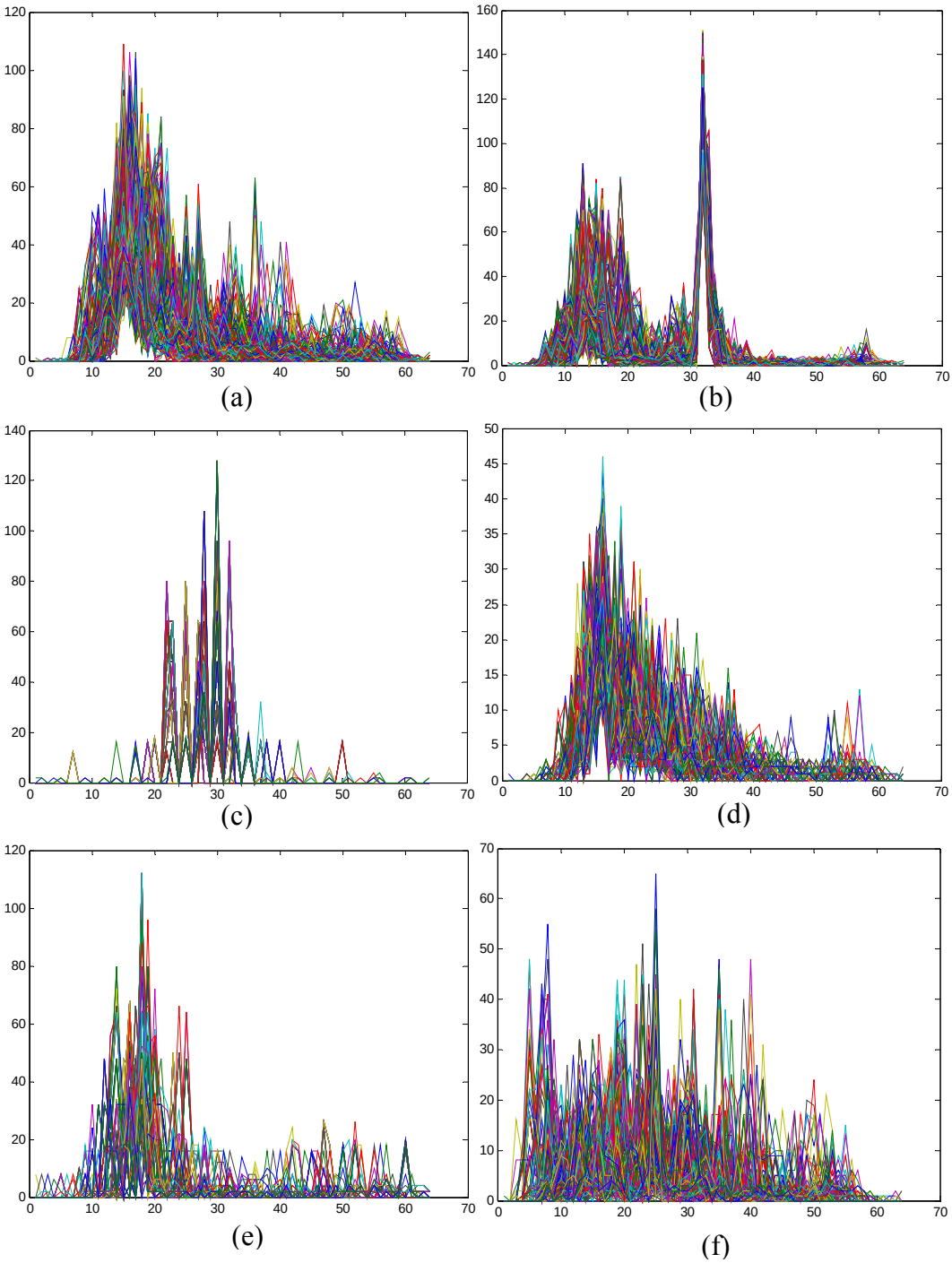


Figure 5.5: 64 bin histogram measure of luminance and chrominance components (a) histogram for y component predecessor (b) histogram for y component successor (c) histogram for u component predecessor (d) histogram for u component successor (e) histogram for v component predecessor (f) histogram for v component successor

As the luminance and chrominance components of yuv color space are being made independent in frame extraction module, therefore we can calculate the histogram

difference of successive video frames on each color component respectively. The maximum histogram difference between two adjacent frames is the second feature being used for computation. This kind of feature helps in identifying the slow motion of object present in the video, but in some cases of the rapid and quick change of objects within a shot will automatically result in false or miss detection. Figure 5.6 represents the results of yuv maximum histogram difference computed by taking 64 bin histograms of luminance and chrominance components as a feature.

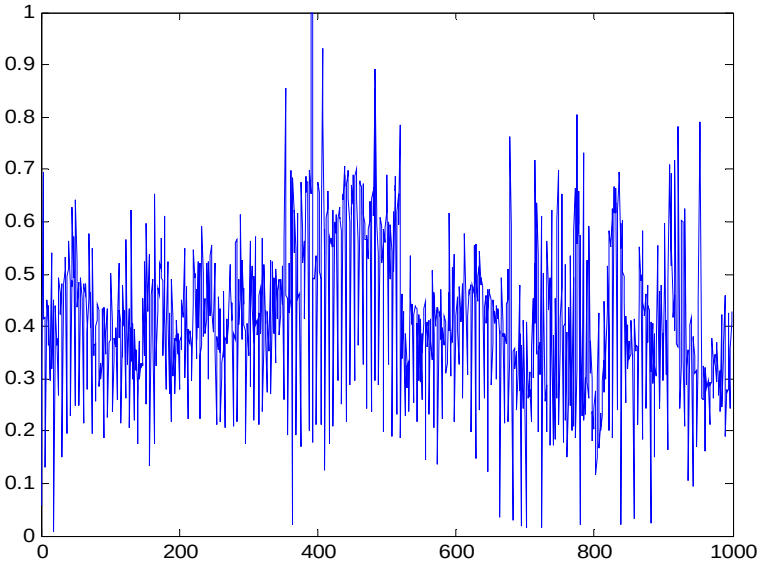


Figure 5.6: 64 bin YUV histogram difference between two consecutive frames

5.2.5 Running average difference

A running average, also called a moving average and sometimes a rolling average, refers to a numerical technique used to examine data points by generating an average of one small and little subset of all the data set present at any time. So a running average is not a single figure, but it is a set of statistics, each of which is the average of the equivalent subset. The simplest form of a running average is known as a simple running average and it is calculated by taking the arithmetic mean of a given set of values in a subset. Thus the running average difference is just the difference in the simple running or moving average values for a given subset. The results of difference between the

consecutive frames of video and their corresponding running average difference for any subset also known as window gathered for one of training video is given below in figure 5.7 and 5.8.

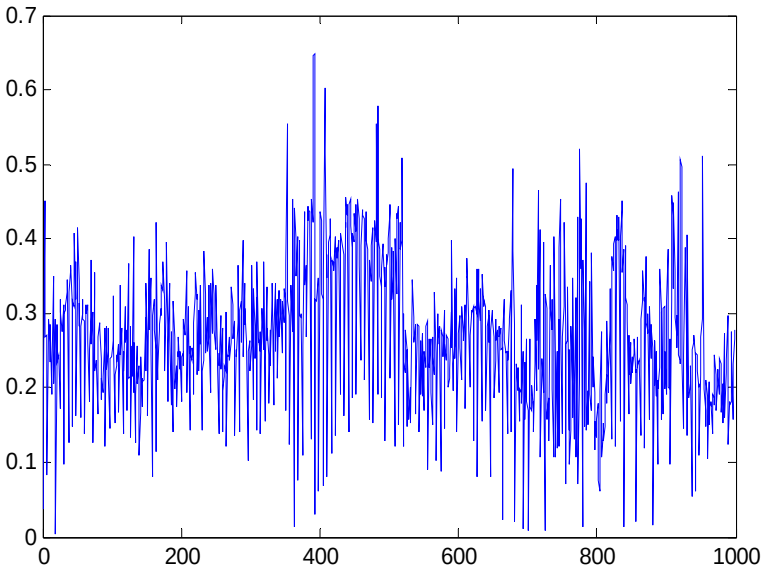


Figure 5.7: Difference between two consecutive frames

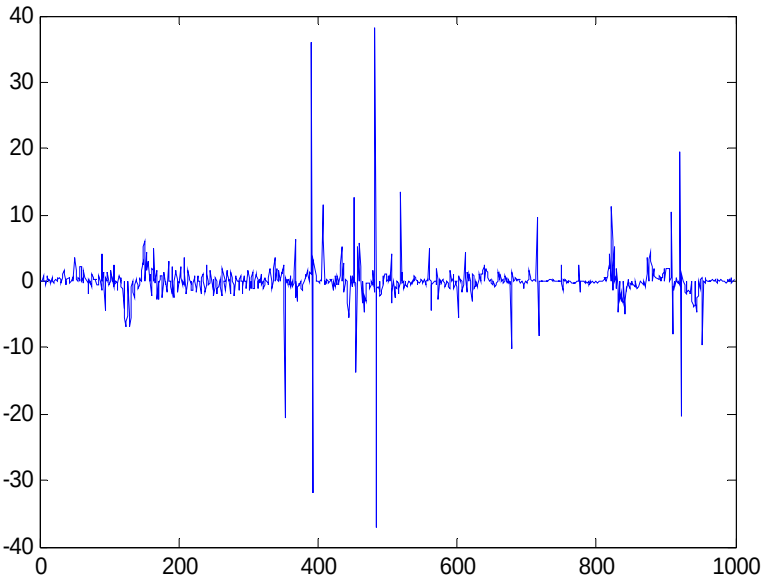


Figure 5.8: Running average difference between two consecutive frames

5.3 Results

To test the purposed framework the starting point constitute the two features i.e. Correlation coefficient and running average difference. The training phase of the framework included the divergent set of basketball and football sequences with either

1000 or 200 frames. For running average difference feature to be included we must be known about the window denoted by *n* in equation 4.4. The window range was started by including two consecutive frames and it was altered in discrete steps until 15.

The behavior of this increase effect in window range drastically affected the performance of shot detection. In the soccer test videos the size of window at 9 gives better results while for basketball video the window size of 15 extracted majority of the shots. The value 15 was chosen to be the window size of the framework as it was found maximum between the two values. The results gathered after training and then testing those videos which were not used in training by using these two features were inappropriate therefore addition of third feature namely yuv histogram difference was indispensible. To include and test this added classifier of yuv histogram difference measure, 8 bins each for y, u and v components are used as first attempt.

The results identified most of the shots but still there were some which were missed for different chroma format of yuv422 and yuv 444. Different bins were tested for {y, u, v} components in combinations of {16,4,4}, {16,16,16}, {32,8,8}, {32,16,16}, {32,32,32}, {64,16,16}, {64,32,32} and {64,64,64}. All shots of the basketball and soccer videos were identified at {32,32,32} and {64,64,64} bins respectively. The {64,64,64} bins was chosen as the measure because it identified both basketball and soccer test videos with the window size of 15 and for all chroma formats of yuv. After enumerating these different statistical features with thresholds as described in framework, winner take-all selection module is called. The table 5.2 summarizes the decision made by two state model and winner take-all selection scheme for one of the basketball video test sequence.

The basic measures of shot detection are recall, precision and accuracy which are used in evaluating the shot detection performance. Recall enumerates the percentage or proportion of shot boundaries that were correctly detected (shot changes in this case) by

the model. Precision quantifies the percentage or proportion of claimed shot boundaries that were actually shot boundaries i.e. what proportion of detected entries is correct. Accuracy reveals the temporal exactness of detected results, and corresponds to the measure of the distance between detected location of a transition by the framework and its true location.

Frame number	**0.4 ≤ Correlation coefficient ≤ 0.6**	**Histogram difference$_i$ ≥ 0.8**	**Running average dfference$_i$**		**Decision**
			≈ average	≈ 0	
209	209	-	-	209	Current shot
392	392	392	-	392	Current shot
408	408	408	408	-	Cut state
483	483	483	-	483	Current shot
776	776	-	776	-	Current shot
858	-	858	858	-	Current shot

Table 5.2: Decision based upon two state model and winner take-all selection

Two out of these three parameters often used to calculate and evaluate the performance of shot cut detection algorithms are recall and precision. Recall and precision can be defined as

$$\text{Recall} = \frac{C}{C + M} \tag{5.1}$$

$$\text{Precision} = \frac{C}{C + F} \tag{5.2}$$

In other words, recall is the fraction of those transitions claimed true transitions detected and precision is the fraction of those detected transitions that are actually correct (precision identifies the level of noise in the changes detected by the algorithm).

The experimental results are shown in table 5.3.

S/No	No. Of frames	No. Of cut detected	No. Of orignal cuts	False postives	Precision	Recall	Type
1	1000	18	17	1	94.44	100	Basketball
2	2000	24	27	2	91.67	81.48	Soccer
3	2000	40	36	5	87.50	97.22	Soccer
4	1000	18	16	3	83.33	93.87	Soccer
5	2000	39	32	7	82.05	100	Basketball
6	1000	13	13	1	92.03	92.03	Soccer
7	1000	18	18	0	100	100	Soccer
8	2000	40	38	4	90	94.73	Basketball
9	2000	61	51	11	81.96	98.03	Soccer
10	2000	102	108	7	93.13	87.96	Basketball
11	2000	32	29	5	84.37	93.10	Basketball
12	2000	34	28	7	79.41	96.42	Basketball
13	1000	9	10	1	88.89	80	Basketball
14	1000	4	4	0	100	100	Basketball
15	1000	12	9	3	75	100	Soccer
16	1000	9	8	1	88.89	100	Soccer

Table 5.3: Summary of cut detection for test data

The videos can be either having 1000 or 2000 frames having 4 to 102 cuts. As it can be seen from the table 5.3, the precision metric ranges from 75% to 100% while recall metric ranges from 80% to 100%. As it can be observed from the table, the precision and recall measures plunge drastically across different types of test videos. The reason behind such a drop is that arrangements of all experiments were done to test the robustness, consistency and stability of the cut detection method with increasing complexity in different events in the videos.

In an ideal condition, the proportion between recall and precision would both be equal to 1:0. However, it is very difficult to attain a high rank of recall without giving up precision. For example, it would be not important to achieve a recall equal to 1:0 by detecting all frame pairs as shot cuts, although the precision would be deprived. On the other hand, a precision equal to 1:0 could be achieved by detecting just one shot cut, resulting in a poor recall measure. Therefore, this algorithm attempts to maximize both recall and precision concurrently. It is worth noting, that the lowest precision reachable is defined by the percentage of total frame pairs that are shot cuts. The overall results for the cut detection for selected videos according to type of dataset used in the testing of the framework are shown in table 5.4

Dataset	Cut	Recall	Precision	Accuracy
Basketball	18	94.74%	94.44%	100%
Soccer	29	96.66%	96.55%	100%
Total	47	95.92%	95.74%	100%

Table 5.4: Experimental results of shot detection

It is not a matter of concern that the performance of an algorithm for any particular genre/type of dataset varies with different games. More importantly, the performance estimation of this method should echo how well it could potentially perform on unseen input video data. In other words, it must be calculated if the performance of a parameter set generalizes well to other sequences not being used in training of an algorithm.

Compared to other shot detection algorithms presented in [46] and [49], the recall, precision and accuracy of cut detection are better and comparable to ekin et.al's work [51]. These precision and recall measures for cut detection are also analogous to the state-of-the-art which use hidden markov models, stochastic context free grammar and dynamic bayesian networks etc. However the gradual transitions are not addressed in this

thesis. The approach of gradual transitions and coverage of other sports videos such as table tennis, baseball and cricket will need to be addressed and will be covered in the future.

Figure 5.9 shows the results of cut detection of one of the test video of 1000 frames of uncompressed data.

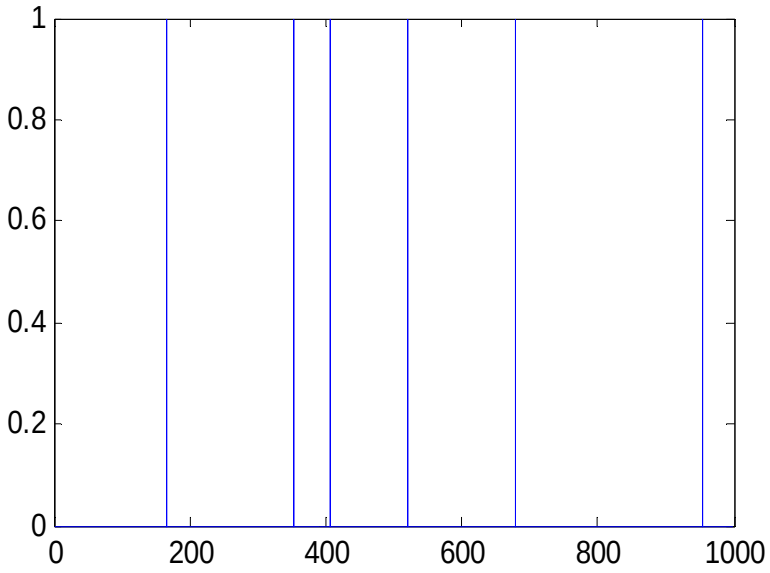


Figure 5.9: Cut detection from a test video

5.4 Summary

Sports videos are different from other kinds of video in the sense that it has extreme use of camera movements, moderately same background, motion of dissimilar objects and plodding shot transitions. Three types of features correlation, histogram difference and running average difference are extracted from the training of diverse sets of raw data and are used for testing a video sequence which was not being used during training phase. The selections of threshold and winner take-all schemes are used from the training phase to the newly input video to compute the cut detection of basketball and soccer sports videos. The recall and precision values show either a significant improvement in comparison with other approaches or are easily comparable given this shot transition. Experimental results reveal the effectiveness of the purposed approach.

Chapter 6

Conclusion

6.1 Overview

This chapter presents a summary of the thesis and concluding remarks followed by some suggestions for new directions and improvements. This thesis has considered two types of uncompressed sports videos viz. Soccer and basketball for the training and evaluation of framework for video shot detection which comes under video segmentation. Video segmentation has a wide range of applications such as pattern recognition, machine vision and content-based image/video retrieval.

The proposed methodology has been implemented on a stand-alone desktop pc and requires MATLAB® support for implementation. With the given size of data and type of content present in an uncompressed video, the time this program takes is directly proportional to the number of frames and type of format used for a .yuv file. Since the limitation is being placed on the number of frames which can be either 1000 or 2000 for evaluation, therefore those videos which have 2000 frames take approximately double the time to the videos with 1000 frames. The experiments were done on a 1.8 GHZ machine with 1024 MB Ram having an Intel architecture on which Microsoft Windows® XP is installed to successfully detect shots in the data.

However the response time by each module is directly proportional to the size of the data and the number of frames in the given video. But the overall results show that the proposed solution gives the optimal performance when it is compared with the other techniques that lie in the same spatial and temporal domains. Experiment justification of the purposed framework shows the versatility and significance of the results obtained in the work.

6.2 Limitation of approach

The research work provides a new framework to detect hard and soft cuts and thus identify different shots in uncompressed sports videos. All the work is based on research as well as exploring ideas from the previous work in existence. To a particular extent, the efforts are thriving by providing good results in comparison with previous approaches for this purpose. However, the proposed technique does have some limitations. At present, most of the media present over the internet consists of huge amount of data and this data also constitutes compressed videos. One constraint is the use of spatial and temporal domain which has been chosen for developing and testing the algorithms that work on un-compressed videos, efforts are needed to address for compact compressed data.

Due to the nature of this research work, the other constraint is the selection of two types of sports videos either basketball or soccer as most of the literature exists on this two sports videos, efforts must be placed for sports videos such as table tennis, baseball and cricket. A part from these another restriction which is being placed after the training of framework was the subdivision of video in multiples of 1000. Though this process works sound but it adds to the cost of the algorithm and must be elevated, moreover there must be a systematic process for selection of different thresholds during the training of framework and later for detecting the shots.

Another limitation is the selection of only three statistical metrics as the features. The use of motion vectors and its magnitude can also be useful for tracking different types of transitions which have not been addressed in this thesis. A part from these currently the fix thresholds are being used which have been identified during the training of framework. This fix threshold should be replaced by the use of Hidden Markov Models or some other statistical model existing in literature.

6.3 Future work

By taking a quick review of modern day applications, almost the competence and precision of all the applications are dictated by their desired accuracy of input and output correspondingly. After a brief analysis, it has been observed that present day applications can be categorized into two major areas. At one side it deals with compressed video (i.e. Working in frequency domain using dct, wavelet) on the other hand it deals with uncompressed videos (i.e. Working in spatial and temporal domain using color spectrum, gray scale computations etc). At present, this approach has been developed and tested for uncompressed sports videos in spatial and temporal domain which has its own limitations as discussed above.

One of the future works can be the enhancement to be made in the purposed approach to introduce better data compression techniques to reduce the size of the embedded data and to implement this technique over compressed videos such as .mpg, .avi, .mov etc. Other formats of yuv file such as yuv400 and yuv411 can be taken as input. Overlaying of different media such as goal scores and advertisements in sports videos can be addressed in future. The use of motion estimation and compensation techniques can be applied to lessen the timing constraints of processing in videos. Other sports videos and different genre types such as cartoons, drama, film and commercials can be taken in consideration for future use.

One problem which was not identified was detecting different events in any sports video involving numerous agents and their interaction. Event models can be learnt from training videos which having variations in the external agents and the temporal order of different events in the sports videos. Event learning can be formulated in a probabilistic framework, and the event models thus learnt after the framework during training can be used for cut detection in novel videos in future.

Bibliography

[1] N. Dimitrova, H.J. Zhang, B.Shahrarary, I. Sezan, T. Huang and N. Zakhor "Applications of video-content analysis and retrieval", IEEE Multimedia, vol 9, no.3, pp42-55, July 2002.

[2] U. Gargi, R. Kasturi and S. H. Strayer, "Performance characterization of video-shot-change detection methods," IEEE Transactions on Circuits and Systems for Video Technology, pp. 1-13, November 2000.

[3] R. Kasturi and R. Jain, "Dynamic Vision," Computer Vision: Principles, Eds. R. Kasturi, R. Jain, IEEE Computer Society Press, Washington, 1991, pp. 469-480.

[4] Arman, F., Hsu A., Chiu M.Y,"Image Processing on Encoded Video Sequences", Multimedia Sysems(1994) Vol 1, No.5, pp.211-219.

[5] Zabih R., Miller J., Mai. K,"A feature based Algorithm for Detecting and Classifying Scene Breaks", Proc. ACM Multimedia 95, San Fransisco, CA, November, 1995, pp. 189-200.

[6] Zhang H.J, Kankanhalli A., Smoliar S.W,"Automatic Partition of Full-Motion Video", Multimedia Systems (1993) vol 1, No.1, pp. 10-28.

[7] Nam J, Cetin E and Tewfik A, "Speaker Identification and Video Analysis for Hierarichal Video Shot Classification", Proc. Int. Conf. of Image Processing, Santa Barbara, CA, October 1997.

[8] Philips M, Wolf W, "Video Segmentation Techniques for News ", in Multimedia Storage and Archiving Systems, C.-C. Jay Kuo Editor, Proc. SPIE 2916, 243-251(1996).

[9] Saraceno, C. and Leonardi, R., "Audio as support to Scene Change Detection and Characterization of Video Sequences", Proc. International Conference on Acoustics, Speech and Signal Processing, Munich, Germany, April 1997, pp 2597-2600.

[10] Shahrary B., "Scene Change Detection and Content- Based Sampling of video Sequences", in Digital Video Compression: Algorithms and Techniques, Rodriguez, Safranek, Delp, Eds., Proc. SPIE 2419, February 1995, pp.2-13.

[11] Z. Cernekova, C. Kotropoulos and I.Pitas , "Video shot Segmentation using singular value decomposition", in Proceedings 2003, International Conference on Multimedia and Expo, Baltimore, Maryland, July 2003, vol 2, pp. 301-302.

[12] J. Nam and A. Tewfik, "Detection of gradual transitions in a video sequence using B-spline interpolation ", IEEE Transactions on Multimedia, vlo7, no.4, pp667-669, August 2005.

[13] R. Zabih, J. Miller and K. Mai, “A feature based algorithm for detecting and classifying production effects”, ACM Multimedia Systems, vol 7, no.1, pp 119-128, January 1999.

[14] R. Lienhart, S. Pfeiffer, and W. Effelsberg. “Video abstracting”, Communications of the ACM, 40(12): pp.54–62, December 1997.

[15] S. Antani, R. Kasturi, and R. Jain. “A survey on the use of pattern recognition methods for abstraction, indexing and retrieval of images and video”, Pattern Recognition, 35(4): pp. 945–965, April 2002.

[16] P. Aigrain, H. Zhang, and D. Petkovic. “Content-based representation and retrieval of visual media: A state-of-the-art review”, Multimedia Tools and Applications, 3(3): pp.179–202, 1996.

[17] R. Brunelli and O. Mich. “Image retrieval by examples”, IEEE Transactions on Multimedia, 2(3): pp. 164–171, 200.

[18] Y. Rui, T. Huang, and S. Chang. “Image retrieval: current techniques, promising directions and open issues”, Journal of Visual Communication and Image Representation, 10(4): pp. 39–62, 1999.

[19] S. Pfeiffer, R. Lienhart, S. Fischer, and W. Effelsberg. “Abstracting digital movies automatically”, Journal of Visual Communication and Image Representation, 7(4): pp. 345–353, 1996.

[20] A. Hanjalic and H. J. Zhang. “An integrated scheme for automated video abstraction based on unsupervised cluster-validity analysis”, IEEE Transactions on Circuits and Systems for Video Technology, 9(8): pp. 1280–1289, December 1999.

[21] F. Idris and S. Panchanathan “Review of image and video indexing techniques” Journal of Visual Communication and Image Representation, 8(2):146–166, 1997.

[22] R. Brunelli, O. Mich, and C. M. Modena “A survey on the automatic indexing of video data” Journal of Visual Communication and Image Representation, 10(2):78–112, June 1999.

[23] A. Hanjalic, “Shot-boundary detection: Unraveled and resolved?” IEEE Trans. Circuits Syst. Video Technol., vol. 12, no. 2, pp. 90–105, Feb. 2002.

[24] F. Idris and S. Panchanathan. “Review of image and video indexing techniques”, Journal of Visual Communication and Image Representation, 8(2): pp. 146–166, 1997.

[25] R. Brunelli, O. Mich, and C. M. Modena. “A survey on the automatic indexing of video data”, Journal of Visual Communication and Image Representation, 10(2): pp.78–112, June 1999.

[26] A. Hampapur, R. Jain, T. E. Weymouth, Production model based digital video segmentation, Multimedia Tools and Applications 1(1) (1995) 9-46.

[27] H.J. Zhang, C.Y. Low, S.W. Smoliar, Video parsing and browsing using compressed data, Multimedia Tools and Applications 1 (1995) 89-111.

[28] D. Lelescu and D. Schonfeld, "Statistical sequential analysis for real-time video scene change detection on compressed multimedia bitstream," IEEE Trans. Multimedia, vol. 5, no. 1, pp. 106–117, Mar. 2003.

[29] J. Nam and A. Tewfik, "Detection of gradual transitions in video sequences using B-spline interpolation," IEEE Trans. Multimedia, vol. 7, no. 4, pp. 667–679, Aug. 2005.

[30] Z. Cernekova, C. Kotropoulos, and I. Pitas, "Video shot segmentation using singular value decomposition," in Proc. 2003 IEEE Int. Conf. Multimedia and Expo, Baltimore, Maryland, July 2003, vol. 2, pp. 301–302.

[31] T. Kikukawa, S. Kawafuchi, "Development of an automatic summary editing system for the audio-visual resources", Transactions on Electronics and Information J75-A (1992) 204-212.

[32] A. Nagasaka, Y. Tanaka, Automatic video indexing and full-video search for object appearances, in Visual Database Systems II (E. Knuth and L.M. Wegner, eds.), pp. 113-127, Elsevier, 1995.

[33] N.H. Nagel, "Formulation of an Object Concept by Analysis of Systematic Time Variation in the Optically Perceptible Environment," Computer Graphics and Image Processing, Vol 7. 1978, pp. 149-194.

[34] M. J. Swain, Interactive indexing into image databases, in: Proc. SPIE Conf. Storage and Retrieval in Image and Video Databases, 1993, pp.173-187.

[35] G. Pass, R. Zabih, Comparing images using joint histograms, Multimedia Systems (1999).

[36] Y. Tonomura, Video handling based on structured information for hypermedia systems, in Proc. ACM Int. Conf. Multimedia Information Systems, 1991, pp. 333-344.

[37] U. Gargi, S. Oswald, D. Kosiba, S. Devadiga, R. Kasturi, Evaluation of video sequence indexing and hierarchical video indexing, in: Proc. SPIE Conf. Storage and Retrieval in Image and Video Databases, 1995, pp. 1522-1530.

[38] J.S. Boreczky, L.A. Rowe, Comparison of video shot boundary detection techniques, Proceedings IS&T/SPIE Intern. Symposium Electronic Imaging, San Jose, 1996.

[39] D. Swanberg, C.-F. Shu, R. Jain, Knowledge guided parsing in video databases, in Proceedings SPIE Conference. 1908, 1993, pp. 13-24.

[40] C.-M. Lee, D. M.-C. Ip, A robust approach for camera break detection in color video sequences, in: Proc. IAPR Workshop Machine Vision Appl., Kawasaki, Japan, 1994, pp. 502-505.

[41] B. Günsel, A. M. Ferman, A. M. Tekalp, Temporal video segmentation using unsupervised clustering and semantic object tracking, Journal of Electronic Imaging, 7(3), (1998) 592-604.

[42] T. N. Pappas, An adaptive clustering algorithm for image segmentation, IEEE Transactions on Signal Processing 40 (1992) 901-914.

[43] A. M. Ferman, A. M. Tekalp, Efficient filtering and clustering for temporal video segmentation and visual summarization, Journal of Visual Communication and Image Representation 9(4) (1998) 3368-351.

[44] P. Aigrain, P. Joly, The automatic real-time analysis of film editing and transition effects and its applications, Computers and Graphics 18(1) (1994) 93-103.

[45] H. Yu, G. Bozdagi, S. Harrington, Feature-based hierarchical video segmentation, in Proc. Int. Conf. on Image Processing (ICIP'97), Santa Barbara, 1997, pp. 498-501.

[46] J.S. Boreczky, L.D. Wilcox, A hidden Markov model framework for video segmentation using audio and image features, in: Proc. Int. Conf. Acoustics, Speech, and Signal Proc., 6, Seattle, 1998, pp. 3741-3744.

[47] Wolf, W. Hidden Markov Model Parsing of video programs, in proceedings international conference of acoustics, speech and signal processing (ICASSP-97), vol 4, April (1997) pp. 2609-2611.

[48] T. M. Bae, S.H. Jin and Y.M. Ro, "Video Segmentation using Hidden Markov Model with Multimodal Features", CIVR 2004, LNCS 3115,2004, pp401-409

[49] W. Zhang, J. Lin, X. Chen, Q. Huang and Y. Liu, "Video shot detection using hidden markov models with complementary features," Proceedings of the First International Conference on Innovative Computing, Information and Control (ICICIC'06), pp. 593-596, August 2006.

[50] H. Xiaodong, M. Huadong and Y. Haidong, "A Hidden Markov Model Approach to Parsing MTV Video Shot", in Proceedings Congress in Image and Signal Processing, 2008.

[51] A. Ekin, A. M. Tekalp and R. Mehrotra, "Automatic soccer video analysis and summarization," IEEE Trans. on Image Processing, July 2003, pp. 796-807