This is to certify that the
dissertation entitled

FUSION-BASED VIDEO SEGMENTATION AND SUMMARIZATION

presented by

John K. Dixon

has been accepted towards fulfillment
of the requirements for the

Doctoral degree in Computer Science & Engineering

Major Professor's Signature

Date

Fusion-Based Video Segmentation and Summarization

By

John K. Dixon

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

Computer Science & Engineering

2003

Abstract

Fusion-Based Video Segmentation and Summarization

By

John K. Dixon

This thesis examines the problem of video segmentation and summarization from a results fusion perspective. Many techniques have been developed for the segmentation and summarization of digital video. The variety of methods is partially due to the fact that different methods work better on different classes of content. Global histogram-based segmentation works best on color video with clean cuts and global intensity changes; local histogram-based segmentation is less sensitive to region changes in the video and therefore works better when scenes consisting of similar content are shot from different angles; DCT-based segmentation algorithms attempt are less sensitive to abrupt intensity changes due to lighting effects such as camera flashes; edge-based segmentation algorithms work well when high quality edge information can be extracted from the video sequence, motion-based summarization works best on video with moving cameras and a minimum of disjoint motion. Results fusion combines the properties of these varying algorithms into a common framework that can benefit from the advantages of each disparate approach. Recognizing that there is no single best solution for each of these problems has led to this work in integrating the variety of existing algorithms using results fusion methods.

The work is divided into four parts. The thesis begins with an in-depth study of the various video segmentation methods. This chapter categorizes the existing shot segmentation and summarization methods, noting their strengths and weaknesses. Next, results fusion based algorithms and implementations from a variety of fields are reviewed and studied so as to understand the methods that can be applied to video segmentation and summarization. This chapter examines results fusion research from the document retrieval and biometric communities and with an eye towards application to the video domain. The third part of this work presents the results of applying results fusion for video segmentation. This section compares and contrasts individual

algorithms with the results fusion implementations. Finally, it is demonstrated that the results fusion methodology used for video segmentation can be extended to video summarization.

Thesis Supervisor: Dr. Charles B. Owen Professor, Michigan State University

This research was supported in part by The MITRE Corporation.

To my mother Faye M. Dixon and brother Ian J. Dixon

ACKNOWLEDGMENTS

To my mother, Faye M. Dixon, thank you for all the support and love that you have shown me throughout my life. You have been a blessing and an angel from heaven. I could not have completed this thesis without you. There have not been words created to express the love and appreciation I have for you.

To my aunt, Dr. Brenda McClain, thank you for all the encouragement and financial support throughout my life. When I thought there was no way, you always provided one. You always have been an inspiration. I could not have finished this thesis without your love and generosity.

To my advisor, Dr. Charles Owen, thank you for being the best advisor a student could possibly have. You are the consummate professional and mentor. Thank you for your untiring support and encouragement in getting this thesis completed. I look forward to continuing to work with you in the future.

TABLE OF CONTENTS

List of Figures

Chapter 1

Introduction

Locating content in digital video continues to be a significant problem as massive quantities of digital video accumulate in corporate libraries, public archives, and home collections. A key element of any method that attempts to index digital video is effective *shot segmentation* and *summarization* of the video. Shot-based segmentation is the first step in determining the structure of the video by breaking it into the components that were originally edited together to form the final product. Summarization presents a pictorial summary of an underlying video sequence in a more compact form, eliminating the massive inter-frame redundancy present in digital video and film. Some researchers combine summarization and an additional process called *abstraction* into one process. Abstraction is the process of creating a series of still or moving images that is shorter in length than the original video and preserves the essential meaning of the video [88]. This thesis considers summarization and abstraction to be separate and unique operations and defines abstraction as the process of reducing the redundancy created in the summarization process due to repeated scenes or shots, maintaining the original structure of the video.

Most video content does not consist of a single continuous recording or filming, but rather a set of discrete recordings that have been edited together. Shot-based segmentation seeks to decompose this edited structure into the components used to construct the video. Shot boundaries can be created by various methods. The most elementary shot boundary is the hard cut. Other methods consist of production edits

such as fades, cross-fades, dissolves, and wipes. Some shot boundaries occur between two frames of video, while others occur between multiple frames. In order for a shot boundary detection method to be effective, it must be as accurate as possible, with few false positives (incorrectly identified shot boundaries) and false negatives (missed shot boundaries).

Indexing methods require minimization of redundancy for effective performance. Were a complete video sequence added on a frame-by-frame basis to an image database, the search mechanism would be forced to contend with hundreds of sequential frames with nearly identical content, making differentiation of results very difficult. Ideally, an indexing method would be operating on a compact *summarization* of the content with only salient elements subject to analysis. Additionally, given the limited performance of indexing methods and the questionable ability of humans to pose exact queries, it is essential that results be presented in a way that allows for very fast browsing by the user with a minimum of redundant results presented.

Great progress has been made on shot segmentation and summarization of digital video. However, it is common for much of this research to focus on specific classes of video or limited content corpuses. Major projects have analyzed news broadcasts, music videos, and late night comedians. Additionally, much of this work tends to focus on small sets of video content and has not been tested on a large video test suite. The current research has answered many questions about how to analyze video where the structure is known in advance. However, any general solution must work for a wide variety of content without the input of manually collected structural knowledge. This fact has been well known in the mature document analysis and the biometric communities for many years [5, 61, 48, 67, 70, 117, 139, 146].

1.1 Questions addressed by this thesis

This thesis examines the problem of creating new algorithms for shot segmentation and video summarization that improve on the performance of existing methods. The specific approach we apply is to examine the wide range of existing segmentation

and summarization methods and blend representative algorithms into a composite system using results fusion. The primary goal is to build a system that can utilize the strengths of the various algorithms, where appropriate to the underlying content, while avoiding the weaknesses when the method is inappropriate to the content.

1.1.1 Shot Segmentation

There has been considerable research on video segmentation techniques [10, 28, 47, 74, 86, 95, 157, 155]. Each of these segmentation techniques is successful at determining shot boundaries for specific *classes of video.* Color histogram-based algorithms build models of the dynamic change in color distribution between successive frames and are successful when applied to video sequences where the color distribution changes abruptly between shots [47, 95, 102, 157, 156]. Model-based algorithms use learning algorithms to construct models of the temporal nature of transitions and are effective when a set of transitions is known and expected in the content [11, 115]. Motion-based algorithms attempt to track content motion through image sequences and are effective when object or camera locations change significantly between shots [95, 144].

The wide variety of techniques forms a *toolbox* for the system designer from which an appropriate segmentation algorithm can be chosen for a given class of video. When the type of video is known a priori, any type of technique can be employed with relative success. However, if the type of video is unknown before analysis, certain assumptions cannot be made about the video and the best choice of algorithms cannot be predetermined, nor can the appropriate parameterization of the algorithms be made (setting thresholds and intervals for example). Research efforts, to date, have predominately focused on the independent implementation of individual segmentation algorithms [105, 106, 155, 157]. Limited research has attempted to combine shot boundary detection algorithms into a composite system [15, 119, 151, 158].

What constitutes good shot segmentation? In developing novel methods for video shot segmentation it is imperative to know what characteristics comprise good shot segmentation. Within a shot, a frame may differ from its neighboring frames by either camera and object movement, focal length changes, or lighting changes

[110]. A good shot segmentation algorithm should disregard frame changes within a shot. In addition, accuracy is an important factor in any shot segmentation method. The algorithm should attempt to maximize correct detections and minimize false and missed detections. In our view, missed detections are more costly than false detections. Missed shot boundaries can never be recovered, however false detections can be corrected during the summarization and abstraction process, which can eliminate the possible redundancy.

This thesis addresses the question about the degree to which the various segmentation algorithms can be integrated to create a composite technique. The new composite technique should be sensitive to the characteristics of the video under analysis and, therefore, applicable to a larger set of content without specific per-video tuning. This research necessarily began with a study of the effectiveness of the various segmentation algorithms on a wide variety of video. The classes of video that will be used for our study are commercial films, home video, news broadcasts, raw news footage, surveillance video, and Unmanned Aerial Vehicle (UAV) military video. The video includes both edited and unedited footage. The results of our algorithm analysis provide important information as to the characteristics that each segmentation technique exhibits for a given type of video. These characteristics are then utilized when developing a novel adaptive shot-based segmentation algorithm.

Our approach to tackling the problem of shot segmentation for a wide variety of video classes adopts results fusion techniques from the document retrieval and biometric community. Results fusion can be thought of as a decision function that has multiple inputs and produces an output that is based on the characteristics of the inputs. This thesis has developed a fusion-based shot segmentation algorithm that fuses together several key features of digital video and performs shot segmentation based on the characteristics of those features. The features were chosen based on their ability to capture important information contained in the video sequence. The key features that are used to characterize a video segment are color, texture, motion, and compressed image characteristics. In our approach, shot segmentation is treated as a binary classification problem in which each frame in a video sequence is or is

not considered as a shot boundary. Results fusion strategies using Support Vector Machines (SVMs), Decision Trees, Rulesets, and Neural Networks are used to fuse the multiple features to determine a higher-quality, more accurate, segmentation. Prior research has proven that these methods produce good performance for solving binary classification problems [19, 39, 75, 108, 111, 139, 149].

1.1.2 Summarization

This thesis also demonstrates that results fusion and adaptation techniques can be applied to video summarization. One of the critical tools of any indexing and browsing environment is effective summarization. Video to be indexed must be presented to an indexing system with a minimum of redundancy so as to avoid redundant retrieval results and to maximize the disparity in the indexing space. Likewise, search results must be presented to human users as compact summaries that allow users to quickly browse through the candidate choices and choose the correct result or adapt the search as quickly as possible. Again, many different approaches for video summarization exist. This toolbox of approaches is utilized as the basis for an adaptive solution for video summarization that draws on the strengths of the different approaches in different application classes.

Video abstraction is the mapping of an entire video segment into a smaller number of representative images [157]. It has been recognized that representing a complete video shot with a single image is an important step towards representing video in a compact meaningful form [131]. These images may be extracted frames from the actual video sequence or composite images constructed from the sequence using salient stills [134] methods or image mosaics [63, 97, 96]. Although these images are single frames, they do not represent one discrete moment in time. Moreover, these images represent the aggregation of temporal changes that occur within a moving image sequence with the salient features preserved [16]. This abstraction has traditionally been done manually in film and video libraries. The huge volumes of video data accumulating today require fully automated techniques to reduce the role of human involvement as much as possible. However, in some instances this representation

may not be enough to capture the dynamic action in complex shots. As a result, researchers have experimented with developing moving image abstracts of shots.

There has been a considerable amount of research on automated video abstraction techniques [23, 24, 25, 55, 73, 85, 97, 96, 112, 126, 127]. Summarization involves reducing the redundancy created in the abstraction process due to repeated scenes or shots, maintaining the original structure of the video. Many techniques exist for developing video summaries. These techniques include moving images abstracts, video skims, and keyframe extraction.

What constitutes good summarization? Good summarization should seek to eliminate redundancy and present the video in a compact form that allows for maximum user retention and comprehension. It should briefly and concisely present the contents of the original video [118]. Its length should be shorter than the original video sequence, but how much shorter? It should only focus on the content that is important to the user, but what is important? One of the main problems with any summarization technique is that the answers to the previous questions vary from user to user. In some cases, users may need to view a few still images of the video sequence, and in others, users may need to view a short clip segment extracted from the longer video. There exists no optimal summary form. Additionally, it is difficult to measure the performance of any summarization technique in terms of a quantifiable result. The best we can do is to compare the summary to what a human user would consider optimal.

This thesis examines the integration of various summarization techniques to develop an effective composite method that, again, is generally applicable to a wide class of content. Determining the most effective summarization technique for a given video source is a difficult research problem. The summarization method should present the user with the most effective means of organizing the data for maximum understanding and saliency. Understanding the summarization output is not a well-defined concept and is likely a user-dependent concept. Additionally, the summarization method must be able to adapt to the underlying video content, presenting the user with the most effective interface for viewing the content. Moreover, a summarization method

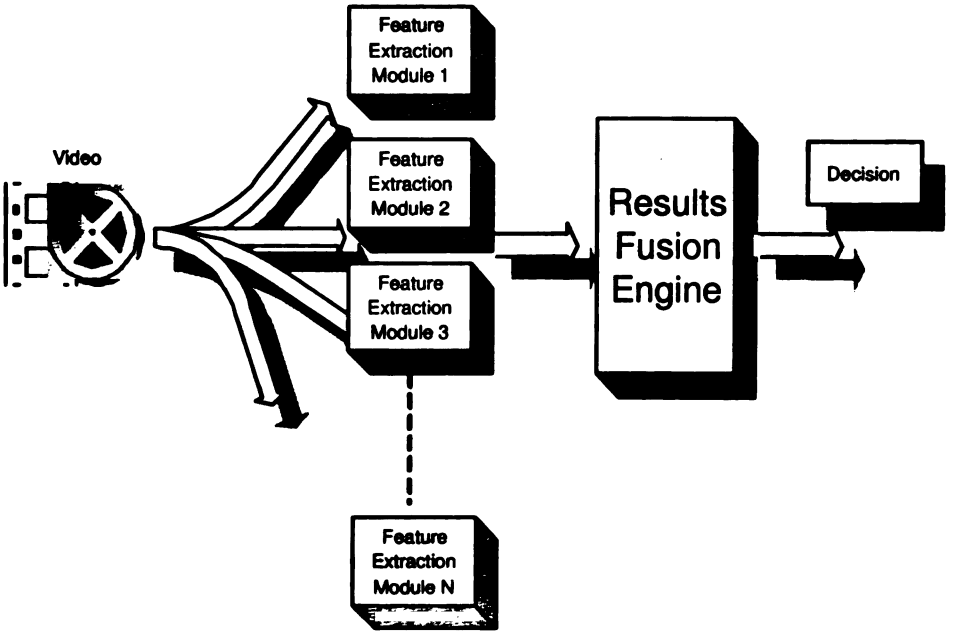

Figure 1.1: Proposed Results Fusion System

should present a concise summary that provides an overview of the video, reducing redundant information created by repeated scenes or alternating shots.

1.1.3 Results Fusion

This thesis focuses on the composition of existing algorithms into a single composite algorithm with considerably improved performance and greater general applicability. The approach that has been chosen is to combine the feature output of multiple algorithms using *results fusion*. Results fusion (also referred to as sensor fusion in the robotics community, classifier fusion in information retrieval, and multimodality fusion in biometrics) is the process of combining multiple evidence or sensors to improve the performance and reliability over utilizing a single evidence or sensor. The performance of any system is predicated upon the reliability of the sensor that is used. If a single sensor is used, it may be subject to errors or unpredictable behavior in certain environments. The result of such a sensor could be unreliable. Results fusion attempts to address this problem by utilizing multiple sensors that capture different aspects of the data under analysis. Each sensor itself may not provide superb performance, however the appropriate combination of these sensors may produce a

reliable and quality result.

An important feature of results fusion methods is the ability of the results fusion engine to generalize or adapt to novel patterns or input data [133]. In order to obtain a system that generalizes well, researchers have experimented with using multiple classifiers with each classifier generalizing differently, utilizing separate features, modalities, or representations. In order to use all the information available, the results or features of the multiple classifiers must be combined to make a final decision.

Several methods exist for results fusion. As an example, considerable research on results fusion has come from the area of document retrieval [6, 44, 61, 81, 82, 83, 94, 123]. In this area, researchers attempt to combine multiple representations of queries and documents or multiple retrieval techniques. Research in this field has shown that significant improvements can be achieved by combining multiple evidence [6, 83]. Results fusion has also become popular for personal identification and authentification, where different methods (fingerprints, retinal scans, and other biometric measures) have limited individual reliability, but allow for greater reliability when used in conjunction, reducing the false acceptance of imposter cases [9, 48, 117, 139]. Research in handwriting and character recognition attempts to combine multiple line segments via results fusion to identify handwritten numerals [80, 145]. Additionally, research in the sensor fusion community has indicated receiving improved performance and reliability over using a single sensor. Researchers in this community attempt to fuse information retrieved from metal detectors, ground penetrating radar, and thermal infrared imagers for the detection of land mines [27, 121].

There are important similarities between the composition of multiple methods in the document filtering community and in creating a fusion-based video segmentation technique. Just as some document filtering algorithms require different document representations [70] to improve retrieval performance, the various video segmentation algorithms utilize different abstract video representations (motion maps, histograms, etc.). Additionally, just as some document filtering algorithms fuse results from previous algorithms to improve performance, the results of various video segmentation

algorithms can be fused to create an improved, unified result.

In general, there is no guarantee that the fusion of multiple strategies will improve performance over individual methods. For example, if an accurate sensor is fused with one that generates random results, then no improvement is realized. However, empirical evidence from the document analysis and biometric community indicate that fusion is beneficial and improves performance. For example, if the optimal strategy is unknown, the fusion of multiple methods can be advantageous even if the fusion results are worse than the best individual strategy.

1.1.3.1 Document Analysis

Some document analysis researchers attempt to combine multiple representations of queries and documents or multiple retrieval techniques. An example application that incorporates results from multiple input sources is the *metasearch engine.* A metasearch engine is a system that provides access to multiple existing search engines. Its primary goal is to collect and reorganize the results of user queries by multiple search engines. There has been considerable research regarding the effectiveness and performance of metasearch engines [8, 21, 34, 49, 52, 60, 79, 99]. The result merging step of a metasearch engine combines the query results of several search engines into a single result. A metasearch engine usually associates a weight or similarity measure to each of the retrieved documents and returns a ranked list of documents based on this value [99]. These weights can be derived from an adjustment of the document rank value of the local search engines or by defining a global value based on all of the retrieved documents. This approach only focuses on the results of multiple searches and not a combination of the multiple search engines functionalities. Metasearch engine research is complicated by the fact that differing search engines produce rank results that are heterogeneous and not easily compared.

Document retrieval methods have also shown increased performance when combining results from various document representations. Katzer, et al. [70] compared text document retrieval performance using different document representation methods. Their results demonstrated that the different document representations retrieved

different sets of relevant documents and that performing information retrieval with multiple document representations improved retrieval performance over using a single method. Additionally, Bartell, et at. [6] and Shaw, et al. [123] combined multiple retrieval algorithms and obtained better performance than using a single method. Hull, et al [61] combined probability estimates of multiple classifiers to improve document filtering performance.

As the document retrieval methods rely on various document representations, the various video segmentation algorithms rely on different characteristics of the underlying video as abstracted into some intermediate representation, such as a feature vector or representative image. Color-based methods create histograms of the frame content color distribution and compute distance metrics between these histograms to search for shot boundaries [47, 86, 95, 102, 156, 157]. Model-based methods create Hidden Markov Models (HMM) of each possible state and transition in a video sequence that are used to locate transitions as temporal events [11, 115]. Edge-based methods utilize derived edge maps of the frame content to search for shot segmentation boundaries [86, 152]. Motion-based methods rely on derived velocity and displacement vectors to compute the amount of motion between video frames to determine shot boundaries [31].

1.1.3.2 Biometrics

Many researchers have focused on the fusion of face and voice data to improve personal identity verification [9, 17, 67, 117, 146]. Several studies have shown that using a multi-modality biometric system can improve on the incompleteness of any single model biometric system [117]. Yacoub [9, 146] uses multi-modal biometric features to fuse face and voice information together via a supervising expert. Person identification is treated as a binary classification problem, with each user either belonging to the imposter class or the client class. Given the scores from the face and voice identification modules, the supervising expert finds the optimal function that separates the two classes to make verification decisions.

Genoud, et al. [48] combined several speaker verification methods to improve per-

formance. The decision functions of each verification method are weighted with a confidence measure. The average confidence measure is tested against a global threshold to determine speaker authentication. Their research concluded that increased performance can be achieved by combining multiple methods.

Ross, et al. [117] fused the results from face, fingerprint, and hand geometry analysis to increase the performance of a biometric system. In their research, the decision function of multiple biometric systems is consolidated via a summation rule, which takes a weighted average of the individual scores. From their research it was concluded that increased performance could be obtained by using multiple modalities.

The results of this research suggest that increased shot segmentation and summarization performance can be achieved by fusing multiple algorithms over using a single method.

1.2 Contributions of this thesis

This thesis contributes to the overall research in digital video in seven distinct ways. Results obtained by the document analysis and biometric communities using fusion of multiple methods to increase performance suggest that an improvement of shot segmentation and summarization could be achieved by fusing together multiple methods. To achieve this goal, we select appropriate shot segmentation algorithms and examine a variety of fusion-based techniques, constructing a composite method for shot segmentation.

We then extend the general ideas and results developed in this work to summarization. The goal has been not only to develop an improved method for video summarization, but also to demonstrate the extensibility of the general concept of results fusion.

Some of the techniques that we have used to extract the features and determine shot boundary detection are not necessarily new, however our implementation of them is. Additionally, we test our newly developed algorithms on a wide variety of different video classes. To date, much of the research in the field of digital video has been done

using small test samples of structured content. Few researchers have attempted to test algorithms on large video suites that encompass the complexity and variety of different classes of content. The classes of video that were used for our study include commercial films, home video, news broadcasts, raw news footage, surveillance video, and Unmanned Aerial Vehicle (UAV) military video.

To summarize, the contributions in this thesis are:

- A Decision Tree and Ruleset based results fusion engine for shot segmentation and summarization that improves performance over using a single algorithm.
- Neural Network results fusion for shot segmentation and summarization that improves performance over using a single algorithm.
- Support Vector Machine-based results fusion for shot and video summarization that fuses key features from video and determines a best segmentation with extensions to summarization that improves performance over using a single algorithm.
- Feature extraction and analysis for results fusion.
- Experimental validation on a large and varied video test suite.
- Adaptability to detect shot boundaries when receiving unreliable data from one or more modalities.
- A novel keyframe-based video summarization technique based on camera motion.

When video enters the proposed system, it is analyzed by each feature module. The feature extraction modules are used to extract low-level image features. These feature modules extract the necessary features and output a measure for each video frame representing the difference metric between successive frames. The outputs of the features modules are combined using a results fusion engine. The output of the results fusion engine is a decision as to whether the current frame under analysis is a shot boundary. Figure 1.1 shows a graphical representation of the proposed system.

1.3 Definitions

This section describes some of the terms and symbols that appear throughout this thesis. Images in this thesis are presented in color.

- *cut*: An abrupt boundary that occurs where there is a change of shots between two consecutive frames. Figure 1.2 shows a graphical representation of a cut between pairs of frames in a music video sequence.

- *dissolve*: The simultaneous occurrence of a fade-in and fade-out, with both effects superimposed over a span of frames [28]. Figure 1.3 depicts a dissolve video edit.

- *fade*: A gradual transition of video content to or from black (or some other fixed color frame). A *fade-out* occurs when there is a gradual change fade to a black screen, while a *fade-in* occurs when there is a gradual fade from a black screen. Figure 1.4 is a graphical representation of a fade-in video effect.

- *keyframe*: A still image that best represents the content of a video sequence in an abstract manner [157]. There is no clear criterion for selecting keyframes and systems vary considerably on what is considered the best choice for keyframe selection or construction.

- *scene*: A logical grouping of shots focusing on certain objects of interest [109].

- *shot*: A sequence of frames captured as a single continuous action in time and space [109].

- *wipe*: A transition from one shot to another by selectively uncovering a contiguous region of the image, often rectangular. The effect is often like a virtual line passing across the image, clearing one picture while it brings in another occurring over a span of frames. Figure 1.5 shows a graphical representation of a wipe video effect.

Figure 1.2: Cut Between Successive Frames

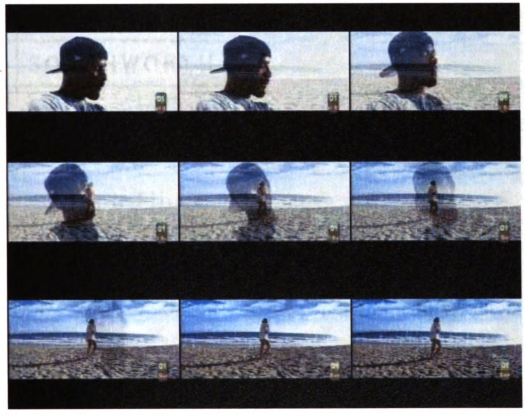

Figure 1.3: Dissolve Video Edit

Figure 1.4: Fade In Video Effect

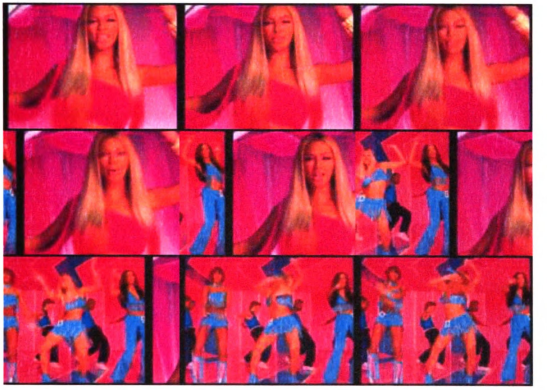

Figure 1.5: Wipe Video Edit

1.4 Stucture of this thesis

The remainder of the thesis is organized as follows. Chapter 2 reviews existing shot segmentation, summarization, and results fusion methods, noting strengths and weaknesses. Chapter 3 examines the shot detection methods implemented in this thesis. Chapter 4 examines result fusion-based algorithms and implementations for video segmentation. Experimental evaluation and testing is detailed in Chapter 5. Chapter 6 discusses how our novel results fusion-based shot segmentation methods can be extended for video summarization. A summary and proposed future work is detailed in Chapter 7.

Chapter 2

Related Work

The widespread distribution and storage of digital video has presented many challenges. The challenges arise because of the nature and characteristics of digital video. It is an inherently voluminous and redundant medium. In order to effectively utilize this medium it must be transformed into a form that is searchable, manageable, and structured. Temporal video shot segmentation is the first step towards achieving this goal. Any method that attempts to index, browse, retrieve, or parse digital video must be segmented. The goal of video shot segmentation is to present a longer video sequence as a set of smaller more manageable segments called shots. A shot can be characterized as a sequence of frames captured as a single continuous action in time and space [109]. Each shot is then mapped into a smaller number of representative images via an abstraction process. The abstraction process creates a series of still or moving images that is shorter in length than the original video and preserves the essential meaning of the video [88]. Summarization attempts to reduce the redundancy created in the abstraction process due to repeated scenes or shots, maintaining the original structure of the video.

Results fusion is the process of combining multiple sensors or classifiers. It is considered a general problem in various application domains such as face recognition, text categorization, person authentification, and optical character recognition. The premise behind results fusion techniques is that better accuracy and reliability can be obtained by fusing multiple evidence or sensors over using a single evidence or

sensor. If a single sensor is used, it may be subject to errors or unpredictable behavior in certain environments. The result of such a sensor could be unreliable. Results fusion attempts to address this problem by utilizing multiple sensors that capture different aspects of the data under analysis. Each sensor itself may not provide superb performance, however the appropriate combination of these sensors may produce a reliable and quality result.

The purpose of this chapter is to give an overview of the existing methods for video segmentation and summarization. The performance and limitations of each algorithm are discussed and compared. Additionally, this chapter will discuss the current research and methods for results fusion.

2.1 Video Systems

There has been much research in the development of composite video systems that allow users to store, search, and browse digital video [56, 105, 106, 141, 140]. The Físchlár project at Dublin City University is a visual indexing system that allows users to store and browse television programs. The Informedia I and II Project at Carnegie Mellon University utilizes speech information, image analysis, and natural language processing on over a terabyte of video data to facilitate video search, navigation, and retrieval [56, 141, 140]. Video segmentation and summarization are important aspects of these systems. Video segmentation is the first step in any digital video analysis system. Its goal is to capture the underlying structure of the video sequence and divide the video stream into logical subunits [156]. Most segmentation algorithms operate on the shot level; however there has been some research on segmenting video on the story level. Figure 2.1 illustrates the hierarchy of a typical video. The second step in any composite digital video system is the summarization of a video sequence. The goal of summarization is to reduce the redundancy in the segmentation process caused by repeated scenes or long shots. Both segmentation and summarization work together as the foundation of any composite video system.

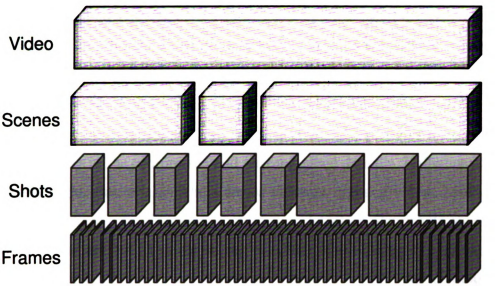

Figure 2.1: Typical Video Structure

2.2 Segmentation

A typical video sequence is composed of scenes, which are composed of shots, which are composed of frames at the lowest level. Figure 2.1 shows the structure of a typical video sequence. Most common segmentation algorithms rely on low-level image features at the shot level to partition the video. Attempting to partition the video at the scene or story level is difficult; there is no standard or universal definition of scenes or stories.

A shot is an unbroken sequence of frames taken from one source [74]. There are two basic types of shot transitions: abrupt and gradual. Abrupt transitions, called cuts, occur when a frame from a subsequent shot immediately follows a frame from the previous shot. Gradual transitions consist of slow change between frames from one shot to frames of a different shot. These types of transitions include cross-dissolves, fade-ins, fade-outs, and other graphical editing effects such as wipes [47]. A fade-in is the gradual increase of intensity starting from one frame to the next. A fade-out is a slow decrease in brightness from one frame to the next. A cross-dissolve is when one frame is superimposed on another, and while one frame gets dimmer, the other frame gets brighter. A dissolve can be considered an overlapping of a fade-in and a

fade-out [28]. Gradual transitions are more difficult to detect because camera and object motion can inhibit the accurate detection of gradual transitions, causing false positives.

There have been several research projects comparing and evaluating the performance of shot detection techniques. Koprinska, et al. [74] provides a survey of the existing approaches in the compressed and uncompressed domain. Dailianas [28] compared several segmentation algorithms across different types of video. Lienhart [86] evaluated the performance of various existing shot detection algorithms on a diverse set of video sequences with respect to the accuracy of each detection method, and the recognition of cuts, fades, and dissolves. Lupatini, et al. [95] compared and evaluated the performance of three classes of shot detection algorithms: histogram-based, motion-based, and contour-based. Boreczsky and Rowe [10] compared various compressed and uncompressed video shot detection algorithms. All of these temporal video segmentation algorithms can be categorized as either a compressed or uncompressed domain technique.

2.2.1 Uncompressed domain techniques

The majority of segmentation algorithms operate in the uncompressed domain. Typically, a similarity measure between successive frames is defined and compared against a predetermined threshold. A cut is determined when the distance value between two images falls below this predetermined threshold. Gradual transitions can be found by using complex thresholding techniques [156] or using a cumulative difference measure. Figure 2.2 categorizes the various uncompressed shot segmentation techniques. The uncompressed algorithms can be organized into the following categories.

2.2.1.1 Pixel Differences

One way to detect the possible changes between successive frames is to compare the corresponding pixel values between the two frames and count how many pixels have changed. If the number of changed pixels is above a predetermined threshold, a shot

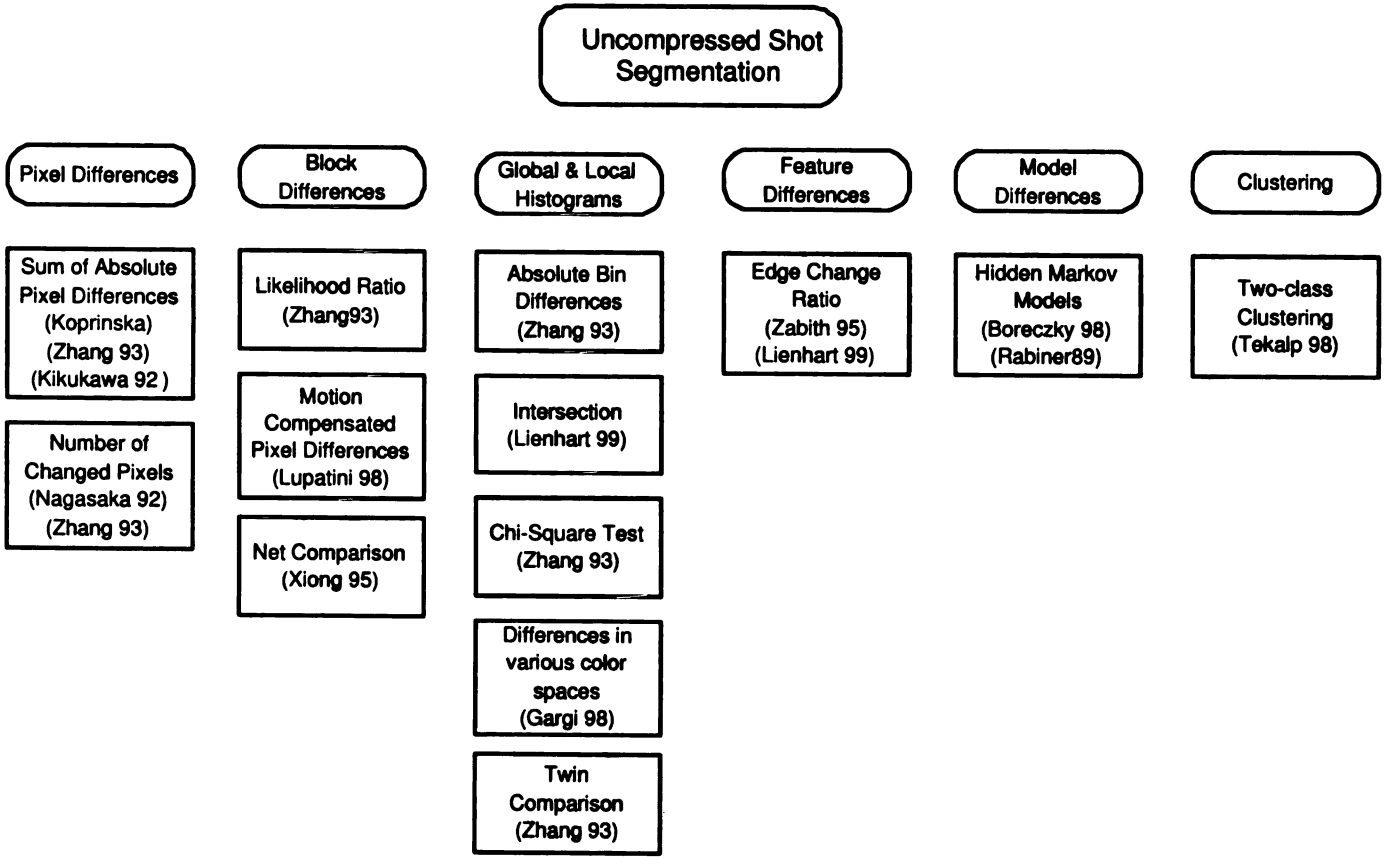

Figure 2.2: Uncompressed Shot Segmentation Techniques

is detected. Koprinska [74] calculates the absolute sum of pixel differences between two successive frames as:

$$D(i,i+1) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}|P_i(x,y) - P_{i+1}(x,y)|}{X \cdot Y} \tag{2.1}$$

where $P_i(x,y)$ and $P_{i+1}(x,y)$ represent pixel intensity values at coordinates (x,y). A cut is determined if the difference value $D(i,i+1)$ is above a predetermined threshold. One potential problem with this implementation is extreme sensitivity to camera motion. If the camera moves a few pixels between successive frames, a large number of pixels will be counted as being changed. Zhang, et al. attempted to reduce this effect with the use of a smoothing filter [156]. Before each pixel comparison the candidate pixel is replaced with the average value of the pixels within a $3x3$ neighborhood. Additionally, this filter reduces noise in the input images.

2.2.1.2 Statistical Differences

Zhang, et al. uses a likelihood ratio to compare successive frames based on the assumption of uniform second-order statistics over regions in each frame [156]. In this algorithm each frame is subdivided into k blocks and the corresponding blocks are compared based on the statistical characteristics of their intensity values. The likelihood ratio that two blocks come from different scenes can be expressed as [156]:

$$\lambda_k = \frac{[\frac{S_i+S_{i+1}}{2} + (\frac{m_i-m_{i+1}}{})^2]^2}{S_i * S_{i+1}} \tag{2.2}$$

where m_i and m_{i+1} are the mean intensity values for the two blocks k and S_i and S_{i+1} are the respective variances in consecutive frames i and $i+1$. The number of blocks whose likelihood ratio exceeds a threshold T_1 is counted as follows:

$$D(i,i+1,k) = \begin{cases} 1 & \text{if } \lambda_k > T_1 \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

If the number of changed blocks exceeds a second threshold T_2 a cut is declared. One advantage that this method has over the pixel difference method is that it improves the tolerance against noise associated with camera and object movement. Additionally, the likelihood ratio has a broader dynamic range than does the percentage used

with the pixel difference method [156]. This broader dynamic range makes it easier to choose a threshold t to distinguish between changed and unchanged areas. One problem with the likelihood ratio algorithm is that it is possible for two different corresponding blocks across a shot boundary to have the same density function, causing no cut to be detected. Another problem with this method is that, compared to the pixel difference method, its performance is slower as a result of the complexity of the statistically-based formula.

2.2.1.3 Histograms

The most commonly used structure for color-based segmentation is the histogram. Ranges of colors are divided into buckets and the number of pixels assigned to each bucket forms the color histogram of the video frame. Histogram-based algorithm techniques use a distance metric between histograms as a similarity measure. The basic assumption is that content does not abruptly change within, but across shots [86]. Once an image has been represented as a histogram there are various distance metrics that can be used. Some of the most commonly used distance metrics are:

- Chi-square:

$$\chi^2 = \sum_{i=1}^{k} \frac{(H_i(j) - H_{i+1}(j))^2}{H_{i+1}(j)} \tag{2.4}$$

- Intersection:

$$Intersection(H_i, H_{i+1}) = 1 - \frac{\sum_i min(H_i(i), H_{i+1}(i))}{N} \tag{2.5}$$

- Absolute Bin Difference:

$$ABD(H_i, H_{i+1}) = \sum_{i=1}^{n} |H_i(i) - H_{i+1}(i)| \tag{2.6}$$

Zhang et al. [156] concluded that the χ^2 method of histogram comparison enhances the difference between two frames across a cut, however it also increases the difference between frames with small camera and object movements. Additionally, the overall performance of the χ^2 method is not much better than the absolute bin difference

method, with χ^2 being more computationally intensive. Zhang et al. [157] computed the distance between color histograms as:

$$D(H_i, H_{i+1}) = \sum_{i}^{N} \sum_{j}^{N} a_{ij}(H_i(i) - H_{i+1}(i))(H_i(j) - H_{i+1}(j)) \tag{2.7}$$

where the matrix a_{ij} is derived from human perception studies. If the distance metric exceeds a threshold, a cut is selected. Histograms are attractive because they are effective at determining abrupt changes between frames. They are also tolerant to translational and rotational motions about the view axis, and change gradually with the angle of view, change in scale, or occlusion [157]. However, they completely ignore the spatial distribution of intensity values between frames. Consecutive frames that have different spatial distribution, but have similar histograms, are considered similar. Zhang et al. [156], and Nagaska and Tanaka [102] are examples of color-based histogram implementations.

One solution to the problems associated with global histograms is to create local histograms. Local histograms segment a frame into blocks and compute histograms for each block. This method is tolerant to local changes in motion, however it is still sensitive to changes in luminance over and entire frame [95]. Nagaska and Tanaka [102] split each frame into 16 blocks of equal size and evaluate the difference between histograms of the corresponding blocks. The χ^2 method is used to compute the distance metric between frames. The largest difference value is discarded in order to reduce the effects of noise, object and camera movements.

Gargi, et al. [47] experimented with computing histograms for various shot detection methods in different color spaces. The color spaces include RGB, HSV, YIQ, XYZ, L*a*b*, L*u*v*, Munsell, and Opponent. They concluded that the Munsell space produced the best performance results. The Munsell space is used because it is close to the human perception of colors. Additionally, Zhang, et al. [157] used a dominant color technique only using the most dominant colors corresponding to the histograms with the most bins. The assumption that is made by the dominant color technique is that small histogram bins are likely to contain noise, distorting shot detection results.

Histogram based methods produce good performance in the presence of abrupt changes between shots. However, in the presence of gradual transitions, such as dissolves, fades, and wipes, the difference between successive frames may be too low to be detected. In order to detect gradual transitions, Zhang, et al. [156] imposed a two threshold technique, consisting of an upper and lower bound. The upper threshold is used to detect abrupt cuts, while the lower threshold is used to detect gradual transitions. Whenever the frame difference exceeds the upper threshold, a cut is detected. If the histogram difference falls between the two thresholds, it was marked as the possible beginning of a gradual transition. The subsequent frames are then compared against the candidate frame to detect the remaining frames in the transition sequence. If the successive differences exceeded the upper threshold before the difference falls below the lower threshold, the sequence is considered a gradual transition.

Lupatini, et al. [95] compared the performance of twelve shot detection methods based on histograms, motion, and contours. They concluded that the best performance was achieved with histogram-based algorithms. Boreczky, et al. [10] compared the performance and evaluation of three histogram-based algorithms, a motion-based algorithm, and an algorithm based on the DCT coefficients. They concluded that the histogram-based algorithms performed better in general than the motion-based and DCT-based algorithms. Zhang, et al. [156] compared pixel differences, statistical differences, and histogram-based methods, and concluded that histogram-based methods offer a good trade-off between accuracy and speed.

2.2.1.4 Clustering

Tekalp, et al. [53] introduced a temporal video segmentation technique based on 2-class clustering to eliminate the subjective nature of selecting thresholds. Video segmentation is treated as a 2-class clustering problem, where the two classes are "scene change" and "no scene change". The K-means clustering algorithm [66] is used to cluster the frames. The χ^2 method and the histogram difference method are used to compute the similarity metric in the RGB and YUV color spaces. From their experiments, the χ^2 method in the YUV color space detected the larger number of

correct transitions, however when the complexity of the distance metric is factored in, the histogram difference method in the YUV color space was the best in terms of overall performance. One major advantage of this technique is that it eliminates the need to select a predetermined threshold. Additionally, multiple features can be used to improve performance of the algorithm. In subsequent research, Ferman and Tekalp [43] utilize two features, histograms and pixel differences for video segmentation via clustering.

2.2.1.5 Edge differences

Zabith, et al. [152] detect cuts, fades, dissolves, and wipes based on the appearance of intensity edges that are distant from edges in the previous frame. A summarization of the edge pixels that appear far from an existing pixel (entering edge pixels) and edge pixels that disappear far from an existing pixel (exiting edge pixels), are used to detect cuts, fades, and dissolves. The method is further improved to tolerate camera motion by applying motion compensation. The global motion between frames is calculated and used to align frames before detecting the entering and exiting edge pixels. One disadvantage of this technique is that it is not able to handle independently moving objects [74]. Another disadvantage of this method is an increase in false positives due to the limitations of the edge detection method. Changes in image brightness, or low quality frames, where edges are harder to detect, may cause false positives. Lienhart [86] experimented with the edge-based segmentation algorithm by Zabith, et al. [152] and concluded that false positives can arrive from abrupt entering and exiting lines of text. In order to reduce the false positive, the classification of hard cuts was extended. Additionally, Lienhart [86] found that hard cuts from monochrome images were being classified as fades. The algorithm was modified to eliminate this misclassification.

2.2.1.6 Model-based

Boreczky and Wilcox [11] used hidden Markov Models (HMMs) to segment video [115]. The features used for segmentation are the distance between gray-level histograms, an audio distance based on the acoustic difference in intervals just before

and just after the frames, and an estimate of the object motion between frames. Separate states in the HMM are used to model fades, dissolves, cuts, pans, and zooms. The arcs between states model the allowable progression of states. From a shot state it is allowable to go to any of the transition states, however from any transition state it is only possible to go back to a shot state. Additionally, since pans and zooms are considered a subset of a shot, they can only be reached from a shot state. The arc from a state to itself models the duration of the state. The transition probabilities and the means and variances of the Gaussian distribution are learned during the training phase based on the Baum-Welch algorithm [115]. Training data consists of features (histograms, audio distance, and object motion) from a collection of video with the data classified as a shot, cut, fade, dissolve, pan, or zoom. After the model is trained, segmenting the video into its shots and camera motions is performed via the Viterbi algorithm [115]. One advantage of this technique is that thresholds are not subjectively determined; they are learned automatically based on the training data. Another advantage of this technique is that it allows for the inclusion of multiple features in the training data.

2.2.1.7 Other Techniques

Vasconcelos and Lippman [138] developed a Bayesian framework for shot segmentation by modeling the shot duration and shot activity. The premise of their research is that the probability that a new shot boundary occurs is highly dependent on how much time has elapsed from the previous one. Gong and Liu [50] created a novel technique for shot segmentation based on Singular Value Decomposition (SVD). Given an input video sequence, SVD is performed on feature vectors derived from each input frame. Silva, et al. [125] devised a video segmentation technique based on volumetric processing of the video sequence. Each video sequence is represented as a volumetric video object. Geometric functions are used to classify shot boundaries. Lei, et al. [84] developed a statistical hypothesis testing framework based on the Hotelling T^2 test to detect shot segmentations. Brunno and Pellerin [18] utilized optical flow measurements based on a global wavelet-based parametric model to determine shot

boundaries. Shot boundaries are detected by locating and analyzing the temporal trajectories of the linear prediction errors of the wavelet coefficients.

2.2.1.8 Gradual Shot Boundary Detection

Hard cuts (abrupt transitions) account for 90% of all transitions between shots [87]. The majority of the research in shot boundary detection has focused on detecting hard cuts because this boundary type can be characterized by a single measure between consecutive frames. The difference between successive frames across a hard cut boundary generally produces a high difference value between the two frames, which can be detected by a variety of algorithms that employ a single threshold.

Gradual transitions such as dissolves, fades, and wipes are more difficult to detect in video sequences. Gradual transitions account for the remaining 10% of shot transitions [87] in commercial video. These transitions occur across a series of frames rather than just a single frame as with hard cuts. Additionally, the difference between successive frames during these types of transitions is relatively small due to the special effects commonly used during a gradual transition. Lowering the threshold does not solve this problem because the differences between successive frames in a gradual transition may be smaller than the difference between frames in a shot, resulting in numerous false detections. Moreover, gradual transitions must be differentiated from camera and object movements that display temporal changes and variances similar to gradual transitions that may also cause false positives.

Fades and wipes are generally easier to detect than dissolves. During a fade-in or fade-out, the video signal is scaled by some mathematically well-defined and simple function. Additionally, during wipes, the two video sequences under analysis are well separable at any time [87]. During a dissolve, the two video sequences under analysis are mixed together temporally and spatially, which makes the problem of detecting dissolves very difficult.

There has been considerable research on the detection of gradual transitions in digital video [69, 87, 98, 104, 156]. The twin comparison method of Zhang et al. [156] was the earliest method that attempted to detect gradual transitions. Zhang et. al,

surmised that a single threshold could not detect all segmentation boundaries. The twin comparison method utilizes two thresholds, one high T_h and one low T_l. The high threshold T_h is used to detect hard cuts and the low threshold T_l is used to detect gradual transitions. The process begins by comparing frames based on a difference metric. If the value of the metric exceeds threshold T_h a hard cut reported as detected. If the difference metric is below threshold T_h and above the lower threshold T_l the frame is marked as the potential start of a gradual transition F_s. An accumulated comparison is then performed on the frame F_s and the subsequent frames. This value usually increases during a gradual transition. The end frame F_e is determined when the difference between successive frames drops below T_l and the accumulated difference value exceeds T_h. If the consecutive difference value drops below T_l before the accumulated difference value exceeds T_h the potential start position F_s is dropped and the process restarts. There are some gradual transitions that exhibit difference metric behavior where the metric temporarily drops below the low threshold T_l. In order to combat this problem a tolerance value can be set that allows a certain number of frames to drop below T_l before eliminating F_s as a potential candidate for the start of a gradual transition.

Meng, et al. [98] detected dissolve transition effects in MPEG compressed video by tracking the temporal characteristics of the frame variance σ^2 of DCT DC coefficients of I and P frames. They specifically watch for parabolic shapes in this value over time. Assuming 2 video sequences $f_1(t)$ and $f_2(t)$ with intensity variances $\sigma_1^2(t)$ and $\sigma_2^2(t)$, gradual transitions can be expressed as a linear combination of the two video sequences. The dissolve region is characterized as:

$$f(t) = f_1(t)[1 - \alpha(t)] + f_2(t)\alpha(t) \tag{2.8}$$

where $\alpha(t)$ is a linear parameter that increases linearly from 0.0 to 1.0 over the range of the dissolve. The parabolic variance curve is described as:

$$\sigma^2(t) = (\sigma_1^2 + \sigma_2^2)\alpha(t) - 2\sigma_1^2\alpha(t) + \sigma_1^2 \tag{2.9}$$

The criteria that is used to detect dissolves based on the parabolic curve is that (1) the depth of the variance valley must be large enough, and (2) the duration of the

suspected dissolve region must be long enough. This algorithm produces good results and the processing speed is very fast [69]. However, false alarms can arise due to fast camera or object motion and some dissolves do not satisfy the second criteria.

Jun, et al. [69] detects dissolve transitions based on the spatio-temporal distribution of macro blocks in partially decoded MPEG video sequences. This algorithm analyses B frames adjacent to anchor frames in order to determine potential candidates for dissolve transition. A forward macro block ratio (FMBR) is computed for all B frames adjacent to anchor frames. The FMBR is defined as [69]:

$$FMBR = \begin{cases} M_{fwd}/(M_{fwd} + M_{bwd}) & \text{if } M_{fwd} + M_{bwd} \neq 0 \\ \text{N/A} & \text{otherwise} \end{cases} \tag{2.10}$$

where M_{fwd} and M_{bwd} represent the number of forward and backward type macroblocks respectively. In a dissolve sequence the FMBR changes either from a high value to a low value or vice versa. The temporal distribution of macro block types is determined by prediction state sequences of B frames adjacent to anchor frames. The prediction states are based on the value of the FMBR. Candidates for dissolve transitions are also determined by the spatial distribution of forward and backward predicted macro blocks. Of course, any method that relies on MPEG macroblock distributions is necessarily dependent on the characteristics of the MPEG encoder utilized to produce the video under analysis.

Drew, et al. [35] detects horizontal wipe video effects by using a comparison of successive frames based on chromaticity histograms. The histograms are created by using only the DC values in the columns of each frame. First, a 2D intensity normalized spatio-temporal image is created as follows: $r = R/(R+G+B)$, $g = G/(R+G+B)$. Then a 2D chromaticity histogram is created for each column. The histogram intersection is used to determine differences between consecutive frames. During a wipe, when the wipe reaches each column, an abrupt change produces a histogram intersection equal to zero. Dissolves are detected by creating 2D color histograms in the Cb-Cr color using only the DC values in the columns of each frame. The algorithm looks for constant behavior in the difference values between histograms.

Lienhart [87] developed a dissolve detection system comprised of a transition de-

tection training system and a multi-resolution transition detection system. The detection system is able to identify fixed-size (16 frames) and fixed-scale dissolves. The training system is able to create infinite dissolve sequences from video databases. It is used to create a video corpus of dissolves. Lienhart notes that during a dissolve the image contrast decreases toward the center of a dissolve and increases toward the end. As a result, the average contrast measure of each frame is used to detect dissolves along with a YUV histogram. A feed-forward neural network is used to train and classify the system to detect dissolves. Lienhart reports receiving favorable results using this technique.

2.2.2 Compressed Techniques

Due to the massive size of uncompressed video content, most video is stored in a compressed format, such as MPEG. As a result, there has been considerable research into performing video segmentation directly on the coded MPEG compressed video [1, 92, 93, 154, 148]. An advantage to performing video segmentation in the compressed domain is not enduring the increased computational complexity of decoding the video before analysis. Additionally, as a result of the lower data rate of compressed video, algorithm operations are often faster [74]. Also, the encoded video inherently contains computed features such as motion vectors and block averages that can be utilized. However, the speed and efficiency of algorithms in the compressed domain comes at the cost of increased implementation complexity.

2.2.2.1 Brief Overview of MPEG Video

An MPEG video stream is composed of three types of interleaved frames: Intracoded (I), Bidirectional (B), and Predicted (P). These frames are combined in a repetitive pattern, with the frames between two I frames labeled as a *group-of-pictures* (GOP). Intracoded (I) frames provide a random access point into the compressed data and are encoded by using lossy DCT, Quantization, Run Length Encoding (RLE), and Huffman entropy coding. An MPEG intra-coded frame is very similar in structure to

a JPEG compressed image. Predicted (P) frames are motion compensated in the forward direction, using the nearest previously reconstructed I or P frame. Bi-directional (B) frames are motion compensated in both directions from I and P frames. Motion compensation is performed on macroblocks, which are a $16x16$ block of pixels. For each macroblock in the current frame the encoder computes a motion vector based on the best matching macroblock in the reference frames, and Discrete Cosine Transform (DCT) encodes the residual error. If the best matching macroblock in the reference frame occupies the same position in the current frame, a zero prediction vector is obtained. If the residual error after motion compensation for a particular macroblock in a B or P frame is too high, meaning that the current frame does not have much in common with the reference frame, the encoder can choose to intracode that macroblock. A macroblock in a B frame can be intracoded, forward predicted, with the prediction vector pointing to a macroblock in a past frame, backward predicted, with the prediction vector pointing to a future frame, or interpolated, where the best matching source macroblocks in the previous and next frame are averaged. A macroblock in a P frame can either be intracoded, or forward predicted, with the prediction vector pointing to a macroblock in a past frame. DCT coding is performed via $8x8$ pixel blocks, creating 64 DCT coefficients per block and 4 blocks per macroblock. Figure 2.3 depicts a typical MPEG data hierarchy. A more detailed description of the MPEG standard can be found at [64].

Algorithms in the compressed domain can be grouped into the following categories: Discrete Cosine Transformation (DCT) coefficients, DC terms, and Hypothesis Testing.

2.2.2.2 Discrete Cosine Transformation (DCT) coefficients

The Discrete Cosine Transform (DCT) is the process used in MPEG compression (and many other standards including JPEG) for separating an image into spectral sub-bands with respect to the images visual quality. It transforms an image from its spatial domain to the frequency domain. The first step in DCT coding is the separation of I frames into $8x8$ or $16x16$ sized blocks. The choice of block size is de-

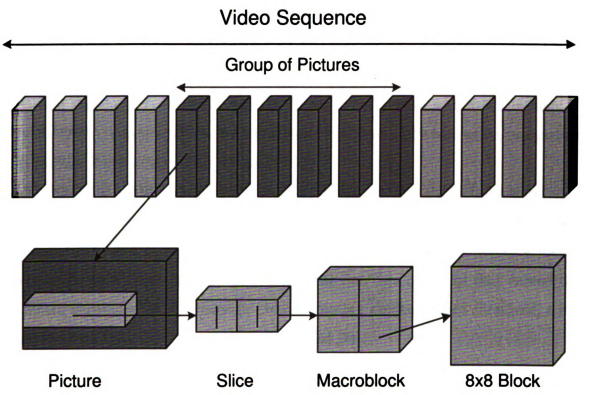

Figure 2.3: Typical Video Structure

termined by a trade-off between compression efficiency and computational complexity. A larger block size produces greater compression efficiency; however it increases the computational complexity of the coding. The DCT converts a block of pixels into a block of transform coefficients. The coefficients represent the spatial frequency components that comprise the block of pixels. Each coefficient can be thought of as a weight that is applied to a basis function. Most blocks will have DCT coefficients of zero because for most images, much of the signal energy lies at low frequencies. Researchers have attempted to use the characteristics of the DCT coefficients to determine shot boundaries. The basic idea is that the DCT coefficients across a shot boundary exhibit significant change in relevant blocks.

Arman, et al. [1] developed a video segmentation algorithm to detect cuts in motion JPEG compressed video data. For each frame, a feature vector is created from a subset of the DCT coefficients. A cut is detected by analyzing the normalized inner product of the vectors of two successive frames. The normalized inner product

is represented as [74]:

$$DP(i, i+\varphi) = \frac{V_i \cdot V_{i+\varphi}}{|V_i||V_{i+\varphi}|} \tag{2.11}$$

If $1 - |DP(i, i+\varphi)| > T_1$, where T_1 is a predetermined threshold, a cut is detected. A second threshold T_2 is used to reduce the effects of camera and object motion. If $T_1 < 1 - |DP(i, i+\varphi)| < T_2$, the frames are decompressed and a histogram comparison is performed for the respective images.

Zhang, et al. [154] extended the work of Arman et al. [1]. In their work, shot boundaries are detected using a pair-wise comparison technique of the DCT coefficients of I frames in the compressed video. A pixel comparison distance metric is compared to a predetermined threshold. The numbers of blocks that exceed the threshold is counted, and if the sum exceeds another predetermined threshold, a cut is determined. The distance metric is defined as [74]:

$$DP(i, i+\varphi, l) = \frac{1}{64} \sum_{k=1}^{64} \frac{|c_{l,k}(i) - c_{l,k}(i+\varphi)|}{max[c_{l,k}(i), c_{l,k}(i+\varphi)]} > T_1 \tag{2.12}$$

$c_{l,k}$ is the DCT coefficient of block l in frame i, with l depending on the size of the frame. If the distance metric is larger than a predetermined threshold T_1, the block l is counted as being changed. If the number of changed blocks exceeds a second predetermined threshold, a cut is detected. Since the algorithm only processes I frames, processing time is reduced, however temporal resolution is also decreased. Gradual transitions are not detected by either of the techniques, and they are subject to produce false positives as a result of camera and object motion. The pair-wise comparison technique is less computationally intensive than the technique proposed by Arman et al. [1].

2.2.2.3 DC Terms

The first DCT coefficient is referred to as the DC term, where DC is derived from *direct current*, an electronics term. The DC term is the average of all pixel values in the block and represents a sub-sampling of the block. Since the DC terms can be considered a reduced-resolution image, segmentation algorithms have been developed that analyze this image in order to detect cuts. Figure 2.4 depicts a full resolution

Figure 2.4: Full Image (352x240) and its DC image (44x30)

image along with its corresponding (tiny) DC image. The DC terms are treated specially in most compression algorithms including MPEG due to their significance as a reduced resolution image and, therefore, are easy to extract.

Yeo and Liu [148] have developed a method for detecting cuts in the compressed domain based on DC frame differences. An image is divided in NxN pixel blocks, with the (i, j) pixel of the DC image corresponding to the average value of the (i, j) block of the original image. The DC terms of I-frames are available from the MPEG stream, while DC terms of P and B frames are re-constructed using motion vectors and DCT coefficients of previous I frames. The reconstruction of the DC terms of P and B frames is a computationally expensive process relative to the I frames, but is still more efficient than reconstructing the entire image. The DC terms of the I, P, and B frames are used to construct a DC frame sequence, which are a sequence of DC images.

Three detection algorithms are applied to a DC sequence. One extracts abrupt changes, another detects plateaus in the frame differences, and another detects flashlight changes. Both the abrupt change detection and plateau detection are combined to give the locations of changes and the beginning of gradual transitions. The distance metric used between successive DC images are simple pixel differences and color histograms. Similar to the uncompressed pixel difference approach, the similarity metric

is based on the sum of the absolute pixel differences of two successive DC images. This similarity metric can be expressed as:

$$D(i,i+1) = \frac{\sum_{x=1}^{X}\sum_{y=1}^{Y}|P_i(x,y) - P_{i+1}(x,y)|}{X \cdot Y} \tag{2.13}$$

where i and $i+1$ are two successive DC images and $P_i(x,y)$ and $P_{i+1}(x,y)$ represent the pixel intensity values. Smoothing can also be used on the DC images to increase the tolerance to small camera and object motions.

A color histogram is created from the first few significant bits of each R,G,B intensity value in the DC images. The RGB color histogram metric is defined as:

$$D_{RGB}(H_i, H_{i+1}) = \sum_{i=1}^{n} |H_i^r(i) - H_{i+1}^r(i)| + |H_i^g(i) - H_{i+1}^g(i)| + |H_i^b(i) - H_{i+1}^b(i)| \tag{2.14}$$

where H_i^r, H_i^g, and H_i^b denote the color histograms of the R,G, and B components respectively for successive images H_i and H_{i+1}.

Yeo and Liu select their cut detection thresholds based on local activity in the candidate images in the temporal domain. A sliding-window is used to examine m successive frame differences. A cut between two successive frames is detected if two conditions are satisfied: 1) the difference is the maximum within a symmetric sliding window of size $2m-1$ and 2) $D(i,i+1)$ is also n times the second largest maximum in the sliding window. The second condition is to reduce the effects of fast panning or zooming scenes and camera flashes to be declared as changes. The parameter m is set to be smaller than the minimum duration between two scenes. Gradual transitions are detected by comparing every frame with the following kth frame, where k is larger than the time allowed for a gradual transition.

2.2.2.4 Other Techniques

Patel and Sethi [110] classify shot transitions in the MPEG compressed domain using statistical hypothesis testing. The first step in the algorithm is to extract I frames from the encoded video sequence. The second step involves creating histograms for each $8x8$ DCT encoded block within an I frame by using the first coefficient of each block. This value represents the average gray level value of the block of pixels. The

final step in the algorithm involves a comparison of successive I frames based on either the Yakimovsky Likelihood ratio test, the Chi-square test, or the Kolmogorov-Smirnov Test [110]. Their study concluded that the Chi-square test gives the best performance. In their later work, Patel and Sethi [111] extended their algorithm to include row and column histograms in addition to the global histograms computed from I frames and the three decisions are fused into a single decision through Boolean logic. Additionally, motion vector analysis is used to characterize shots originating from camera and object motion.

Meng, et al. [98] detected shot boundaries in MPEG compressed video by using the frame variance σ^2 of DCT DC coefficients of I and P frames. Gradual transitions are detected by looking for parabolic shapes in a curve that is derived from the variance of DC term sequence of I and P frames. Hard cuts are detected by an analysis of motion vectors.

Zhang, et al. [153] uses a two-pass technique to detect shot boundaries. The first step involves extracting I frames and performing a pair-wise comparison of DCT coefficients of the extracted I frames. The second step involves verifying the results obtained in the first step. The motion vectors for selected areas are checked to determine the exact cut locations. Further analysis of motion vectors is able to distinguish camera motion from gradual transitions. The twin-comparison technique is used on the DCT differences of I frames to determine gradual transitions.

Koprinska and Carroto [75] developed a hybrid two-pass approach to shot detection in the compressed domain using a rule-based and neural network system. The first pass locates possible shot boundaries and the second pass confirms the potential boundaries. During the first pass the algorithm looks for peaks in intracoded macro blocks of P frames. Peaks indicate an abrupt change in either the P frame with the peak or in one of the two B frames before it. Gradual transitions are detected by locating patterns in the intracoded macro blocks. The second pass of the algorithm is used to confirm the potential candidates of the first pass. Cuts are detected by using rules that check the forward and backward macro blocks of 2 B frames that are near potential transitions. Gradual transitions are detected with a neural network trained

on the motion vector patterns. The neural network is also used to distinguish camera and object motion from dissolves.

2.3 Summarization

Video summarization attempts to present a pictorial summary of an underlying video sequence in a more compact form, limiting or eliminating redundancy. Video summarization focuses on finding a smaller set of images to represent the visual content, and presenting these keyframes to the user. Most summarization research involves extracting keyframes and developing a browser-based interface that best represents the original video. Video abstraction refers to the short summary of the content of a longer video document [85]. It is the process of mapping an entire segment of video into a small number of representative images [157]. A video abstract is a sequence of still or moving images that represent the content of the video in such a way that the original meaning of the video is well preserved. There are two types of video abstracts, still and moving abstracts. The still-image abstract, also known as the static storyboard, is a collection of salient still images generated from the underlying video. The moving abstract, also known as moving storyboard or video skim, is a collection of short image sequences or video clips that are considerably shorter in length than the underlying video source. Additionally, moving storyboards usually have information in the audio track associated with the video sequence.

Li et al. [85] describes three advantages of a still-image Representation: 1) A still-image abstract can be created much faster than a moving image abstract, since no manipulation of the audio or text information is necessary. As a result, there are no synchronization or timing issues that need to be addressed. 2) The temporal order of the representative frames can be displayed so that users can grasp the concept of the video more quickly. 3) Additionally, all extracted still images are available for printing, if desired.

Li et al. [85] also describes three advantages of video Skims: 1) The audio track may contain valuable information that is lost in the still-image representation. 2) It

is more natural and interesting for users to view a short trailer, than a sliding window of pictures. 3) The motion in the video sequence conveys information that is lost in the still-image representation.

2.3.1 Still-Image Representation

2.3.1.1 Keyframe Extraction

Keyframes are still images that best represent the content of the video sequence in an abstracted manner [157]. The goal of extracting keyframes from a video segment is to retain the important content of the video while removing redundancy. There have been numerous research efforts with respect to keyframe extraction [157].

2.3.1.2 Shot-Based keyframe extraction

In some commercial products and modeled systems the first and last frame of each shot are selected to represent shot content [40, 122, 136, 157]. This procedure is often referred to as *temporal keyframe extraction.* Although this may reduce the total number of keyframes and provide information about the total number of keyframes a priori, this method is not an accurate and sufficient representation of the shot content. It does not characterize or capture dynamic action or motion within a shot, therefore keyframes should be extracted based on the underlying semantic content. Semantic analysis of a video is a difficult research problem. As a result, most keyframe extraction techniques rely on low-level image features, such as color and motion.

2.3.1.3 Color-Based Keyframe Extraction

Zhang et al. [157] extracts keyframes based on their color content. Keyframes are extracted in a sequential manner. The density of the keyframe selection process can be adjusted; however the default is that the first and last frames of each shot are considered keyframes. Once a keyframe has been selected, a color histogram comparison method is employed on subsequent frames and on the previous keyframe. If the distance metric exceeds a predetermined threshold, a keyframe is selected. The

Munsell space was chosen to define keyframes, because of its closeness to human perception [157]. The color space is quantized into 64 "super-cells" using a standard minimum of squares clustering algorithm. A 64 bin histogram is calculated for each keyframe, with each bin having the normalized count of the number of pixels that fall in the corresponding supercell. The distance metric between two histograms is defined as follows:

$$D(H_i, H_{i+1}) = \sum_{i}^{N} \sum_{j}^{N} a_{ij}(H_i(i) - H_{i+1}(i))(H_i(j) - H_{i+1}(j)) \tag{2.15}$$

where the matrix a_{ij} is derived from human perception studies.

2.3.1.4 Clustering-Based Keyframe Extraction

Ferman and Tekalp [42] propose a keyframe extraction method based on the clustering of frames within a shot. Frames within a shot are clustered into a certain number of groups based on a color histogram similarity measure. The frame closest to the center of the largest cluster is selected as the keyframe for the shot.

Zhuang et al. [159] proposed a method for keyframe extraction based on unsupervised clustering. The color histogram is used to represent the visual content within the frame. A $16x82D$ HS histogram is used in the HSV color space. The similarity metric between successive frames i and j is defined as follows:

$$\sum_{h=1}^{16} \sum_{s=1}^{8} min(H_i(h,s), H_j(h,s)) \tag{2.16}$$

A keyframe is only selected from clusters that are bigger than N/M, the average size of clusters, where N is the total number of frames in a shot, and M is the number of clusters. For each cluster, the frame that is closest to the centroid is selected as the keyframe.

2.3.1.5 Motion-Based Keyframe Extraction

Wolf proposes a motion based keyframe selection algorithm based on optical flow [144]. The algorithm computes the flow field for each frame based on the Horn and

Schunk optical flow algorithm [4]. The sum of the magnitudes of the optical flow components are used as the motion metric. The keyframe selection process selects keyframes that are at the local minima of motion between two local maximas.

2.3.2 Moving Image Representation

2.3.2.1 Video Skims

Video Skims are short video clips consisting of a collection of image sequences and the corresponding audio, extracted from the original longer video sequence. Video skims represent a temporal multimedia abstraction that is played rather than viewed statically. They are comprised of the most relevant phrases, sentences, and image sequences. The goal of video skims is to present the original video sequence in an order of magnitude less time [140].

There are two basic types of video skims: summary sequences and highlights [85]. Summary sequences are used to provide a user with an impression of the video sequence, while a highlight video skim contains only the most interesting parts of a video sequence.

Omoigui, et al. [107] develops summary video skims by increasing the speed of playback of the original video. Speeding up the playback allows the entire video to be displayed in a shorter amount of time. Summary video skims have the advantage of allowing the user to view an entire segment of video in less time, however they only allow for a maximum time compression of 1.5 to 2.5 depending on the speech speed [85].

The MoCA project [88] developed automated techniques to extract highlight video skims to produce movie trailers. Scenes containing important objects, events, and people are used to develop the video skims. Selecting highlights from a video sequence is a subjective process; as a result most existing video-skimming work focuses on the generation of summary sequences [85].

The Informedia I & II Project at Carnegie Mellon University [140, 24] utilizes speech, closed caption text, speech processing, and scene detection to automatically

segment news and documentary video. They have created a digital library with over a terabyte of video data. One aspect of the summarization method of the Informedia Digital Video Library System is partitioning the video into shots and keyframes. Multi-level video summarization is facilitated through visual icons, which are keyframes with a relevance measure in the form of a thermometer, one-line headlines, static filmstrip views, utilizing one frame per scene change, active video skims, and the transcript of an audio track. Text keywords are extracted from the transcript and closed captioning text by using a Term Frequency/Inverse Document Frequency (TF-IDF) technique. Audio data is extracted from the segments corresponding to the selected keywords and its neighboring areas to improve audio comprehension. Image extraction is facilitated by selecting frames with faces and text, frames following camera motion, frames with camera motion and faces or text and frames at the beginning of a video sequence. Video Skimming is created by the confluence of extracted audio and image extraction. Experiments using this skimming approach have shown impressive results on limited types of documentary video that have explicit speech or text content [85]. It remains unclear whether this technique may produce similar results with video containing more complex audio content.

2.4 Other Techniques

Bagga, et al. [3] developed a novel video summarization technique that uses image analysis, closed caption text, and hierarchical scene clustering. First, shot segmentation is performed via a statistical analysis method and then each of the potential scene boundaries are clustered via a hierarchical clustering method. The midpoint between two consecutive scene changes is chosen as the keyframe for the scene. The dominant color in the $L * a * b$ color space is then used to describe each keyframe. The difference between dominant color components is used as the distance metric between keyframes. A second distance metric is also computed for the closed caption text of each scene. The two distance metrics are then combined to form a matrix D and hierarchical clustering is performed on this new matrix. A recursive clustering

process is performed to merge similar clusters until a predefined cluster distance is met. This method has the advantage of utilizing text and image data simultaneously to cluster the video data.

Dementhon, et al. [29] represents a video sequence as a polygonal trajectory curve in a high dimensional feature space. This curve is created by mapping each frame to a time dependent feature vector and representing these feature vectors as points. The curve is segmented into regions of linearity or low dimensionality via binary curve splitting and the frames that appear at boundaries between curve segments can be used as keyframes to summarize the video sequence. Additionally, the algorithm builds a binary tree of the video sequence under analysis, where the branches represent a more detailed representation of the video sequence. This representation allows users to view a video sequence at varying levels of granularity. The authors claim that this method is less sensitive to differences in pair-wise frames as with other traditional approaches that utilize frame-to-frame distance measures. Stefanidis, et al. [128] also used trajectories to develop meaningful video summaries. Their research focused on locating objects within a video sequence and creating object trajectories. These trajectories were further analyzed for critical points that describe the motion of an object over time.

Dixon and Owen [31] developed a novel video summarization technique based on the amount of camera motion for raw unedited video sequences in bandwidth-constrained client-server environments. This method was primarily developed to support summarization of content from Unmanned Aerial Vehicles (UAV). The algorithm selects keyframes that exhibit significant camera motion between frames. An affine motion model is used to characterize the camera motion. Once keyframes are selected they are placed in a keyframe pool. Keyframes are transmitted from the keyframe pool based on the amount of available transmission bandwidth.

Doulamis, et al. [32] devised a summarization algorithm based on a fuzzy representation of a video sequence. Each frame is segmented and size, location, color, and motion are used to form feature vectors for each segment. The feature vector is made up of the horizontal and vertical locations of the center of the segment,

the average values of the three color components, and the average motion vector of the motion segment. Color and motion properties are classified into predetermined classes with a value representing the degree to which the feature vectors belong to each class. This value forms the fuzzy membership and is assigned to each class. A multi-dimensional fuzzy histogram is computed for each segment in a frame. The histogram represents the degree to which an entire frame belongs to a specific class. Keyframes are selected via a method that minimizes the cross-correlation criteria. In subsequent research Doulamis, et. al [33] developed a video summarization technique that estimats points of a feature vector trajectory curve. The feature vector is built of both *global* and *object* features. The global features include color, texture, and global motion. The object features are extracted similar to how the segments were extracted in this first research effort[32]. The plot of all the feature vectors forms a trajectory. Keyframes are selected by selecting appropriate junction points along this trajectory. Interpolation theory is used to select optimal points along this curve.

Farin, et al. [41] utilizes two-stage clustering and user specified domain knowledge to summarize a video sequence. The first step involves the creation of a feature vector based on luminance histograms for each frame. The second step organizes the feature vectors in temporal segments. These segments are user defined (4 sec.) and are generally smaller than the length of a shot. The purpose of segmenting the video into segments is to eliminate the possibility of gradual transitions appearing in the summaries. When two subsequent shots are clustered together having gradual transitions, the clustering algorithm generally selects the transition frame as the cluster center. A time-constrained clustering approach is used as a first pass clustering technique to detect possible shot boundaries. Keyframes are selected via a second clustering algorithm based on k-means clustering. Users can also exclude certain scenes deemed as irrelevant or unimportant by feeding information about these scenes into the algorithm. Feature vectors are computed for the unwanted scenes and the second pass clustering algorithm is augmented to remove the unwanted clusters. One advantage of this technique is that it does not rely on accurate cut detection. Additionally, the use of domain knowledge by the user over automated pre-filtering techniques is a

desirable feature.

Fujimura, et al. [46] developed a summarization algorithm based on color and closed caption information. The first step involves detecting shot boundaries via a shot probability model. In the second step, a color histogram is used to detect unchanged scenes and a multiplexing technique based on color histograms is used to extract important features from each scene. The third step involves developing rules to extract important closed caption text from each scene. The properties of the unchanged analysis, multiplexing, and closed caption rules are used to create summaries of the video sequence. This system is highly rule based and the authors claim to achieve good performance in terms of time compression for some movie sequences.

Gong and Liu [51] developed a moving image summarization algorithm that is aimed at reducing the visual redundancy in a video summary. This approach differs from previous approaches that attempt to develop moving summaries in that its goal is not to select the most important scenes to add to the summary, but to minimize duplicated and redundant content. They have developed a redundancy metric based on the entropy metric from information theory. The first step of the algorithm segments the video into shots via local color histograms and singular value decomposition (SVD). The second step clusters the shots based on their visual similarity. The shot with the longest length is selected as the cluster center. Shots are discarded if their length is less than 1.5 seconds. The final video summary is created by concatenating a condensed version of each cluster center in temporal order. In experimentation, each cluster center was condensed to 1.5 seconds before concatenation. The authors concluded that their redundancy metric produced good results with videos containing many long static shots or visually similar shots and produced poor results when the video contained short shot and diverse sequences.

Sugano, et al. [129] developed a video summarization algorithm that operates in the MPEG compressed domain using MPEG-7 video sequences. Their research creates digests as well as highlight video summaries. Digest summaries are based on the analysis of audio level and visual information. Highlight summaries of sports

broadcasts are developed by analyzing the audio class and audio level.

Uchihashi and Foote [135] developed a video summarization algorithm based on a shot importance measures. This measure is developed through hierarchical clustering. A shot is deemed important if it is considered long and rare. A keyframe is selected for each cluster, and a keyframe packing algorithm is used to display the frames, with the size of the frame corresponding to its importance.

2.5 Results Fusion

Researchers in various disciplines have attempted to use results fusion to improve performance and reliability for their applications [6, 48, 61, 70, 117, 9, 146]. They attempt to fuse together relevant evidence or sensors that when used alone are considered unreliable in certain instances. Results fusion is a general problem that is interesting in many domains. Research from the document retrieval and biometric community has shown that success and reliability can be improved from using results fusion. Recently, the digital video community has recognized the need to incorporate multiple evidence to improve performance. The types of results fusion strategies and methodologies used depend on the amount of knowledge known at fusion time.

In general, there are a variety of methods that can be employed for results fusion. In the document retrieval community, some methods that have been used are simple unweighted Boolean retrieval, Bayesian inference networks, and logistic regression. Additionally, weighting strategies in which weights are determined based on experimental or test searches have been used.

2.5.1 Sum Rule

The sum rule is the simplest form of fusion. Let i denote the number of methods, w_m represent the number of possible m classes, and x_i denote the measurement vector of method i. An input is assigned to class w_j if:

$$\sum_{i=1}^{N} P(w_j|x_i) = max_{k=1}^{N} \sum_{i=1}^{N} P(w_k|x_i) \tag{2.17}$$

where N is the number of methods. Ross, et al. [117] fused the results from face, fingerprint, and hand geometry analysis to increase the performance of a biometric system. In their research, the decision function was based on the sum rule. From their research, it was concluded that increased performance could be obtained by using multiple modalities.

2.5.2 Voting Strategies

Various voting strategies have frequently been utilized to fuse multiple results [7]. The most common voting technique used by researchers is majority voting. Let Δ_{mi} represent the decision for each class w_m by the *ith* method. An input is assigned to class w_j if:

$$\sum_{i=1}^{N} \Delta_{ji} = max_{k=1}^{l} \sum_{i=1}^{N} \Delta_{ki}, \text{where}$$
$$\Delta_{ki} = \begin{cases} 1 & \text{if } P(w_k|x_i) = max_{j=1}^{l} P(w_j|x_i) \\ 0 & \text{otherwise} \end{cases} \tag{2.18}$$

where l is the number of classes. In this method, each sensor is given equal weight to make decisions to determine the outcome. The problem with this strategy is that all the sensors are given equal weight in all conditions. A better implementation would be to give more weight to the sensors that are more reliable. Hull, et al. [62] developed 11 classifiers based on template matching and probabilistic strategies. The final decision was made if 6 out of the 11 sensors made the same decision. Xu, et al. [145] experimented with combining methods for handwriting recognition based on various voting strategies.

2.5.3 Probabilistic Strategies

Kittler, et al. [72] have developed a theoretical framework based on Bayesian theory for combining classifiers that use different representations. Let x_i denote the feature vector that the *ith* classifier observes where $i \in 1, \ldots, k$ and where there are m possible classes $w_1, \ldots, w_m$. The class w_j is selected with the maximum posterior probability $P(w_j|x_1, \ldots, x_k$. According to Bayes theorem, the posterior probability

can be expressed as:

$$P(w_j|x_1,\ldots,x_k) = \frac{P(w_j)P(x_1,\ldots,x_k|w_j)}{P(x_1,\ldots,x_k)} \tag{2.19}$$

If the feature vectors extracted from each sensor are conditionally independent then the decision rule is defined as:

$$P(w_j)\prod_{i=1}^{k} P(x_i|w_j) = \max_{l=1}^{m} P(w_l)\prod_{i=1}^{k} P(x_i|w_l) \tag{2.20}$$

The decision rule in Equation 2.20 in terms of the posterior probabilities is defined as the product rule, because the final decision is based on the product of the probability of all the classifiers. The feature vectors $(x_1,\ldots,x_k)$ are assigned to class w_j if:

$$P^{1-k}(w_j)\prod_{i=1}^{k} P(w_j|x_i) = \max_{l=1}^{m} P^{1-k}(w_l)\prod_{i=1}^{k} P(w_l|x_i) \tag{2.21}$$

If equal prior probabilities are assumed for each of the classifiers, then Equation 2.21 is reduced to:

$$\prod_{i=1}^{k} P(w_j|x_i) = \max_{l=1}^{m}\prod_{i=1}^{k} P(w_l|x_i) \tag{2.22}$$

Hull, et al. [61] combined probability estimates of multiple classifiers to improve document filtering performance. Multiple statistical classifiers were fused by a meta-classifier to accept or reject retrieved documents based on their similarity to previous or subsequently retrieved documents.

2.5.4 Machine Learning and Data Mining

The main advantage of the sum rule, voting strategies, and probabilistic methods are that they do not require any training. However, researchers have also experimented with various machine learning and data mining techniques for results fusion. Some of these methods include utilizing neural networks, decision trees, rulesets, and Support Vector Machines (SVMs) (see Section 4.2.4, Section 4.2.2, Section 4.2.3, and Section 4.2.1 for a detailed descriptions of these methods). The strengths of these methods are based on their discriminative ability to learn and represent the underlying patterns of the input data.

Cho [22] experimented with neural networks to recognize patterns in handwritten numerals. Several independent neural-networks were used to classify input patterns of handwritten numeral features and the final fusion decision was made via the majority voting method. Lin and Hauptmann [91] experimented with SVMs for news video classification. In this research, multiple SVM classifiers were fused via a metaclassifier also based on SVMs. Their strategy fused text and image features to classify news segments as either weather or non-weather scenes.

2.6 Summary

This chapter has presented a broad view of the current research related to shot segmentation, summarization, and results fusion. Most video segmentation and summarization research focuses on the creation of algorithms for specific classes of content. When the type of video is known a priori, any number of algorithms can be chosen with relative success. Color histogram-based algorithms are successful when applied to video sequences where the distribution of color changes between shots [47, 95, 102, 157, 156]. Model-based algorithms are effective when a set of transitions is known and expected in the content [11, 115]. Motion-based algorithms are effective when object or camera locations change significantly between shots [95]. However, when the type of video is unknown, an adaptive method is needed to adjust to the type of content for the best possible segmentation and summarization result.

There is a variety of methods available for results fusion. The choice of method usually depends on how much information is available at fusion time. Probabilistic strategies, data mining and machine learning strategies, and Boolean strategies have all been successfully used to fuse multiple modalities.

Chapter 3

Shot Detection Methods

Video shot detection is the automatic determination of the boundaries between video shots, segments of video captured as a continuous sequence. Shot detection segments the video into sequences delineated by the shot boundaries. Section 2.2 described a wide variety of methods to perform segmentation using video shot detection with a description of the relative strengths and weaknesses of each algorithm. Different algorithms have been designed and presented that perform differently on varying types of shot transitions. As the goal of this research is to construct a *fusion-based* shot detection method that incorporates the strengths of different algorithms simultaneously, four diverse and representative shot boundary detection methods were selected for detailed study. Three algorithms were chosen from the uncompressed video domain and one algorithm was chosen from the MPEG compressed video domain. This chapter describes each of these methods in detail.

3.0.1 Gradual Shot Boundary Detection Issues

Reliable gradual transition detection is a difficult research problem. As discussed, there are several different highly-optimized techniques for detecting various types of gradual transitions. Each algorithm attempts to solve the problem in a totally different manner. One algorithm utilizes multiple thresholds, another utilizes I and B frames in compressed MPEG video sequences, and another uses machine learning.

To date, there had not been a systematic approach to detecting gradual transitions. Our research does not focus on detecting gradual transitions. In order to reliably and accurately detect gradual transitions, we would have to characterize every type of gradual effect for training and testing. That is beyond the scope of this work. Since gradual transitions make up about 10% of all shot boundaries our research focuses primarily on hard cut shot transitions.

3.1 Shot Boundaries

The four shot boundary detection methods that are used in our study are (a) global color histograms, (b) local color histograms, (c) edge features, and (d) DCT coefficients. It is important to note that researchers have experimented with various flavors of each of the implemented algorithms. Some researchers have reported receiving better performance for each individual algorithm based on using different algorithm parameters, thresholds, and color spaces. The goal of this research is not to optimally improve the shot detection of each individual algorithm, but rather to fuse the algorithms to determine a best segmentation. Clearly the component algorithms can be further tuned to produce better individual results.

3.1.1 Global Color Histogram

The global color histogram shot detection algorithm works by searching for peaks in the frame difference values, which represent cuts. The most common video shot boundary detection techniques utilize color histograms of each frame to segment the video. Color histograms are described in detail in Section 2.2.1.3. The idea supporting global color histograms as a shot detection method is that successive frames within a shot will have highly correlated color distributions due to the similar color content in the temporally adjacent frames. Color histogram-based algorithms provide a good tradeoff between performance and complexity [10]. Color histograms are attractive because they are invariant to scaling and rotation of the content of video frames. Consequently, they are invariant to most common camera motion. Additionally, the

representation is compact in size with respect to the size of its respective video frame. Once a frame has been represented via a histogram a number of distance metrics can be employed to detect dissimilarity between frames [156].

The work in this thesis is based on an implementation of a 64 bin RGB histogram over the entire frame. The histogram is created by using the top 2 bits of each color value. Thus, each color component (R, G, B) is discretized to 4 values. Euclidean distance is used as the dissimilarity metric used between consecutive frames. The difference metric between two histograms H_i and H_{i+1} is defined as follows, where N is the number of histogram bins:

$$D(H_i, H_{i+1}) = \sqrt{\sum_{j=0}^{N}(H_i(j) - H_{i+1}(j))^2} \tag{3.1}$$

A shot boundary is detected if the dissimilarity metric between consecutive frames exceeds a threshold.

3.1.2 Region-based or Local Histograms

Global histograms are tolerant of object and camera motion, however they ignore the spatial information between frames and thus two frames with different spatial distributions can have similar color histograms. Two shots having the color distribution is particularly a problem when shots represent the same content from different angles, as is common in motion pictures (as in establishing shots, masters, and close-ups). Local histograms attempt to solve the problems associated with global histograms. Local histograms were described in Section 2.2.1.3. Local histograms segment a frame into smaller blocks and compute histograms for each region. This method is tolerant to local changes in motion, however it still sensitive to changes in luminance over the entire frame[95]. The research presented in this thesis utilizes a region-based histogram technique wherein each frame is divided into 12 blocks in a $4x3$ pattern. A 64 bin RGB histogram is computed for each block. A distance metric is computed for each block between successive frames. The sum of the 12 dissimilarity metrics is used as the distance metric to determine shot boundaries. Figure 3.1 depicts a video

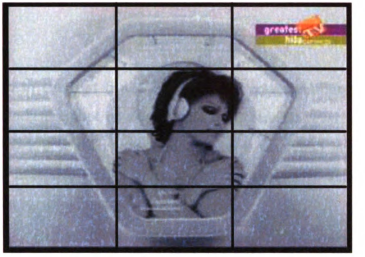

Figure 3.1: Still Frame Segmented into Regions

frame divided into the 12 regions.

3.1.3 Edge Features

Zabith, et al. [152] detects cuts, fades, dissolves, and wipes based on an edge ratio that is computed from the appearance and disappearance of intensity edges between subsequent frames. We have implemented an edge-based shot boundary detection algorithm based on Zabith et al. [152]. This algorithm was described in Section 2.2.1.5. Given two input image frames, both images are re-scaled to 88 by 60 and a robust statistical edge detection algorithm adapted from Kundu [77] is used to compute the edge maps of both frames. Re-scaling the images decreases the execution time of the algorithm; however it also lowers the accuracy of results. From experimentation we have achieved favorable performance using 88 by 60 scaled images. Figure 3.2 depicts a video frame along with its edge image. Each edge image is then dilated with a diamond shaped kernel of size 7 by 7 to allow for movement that may occur between frames. Motion compensation based on pyramidal Lucas and Kanade optical flow [4] is used to register the two images. The edge change ratio is defined as follows: Let E and E' represent the dilated current and next edgemaps of the current and next frames. Additionally, let $\overline{E}^{'}$ be the motion compensated next frame that is aligned

with the previous frame. The exiting edge ratio ρ_{out} is defined as:

$$\rho_{out} = \frac{\sum_{x,y} E[x,y]\overline{E}'[x,y]}{\sum_{x,y} E[x,y]} \tag{3.2}$$

If a pixel is found in the first frame $E[x,y]$ and a corresponding pixel is found in $\overline{E}'[x,y]$ then this indicates that no changes has occurred. However, if a pixel is found in the first frame $E[x,y]$ and a corresponding pixel is not found in the second image $\overline{E}'[x,y]$ then this indicates that a pixel has exited the first frame. Similarly, the entering edge ratio ρ_{in} is defined as:

$$\rho_{in} = \frac{\overline{E}'[x,y]\sum_{x,y} E[x,y]}{\sum_{x,y} \overline{E}'[x,y]} \tag{3.3}$$

The edge change ratio is defined as the maximum of ρ_{out} and ρ_{in}.

$$max(\rho_{out}, \rho_{in}) \tag{3.4}$$

3.1.4 DCT Coefficients

Zhang et al. [154] perform video shot detection using the DCT coefficients of candidate blocks of I frames in compressed video. DCT-based algorithms were discussed in Section 2.2.2.2. Figure 3.3 depicts a video frame along with its DCT image. We have implemented a DCT based shot detection technique based on Zhang et al. [154]. A pair-wise frame comparison between corresponding $16x16$ DCT blocks is computed as follows:

$$D(i,i+1) = \sum_{x=1}^{M}\sum_{y=1}^{N} \frac{|c_{x,y}(i) - c_{x,y}(i+1)|}{max[c_{x,y}(i), c_{x,y}(i+1)]} > T_1 \tag{3.5}$$

where M and N represent the number of rows and columns of blocks respectively, and $c_{x,y}(i)$ and $c_{x,y}(i+1)$ represent the DCT coefficients of block x,y in frames i and $i+1$ respectively. The sum of all the MxN blocks is used as the distance metric between frames. If this total is larger than a predetermined threshold, a shot boundary is detected.

Figure 3.2: Still Frame and Edge Image

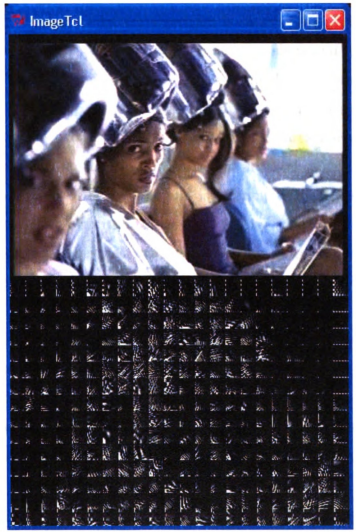

Figure 3.3: Still and DCT Image

3.2 Summary

We have chosen to implement global histograms, local histograms, edge features, and DCT coefficients to extract features for video segmentation of hard cuts. These algorithms have been chosen as a set because they are good general examples of the common classes of shot detection algorithms in the literature and have unique characteristics. The global histogram algorithm is chosen because it produces good performance in the presence of abrupt changes between shots and is very commonly used in practice. Many researchers have received good performance using global histograms [10, 47, 95, 156]. Local histograms are the most common approach to combating the problems exhibited by global histograms, particularly when the shots are of the same scene from differing angles. This method takes into account the spatial distribution of pixels in a frame and is tolerant to local changes in camera and objects motion. The algorithm based on intensity edges was used in order to provide a non-color technique that was based on features and texture, rather than global frame characteristics. Although the assumption that frames within a shot will have similar color content is a widely held and valid assumption, sometimes frames across a shot exhibit similar color properties. For example, if a grayscale video is under analysis, color histogram-based methods will not be as effective. The edge change ratio algorithm combats this problem. Additionally, the edge ratio algorithm claims to detect hard cuts as well as some gradual transitions; however we did not receive good performance detecting gradual transitions. DCT coefficients are used to provide a fast and efficient algorithm in the compressed domain.

Gradual transition detection is a difficult research problem and in our research we do not specifically attempt to detect them. They make up about 10% of all shot boundary transitions. Many researchers have attempted different techniques with differing results; however there has not been a systematic approach to gradual transition detection to date. Most of the gradual transition algorithms can be adapted to a fusion method as described in this thesis.

Chapter 4

Results Fusion

An old maxim states that "two heads are better than one." It seems logical that this maxim can be extended to the concept of fusing results of algorithms such that the strengths of different algorithms can be of advantage in appropriate circumstances; hence, the idea of *results fusion*, the combination of multiple algorithms together to produce a new, composite, algorithm that is better than any of its parts.

Results fusion is the process of utilizing multiple classifiers or representations to improve performance and reliability over using a single classifier or representation. Many researchers have recognized the need to improve individual results by utilizing multiple classifiers. Several methods exist for results fusion (also referred to as classifier fusion or sensor fusion in the robotics community). Research in the document retrieval community has shown improved performance using results fusion for document retrieval [8, 21, 34, 49, 52, 60, 79, 99]. Additionally, research in the biometrics community has shown improved performance utilizing results fusion to improve person identification and authentication [9, 17, 67, 117, 146]. The results of this research suggest that increased performance (better shot boundary retrieval results and accuracy) can be achieved for shot-based segmentation by fusing multiple classifiers into a single composite solution and that the resulting algorithm will perform better any of its constituent parts.

Video shot segmentation algorithms use a variety of *features* to determine shot boundaries. These features include histograms, edge images, motion vectors, DCT

coefficients, and DC Terms. Traditional shot segmentation methods have extracted and analyzed the temporal characteristics of a single feature to determine shot segmentation boundaries. The reliability and performance of these various algorithms is based on the ability to extract the meaningful and important features from a video sequence. If the video sequence under analysis does not lend itself well to extraction of the selected features or the features convey limited information, the results of the segmentation method may not be reliable. For example, if a video sequence consists of content with no clear edges or edges that are difficult to extract, any method based on the comparison of edgemaps between sequential frames would not be reliable. To combat this problem, research in the video shot segmentation community have attempted to improve performance and reliability by combining multiple algorithms [43, 103, 132, 53, 151].

Section 4.1 presents the levels of results fusion and known work in fusion-based shot detection. Section 4.2 describes the novel results fusion shot-based segmentation algorithms designed and implemented. Section 4.3 describes the features utilized by the results fusion shot segmentation methods. Section 4.5 describes our baseline testing methods to determine the performance of our fusion-based methods.

4.1 Levels of Results Fusion

The purpose of results fusion is to fuse together multiple evidences into a combined framework to improve performance and reliability. The classifier can employ various methods of fusion based on the type of information available. Research in sensor fusion, information retrieval, and biometrics have generally classified results fusion into three categories depending upon the amount of information they attempt to combine [145].

1. Abstract or Decision Level: each classifier outputs a result that is a unique label or decision.

2. Ranked Level: each classifier outputs a queue of all the available labels sorted

by decreasing confidence order and the queues of all the classifiers are combined.

3. Measurement Level: each classifier outputs a unique label or result along with a score or confidence value.

Ross and Jain [117] added another level of results fusion, the feature extraction level, where features extracted from multiple algorithms are concatenated to form a new synthesized higher-dimensional feature vector.

Since the result of a video segmentation algorithm is ultimately a frame denoting the end of one shot and the beginning of another, some researchers have developed algorithms based on fusing the output frames of various segmentation algorithms. These algorithms utilize clustering or merging methods to group similar shots together while preserving temporal order to determine a best segmentation. This type of fusion can be labeled *output level fusion* [43, 103, 53].

As discussed, the types of fusion available are abstract, ranked, measurement, feature extraction, and output level fusion. In order of complexity, abstract level fusion provides the least amount of information to determine classification. This level of fusion is one of the most basic levels of fusion where usually the accept/reject result decisions of multiple classifiers are combined via the meta-classifier. Only the outputs of the classifiers are used to fuse the results, regardless of the type of inputs. As a result of the lack of information being available at this level, the classifier usually employs simple majority voting or linear combination [151, 160]. Output level fusion also provides a minimal amount of information to determine classification. This level of fusion attempts to fuse the output results of the multiple evidences. At this level, the classifier utilizes merging and clustering techniques of output frames [43, 100, 103, 53]. Feature extraction level fusion involves creating new feature vectors in a higher dimensional space to determine classification. The goal is that the new feature vector will create a more discriminatory feature upon which a classification decision can be made. Neural networks, support vector machines, decision trees, generalized trace, and Bayesian methods have been used by meta-classifiers to facilitate fusion on this level [119, 132, 158]. Measurement level fusion involves the fusion of scores

or confidence values from the multiple classifiers. This level involves associating probabilities, weights, or scores to the output of each classifier and fusing together these scores. Meta-classifiers at this level use Boolean logic and summing or averaging probabilities to facilitate fusion on this level [15]. Ranked level fusion involves each evidence outputting all possible decision labels in a queue in decreasing confidence order. Bartell, et al. [6] developed a multiple expert system using multiple ranked retrieval systems. The method uses the results of past queries to learn an optimized combination strategy to rank relevant documents. This level of fusion is commonly used for problems with several classes of outputs [59]. This type of fusion does not apply to shot-based segmentation which only utilizes two distinct classes: scene change and no scene change.

4.1.1 Abstract Level Fusion

Abstract or decision-level fusion involves each classifier outputting a label or decision value based on the input data. A final decision is made by the combination of the decision values of the multiple classifiers. Combining multiple outputs on the abstract level usually involves majority voting methods, linear combination, nearest neighbors, or winner-take-all methods [30].

Yusoff, et al. [151] developed a cut detection technique that combines multiple experts using a voting scheme. The five algorithms used for their research are (a) average intensity measurement, (b) Euclidean distance, (c) global histogram comparison, (d) likelihood ratio, and (e) motion estimation/prediction error. A *receiver operating characteristics* (ROC) curve is created for each method, by setting thresholds for each individual method. Calculating the percentage of undetected true shot boundaries, against the percentage of incorrectly detected shot boundaries creates the ROC curve [151].

$$p_u = \frac{S_u}{S_a}, p_f = \frac{S_f}{S_a}, \tag{4.1}$$

S_u is the number of missed shots, S_f is the number of false positives, and S_a is the number total number of shot boundaries. The fusion system is designed by setting

the operating points of the different algorithms at different levels.

An operating point is described as the (p_u, p_f) values for each algorithm. The threshold for each individual algorithm is calculated by $T(p_u^i, p_f^i)$ for each expert i. For each pair-wise comparison, each expert computes the frame distance metric S_i and determines shot boundaries based on the following:

$$d(S_i, p_u^i, p_f^i) = \begin{cases} \text{cut} & \text{if } S_i \geq T(p_u^i, p_f^i) \\ \text{no cut} & \text{otherwise} \end{cases} \tag{4.2}$$

The fusion system determines if there is a cut as follows:

$$D = \begin{cases} \text{cut} & \text{if } n_c \geq n_n \\ \text{no cut} & \text{otherwise} \end{cases} \tag{4.3}$$

where n is the number of experts, n_c is the number of experts that detect a shot boundary, and n_n is the complement $n_n = n - n_c$. In order to determine the optimal values for (p_u, p_f), the thresholds for each individual algorithm are selected from five points on the ROC curve. n experts and N possible thresholds per expert generates N^n possible threshold combinations. The fusion engine determines a shot boundary when at least three experts identify a shot boundary. Yusoff, et al. concluded that their method can significantly improve shot boundary detection results.

This algorithm performs abstract or decision level fusion. The outputs of multiple algorithms are fused together via a majority voting scheme. Majority voting schemes have proven to operate well in some instances of information fusion [160]. The idea behind this algorithm is that if the majority of the implemented algorithms determine that a shot boundary exists, there is a high probability that an actual shot boundary exists. One problem with the Yusoff implementation is that each expert is given equal weight in determining a shot boundary. The shot boundaries are detected if any three experts signal a shot boundary and do not take into account which algorithm or expert may be more appropriate for a given video class. Another problem with this method is that each expert utilizes static thresholds in determining local shot boundaries. In developing a solution that can be utilized on a wide variety of content it is important that the algorithm adapt to the different types of video under analysis. As a result,

a dynamic threshold scheme would be more appropriate. Additionally, the coarse manner in which thresholds were chosen needs to be refined in order to be effective on different classes of video.

4.1.2 Measurement Level Fusion

Measurement or confidence level fusion involves each classifier producing a vector with a confidence score denoting the degree to which the classifier believes that input data belongs to a specific class. A decision is then made based on the combination of these scores. An important aspect of measurement level integration is normalization [17]. The responses from the different classifiers usually have varying scales and offsets. Normalization must be implemented to map the scores from the multiple domains into a common domain before combining them. The *tanh-estimators* have been used for normalization. Given a list of scores S_{ij} where j denotes classifier j and i represents a feature, the scores can be normalized as follows [17]:

$$S_{ij} = \frac{1}{2}[tanh(.01\frac{S_{ij} - \mu_{tanh}}{\theta_{tanh}}) + 1] \in (0, 1) \tag{4.4}$$

where μ_{tanh} and σ_{tanh} are the average and standard deviation estimates of the scores S_{ij} where $i = 1, \ldots, I$ with I representing the number of features. Some combination strategies that have been used at the measurement level are the geometric average [17], the sum rule, decision trees, and linear discriminant analysis [117].

Browne, et al. [15] experimented with combining three shot boundary algorithms. The shot detection algorithms used for their research are color histograms, edge detections, and encoded macro blocks. It was concluded from their experiments that a dynamic threshold implementation of each algorithm improved shot boundary detection performance, though this is an individual algorithm improvement (tuning) rather than a fusion result. Weighted Boolean logic was used to combine the three shot boundary detection algorithms. Figure 4.1 illustrates the functionality of this method. The three shot detection algorithms are executed in parallel using dynamic thresholds for each algorithm. A shot is determined in a hierarchical manner. The algorithm works as follows:

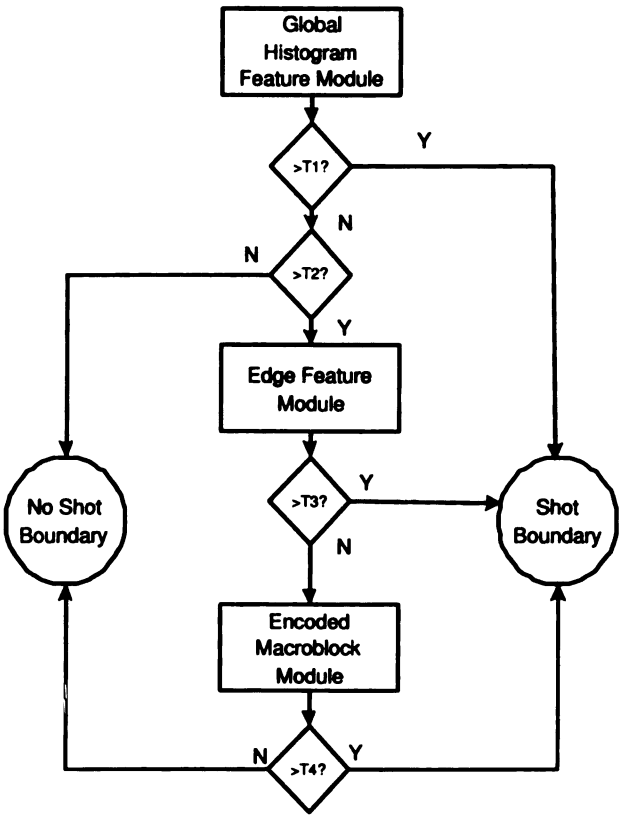

Figure 4.1: Browne Boolean Logic Method

- If the color histogram algorithm is above its adaptive threshold $T1$, a shot boundary is detected.
- If the edge detection algorithm is above its adaptive threshold $T3$ and the color histogram algorithm is above an alternative, minimum, threshold $T2$, then a shot boundary is detected.
- If the encoded macro block algorithm is above its threshold $T4$ and the color histogram algorithm is above its minimum threshold $T2$, a shot boundary is detected.

This algorithm performs fusion at the measurement level. The multiple scores from each algorithm are fused together using Boolean logic in a hierarchical manner. The reasoning behind this implementation is that color histograms offer the best performance in terms of speed and computational complexity. As a result, the color histogram method takes precedence in the hierarchy. It is not clear why the researchers chose to have the edge-based algorithm operate at the second level of the hierarchy, with the encoded macro block algorithm functioning at the bottom level. The video corpus used for testing was broadcast TV video. Their tests showed mixed results when compared to single method implementations. One problem with this algorithm is that the histogram method is always given the highest weight to determine shot boundaries. During some shot transitions and gradual effects, the color histogram may produce unreliable results. One of the main reasons why researchers have implemented algorithms that utilize multiple methods is to compensate for the unreliability of a single method. Since this algorithm relies on the performance of the color histogram algorithm, when this algorithm produces unreliable and incorrect results, the errors are incorporated into the decisions of the other algorithms.

4.1.3 Feature Extraction Level Fusion

Feature extraction level fusion involves using the data from each classifier to form a new, composite, feature vector. Since the features extracted from each classifier

are independent, these values are concatenated to form a new feature vector in a higher dimensional space. Methods such as neural networks, decision trees, SVMs, and Bayesian models can be used to classify the new vector.

Taskiran, et al. [132] developed a shot detection algorithm in the compressed domain-based on luminance histograms and the standard deviation differences from the luminance component of DC images which are subsampled images available from MPEG and motion-JPEG video sequences. DC images are effectively images constructed by averaging the pixel values in each block. A generalized trace or feature vector is extracted from each frame. The generalized sequence trace is defined as [132]:

$$d_i = \| \vec{x}_i - \vec{x}_{i+1} \|_2 \tag{4.5}$$

where $\vec{x}_i$ is a feature vector computed from 2 extracted features. The features that are used in the generalized trace are the histogram intersection of DC images and the standard deviation differences between successive frames.

DC images are computed from the I, B, and P frames of the video sequence. The histogram intersection is defined as follows:

$$D(H_i, H_{i+1}) = 1 - \frac{\sum_i min(H_i(i), H_{i+1}(i))}{N} \tag{4.6}$$

where $H_i(i)$ and $H_{i+1}(i)$ are the luminance histograms for frames f_i and f_{i+1} respectively.

The standard deviation differences between frames i and $i+1$ is computed as $D(i, i+1) = |\sigma_i - \sigma_{i+1}|$ where σ_i^2 is defined as:

$$\sigma_i^2 = \frac{1}{T-1} \sum_i \sum_j (Y_i(i,j) - \mu)^2 \tag{4.7}$$

μ is the mean value of the luminance image, and T is the number of pixels.

Shot detection is performed by locating observed edges in the generalized sequence trace. The edges in the generalized sequence trace are found using the morphological Laplacian. If the number of edges exceeds a predefined threshold, a shot boundary is declared.

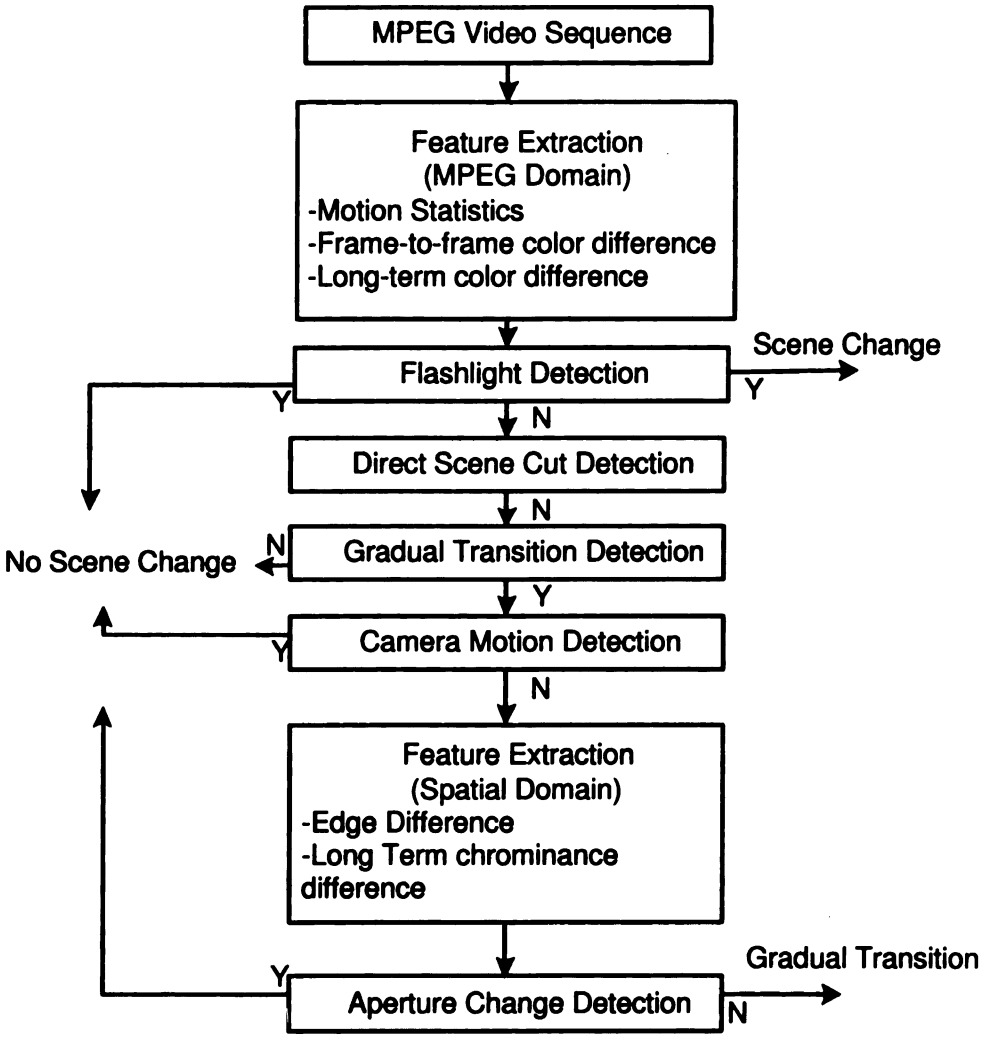

Figure 4.2: Zhong Multi-stage Shot Detection

The fusion method of Taskiran, et al. is preformed at the feature extraction level. A feature vector is extracted from each frame based on the luminance histograms and the standard deviation differences from the luminance component of dc-images. The feature vector is further processed by analyzing the edges in the generalized sequence trace. The reasoning behind this approach is that it is able to utilize multiple features and its performance is fast as a result of operating in the compressed domain.

Zhong, et al. [158] experimented with various heuristics utilizing multiple features to detect shot boundaries in the compressed domain. The multi-stage algorithm combines motion, color, and edge information. Figure 4.2 depicts the heuristic multi-stage method.

The frame-to-frame color difference between two I or P frames is defined in the YUV color space. The difference metric between two frames i and j is defined as [158]:

$$D(i,j) = |\bar{Y}_i - \bar{Y}_j| + |\sigma_Y^i - \sigma_Y^j| + w*(|\bar{U}_i - \bar{U}_j| + |\sigma_U^i - \sigma_U^j| + |\bar{V}_i - \bar{V}_j| + |\sigma_V^i - \sigma_V^j|) \quad (4.8)$$

where $\bar{Y}, \bar{U}, \bar{V}$ are the mean YUV values from the dc-images, and $\sigma_Y, \sigma_U, \sigma_V$ are the standard deviations of the YUV components.

The long-term color difference metric between the current I frame and its kth previous P or I frame is defined as [158]:

$$D_{long-term}(i) = D(i, i - (M+1)*k) \quad (4.9)$$

where M is the number of B frames between a pair of successive I or P frames.

Motion statistics are computed from the P and B frames. In P frames, a motion measure is determined from the ratio of intra-coded blocks to forward motion vectors. In B frames two motion statistics are extracted, the ratio of backward to forward motion vectors, and the ratio of forward to backward motion vectors.

Flashlight detection involves detecting abrupt changes in brightness patterns that are not at shot boundaries. To detect flashes the authors use the ratio of frame-to-frame color differences to the long-term color difference. The ratio for a current frame i is defined as [158]:

$$Fr(i) = D(i, i-1)/D(i+\delta, i-1) \quad (4.10)$$

where δ is the average length of the aperture change of the video. A flashlight is detected if $Fr(i)$ is greater than a predetermined threshold.

Shot detection is performed by identifying maximas from a temporal window in the color and motion feature difference values. A peak-to-average ratio (PA) is computed for each feature within the temporal window. In order to determine the optimal way to combine PA ratios consisting of many thresholds and possible combinations, a decision tree-based learning algorithm is computed for I, P, and B frames separately. The Oblique Classifier OC1 [101] is used to build the decision tree. Separate decision

trees are created for each type of frame because the different frames have different characteristics and features.

Gradual scene change detection is computed using the twin-comparison method described in 2.2.1.3. For long sequences, the beginning and ending edges of gradual transitions are found using a heuristic based on the induction results from the decision tree learning.

A multi-level thresholding scheme is used to reduce the problems occurring from false positives and false negatives. Instead of using one optimized threshold, multiple thresholds are used in a hierarchical manner. The process terminates when a cut is detected or the number of levels has been traversed. In order to obtain increased detection accuracy, camera motion detection and aperture change detection are implemented. The authors indicate receiving favorable results on various types of videos.

This algorithm utilizes fusion at the feature extraction level. Shot boundaries are determined via decision trees, which utilize a combination of multiple thresholds and multiple decision ratios. The reasoning behind this algorithm is that it utilizes a machine learning algorithm to fuse multiple features. It allows decision trees to determine scene boundaries. The problem with this algorithm is that it is very heuristic in nature. Additionally, it utilizes many thresholds that have to be predetermined through alternate testing outside of the system.

Sabata et al. [119] utilized a Bayesian framework to fuse multiple segmentation algorithms to improve shot detection performance. This algorithm utilizes color histograms, texture features using Gabor filters, and motion features extracted from spatiotemporal volumes.

The color histogram is computed for a candidate frame b over temporal intervals $[b-\epsilon, b]$ and $[b, b+\epsilon]$. The color histogram is then weighted using a Gaussian mask. The weighted histogram is defined as follows [119]:

$$h_{[t_s,t_e]}(i) = \sum_{t \in [t_s,t_e]} w_t \cdot h_t(i) \tag{4.11}$$

where w_t is the weight computed from the Gaussian function $G(x) = e^{-\frac{x^2}{2\pi\sigma^2}}$ The distance measure for the Gaussian weighted histograms is defined as follows [119]:

$$D^{color}(s,t) = \sum_i \frac{(h_{[l]}(i) - h_{[r]}(i))^2}{(h_{[l]}(i) + h_{[r]}(i))^2} \tag{4.12}$$

where $[l] = [t - \epsilon(s), t]$ represents the previous frames in the interval and $[r] = [t, t + \epsilon(s)]$ represents the remaining frames in the interval.

Texture features are used in the form of texture energy to compute segmentation boundaries. Gabor filters are used to determine the texture energy for each frame. The Gabor filter is defined as follows [119]:

$$G(x,y) = e^{-\frac{(x-X)^2+(y-Y)^2}{2\sigma^2}} sin(\frac{2\pi(x\cos\theta - y\sin\theta)}{\lambda} + \phi) \tag{4.13}$$

where λ is the wavelength, θ is the orientation, ϕ the phase shift, and σ the standard deviation of the Gaussian windowed sinusoidal waveform. Twelve filters are generated by quantizing θ into four values and λ into three values. The distance metric for texture energy is calculated over a temporal window using a scaled Gaussian window.

Motion features are also utilized to detect shot boundaries. The motion algorithm selects features to track based on the research of Shi, et al. [124]. A score is assigned to the tracked features based on the contribution of each feature that is tracked from the previous frame. The weight for a feature i is defined as:

$$w_i = 1 - e^{\frac{p_i}{k}} \tag{4.14}$$

where p_i is the number of previous frames that the ith feature was tracked and k is a constant that determines how sensitive the weight is to the number of previous frames. The distance is computed from the average of the tracked scores in a temporal window and a fraction of the missed tracked features.

Edges in the spatiotemporal volume are used to detect shot boundaries by projecting the video data along the $x - t$ and $y - t$ planes. The result of the projection is an image that can be analyzed to detect shot boundaries. Segment boundaries are determined by the number of horizontal edge pixels, perpendicular to the t axis,

in the spatiotemporal volume. A probability measure is calculated by averaging the number of horizontal edges across many sections.

A Bayesian framework [57, 58] is utilized for information fusion. The Bayesian classifier is defined as follows:

$$\overset{argmax}{O} \; Pr(Outcome = O|a_1, \ldots, a_n) \tag{4.15}$$

where the variable *Outcome* can take the value from the set $0, 1, 2$ where the set values indicate a regular frame, a shot boundary, or a flash. Naive and TAN Bayesian classifiers [45] were used for fusing the multiple features.

This algorithm performs fusion on the feature extraction level. The multiple features are fused using a Bayesian network. The problem with this implementation is that this algorithm was only tested on a limited set of videos (broadcast news). Additionally, the researchers focused their research efforts trying to eliminate false positives instead of also looking to increase accuracy.

4.1.4 Output Level Fusion

Output level fusion involves merging the output results of each of the individual classifiers. Output level fusion attempts to fuse shot boundaries based on the similarity among the output frames. Shot boundaries that are considered similar and are temporally close to one another are grouped together and the unified result of the merging should represent all the shot boundaries.

Ferman, et al. [43, 53] utilize two features, histograms and pixel differences for video segmentation via 2-class clustering. Video segmentation is treated as a 2-class clustering problem, where the two classes are "scene change" and "no scene change". The K-means clustering algorithm [66] is used on the similarity measure of color histograms between successive frames. Additionally, the sensitivity of the pixel difference method to object and camera motion caused the authors to filter the features before clustering. The authors concluded that the use of multiple features simultaneously can improve the shot detection performance.

Naphade, et al. [103] also utilizes a K-means clustering algorithm [66] on multiple features to detect shot boundaries. The algorithm utilizes histogram differences and a spatial difference metric. The spatial difference operator is defined as:

$$d_{i,j}(f_k, f_{k+1}) = \begin{cases} 1 & \text{if } |I_{i,j}(f_k) - I_{i,j}(f_{k+1})| > 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.16}$$

where $I_{i,j}(f_k)$ and $I_{i,j}(f_{k+1})$ denote the intensity of pixels at location (i, j) in frames f_k and f_{k+1}. The spatial difference metric D_s is defined as:

$$D_s(f_k, f_{k+1}) = \frac{1}{MxN} \sum_{k=1}^{M} \sum_{j=1}^{N} d_{j,k}(f_k, f_{k+1}) \tag{4.17}$$

An unsupervised 2-class clustering algorithm based on K-means clustering is used to classify shots.

The clustering algorithms of Ferman, et al. and Naphade, et al. both utilize output level fusion. The output frames of multiple algorithms are fused via K-means clustering. The cluster centers are used as the shot detection boundaries. The reasoning behind using clustering methods for shot segmentation is two-fold. First, they alleviate the need to develop pre-determined thresholds. Determining the proper algorithm thresholds to use is not a straightforward task. Moreover, this process is highly dependent upon the type of video under analysis. For example, the optimal set of thresholds for one video sequence may be suboptimal for another. Eliminating this procedure of selecting thresholds was a key factor in developing clustering-based methods. Secondly, they allow for the inclusion of multiple features to determine a best segmentation. Researchers have long recognized the need to use multiple features to increase reliability and shot segmentation performance. The negative aspects of clustering-based algorithms are that false positives can result in missed shot boundaries. Frames from scenes that cause numerous false positive detections will appear more in clusters and could eliminate true cut frames from being selected as the cluster center. This phenomenon can lead to misdetections during the clustering process which often leads to missed shot detection boundaries. Additionally, the clustering algorithm all require the entire video to be processed before segmentation which can be problematic for long video sequences. To combat this problem, Ferman, et al. has

developed an ad hoc method for computing clustering on the fly by processing the video in temporal blocks and performing clustering on those blocks.

Miene et al. [100] combined a frequency-domain approach based on FFT features, gray-level histograms, and a technique based on the changes in luminance values to detect shot boundaries. Each video frame is converted into a gray-scale image and is transformed using an FFT. The sum of the absolute differences of the real and imaginary parts for consecutive frames n and $n-1$ is calculated as follows:

$$F_{Total}(n, n-1) = |R_{Sum}(n) - R_{Sum}(n-1)| + |I_{Sum}(n) - I_{Sum}(n-1)| \quad (4.18)$$

where R_{Sum} is calculated by adding the lower frequencies of the real part and I_{Sum} is calculated as the values of the imaginary-part.

Miene also uses the difference of luminance values between consecutive frames to determine shot boundaries. Each frame in converted to YUV 1:1:1 format. The absolute differences of luminance (Y) values of consecutive frames are calculated as follows:

$$Y_{Diff}(n, n-1) = |Y_{Sum}(n) - Y_{Sum}(n-1)| \quad (4.19)$$

where Y_{Sum} is calculated by summing all the luminance values of each frame:

$$Y_{Sum} = \sum_{x=0}^{w-1} \sum_{y=0}^{h-1} Y(x, y) \quad (4.20)$$

w and h and the width and height of the input frame.

Gray-level histogram differences are also used to detect shot boundaries. The difference between consecutive histograms is calculated as follows:

$$H_{G_Diff}(n, n-1) = \sum_{i=0}^{255} \frac{(H_G(n)(i) - H_G(n-1)(i))^2}{MaxH_G(n)(i), H_G(n-1)(i)} \quad (4.21)$$

where H_G is the gray-level histogram, i is the histogram index, and n is the frame number.

This implementation illustrates output level fusion. Each algorithm detects shot boundaries individually and boundaries within a pre-determined temporal threshold are merged. Each algorithm creates a boundary list consisting of all the merged shots. The three boundary lists are merged into a single shot boundary list by merging the

overlapping boundaries. All detected boundaries within a pre-determined threshold are reduced to one boundary. One problem with this implementation is the errors that arise due to camera motion. There is no inherent algorithm to handle shot boundaries that arise from camera motion. Another problem with this implementation is that each algorithm is given equal weight which does not take into account that fact that different algorithms perform better for various classes of content. Additionally, the authors provide a pre-determined threshold to determine the length of a shot. This manual threshold could conflict with the actual shot boundaries and could eliminate shots from analysis.

4.1.5 Fusion Level Comparisons

When developing our solution for a fusion-based shot detection algorithm we analyzed the various fusion levels in digital video and concluded that the best levels to achieve effective shot segmentation performance was at the *measurement and feature extraction levels.* The reasoning behind this decision was that abstract and output level fusion offer very little information with respect to the decision making process. With only the output decisions or the output frames from the classifiers available there are only a limited amount of fusion strategies that one can apply and determining a false positive or negative for a presented algorithm results is problematic. These implementations can be considered ad-hoc, because they lack underlying theory, and the relative importance of each classifier is ignored or arbitrarily assigned [90]. Additionally, the fusion strategies that are available at these levels do not differ by a significant amount. Fusion at the measurement and feature extraction levels provide the most information available to make decisions regarding a best segmentation. These levels attempt fusing algorithm thresholds, scores, and confidence measures and offer a wide variety of combination methods.

4.2 Results Fusion for Video Shot Segmentation

This research models shot-based video segmentation as a binary classification problem. Binary classification problems classify data into two distinct groups, usually accept or reject. In the digital video domain the two distinct classes would be shot change and no shot change. There has been a lot of research with respect to solving binary classification problems in the machine learning community [117, 9, 146]. Some methods that have been used to solve binary classification problems in the biometric community are support vector machines, Bayesian classifiers, linear discriminate analysis, decision trees, and neural networks. Researchers in the biometric community have reported receiving optimal performance using support vector machines for binary classification [9, 146]. As a result, one of the results fusion engines designed is based on SVMs. SVMs fuse multiple classifiers at the feature extraction level, with the outputs of the multiple classifiers synthesized in a vector in a high dimensional space. This new vector is then analyzed by the SVM to make a judgment as to its class. We extract key features from a video sequence and utilize SVMs for classification.

4.2.1 Support Vector Machines

Support Vector Machines (SVM) are used to solve binary classification problems by mapping the training data onto a higher dimensional space and then determining the optimal separating hyperplane within that space by solving a Quadratic Programming (QP) problem [137]. SVMs are based on the principle of Structural Risk Minimization (SRM), which differs from classical statistical learning approaches. Classical statistical learning approaches are designed to minimize the empirical risk, by reducing the misclassification errors on the training set. The principal of SRM states that better generalization capabilities are achievable through a minimization of the bound on the generalization error. Thus, they attempt to minimize the probability of misclassifying a previously unseen data point drawn randomly from a fixed but unknown probability distribution. They provide an upper bound for the probability of misclassification of the test set for any possible probability distributions of the

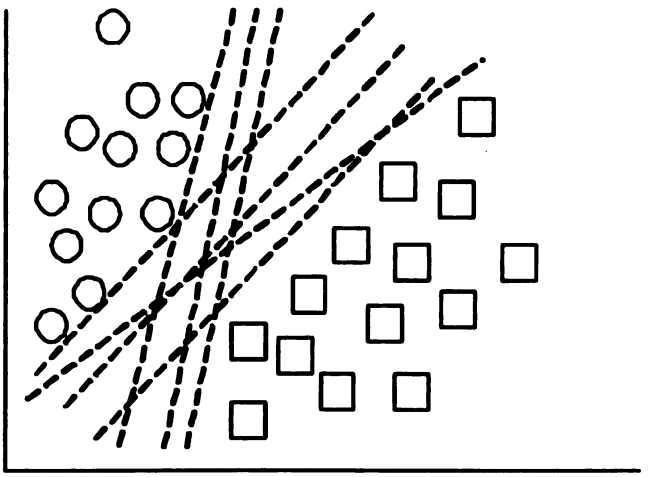

Figure 4.3: Multiple Separating Hyperplanes in 2D space

data points [113]. SVMs have exhibited good generalization performance for face recognition [108], text categorization [39], multi-modal person authentication [139], and optical character recognition [19].

Given a data set D which consists of m points in an n-dimensional space belonging to two different classes $+1$ and -1, D is defined as:

$$D = \{(\mathbf{x}_i, y_i) | i \in \{1 \ldots m\}, \mathbf{x}_i \in \Re^n, y_i \in \{+1, -1\}\} \tag{4.22}$$

A binary classifier finds a function $f : \Re^n \rightarrow \{+1, -1\}$ that maps points from their domain to their label space. Although there exists an infinite number of hyperplanes that could partition the data into two states (see Figure 4.3), the principle of SRM states that there will be one hyperplane with the maximal margin, with the margin being defined as the sum of the distances from the hyperplane to the closest points of the two classes. In order to detect the optimal hyperplane only a small amount of training data is needed. This data is in the form of support vectors that determine the margin. Figure 4.4 depicts the support vectors and the optimal hyperplane between two classes, squares and circles in 2 dimensions. A set of training patterns is said to be linearly separable if there exists a vector $\mathbf{w}$ and a scalar b such that

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b &\geq 1 \quad \text{if} \quad y_i = 1 \\ \mathbf{w} \cdot \mathbf{x}_i + b &\leq -1 \quad \text{if} \quad y_i = -1 \end{aligned} \tag{4.23}$$

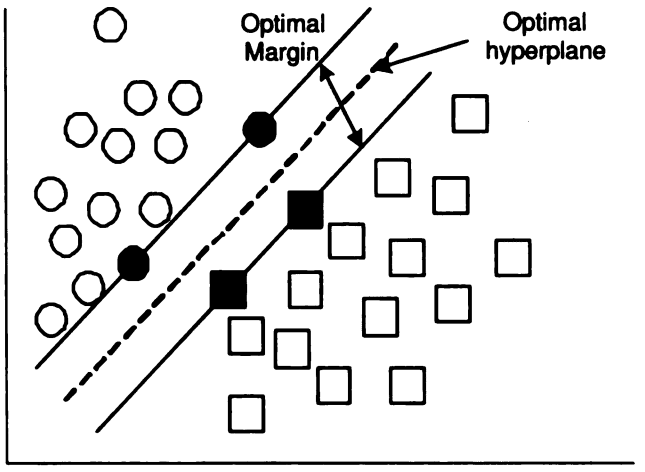

Figure 4.4: The decision boundary and optimal hyperplane in 2D space [78]. The support vectors in red denote the margin of the largest separation between the two classes.

are valid for all elements of the training set [26]. The distance between hyperplane $\mathbf{w} \cdot \mathbf{x}_i + b \geq 1$ and hyperplane $\mathbf{w} \cdot \mathbf{x}_i + b \leq -1$ is $2/\|\mathbf{w}\|$, where $\|\mathbf{w}\|$ is the Euclidean norm of $\mathbf{w}$. $2/\|\mathbf{w}\|$ represents the minimum distance between points of different classes. Equation 4.23 can be rewritten as:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1, i \in \{1 \ldots m\} \tag{4.24}$$

By minimizing $\frac{1}{2}\|\mathbf{w}\|^2$ subject to the constraints of Equation 4.24 we obtain two hyperplanes with the maximum margin. The optimal hyperplane is defined as [26]:

$$\mathbf{w}_0 \cdot \mathbf{x} + b_0 = 0 \tag{4.25}$$

The optimization problem of minimizing $\frac{1}{2}\|\mathbf{w}\|^2$ can be solved via quadratic programming, which is guaranteed to find the global maximum.

The dual form of Equation 4.24 can be written as:

$$\text{maximize } W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{4.26}$$

where $\alpha_i \geq 0$ and $\sum_{i=1}^{m} \alpha_i y_i = 0$. Classifying a new data set $\mathbf{z}$ with s data points is defined as follows:

$$\mathbf{w} \cdot \mathbf{z} + b = \sum_{j=1}^{s} \alpha_j y_j (\mathbf{x}_j \cdot \mathbf{z}) + b \tag{4.27}$$

where $\mathbf{z}$ is defined as class 1 if the sum is positive and class 2 otherwise.

If the dataset is linearly non-separable we can allow for error in the classification (see Figure 4.5). ξ represents the slack variables in optimization theory such that Equation 4.23 can be rewritten as [26]:

$$\begin{aligned} \mathbf{w} \cdot \mathbf{x}_i + b &\geq 1 - \xi_i \quad \text{if} \quad y_i = 1 \\ \mathbf{w} \cdot \mathbf{x}_i + b &\leq -1 + \xi_i \quad \text{if} \quad y_i = -1 \end{aligned} \tag{4.28}$$

where $\xi_i \geq 0$. Equation 4.24 can be rewritten as:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, i \in \{1 \ldots m\} \tag{4.29}$$

Now the optimization problem becomes $\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n} \xi_i$ where C is a tradeoff parameter between the error and the margin, subject to the constraints of Equation 4.29 [26]. The dual form of Equation 4.29 can be written as [26]:

$$\text{maximize } W(\alpha) = \sum_{i=1}^{M} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{M} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j), \text{where } 0 \leq \alpha_i \leq C \tag{4.30}$$

The only difference between the linearly separable case in Equation 4.26 and the linearly non-separable case in Equation 4.30 is the upper bound C on α_i. Just as in the linear case a quadratic problem solver can be used to solve for α_i [78].

Thus far, we have described SVMs in terms of linear separable decision surfaces in the input space, however their generality allows for more diverse decision surfaces. The key is to map the original input patterns in $\mathbf{x}_i$ into a higher dimensional features space $\phi(\mathbf{x}_i)$. To develop a hyperplane in feature space one has to transform the M dimensional input vector $\mathbf{x}$ into an m dimensional vector through an M dimensional vector function $\phi : \Re^m \rightarrow \Re^M$ [120]. The reasoning behind this transformation is that linear operations in the feature space are equivalent to non-linear operations in the input space (see Figure 4.6) [78]. Maximizing Equation 4.26 in feature space requires the computation of dot products: $\phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ in a high dimensional space [120]. This computation can be done via kernel functions such that:

$$K(\mathbf{x}, \mathbf{y}) = (\phi(\mathbf{x}), \phi(\mathbf{y})) \tag{4.31}$$

The kernel function $K(\mathbf{x}, \mathbf{y})$ denotes a prior knowledge about the similarity between data $\mathbf{x}$ and $\mathbf{y}$. Some kernel functions that are used are [9]:

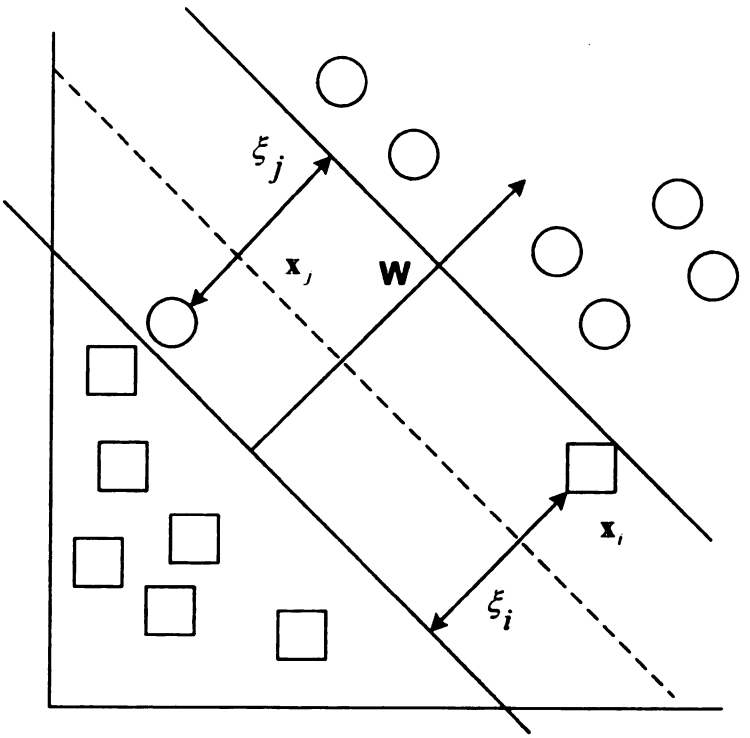

Figure 4.5: SVM Theory with slack variables ξ_i and ξ_j [78]

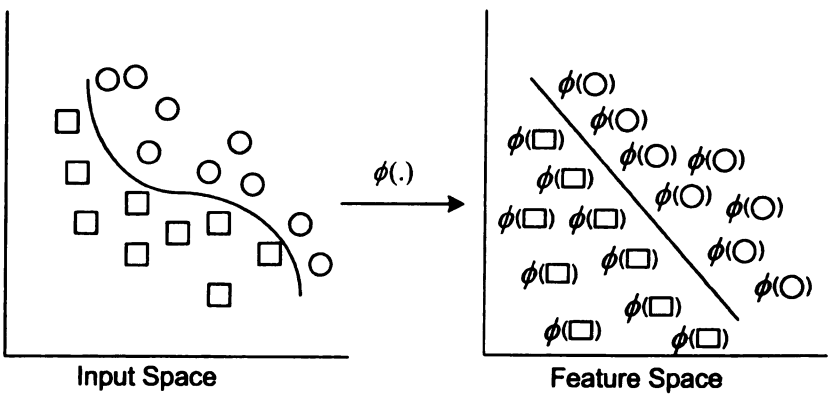

Figure 4.6: SVM Transformation [78]

- $K(\mathbf{x}, \mathbf{y}) = \mathbf{x}^t\mathbf{y}$ Linear Kernel
- $K(\mathbf{x}, \mathbf{y}) = (\mathbf{x}^t\mathbf{y} + 1)^p \quad p \in N$ Polynomial Kernel
- $K(\mathbf{x}, \mathbf{y}) = tanh(a\mathbf{x}^t\mathbf{y} + b)$ with $(a, b) \in \Re^2$ Muli-Layer Perceptron Classifier
- $K(\mathbf{x}, \mathbf{y}) = e^{-\|\mathbf{x}-\mathbf{y}\|^2/2\sigma^2}$ Radial Basis Function Classifier

In Equation 4.27 new data was classified in the input space. Classifying new data set z with s data points in the feature space using a kernel function is defined as:

$$f = (\mathbf{w}, \phi(\mathbf{z})) + b = \sum_{j=1}^{s} \alpha_j y_j K(\mathbf{x}_j, \mathbf{z}) + b \tag{4.32}$$

where the new data $\mathbf{z}$ is classified as class 1 if $f \geq 0$ and class 2 if $f < 0$.

Using SVMs involves a user specifying the type of kernel and the parameters associated with that kernel type which is a major advantage because no special knowledge or expensive tests are needed to set the values of the parameters. Additionally, the complexity of the SVM during training is not dependent on the dimensionality, but on the number of data points. The number of computation steps required for SVMs are $O(n^3)$ where n is the number of data points [9]. During execution the classification of the data points in just a weighted sum (see Equation 4.32). Additionally, during the training phase only the relevant information is needed in the form of support vectors. By only utilizing the important points, in the form of support vectors, this reduces the size of the training set. Also, this provides for an efficient means of classification.

4.2.2 Decision Trees

Decision trees are a statistical machine learning approach that creates a series of *if-then-else* rules in the form of a tree-like structure from the training data set in order to make decisions regarding class labels. It uses a top-down induction strategy from a training set of data to construct the tree structure. Decision trees do this by analyzing the attributes or features to determine values that maximize the information gain at a particular node [117]. The information gain is usually the decrease in entropy as a result of making a decision as to which attribute to use and at what level in the tree.

Decision trees are designed for problems whose instances are represented in *key-value pairs*. Given a set of examples, the decision tree assigns classification to each example. Although decision trees were primarily designed for discrete valued data, they have been extended to incorporate real-valued floating-point data. At each internal node of the tree, a binary decision is made based on a threshold value. The threshold is chosen that best yields the greatest information gained by partitioning the set into two subsets based on the threshold.

A decision tree is constructed from the root node. From the root, a decision tree grows by splitting the data at each level based on maximizing a cost function to form new nodes. This cost function usually measures the impurity or variance of the example data sets. After choosing on a split, the subsamples are then mapped to the two children nodes. This procedure is then recursively applied to each child node until a stopping criteria is met. The nodes are connected by branches with leafs being the nodes at the end of the branches. Each internal node of the decision tree is a classifier, with the classification determined at each leaf node. The nodes in the tree contain information about the number of instances and the distribution of dependent variables at that node. The root node contains all the instances of the training set. Once constructed, a tree predicts a new case by starting at the root and following a path until a leaf node is reached. The outputs of the internal nodes determine a unique path from the root to the leaf of the decision tree. The path is determined by the splitting rules on the values of the independent variables in the new instance.

4.2.3 Rulesets

Rulesets are an unordered collection of *if-then-else* rules generated from decision trees. They are designed to help make a decision tree more readable and understandable. The set of rules generated from decision trees consists of at least one default rule, which is used to classify unseen instances when no other rules apply. Generally, a ruleset will have fewer rules than a decision tree has leaf nodes, which aides in its understandability. Additionally, rulesets are often more accurate predictors than decision trees. Each generated rule consists of an attribute value pair, the resulting

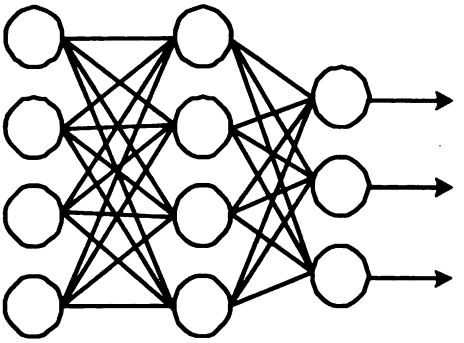

Figure 4.7: Two-Layer Neural Network Architecture

classification, the number of instances that the rule covers, followed by a percentage that represents the confidence measure for that rule.

4.2.4 Neural Networks

Neural networks have frequently been used for the recognition and estimation of input patterns. The strength of neural networks is based on its discriminative ability to learn and represent the underlying patterns of the input data. Neural networks can utilize numerous learning algorithms with a seemingly infinite amount of network architectures. Figure 4.7 illustrates a two-layer neural network architecture.

This network can be thought of as a non-linear decision making process. Let X denote the input pattern $(x_1, \ldots, x_i)$ and W denote the set of outputs $(w_1, \ldots, w_j)$. The output y_i determined from the output nodes as follows [22]:

$$y_i = f\left\{\sum_k w_{ik} f(\sum_j w_{kj} x_j)\right\} \tag{4.33}$$

where w_{ik} and w_{kj} denote the weights between kth hidden node to the ith class output and the weight between the jth input node to the kth hidden node respectively. The function f is the transfer function such as the log-sigmoid function, tan-sigmoid function, or a pure linear transfer functions. The transfer function forms the decision boundary between the classes. The node with the maximum value is selected as the class node. Figure 4.8 illustrates the log-sigmoid, tan-sigmoid, and linear transfer functions.

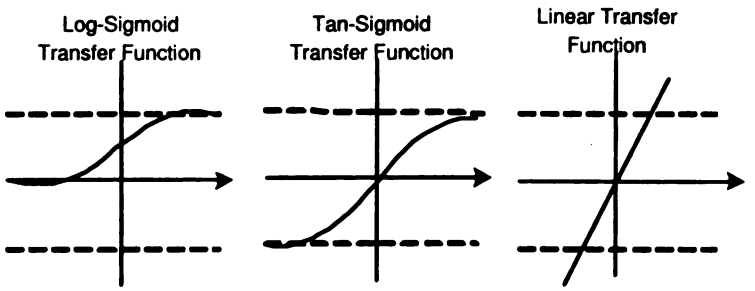

Figure 4.8: Neural Network Transfer Functions

Training of a neural network can be facilitated through supervised learning. Supervised learning involves the use of known input patterns and classes. The weights and biases of the network are chosen to minimize a squared-error cost function. This cost function can be defined as:

$$E[\sum_{i=1}^{k}(y_i(X) - d_i)^2] \tag{4.34}$$

where E is the expected value operator, $y_i(X)$ are the actual outputs of the network and d_i are the desired outputs of the network. It has been shown that the outputs of the neural network can be estimates of Baysian posterior probabilities if configured correctly[116]. Thus Equation 4.34 can be generalized to the form [22]:

$$E[\sum_{i=1}^{k}(y_i(X) - E[d_i|X])^2] + E[\sum_{i=1}^{k} var[d_i|X]] \tag{4.35}$$

where $E[d_i|X]$ and $var[d_i|X]$ are the conditional probability and conditional variance of the desired output d_i. The second term in Equation 4.35 is not dependent on the network outputs, the minimization of the cost function can be described in terms of the the mean-squared error (MSE) between the network outputs and the conditional expectation of the desired outputs [22]. Thus, the desired output d_i is assigned to class w_i by the following Bayesian probabilities [22]:

$$E[d_i|X] = \sum_{i=1}^{k} d_i P(w_j|X) = P(w_i|X) \tag{4.36}$$

4.3 Features

The features used for results fusion can come in very diverse forms. Some of these forms include continuous values, such as with numerical data, binary values, discrete labels, timestamps or dates. In Section 3.1 we discussed the implemented shot boundary detection algorithms: global histograms, local histograms, edge features, and DCT coefficients. As a result of the implemented algorithms we obtain the following features between pair-wise frames, global histograms differences, local histogram differences, the edge change ratio, and DCT coefficient differences. Each one of these values represents continuous data. It has been shown that when developing shot segmentation strategies that utilize multiple features, improved results can be obtained when features are chosen that complement one another [53]. Thus, in a situation in which extracting one feature may be difficult or unreliable, extracting another feature may be more appropriate. The features that are computed for our shot segmentation algorithm are based on their overall performance exhibited by a history of research in the video shot segmentation community. Additionally, the chosen features are based on how the strengths and weaknesses of each algorithm complement one another.

Global histograms are the most common feature used for color-based video shot segmentation (see Section 2.2.1.3). One of the strengths of this method is that it does a good job of detecting abrupt changes between shots. Additionally, histograms are easy to implement and computationally fast. One of the weaknesses of global histogram methods is that they ignore spatial content within the video frame. (Indeed, two images that are very dissimilar can have identical global color histograms.) Thus, consecutive frames that have different spatial distributions, but have similar histograms, are considered similar. Another weakness is that global histograms are not tolerant of local changes within a video frame.

Local histograms combat the weaknesses of global histograms. Methods based on local histograms are tolerant of local changes, however they are still sensitive to changes in luminance over the entire frame [95]. Additionally, both histogram-based methods do not inherently detect gradual transitions.

When color information cannot be easily extracted from a video frame, edge features can be used to determine shot boundaries. Zabith, et al. [152] detect cuts, fades, dissolves, and wipes based on the appearance of intensity edges that are distant from edges in the previous frame. One advantage of this feature is that it detects abrupt, as well as gradual, transitions. One disadvantage of this feature is that it is less reliable in the presence of multiple independently moving objects [74] because multiple moving objects cause the global motion compensation aspect of this method to produce errors during registration and prevent any registration method from allowing edges to correspond (see Section 2.2.1.5). Another disadvantage of this method is an increase in false positives. Changes in image brightness or low quality frames, where edges are harder to detect or appear and disappear due to noise, may cause false positives.

DCT coefficients represent the spatial frequency components that comprise the block of pixels in a video frame (see Section 2.2.2.2). One advantage of using DCT coefficients is that they are often available as a byproduct of the MPEG compressed video stream. Hence, their extraction is computationally fast. One disadvantage of this method is that since the algorithm only processes I frames, temporal resolution is decreased, though this can be alleviated using partial decompression in the analysis.

4.4 Results Fusion Shot Segmentation

We have implemented results fusion shot segmentation strategies based on support vector machines (SVM), Decision Trees, Rulesets, and Neural Networks. The result of each individual feature extraction module is used to form a feature vector, and the new feature vector is fed to the results fusion engine to make a final decision as illustrated in Figure 4.9.

Each feature module extracts pair-wise frame differences and these metrics are used to form a new feature vector. The new feature vector consists of global histogram differences, local histogram differences, the edge change ratio, and DCT coefficient differences. The goal of the results fusion engine is to classify each feature vector into one of two classes: shot boundary or no shot boundary.

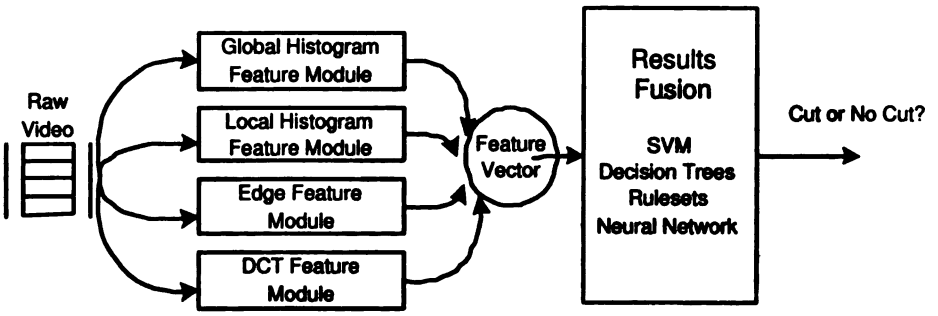

Figure 4.9: Results Fusion Video Segmentation Methods

The SVM results fusion implementation is based on the libsvm software [20]. We used the C-SVM implementation of SVM [137]. C-SVM solves the following quadratic programming problem:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\mathbf{w}\cdot\mathbf{w} + C\sum_{i=1}^{l}\xi_i \\ & y_i(\mathbf{w}\cdot\phi(\mathbf{x}_i)+b) \geq 1-\xi_i, \\ & \xi_i \geq 0, i = 1,\ldots,l. \end{aligned} \tag{4.37}$$

It has been shown that the linear kernel is a special case of the Radial Basis Function (RBF) kernel and the polynomial kernel has some numerical difficulties. As a result, the RBF kernel was used because it is the most appropriate kernel to use for a wide variety of data sets [89].

The decision tree results fusion method is based on the C5.0 data mining software [114]. Figure 4.10 illustrates a sample partial output of the decision tree created by C5.0 on a training data set. The training data consisted of actual shot boundaries and regular frames. The ruleset-based results fusion method is also based on the C5.0 data mining software [114]. Figure 4.11 depicts a partial sample of the rules generated by C5.0 on a sample data set. The ruleset output displays the number of testing and training instances that each rule covers, the *if-then-else* rule, and a confidence measure.

The neural network results fusion method is based on the feed-forward neural network. The Matlab Neural Network Toolbox was used to design the network architecture. The network architecture consisted of 10 neurons in the hidden layer and 1 neuron in the output layer. The hidden layer uses the log sigmoid transfer func-

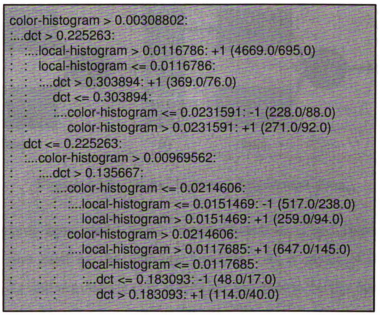

Figure 4.10: Partial Decision Tree Output of C5.0

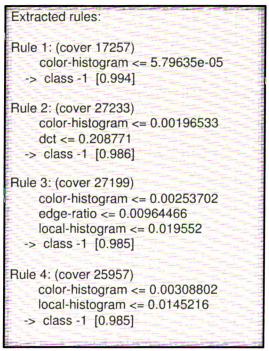

Figure 4.11: Partial Ruleset Output of C5.0

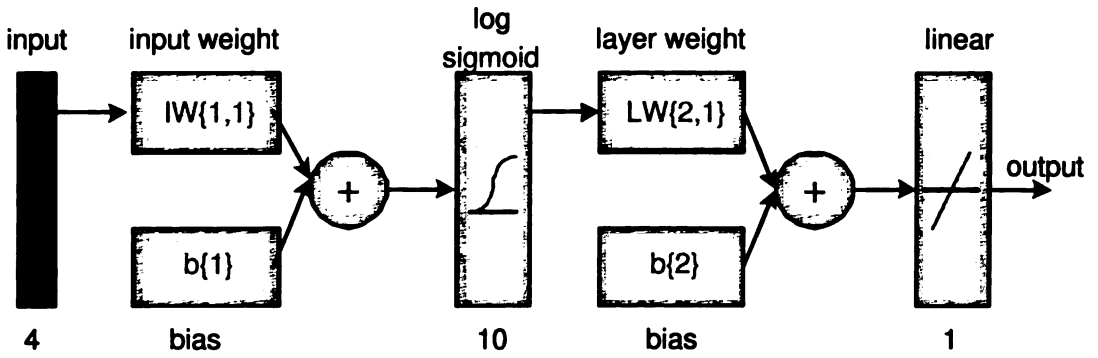

Figure 4.12: Neural Network Implementation

tion and the output layer uses a linear transfer function. The Levenberg-Marquardt algorithm [54] was used as the training function. As a result of its optimization techniques, the Levenberg-Marquardt algorithm appears to be the fastest method for training feed-forward neural networks [147]. Training was stopped when the training error went below .0001 or after 400 learning epochs. Figure 4.12 illustrates our neural network with 4 input nodes, 1 hidden layer with 10 neurons, and 1 output layer.

4.5 Baseline Testing Methods

Our results fusion-based segmentation strategies were baseline tested against a static and dynamic implementation of each individual shot detection method. Additionally, each one of our results fusion strategies was tested against a unimodal version of its implementation. The purpose of implementing a unimodal decision tree, ruleset, neural network, and SVM classifier was to determine how much or if fusion was actually being applied. Lastly, we implemented two well-known combination strategies in the video domain based on Boolean logic and majority voting.

4.5.1 Boolean Logic

Our Boolean logic method was based on the hierarchical method of Browne, et al. [15]. If the color histogram feature module is above its high threshold $T1$, a shot boundary is detected. If the local histogram feature module is above its threshold

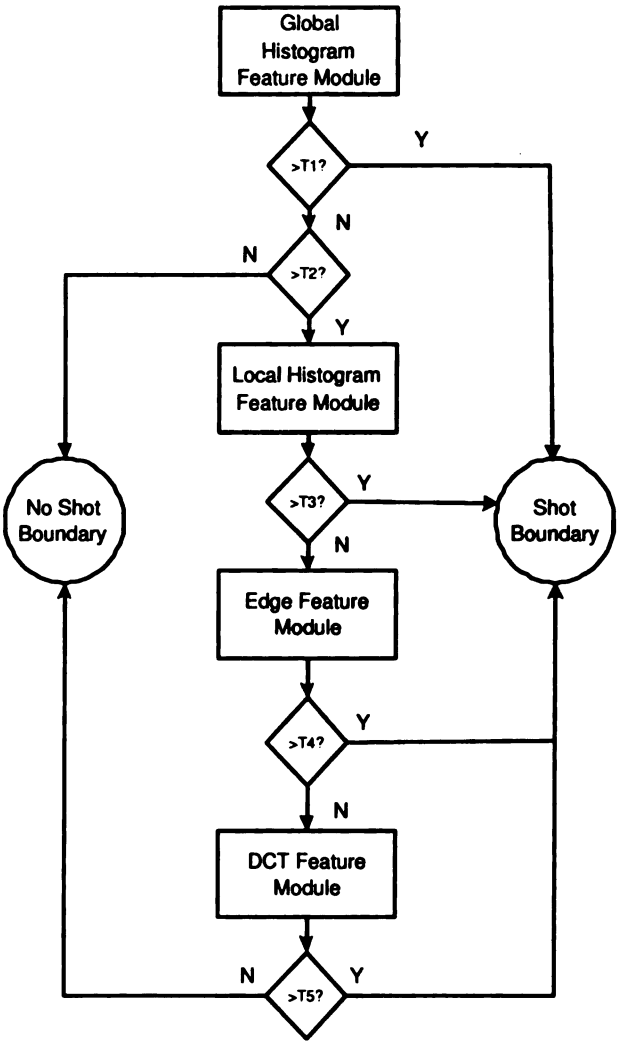

Figure 4.13: Boolean Logic Method

$T3$ and the color histogram feature module is above a minimum threshold $T2$, then a shot boundary is detected. If the edge feature module is above its threshold $T4$ and the color histogram feature module is above a minimum threshold $T2$, then a shot boundary is detected. Lastly, if the DCT feature module is above its threshold $T5$ and the color histogram algorithm is above its minimum threshold $T2$, a shot boundary is detected.

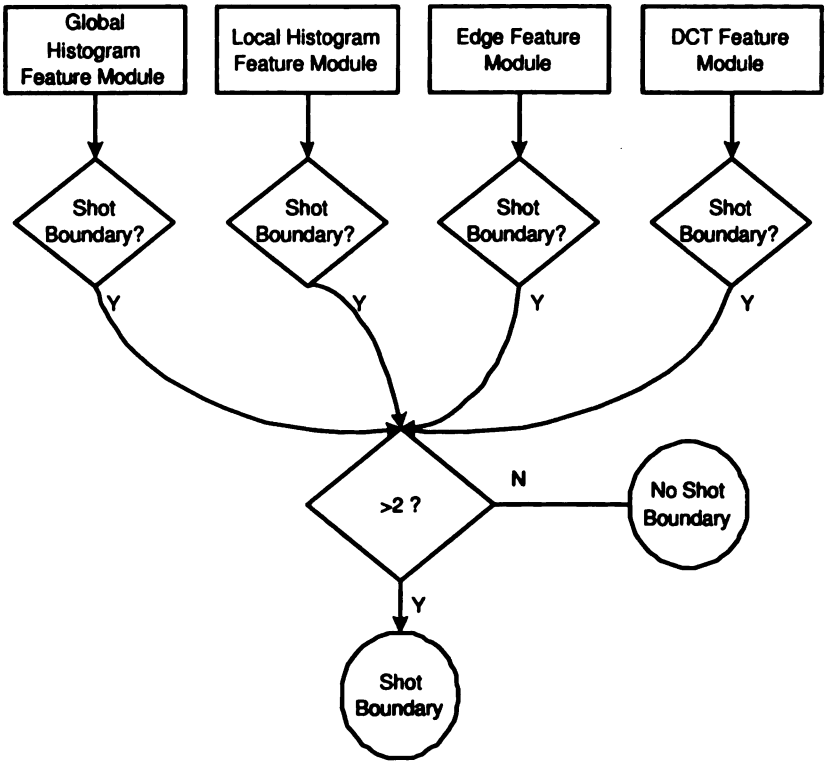

Figure 4.14: Majority Voting Method

4.5.2 Majority Voting

We have also implemented a majority voting algorithm similar to the research of Yusoff, et al. [151]. Each feature detection module outputs a decision value to determine if the current video frame under analysis is a shot boundary. If at least three of the four feature detection modules indicate a shot boundary then the fusion system determines that a shot boundary is present. This method is illustrated in Figure 4.14.

4.6 Cross Validation

Cross validation allows one to get a more reliable estimate of the predictive accuracy of the classifier. We performed ten-fold cross validation with the SVM, decision trees, rulesets, and neural network implementations. Ten-fold cross validation has been statistically proven to be good enough in evaluating the robustness and performance of

classifiers [130, 143]. The training data is decomposed into 10 equally sized randomly generated subsets. The classes are distributed evenly among the 10 subsets. Nine of the subsets are used to train the learner and the tenth hold-out subset is used for testing. This procedure is repeated ten times, with a different randomly generated subset used for testing each repetition.

4.7 Summary

This section described the levels of results fusion noting each levels strengths and weaknesses. As a result, this research focused on developing methods for results fusion-based on the measurement and feature extraction levels. These levels offer the most flexibility in developing a composite system. It then described some current attempts at developing results fusion techniques for digital video. Each of the described techniques was ad hoc and not practical on a large test suite of video. This chapter then described the novel results fusion-based methods developed for shot segmentation. Each one of the strategies has been used in information and biometric retrieval systems. This research has adapted their approaches to the video domain. Additionally this chapter described a new classification method in the area of pattern classification, support vector machines. This method has been receiving a lot of attention in solving binary classification problems and is used as one of our results fusion shot segmentation methods.

Chapter 5

Experimental Evaluation

Key to any new development in digital video segmentation is the validation of the performance of the proposed method on a collection of content. For this reason, the research presented in this thesis includes the construction of a video corpus with associated ground-truth segmentation data. This chapter describes the video corpus and the method for experimental evaluation of fusion-based segmentation algorithms. Images in this thesis are presented in color.

5.1 Video Corpus

A video corpus consisting of various classes of video content has been collected for this experimental evaluation and ground truth data manually generated. The classes of video include motion pictures, TV sitcoms, cartoons, Unmanned Aerial Vehicle (UAV) footage and music videos. The collection consists of over 8 hours of video.

Figure 5.1 describes some characteristics of the video corpus including the type, title, duration, and frame-to-shot ratio. All videos were digitized at a size of 352 by 240 at a frame rate of 30fps. The videos were decoded for analysis using the CODEC supplied with Microsoft DirectShow. Some characteristics of the video corpus are as follows:

1. *Television Programs*: This class of videos included 2 one hour episodes of the television program *24*. This test class includes over 84 minutes of video.

Video Type	Video Title	Length	# of Frames	# of Cuts	Ratio
Cartoon	The Family Guy A	21:49	39270	261	150:1
Cartoon	The Family Guy B	21:39	38970	326	119:1
Cartoon	The Family Guy C	21:48	39240	246	159:1
Cartoon	The Family Guy D	21:49	39270	260	151:1
Cartoon	The Family Guy E	21:38	38940	310	125:1
Movie	Blade 2	109:45	197550	2301	85:1
Movie	The Royal Tenenbaums	50:43	91290	479	190:1
Music Video	Destinys Child A	4:06	7380	233	31:1
Music Video	Destinys Child B	3:40	6600	239	27:1
Music Video	Destinys Child C	3:18	5940	208	28:1
Music Video	Jay Z	4:23	7890	208	37:1
Music Video	Michael & Janet Jackson	4:49	8670	250	34:1
Music Video	Mya	4:00	7200	111	64:1
Music Video	R Kelly A	4:20	7800	123	63:1
Music Video	R Kelly B	7:20	13200	193	68:1
TV Drama	24 A	42:32	76560	564	135:1
TV Drama	24 B	42:18	76140	700	108:1

Figure 5.1: Video Corpus

2. *Movies*: This class of videos included the movie *Blade 2* and the first 50 minutes of *The Royal Tenenbaums*. This test class includes over 160 minutes video.

3. *Cartoons*: This class of videos included 5 episodes of *The Family Guy*. Over 110 minutes of the video are included in this test class of video.

4. *Music Videos*: This class of music videos includes content by Destinys Child, Jay Z, Mya, Michael and Janet Jackson, and R. Kelly. This test class included over 30 minutes of video.

The ground-truth data has been collected using a custom video file scripting tool that enables a user to manually select and annotate shot boundaries in video sequences. The focus of any shot-boundary detection method is the determination of boundaries between camera sequences, something that, while difficult for computers to currently determine, is easy for humans to assess. Figure 5.2 shows the interface for the video file scripting tool. The ground truth data was collected by a student assistant in the lab.

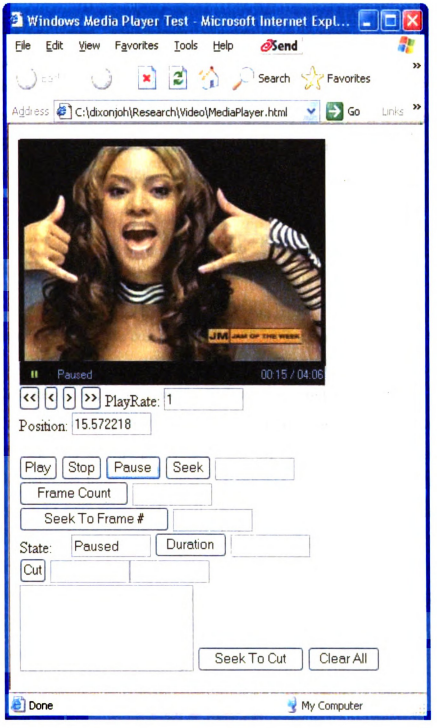

Figure 5.2: Video Scripting Tool

5.2 Performance measures

During experimentation, we wanted to develop a statistically acceptable framework to assess the performance of each algorithm. Researchers have used various methods to compare performance between algorithms using the confusion matrix. Figure 5.3 illustrates the confusion matrix with TP indicating the true positives, FP indicating the false positives, TN indicating the true negatives, and FN indicating the false negatives. Some researchers have classified performance in terms of two statistics, precision and recall [47, 100, 109]. Precision and recall are expressed as:

$$Precison = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \tag{5.1}$$

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \tag{5.2}$$

Other researchers compare competing algorithms in terms of the three basic metrics: correct detections (true positives), false detections (false positives), and missed detections (false negatives) [18]. However, it should be noted that these are often competing metrics. Many algorithms based on a thresholding mechanism can be adjusted to increase the number of true positives at the expense of increased false positives or adjusted to decrease false positives at the expense of false negatives. This same observation applies to precision and recall, though this is typically accommodated by stating precision and recall together in a precision/recall graph (also called a receiver operating curve or ROC).

Research in the biometric community utilizes the false acceptance rate (FAR) and the false rejection rate (FRR) to compare performance between biometric systems. These statistics are defined as follows:

$$FAR = \frac{\mathrm{FP}}{\mathrm{FP+TN}} \tag{5.3}$$

$$FRR = \frac{\mathrm{FN}}{\mathrm{FN+TP}} \tag{5.4}$$

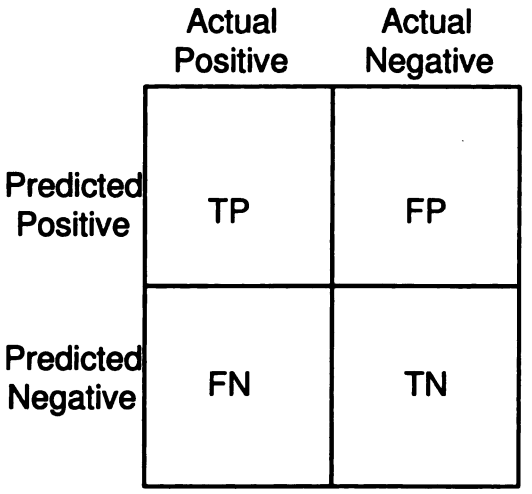

Figure 5.3: Confusion Matrix

FAR is defined as the ratio of the total number of false acceptances to the total number of imposter accesses and FRR is defined as the ratio of the number of false rejections and the total number of client accesses.

Lienhart [86] classifies shot detection performance in terms of the hit rate, miss rate, and false hits. The hit rate h is the ratio of correctly detected shots to the actual number of shots. This measure is the same as recall. The miss rate is defined as $1 - h$. False hits are defined as the ratio of falsely detected shot boundaries to the actual number of shots.

Some researchers have also attempted to compare competing algorithms in terms of a single statistic: accuracy. In terms of the confusion matrix (see Figure 5.3), accuracy is defined as:

$$Accuracy = \frac{\text{TN+TP}}{\text{TN+TP+FN+FP}} \tag{5.5}$$

Accuracy is the measurement of correctly classified instances. This measure has been shown to be insufficient to evaluate the performance of different algorithms [130]. For example, if a data set consists of 95 TNs and 5 TPs, classifying all 100 instances into the negative class (95 TNs and 5 FNs) would achieve a 95% accuracy measure. However, the ability of the system to predict the positive class would be 0%. Although the accuracy measure is an apparently high 95%, the classifier

cannot discriminate between the two classes. This is particularly a problem in video shot detection, where the vast majority of frames boundaries do not represent shot boundaries. As an example, *Blade 2* has 197,549 frame boundaries and 2,300 shot boundaries (a frame boundary is a pair of frames that represent a transition in time; a shot boundary is a frame boundary where the two frames are from different shots). Failing to detect any cuts in this entire sequence would still yield an accuracy measure of 98.8%.

We have chosen to base our performance in terms of precision and recall values. Theoretically, we strive to achieve perfect precision and recall, however in practice there is a tradeoff. Increasing the precision measure past a certain point usually results in lowering the recall and vice versa. We seek to obtain results that provide a balance, maximizing both precision and recall simultaneously. Additionally, we determine the overall performance of an algorithm with respect to the video corpus in terms of composite precision and recall. Composite precision and recall for the set N of videos in the corpus is defined as:

$$Recall_{Composite} = \frac{\sum_i^N \mathrm{TP}_i}{\sum_i^N (\mathrm{TP}_i + \mathrm{FN}_i)} \tag{5.6}$$

$$Precison_{Composite} = \frac{\sum_i^N \mathrm{TP}_i}{\sum_i^N (\mathrm{TP}_i + \mathrm{FP}_i)} \tag{5.7}$$

In evaluating the performance of any algorithm based on precision and recall statistics, it is important to know what level of information one wants to obtain from the data [10]. If the intent of the video is to be further observed by a human analyst, then a high recall value is important. Humans are good at recognizing errors in the shot boundary detection and can disregard the redundant information relatively quickly. A system that is fully automated places more emphasis on precision. Depending on the goal of the segmentation, a trade-off must be made between precision and recall. It may or may not be acceptable to retrieve a few extra shot boundaries that would otherwise be missed at the expense of retrieving an enormous amount of incorrectly identified shot boundaries [10]. This research only considers the thresholds

that produced results of at least 50% recall.

5.3 Training Data

The division of the training data plays a critical role in determining the performance of supervised learning-based algorithms. However, in the video domain, the ratio of shot to non-shot boundaries does not lend itself well to developing a balanced training set. For example, in Figure 5.1 the Royal Tenenbaums video has a frame-to-shot ratio of 190 : 1 in 91290 frames. It has been shown that balancing the number of classes in the training data leads to a more robust classifier [130]. Additionally, the classifiers generated from equally balanced data sets consider all the discriminating attributes that separate the two classes. Unbalanced data sets can lead to overfitting problems and poor cross validation. The total number of shots in this video sequence totals 479, but sampling only a balanced 479 negative samples in this video does not adequately characterize the negative cases.

Theoretically, in order to train a robust classifier it must be trained on every possible type of input. Utilizing all the available data to train a machine learning system is impractical. In practice, one must attempt to train the classifier with a set of data that can characterize all possible input patterns or at least all the extremas. Initially we trained the various results fusion engines with a balanced data set. The positive examples in the set were chosen from all the manually collected shot boundaries for each video sequence. An algorithmic and a random solution were then utilized to sample the possible negative cases. The results of training with balanced data led to poor classification for all the implemented methods for both the random and algorithmic solutions. As a result, we incorporated more negative examples into the training set. For every shot in the training data, we randomly sampled 30 negatives cases. This approach proved to lead to a more robust classifier. *Leave out one* testing was performed for each video under analysis, where all the training data from the video corpus was used to train each method, except the video that will be used for testing.

Normalization is an important aspect of results fusion [17]. The variety of videos contained in the video corpus exhibit varying characteristics. Moreover, the implemented algorithms exhibit different behaviors at shot boundaries. All of the implemented methods attempt to look for peaks in the inter-frame comparison method distributions. As an example, a simple shot boundary detection method might seek to find peaks in the comparison between the color histograms of two adjacent frames. However, these measures vary depending on the underlying content and one video's peak can be drastically different from another's. As a result, each of the video features was locally normalized with respect to the information contained in its own sequence. Specifically, each video's pair-wise frame difference metric was normalized by the maximum difference metric observed in the entire video sequence, mapping the measures into a common $[0, 1]$ interval.

5.4 Filtering

Initial tests of the implemented algorithms showed that many shot transitions resulted in high false positive measurements. One of the reasons for this occurrence is that during a shot transition that occurs over a series of frames (gradual transitions), the threshold would be crossed several times. Our research does not focus on detecting gradual transitions. In order to reliably and accurately detect gradual transitions, we would have to characterize every type of gradual effect for training and testing. That is beyond the scope of this work. Researchers have experimented with various smoothing methods to reduce the effects of gradual shot transitions [28, 86]. Zabith, et al. [152] and Lienhart [86] used a gliding mean value to smooth the results of the edge change ratio method. Any dissimilarity metric value greater than a predetermined threshold was smoothed, any other dissimilarity metric value was set to zero.

Our research uses a filtering strategy based on the research of Dailianas, et al. [28]. This algorithm works by processing the sequence of dissimilarity metrics between successive frames and computing a new sequence that is analyzed for shot transitions. Given a candidate frame, the previous and next k frames in the video sequence are

analyzed. If any of the $2k$ dissimilarity metrics is greater than the candidate frame's dissimilarity metric, the candidates dissimilarity metric is replaced with the last local minimum detected in the sequence. From experimentation k was set to 6 in this research.

5.5 Existing Method Results

In order to properly assess the performance of the proposed results-fusion segmentation algorithms, it was necessary to compare to baseline implementations of common existing algorithms including both single method and fusion approaches. We have implemented and analyzed several existing methods as described in this section. These methods include the four basic single method approaches (global histograms, local histograms, DCT, and edge change ratio) with single thresholds and adaptive thresholds, as well as the existing majority voting method and Boolean logic method for results fusion.

5.5.1 Single Threshold

It is a difficult task to determine a single best threshold to use for a diverse set of video sequences. Theoretically, utilizing an optimal threshold would be ideal for each video sequence; however this is functionally not possible, as it would require an oracle to supply the threshold value. There is no single ideal threshold that can be used for a variety of video classes. Moreover, not only do different classes of video have different characteristics, but different videos within the same classes do not always exhibit the same properties. A threshold that may be optimal for one video sequence will probably not be optimal for another. As a result, a compromise must be made in determining a single threshold. This compromise sacrifices performance and reliability for each video sequence under analysis. In our testing, to determine a single threshold, we averaged the best performing thresholds in terms of precision and recall values for each individual algorithm. Figure 5.4, Figure 5.5, and Figure 5.6 illustrate the *precision vs. recall* graphs of the global histogram, local histogram, and

DCT coefficient algorithms respectively. The *precision vs. recall* graph was not shown for the edge change ratio because this method maps all the ratios into the interval $[0, 1]$. All the peaks in the algorithm denote cuts, and all the other edge change ratios are close to zero. Deciding on a single threshold involves maximizing the precision and recall. Figure 5.9 displays the single thresholds used for each algorithm based on the precision and recall curves. As a result of using a single threshold for a wide variety of content, the single threshold method for each individual algorithm shows a mixed bag of results. Figure 5.7 shows the graph of precision and recall values for the single threshold implementation.

An analysis of the precision and recall table in Figure 5.7 shows that the music video *RKelly A* performs poorly for all the implemented algorithms. This is a direct result of the numerous editing effects used throughout the video that could not be removed by the pre-filtering technique. In this research, gradual transitions are not detected, therefore videos having numerous editing effects and gradual transitions can lead to erroneous results. Figure 5.10 shows some frames from the *RKelly A* music video sequence. Additionally, the *Blade2 A* and *Blade2 B* videos exhibit poor performance as well. The *Blade2 A* and *Blade2 B* videos consists of dark scenes and manually edited lighting effects. The global and local histogram methods achieved the best performance using a single threshold in terms of maximizing the precision and recall. The highlighted cell values in the table show the method that produced the best performance in terms of increasing precision and recall. These results are consistent with what other researchers have concluded when using histogram-based algorithms along with other methods [10, 95, 156].

5.5.2 Adaptive Threshold

OṪoole et al. [109] concluded that fixed thresholds are inadequate to deal with a variety of different types of video content. Additionally, researchers have reported receiving better performance using adaptive threshholding methods [15, 100, 150]. As a result, we have implemented 3 adaptive shot boundary techniques based on the research of Yusoff, et al [150]. The three adaptive shot detection methods are the

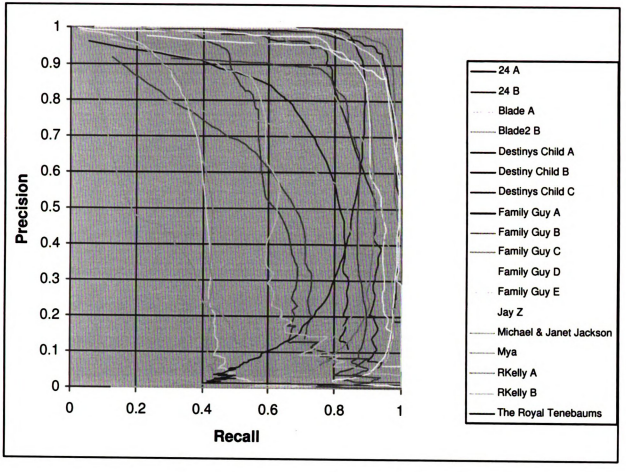

Figure 5.4: Color Histogram *Precision vs. Recall* Graph

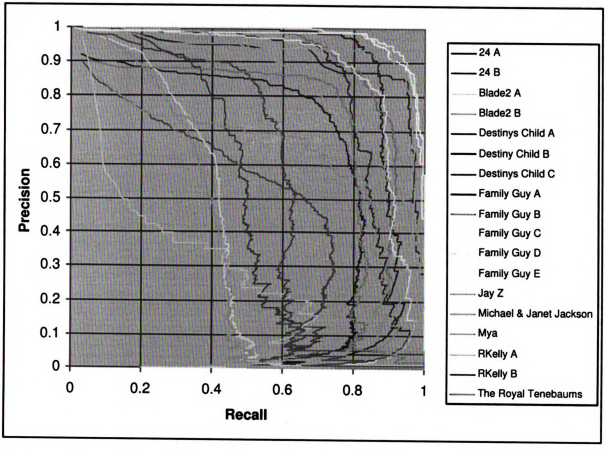

Figure 5.5: Local Histogram *Precision vs. Recall* Graph

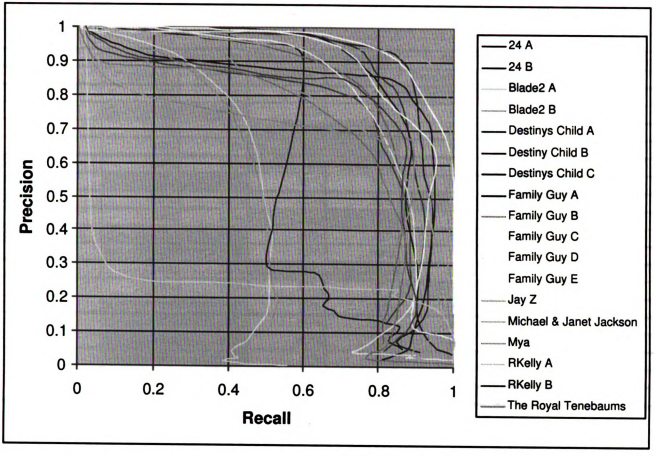

Figure 5.6: DCT *Precision vs. Recall* Graph

	Static Global Histogram		Static Local Histogram		Static DCT		Static Edge Change Ratio	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
24 A	72%	73%	65%	80%	94%	63%	27%	63%
24 B	64%	55%	53%	57%	87%	45%	18%	63%
Blade 2 A	37%	72%	39%	64%	51%	26%	40%	51%
Blade 2 B	38%	70%	35%	65%	50%	28%	35%	54%
Destinys Child A	84%	79%	77%	88%	90%	73%	82%	68%
Destinys Child B	95%	86%	95%	82%	66%	22%	89%	44%
Destinys Child C	56%	63%	48%	56%	77%	93%	57%	88%
Jay Z	99%	65%	95%	86%	86%	91%	88%	75%
Michael & Janet Jackson	81%	80%	78%	76%	80%	66%	75%	76%
Mya	85%	54%	75%	59%	71%	11%	76%	61%
R Kelly A	34%	36%	43%	33%	84%	21%	41%	27%
R Kelly B	59%	76%	55%	77%	54%	85%	5%	79%
The Family Guy A	87%	93%	76%	98%	100%	4%	64%	78%
The Family Guy B	83%	94%	68%	96%	82%	8%	57%	70%
The Family Guy C	94%	96%	83%	98%	90%	18%	79%	78%
The Family Guy D	88%	89%	71%	95%	69%	3%	61%	74%
The Family Guy E	92%	91%	84%	97%	100%	11%	62%	78%
The Royal Tenenbaums	86%	48%	82%	43%	100%	1%	51%	31%

Figure 5.7: Single Threshold Precision and Recall Table

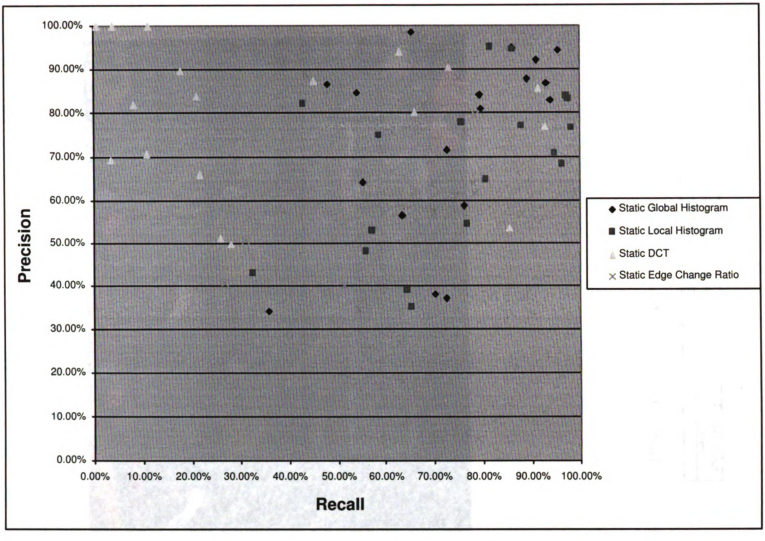

Figure 5.8: Single Threshold Precision Recall Graph

	Color Histogram	Local Histogram	DCT	Edge Change Ratio
Threshold	1938.59	7600.11	129.61	1

Figure 5.9: Single Thresholds used for each method

Figure 5.10: *RKelly A* Music Video Sequence

constant variance model, proportional variance model, and the Dugad model [38]. The constant variance model sets a threshold at some fixed distance away from the average of the metric values within a sliding window. The constant variance model can be expressed as:

$$m_T = \mu_N + T_c \tag{5.8}$$

where the value T_c defines a constant, and μ_N reflects the average of the samples within the window N.

The proportional variance model sets a local threshold at some multiple of the average of the frames within the sliding window. The proportional variance model can be expressed as:

$$m_T = \mu_N T_p \tag{5.9}$$

where the value of T_p is determined from experimentation.

The Dugad model sets a local threshold at some multiple of the standard deviation and the average of the samples within the window. The Dugad model can be expressed as:

$$m_T = \mu_N + T_d\sqrt{\sigma_N} \tag{5.10}$$

where T_d is calculated from experimentation.

Various window sizes and parameters were used to determine the best performing dynamic method based on the total error rate. The total error rate is defined as:

$$Error = \frac{\text{FP} + \text{FN}}{\text{FP+FN+TP+TN}} \tag{5.11}$$

The global histogram method produced the best results using the proportional variance model with 11 as the window size and 6 as the constant. The local histogram method produced the best results using the proportional variance model with 21 as the window size and 6.5 as the constant. The DCT method produced the best results with the Dugad model with 11 as the window size and 2.5 as the constant. The Edge Change Ratio method produced the best results with the proportional variance model with 29 as the window size and 2.5 as the constant.

Experiments with adaptive thresholds indicate that adaptive thresholds improve algorithm performance over static thresholding methods. These results are consistent

	Global Histogram		Local Histogram		DCT		Edge Change Ratio	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
24 A	82%	88%	74%	89%	87%	84%	71%	62%
24 B	68%	84%	57%	83%	95%	78%	64%	45%
Blade 2 A	71%	66%	74%	67%	81%	47%	62%	23%
Blade 2 B	49%	76%	52%	81%	53%	55%	62%	39%
Destinys Child A	91%	83%	90%	80%	92%	74%	92%	58%
Destinys Child B	93%	68%	94%	77%	80%	15%	86%	26%
Destinys Child C	86%	74%	87%	76%	96%	64%	85%	31%
Jay Z	95%	82%	92%	86%	98%	81%	92%	70%
Michael & Janet Jackson	82%	85%	90%	84%	86%	71%	85%	57%
Mya	89%	69%	59%	74%	94%	66%	70%	45%
R Kelly A	55%	29%	53%	34%	100%	20%	64%	28%
R Kelly B	83%	66%	81%	69%	83%	55%	87%	28%
The Family Guy A	59%	91%	46%	94%	55%	88%	20%	74%
The Family Guy B	52%	90%	45%	95%	64%	85%	26%	74%
The Family Guy C	60%	86%	47%	97%	59%	87%	25%	89%
The Family Guy D	47%	88%	38%	95%	54%	88%	22%	75%
The Family Guy E	51%	89%	42%	96%	61%	87%	27%	79%
The Royal Tenenbaums	83%	82%	75%	90%	71%	50%	24%	74%

Figure 5.11: Adaptive Threshold *Precision vs. Recall* Table

with other shot segmentation research using adaptive thresholds [15, 38, 109, 150]. Figure 5.11 and Figure 5.12 display the *precision vs. recall* table and graph for the adaptive threshold implementation. The highlighted cell values in the table show the method that produced the best performance in terms of increasing precision and recall. From a comparison of Figure 5.8 and Figure 5.12 one can see that the overall precision and recall statistics for all the implemented methods has increased due to the use of dynamic thresholds.

The dynamic local histogram method produced the best results in terms of maximizing the precision and recall values for all the dynamic methods. It produced the maximum precision and recall values for 8 of 18 videos. As a result, this algorithm implementation will be used to baseline test the fusion-based methods.

5.5.3 Majority Voting Method

The majority voting method of Yusoff, et al. [151] was implemented with adaptive and static thresholds (see Section 4.5.2). Figure 5.13 displays the *precision vs. recall* table of this method. The highlighted cell values in the table show the method that produced the best performance in terms of increasing precision and recall.

The method yielded the intuitively expected results. The algorithm produces a high precision for low recall values. If a majority of the algorithms determine that a cut exists, there is a high probability that a cut does exist. As a result, the majority voting method produces relatively high precision values for each algorithm. One major problem with this implementation is that since all the algorithms are given equal weight to determine the outcome, unreliable algorithms can lead to numerous missed detections.

Yusoff, et al. [151] claims to receive good performance using this algorithm; however this implementation was only tested on two video sequences. Additionally, the descriptions and characteristics of the two video sequences that were used in their study were not given. This thesis provides a more thorough analysis of this implementation. Figure 5.14 shows a comparison table of the voting methods and the adaptive local histogram method. Additionally, the composite performance of the

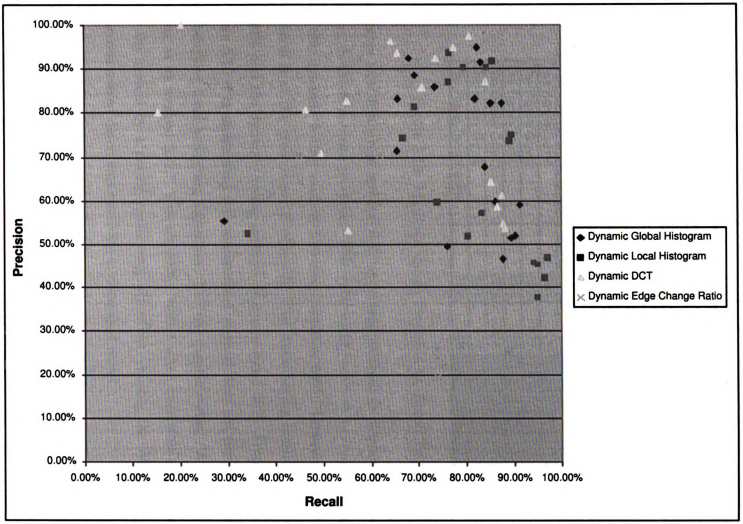

Figure 5.12: Adaptive Threshold *Precision vs. Recall* Graph

	Voting Method Static		Voting Method Dynamic	
	Precision	Recall	Precision	Recall
24 A	90%	63%	96%	80%
24 B	77%	41%	97%	72%
Blade 2 A	48%	27%	96%	30%
Blade 2 B	47%	39%	67%	47%
Destinys Child A	95%	73%	95%	68%
Destinys Child B	94%	41%	92%	9%
Destinys Child C	75%	71%	96%	54%
Jay Z	99%	80%	100%	79%
Michael & Janet Jackson	91%	70%	98%	68%
Mya	84%	42%	95%	63%
R Kelly A	85%	24%	96%	21%
R Kelly B	71%	78%	99%	50%
The Family Guy A	94%	72%	92%	82%
The Family Guy B	92%	67%	93%	84%
The Family Guy C	97%	45%	93%	72%
The Family Guy D	88%	32%	86%	81%
The Family Guy E	99%	47%	92%	79%
The Royal Tenenbaums	84%	18%	90%	58%

Figure 5.13: Majority Voting *Precision vs. Recall* Table

adaptive local histogram method and the voting methods is displayed. Overall, the voting methods increase precision. The static voting method increases precision 21% and the dynamic voting method increases precision 50%. However, this increased precision is at the great expense of recall. The static voting method reduces recall by 44% and the dynamic voting method decreases recall 30%. From the analysis of Figure 5.15 it is shown that this combination method does not perform better than the adaptive local histogram algorithm. The old maxim states, "A chain is only as strong as its weakest link." This algorithm suffers when unreliable estimates are fed into the system by a poorly performing algorithm, causing the system to produce numerous missed detections resulting in decreased recall.

5.5.4 Boolean Logic

The Boolean logic combination algorithm of Browne, et al. [15] has been implemented using static and dynamic thresholds (see Section 4.5.1). Figure 5.16 displays the *precision vs. recall* data table of the Boolean logic static and dynamic method. The

	Dynamic Local Histogram		Voting Method Static		Voting Method Dynamic	
	Precision	Recall	Precision	Recall	Precision	Recall
24 A	74%	89%	90%	63%	96%	80%
24 B	57%	83%	77%	41%	97%	72%
Blade 2 A	74%	67%	48%	27%	96%	30%
Blade 2 B	52%	81%	47%	39%	67%	47%
Destinys Child A	90%	80%	95%	73%	95%	68%
Destinys Child B	94%	77%	94%	41%	92%	9%
Destinys Child C	87%	76%	75%	71%	96%	54%
Jay Z	92%	86%	99%	80%	100%	79%
Michael & Janet Jackson	90%	84%	91%	70%	98%	68%
Mya	59%	74%	84%	42%	95%	63%
R Kelly A	53%	34%	85%	24%	96%	21%
R Kelly B	81%	69%	71%	78%	99%	50%
The Family Guy A	46%	94%	94%	72%	92%	82%
The Family Guy B	45%	95%	92%	67%	93%	84%
The Family Guy C	47%	97%	97%	45%	93%	72%
The Family Guy D	38%	95%	88%	32%	86%	81%
The Family Guy E	42%	96%	99%	47%	92%	79%
The Royal Tenenbaums	75%	90%	84%	18%	90%	58%
Composite	60%	81%	73%	45%	91%	57%
Comparison			21%	-44%	50%	-30%

Figure 5.14: Majority Voting vs. Adaptive Local Histogram *Precision vs. Recall* Table

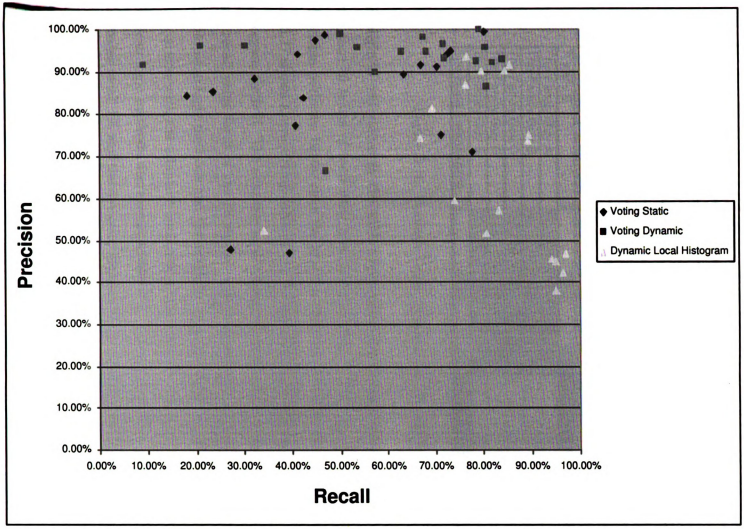

Figure 5.15: Majority Voting *Precision vs. Recall* Graph

	Boolean Logic Static		Boolean Logic Dynamic	
	Precision	Recall	Precision	Recall
24 A	67%	82%	77%	88%
24 B	58%	64%	63%	86%
Blade 2 A	36%	78%	65%	72%
Blade 2 B	37%	80%	45%	82%
Destinys Child A	84%	88%	87%	85%
Destinys Child B	94%	87%	91%	80%
Destinys Child C	59%	80%	85%	78%
Jay Z	97%	85%	95%	84%
Michael & Janet Jackson	77%	87%	80%	90%
Mya	77%	62%	74%	71%
R Kelly A	34%	37%	46%	33%
R Kelly B	58%	85%	80%	66%
The Family Guy A	84%	99%	38%	93%
The Family Guy B	74%	96%	35%	92%
The Family Guy C	90%	97%	37%	90%
The Family Guy D	81%	95%	31%	93%
The Family Guy E	88%	97%	35%	93%
The Royal Tenenbaums	85%	58%	76%	89%

Figure 5.16: Boolean Logic *Precision vs. Recall* Table

highlighted cell values in the table show the method that produced the best performance in terms of increasing precision and recall. The dynamic method of this algorithm performs better than the static method on 11 of the 18 videos tested. Figure 5.17 shows a comparison table of the Boolean logic method and the adaptive local histogram method. The adaptive local histogram method produces a higher precision and recall value for 8 of the 18 videos. Additionally, the composite performance of the Boolean logic methods and the adaptive local histogram method is displayed. In comparison with the adaptive local histogram method, both the adaptive and static Boolean logic methods decrease precision 9%. The Boolean logic static method also decreases recall 1% and the dynamic Boolean logic method increases performance 1%. From the analysis of Figure 5.18 it is shown that this combination method does not perform much better than the adaptive local histogram algorithm.

The majority voting and Boolean logic methods did not significantly improve performance over the adaptive local histogram implementation. When the type of video is known in advance, these algorithms can be tuned to achieve good performance

	Dynamic Local Histogram		Boolean Logic Static		Boolean Logic Dynamic	
	Precision	Recall	Precision	Recall	Precision	Recall
24 A	74%	89%	67%	82%	77%	88%
24 B	57%	83%	58%	64%	63%	86%
Blade 2 A	74%	67%	36%	78%	65%	72%
Blade 2 B	52%	81%	37%	80%	45%	82%
Destinys Child A	90%	80%	84%	88%	87%	85%
Destinys Child B	94%	77%	94%	87%	91%	80%
Destinys Child C	87%	76%	59%	80%	85%	78%
Jay Z	92%	86%	97%	85%	95%	84%
Michael & Janet Jackson	90%	84%	77%	87%	80%	90%
Mya	59%	74%	77%	62%	74%	71%
R Kelly A	53%	34%	34%	37%	46%	33%
R Kelly B	81%	69%	58%	85%	80%	66%
The Family Guy A	46%	94%	84%	99%	38%	93%
The Family Guy B	45%	95%	74%	96%	35%	92%
The Family Guy C	47%	97%	90%	97%	37%	90%
The Family Guy D	38%	95%	81%	95%	31%	93%
The Family Guy E	42%	96%	88%	97%	35%	93%
The Royal Tenenbaums	75%	90%	85%	58%	76%	89%
Composite	60%	81%	55%	80%	55%	82%
Comparison			-9%	-1%	-9%	1%

Figure 5.17: Boolean Logic vs. Adaptive Local Histogram *Precision vs. Recall* Table

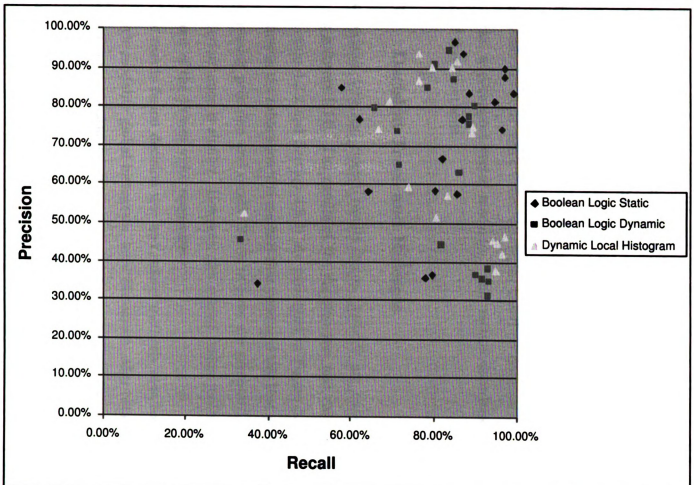

Figure 5.18: Boolean Logic *Precision vs. Recall* Graph

on a limited video class. The majority voting algorithms suffer from the fact that each method is given equal weight in making shot boundary decisions. Perhaps a weighted voting strategy would be more appropriate. The Boolean logic method is an ad hoc attempt at combining results. The color histogram method is given the most weight in all decisions to make shot boundaries. When the color histogram method produces unreliable results, the algorithm will produce poor performance.

5.6 Results Fusion Engine Results

This thesis develops result fusion strategies based on decision trees, rulesets, neural networks, and support vector machines. This section describes the performance of each of these methods.

5.6.1 Decision Trees and Rulesets

The decision tree and ruleset results fusion methods are based on the C5.0 data mining software [114]. The results of the decision tree and ruleset implementations (see Section 4.2.2 and Section 4.2.3) are presented in Figure 5.19. The highlighted cell values in the table show the method that produced the best performance in terms of increasing precision and recall. Since rulesets are generated from decision trees, their performance is similar. However, the ruleset fusion method performed better on 11 out of the 18 test video sequences.

The results fusion decision tree and ruleset methods were baseline tested against the single modality global histogram, local histogram, edge ratio, and DCT decision tree and ruleset methods. The single modality decision tree and ruleset methods produced the exact same results. Figure 5.20 shows a table of the precision and recall values of the results fusion decision tree, results fusion ruleset, and single modality methods. The results fusion methods outperform the single modality methods in 13 out of the 18 test video sequences.

Figure 5.21 shows a comparison table of the decision tree and ruleset methods and the adaptive local histogram method. The decision tree and ruleset methods

	Decision Tree		RuleSet	
	Precision	Recall	Precision	Recall
24 A	70%	92%	72%	91%
24 B	61%	83%	68%	83%
Blade 2 A	58%	83%	55%	86%
Blade 2 B	43%	94%	45%	94%
Destinys Child A	79%	95%	81%	95%
Destinys Child B	86%	96%	86%	96%
Destinys Child C	44%	97%	49%	99%
Jay Z	84%	93%	87%	93%
Michael & Janet Jackson	83%	96%	83%	95%
Mya	68%	74%	73%	73%
R Kelly A	28%	57%	31%	54%
R Kelly B	55%	85%	57%	85%
The Family Guy A	80%	100%	85%	100%
The Family Guy B	68%	99%	68%	99%
The Family Guy C	82%	99%	84%	99%
The Family Guy D	69%	100%	67%	99%
The Family Guy E	81%	100%	81%	100%
The Royal Tenenbaums	74%	91%	74%	91%

Figure 5.19: Decision Tree and Ruleset *Precision vs. Recall* Table

produce the highest precision and recall values for 14 of the 18 test video sequences. Additionally, the composite performance of the decision tree, ruleset, and adaptive local histogram methods are illustrated. The decision tree method increased precision 2% and recall 11% when compared with the adaptive local histogram method. The ruleset method increased precision 4% and recall 11% when compared with the adaptive local histogram method. From the analysis of Figure 5.22 it is shown that these combination methods do perform better than the adaptive local histogram algorithm.

5.6.2 Feed-Forward Neural Network

The Matlab Neural Network Toolbox was used to design a results fusion shot segmentation algorithm based on a feed-forward neural network. The network architecture consisted of 10 neurons in the hidden layer and 1 neuron in the output layer. The hidden layer uses the log sigmoid transfer function and the output layer uses a linear transfer function. The Levenberg-Marquardt algorithm [54] was used as the training

	Decision Tree		RuleSet		DT Color Histogram		DT Local Histogram		DT DCT		DT Edge	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
24 A	70%	92%	72%	91%	59%	91%	66%	86%	80%	87%	24%	52%
24 B	61%	83%	68%	83%	57%	79%	52%	72%	74%	86%	10%	41%
Blade 2 A	58%	83%	55%	86%	48%	89%	39%	64%	54%	58%	7%	9%
Blade 2 B	43%	94%	45%	94%	41%	92%	32%	68%	35%	73%	17%	75%
Destinys Child A	79%	95%	81%	95%	79%	94%	78%	93%	81%	89%	70%	64%
Destinys Child B	86%	96%	86%	96%	87%	96%	63%	63%	46%	43%	40%	33%
Destinys Child C	44%	97%	49%	99%	52%	97%	47%	81%	26%	89%	43%	77%
Jay Z	84%	93%	87%	93%	89%	90%	86%	92%	38%	94%	83%	68%
Michael & Janet Jackson	83%	96%	83%	95%	83%	95%	76%	84%	73%	70%	64%	80%
Mya	68%	74%	73%	73%	65%	75%	66%	70%	91%	58%	55%	65%
R Kelly A	28%	57%	31%	54%	27%	59%	26%	40%	28%	28%	29%	15%
R Kelly B	55%	85%	57%	85%	60%	85%	55%	76%	14%	79%	4%	71%
The Family Guy A	80%	100%	85%	100%	79%	100%	67%	92%	84%	73%	48%	80%
The Family Guy B	68%	99%	68%	99%	64%	99%	59%	99%	88%	81%	47%	75%
The Family Guy C	82%	99%	84%	99%	72%	99%	63%	90%	74%	78%	24%	33%
The Family Guy D	69%	100%	67%	99%	65%	99%	58%	93%	73%	83%	19%	31%
The Family Guy E	81%	100%	81%	100%	78%	100%	59%	94%	84%	87%	26%	44%
The Royal Tenenbaums	74%	91%	74%	91%	62%	91%	60%	76%	60%	75%	45%	65%

Figure 5.20: Decision Tree, Ruleset, and Single Modality Decision Tree/Ruleset Table

	Dynamic Local Histogram		Decision Tree		RuleSet	
	Precision	Recall	Precision	Recall	Precision	Recall
24 A	74%	89%	70%	92%	72%	91%
24 B	57%	83%	61%	83%	68%	83%
Blade 2 A	74%	67%	58%	83%	55%	86%
Blade 2 B	52%	81%	43%	94%	45%	94%
Destinys Child A	90%	80%	79%	95%	81%	95%
Destinys Child B	94%	77%	86%	96%	86%	96%
Destinys Child C	87%	76%	44%	97%	49%	99%
Jay Z	92%	86%	84%	93%	87%	93%
Michael & Janet Jackson	90%	84%	83%	96%	83%	95%
Mya	59%	74%	68%	74%	73%	73%
R Kelly A	53%	34%	28%	57%	31%	54%
R Kelly B	81%	69%	55%	85%	57%	85%
The Family Guy A	46%	94%	80%	100%	85%	100%
The Family Guy B	45%	95%	68%	99%	68%	99%
The Family Guy C	47%	97%	82%	99%	84%	99%
The Family Guy D	38%	95%	69%	100%	67%	99%
The Family Guy E	42%	96%	81%	100%	81%	100%
The Royal Tenenbaums	75%	90%	74%	91%	74%	91%
Composite	60%	81%	62%	90%	63%	91%
Comparison			2%	11%	4%	12%

Figure 5.21: Decision Tree, Ruleset, and Adaptive Local Histogram Table

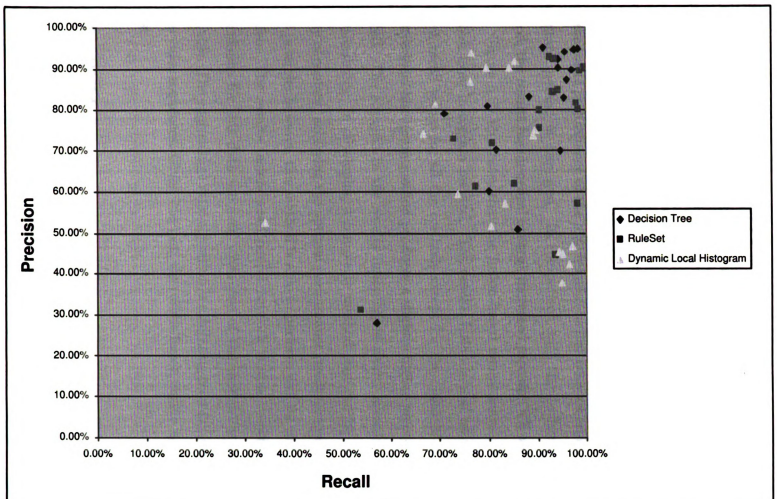

Figure 5.22: Decision Tree and Ruleset vs. Adaptive Local Histogram *Precision vs. Recall* Graph

function. The results fusion neural network was baseline tested against the single modality global histogram, local histogram, edge ratio, and DCT neural network-based methods. Figure 5.23 shows a table of the precision and recall values of the results fusion neural network and single modality methods. The results fusion method outperforms the single modality methods in 13 out of the 18 test video sequences.

Figure 5.24 shows a comparison table of the fusion neural network and the adaptive local histogram method. The results fusion method outperforms the adaptive local histogram threshold method in 13 out of the 18 test video sequences. Additionally, the composite performance of the fusion-based neural network and the adaptive local histogram method is displayed. In comparison with the adaptive local histogram method, the results fusion neural network increased precision 4% and increased recall 12%. Figure 5.25 illustrates that the fusion-based neural network outperforms the adaptive local histogram method. The results fusion neural network produces the highest recall values for all the methods except for the video *24 A*. In general, the adaptive local histogram method produces a slightly higher precision at the expense of recall. The strength in the results fusion-based neural network method is in its generalizability. In developing a composite method that can be utilized for a wide variety of content, achieving the optimal performance result for every individual video is not possible; however, this method is able to perform well for many different types of video. The results fusion-based neural network is able to achieve near optimal performance for this data set even when one of the modalities used for fusion is unreliable. The edge change ratio algorithm is exhibiting poor performance for almost all the video sequences. Given this fact, the results fusion engine is still able to remain robust.

5.6.3 SVM

This thesis has implemented a results fusion method based on the support vector machine (see Section 4.2.1). The SVM results fusion implementation is based on the libsvm software [20]. We used the C-SVM implementation of SVM [137]. As with the other fusion methods, the SVM results fusion method was baseline tested against the

	Neural Network		Color Histogram		Local Histogram		Edge		DCT	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
24 A	71%	92%	63%	91%	64%	86%	23%	56%	80%	87%
24 B	63%	83%	58%	77%	56%	69%	11%	41%	74%	86%
Blade 2 A	55%	88%	49%	88%	41%	62%	7%	11%	53%	61%
Blade 2 B	45%	93%	43%	91%	35%	66%	17%	75%	40%	71%
Destinys Child A	84%	95%	80%	94%	82%	93%	69%	64%	80%	89%
Destinys Child B	87%	95%	89%	96%	64%	63%	40%	34%	46%	43%
Destinys Child C	54%	98%	53%	97%	50%	79%	42%	77%	24%	89%
Jay Z	90%	92%	89%	89%	88%	91%	83%	68%	36%	94%
Michael & Janet Jackson	84%	94%	84%	94%	80%	82%	64%	83%	73%	70%
Mya	73%	74%	68%	75%	69%	68%	53%	65%	92%	52%
R Kelly A	28%	54%	28%	59%	29%	39%	28%	15%	27%	28%
R Kelly B	54%	85%	62%	85%	58%	76%	4%	72%	12%	79%
The Family Guy A	87%	100%	80%	100%	64%	92%	47%	80%	84%	79%
The Family Guy B	71%	98%	67%	99%	57%	99%	46%	77%	86%	86%
The Family Guy C	86%	99%	78%	99%	66%	90%	23%	34%	75%	76%
The Family Guy D	74%	100%	68%	99%	63%	92%	18%	32%	76%	79%
The Family Guy E	86%	100%	80%	100%	64%	94%	25%	45%	84%	87%
The Royal Tenenbaums	73%	91%	65%	91%	64%	76%	43%	65%	59%	76%

Figure 5.23: Results Fusion Neural Network and Single Modality Neural Netowrk Table

	Dynamic Local Histogram		Neural Network	
	Precision	Recall	Precision	Recall
24 A	74%	89%	71%	92%
24 B	57%	83%	63%	83%
Blade 2 A	74%	67%	55%	88%
Blade 2 B	52%	81%	45%	93%
Destinys Child A	90%	80%	84%	95%
Destinys Child B	94%	77%	87%	95%
Destinys Child C	87%	76%	54%	98%
Jay Z	92%	86%	90%	92%
Michael & Janet Jackson	90%	84%	84%	94%
Mya	59%	74%	73%	74%
R Kelly A	53%	34%	28%	54%
R Kelly B	81%	69%	54%	85%
The Family Guy A	46%	94%	87%	100%
The Family Guy B	45%	95%	71%	98%
The Family Guy C	47%	97%	86%	99%
The Family Guy D	38%	95%	74%	100%
The Family Guy E	42%	96%	86%	100%
The Royal Tenenbaums	75%	90%	73%	91%
Composite	60%	81%	63%	91%
Comparison			4%	12%

Figure 5.24: Results Fusion Neural Network and Adaptive Local Histogram Table

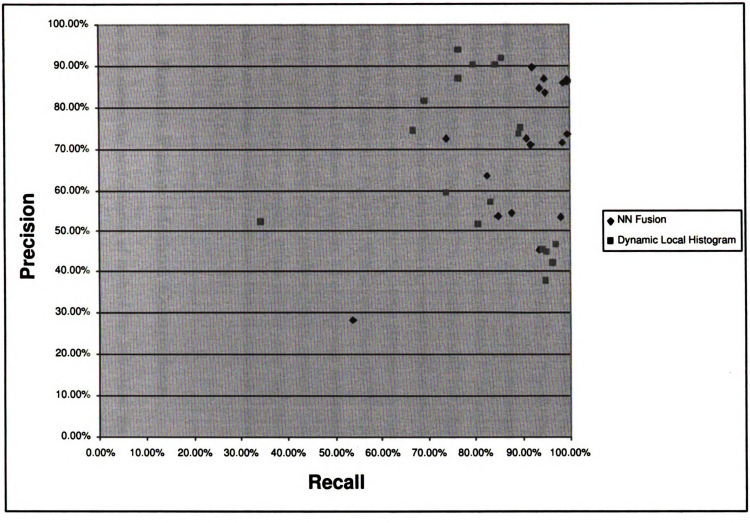

Figure 5.25: Results Fusion Neural Network vs. Adaptive Local Histogram *Precision vs. Recall* Graph

single modality global histogram, local histogram, edge ratio, and DCT SVM-based methods. Figure 5.26 shows the results of the results fusion SVM and single modality methods. The results fusion SVM method performs the best on 10 of 18 test videos sequences. The results of the results fusion SVM method and the single modality SVM methods clearly show that fusion is taking place. As is the case with the other fusion methods, the results fusion SVM method produces high recall values for almost all the video sequences. Although some of the single modality algorithms produce higher precision than the results fusion SVM, it is at the expense of recall.

Figure 5.27 displays a comparison table of precision and recall values of the results fusion SVM and the adaptive local histogram method. The SVM method outperforms the adaptive threshold method in 13 of the 18 video test sequences. Additionally, the composite performance of the results fusion-based SVM method and the adaptive local histogram method is displayed. In comparison with the adaptive local histogram method, the results fusion SVM method increases precision and recall 8%. The results are consistent with how the other results fusion strategies compared to the adaptive local histogram technique. Figure 5.28 illustrates that the SVM results fusion method produces high recall values for almost all of the video sequences. Although some of the adaptive local histogram tests produce higher precision than the results fusion SVM, it is at the expense of recall. These results show that the SVM provides more generality and better overall performance than the adaptive local histogram and the single modality SVM methods.

5.6.4 Results Fusion Method Comparison

Figure 5.29 shows the performance of each of the individual results fusion methods with the video test suite. The highlighted cells in the table indicate the method that performed the best for a specific video based on maximizing the precision and recall. The SVM results fusion engine performed the best on the most video sequences. It performed the best on 8 out of the 18 video sequences.

Each fusion method compared similarly against the adaptive local histogram method. Each method was able to significantly increase recall on most of the video

	SVM		Color histogram		Local histogram		DCT		Edge	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
24 A	79%	89%	85%	66%	80%	81%	83%	87%	25%	47%
24 B	73%	79%	81%	48%	71%	57%	77%	84%	15%	32%
Blade 2 A	57%	78%	61%	54%	52%	50%	53%	61%	6%	8%
Blade 2 B	46%	89%	53%	57%	41%	54%	40%	70%	25%	66%
Destinys Child A	81%	95%	88%	61%	87%	85%	82%	70%	72%	62%
Destinys Child B	87%	93%	96%	85%	69%	62%	48%	43%	43%	31%
Destinys Child C	58%	96%	68%	74%	62%	64%	25%	89%	45%	76%
Jay Z	90%	93%	96%	56%	94%	84%	40%	94%	84%	67%
Michael & Janet Jackson	82%	89%	91%	76%	85%	73%	74%	69%	68%	76%
Mya	82%	69%	82%	61%	78%	64%	92%	49%	64%	62%
R Kelly A	32%	50%	43%	43%	42%	34%	28%	28%	30%	15%
R Kelly B	36%	85%	77%	71%	67%	70%	13%	79%	4%	69%
The Family Guy A	86%	100%	95%	85%	79%	90%	84%	76%	53%	74%
The Family Guy B	78%	98%	92%	79%	75%	98%	88%	84%	53%	71%
The Family Guy C	83%	98%	97%	95%	82%	90%	76%	74%	27%	31%
The Family Guy D	80%	98%	93%	86%	77%	89%	76%	76%	22%	28%
The Family Guy E	86%	99%	96%	86%	84%	93%	86%	85%	30%	41%
The Royal Tenenbaums	79%	91%	87%	83%	75%	70%	64%	74%	50%	53%

Figure 5.26: Results Fusion SVM and Single Modality SVM Table

	Dynamic Local Histogram		SVM	
	Precision	Recall	Precision	Recall
24 A	74%	89%	79%	89%
24 B	57%	83%	73%	79%
Blade 2 A	74%	67%	57%	78%
Blade 2 B	52%	81%	46%	89%
Destinys Child A	90%	80%	81%	95%
Destinys Child B	94%	77%	87%	93%
Destinys Child C	87%	76%	58%	96%
Jay Z	92%	86%	90%	93%
Michael & Janet Jackson	90%	84%	82%	89%
Mya	59%	74%	82%	69%
R Kelly A	53%	34%	32%	50%
R Kelly B	81%	69%	36%	85%
The Family Guy A	46%	94%	86%	100%
The Family Guy B	45%	95%	78%	98%
The Family Guy C	47%	97%	83%	98%
The Family Guy D	38%	95%	80%	98%
The Family Guy E	42%	96%	86%	99%
The Royal Tenenbaums	75%	90%	79%	91%
Composite	60%	81%	65%	87%
Comparison			8%	8%

Figure 5.27: Results Fusion SVM and Adaptive Local Histogram Table

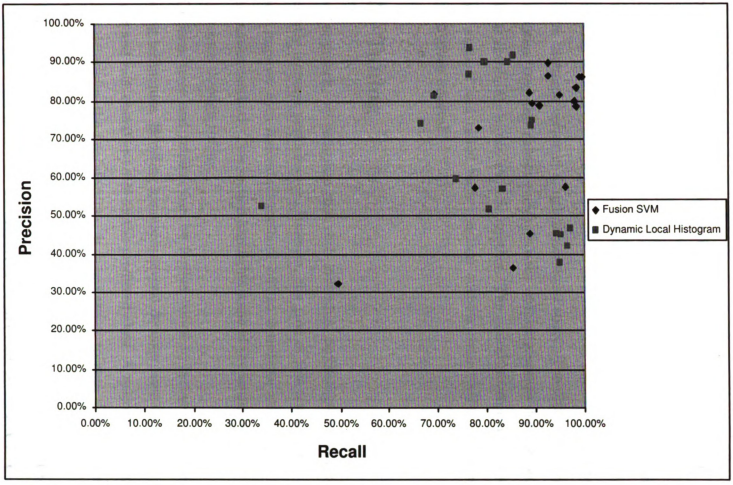

Figure 5.28: Results Fusion SVM vs. Adaptive Local Histogram *Precision vs. Recall* Graph

sequences. Although in some cases, the baseline testing algorithms may produce higher precision values, it is usually at the expense of low recall. Also, the results fusion strategies did not always show the best precision and recall values for every video sequence. However, results fusion did produced the best overall composite performance.

Figure 5.30 displays a graph of the *precison vs. recall* values of the results fusion strategies. It is important to note that most of the methods perform similarly on the same video sequences. This phenomenon is a result of each method attempting to partition the same vector in a high dimensional space. The differences arise because the decision tree and ruleset methods both output absolute decision values. Any vector is absolutely in one class or another. The neural network and SVM methods output sigmoid and radial basis function outputs respectively. These values vary within a given interval.

5.7 Tuning

In our approach, we recognize that each individual method could be further tuned to slightly increase performance. However, the goal of this research was not to optimally improve each individual method, but to determine an improved composite segmentation method. This section highlights some of the issues related to increasing performance through tuning.

Decision trees and rulesets can be further tuned to increase performance using boosting techniques. Boosting generates several decision trees or rulesets for a given dataset and each classifier votes on the predicted class. A majority voting method is used to determine the final decision. It is important to note that boosting does not always increase discriminatory results.

The feed-forward neural network design was not carefully optimized for comparison with the other methods. The reported results were solely intended as proof of concept rather than to show optimal performance. In order to fully experiment and test with neural networks one could use more complex network architectures using

	Decision Tree		RuleSet		Neural Network		SVM	
	Precision	Recall	Precision	Recall	Precision	Recall	Precision	Recall
24 A	70%	92%	72%	91%	71%	92%	79%	89%
24 B	61%	83%	68%	83%	63%	83%	73%	79%
Blade 2 A	58%	83%	55%	86%	55%	88%	57%	78%
Blade 2 B	43%	94%	45%	94%	45%	93%	46%	89%
Destinys Child A	79%	95%	81%	95%	84%	95%	81%	95%
Destinys Child B	86%	96%	86%	96%	87%	95%	87%	93%
Destinys Child C	44%	97%	49%	99%	54%	98%	58%	96%
Jay Z	84%	93%	87%	93%	90%	92%	90%	93%
Michael & Janet Jackson	83%	96%	83%	95%	84%	94%	82%	89%
Mya	68%	74%	73%	73%	73%	74%	82%	69%
R Kelly A	28%	57%	31%	54%	28%	54%	32%	50%
R Kelly B	55%	85%	57%	85%	54%	85%	36%	85%
The Family Guy A	80%	100%	85%	100%	87%	100%	86%	100%
The Family Guy B	68%	99%	68%	99%	71%	98%	78%	98%
The Family Guy C	82%	99%	84%	99%	86%	99%	83%	98%
The Family Guy D	69%	100%	67%	99%	74%	100%	80%	98%
The Family Guy E	81%	100%	81%	100%	86%	100%	86%	99%
The Royal Tenenbaums	74%	91%	74%	91%	73%	91%	79%	91%

Figure 5.29: Results Fusion Methods Table

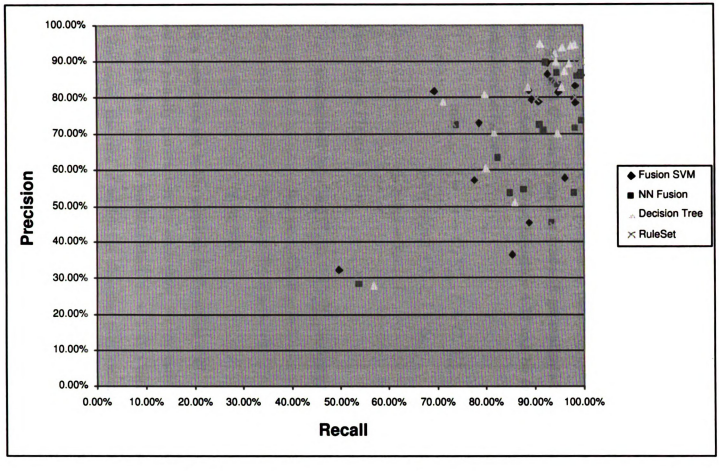

Figure 5.30: Results Fusion Methods *Precision vs. Recall* Graph

several hidden layers and shared weights. Additionally, the training procedure could be optimized by using different transfer functions and training algorithms. Regularization and early stopping methods could be employed for improving network generalization. Weight decaying and pruning could also be used to deal with overtraining the neural network [120]. Our observation of the test error and the learning epochs allowed us to ascertain that the networks were not overtrained. Also, we noticed that decreasing the number of nodes in the hidden layer decreased performance.

SVMs can also be tuned for the purpose of handling unbalanced data sets. Two parameters, C_+ and C_- can be used to trade-off generalization ability and misclassification error for the data set. Thus, Equation 4.37 can be extended to include the tuning parameters as follows:

$$\begin{aligned} \min_{\mathbf{w},b,\xi} \quad & \tfrac{1}{2}\mathbf{w}\cdot\mathbf{w} + C_+ \textstyle\sum_{i=1}^{l} \xi_i + C_- \sum_{i=-1}^{l} \xi_i \\ & y_i(\mathbf{w}\cdot\phi(\mathbf{x}_i) + b) \geq 1 - \xi_i, \\ & \xi_i \geq 0, i = 1, \ldots, l. \end{aligned} \tag{5.12}$$

The dual form of Figure 5.12 is:

$$\begin{aligned} \text{maximize } W(\alpha) \quad & = \textstyle\sum_{i=1}^{M} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{M} \alpha_i\alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ & 0 \leq \alpha_i \leq C_+, \text{if } y_i = 1 \\ & 0 \leq \alpha_i \leq C_-, \text{if } y_i = -1 \\ & \mathbf{y}\cdot\alpha = 0. \end{aligned} \tag{5.13}$$

Classifying a new data set is the same as in Equation 4.27. The ratio of C_+ to C_- determines the penalty parameters for each class. As the ratio of C_+ to C_- increases, the rate at which the classifier predicts class $x_i = 1$ also increases. As the ratio of C_+ to C_- decreases, the rate at which the classifier predicts class $x_i = -1$ increases. Drish [36] uses the $F1$ value to determine the best possible C_+ value for a fixed C_-. The $F1$ value is used to determine the maximum balance between precision and recall. The $F1$ statistic is defined as:

$$F1 = 2/((1/\text{precision}) + (1/\text{recall})) \tag{5.14}$$

Additionally different kernels and parameters could be used to train and test the

network. This research did not attempt to find the optimal performance for each dataset.

5.8 Conclusion

This research approached the problem of shot based segmentation as a binary classification problem. As a result, we utilized well known strategies from the information retrieval and biometric community and adapted them to the video domain. Decision Trees, rulesets, neural networks, and support vector machines were all used to show that results fusion-based shot segmentation could improve shot detection performance. Results fusion was applied on the measurement and feature extraction level because these levels offer the most flexibility in determining the results fusion strategies to employ. In this research, key low-level image features were used to provide input to the results fusion methods. Global histograms, local histograms, DCT coefficients, and edge features were extracted from the video frames. These features were chosen because they have historically been known to accurately predict shot boundaries in certain conditions. Additionally, their features complement each other's strengths and weaknesses. The distance metric delta δ between successive frame pairs was used as input to the results fusion methods. *Leave out one* testing was performed on a video corpus of over 8 hours in length from a wide variety of sources. The results of this research shows that results fusion can be realized in the video domain to improve performance. We have developed over 24 baseline testing methods to assess performance of our results fusion algorithms. Within this baseline set two known combination strategies in the video community, majority voting and Boolean logic were implemented and both strategies did not improve performance over using a single method. The results in this research also show the ability of our results fusion engine to detect shot boundaries when receiving unreliable data. The edge change ratio algorithm performed poorly for almost all the methods; however its inclusion in our results fusion strategy did not reduce performance.

In comparing the results fusion strategies to one another the SVM method per-

formed the best on 8 out of 18 videos. This result is not surprising. Research in the information retrieval and biometric communities have reported receiving favorable performance using this classification method [149, 9].

The results fusion approach in this thesis can be extended to include other features or attributes, such as text and speech processing metrics. The power of using SVM is that its computational complexity is not dependent on how many attributes are used, but how many data points are in the training set.

Another key aspect of this research is performance. Supervised learning algorithms take time to learn patterns in the dataset. Decision Trees and Rulesets are fast at developing rules from the training data. No test took longer than a few seconds to train the classifier on the training data and evaluate the test data. The neural network took about 10 minutes to train the videos on the training data and after the model was trained all tests ran in seconds. The SVM results fusion strategies took the longest to train. SVM training time depends on the type of kernel that is used and how many data points are in the training data. We used the RBF kernel and training took about 15 minutes per video. Again, once the model was created the tests ran in seconds. The importance of a generalizable shot-detection method is that training need only be performed once when the system is built using a large training corpus. Henceforth, the system will only be used in the classification mode, which is very fast for all of the results-fusion methods presented.

The promising results contained in our experiments show that results fusion strategies from the sensor fusion, biometric, and information retrieval communities can be adapted to the video domain.

Chapter 6

Extensions to Summarization

One of the critical tools in any indexing and browsing environment is effective summarization. Figure 6.1 illustrates the basic components for a typical digital video analysis system. The video to be indexed must be presented to an indexing system with a minimum of redundancy to avoid redundant retrieval results and to maximize the disparity in the indexing space. Likewise, search results must be presented to human users as compact summaries that allow users to quickly browse through the candidate choices and choose the correct result or adapt the search as quickly as possible. After shot segmentation, summarization attempts to eliminate or limit the redundancy in the video while preserving the most important aspects of the video.

This process leads to developing a pictorial summary of an underlying video sequence that represents the original video in a more compact form. This summary usually consists of utilizing a smaller set of images to represent the visual content, and presenting these keyframes to the user. Most summarization research focuses on keyframe extraction. Section 2.3 described numerous methods for summarization and keyframe extraction. Researchers have achieved great progress in developing summarization techniques for digital video. However, it is common for much of this research to focus on specific classes of video or limited content corpuses. Major projects have analyzed news broadcasts, television programs, and commercials where the video structure is known in advance. There has been limited research on developing composite techniques that can be used on a wide variety of video classes. One of the

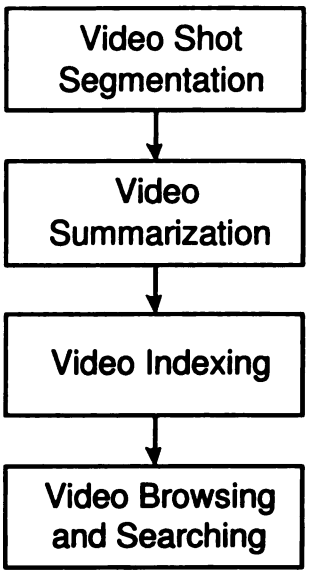

Figure 6.1: Video System Components

strengths of any summarization method should be its generalizability, or its ability to adapt to the video under analysis.

Another key aspect of summarization is performance evaluation. Evaluation is a difficult task because there is no standard way to determine the performance of any summarization method. The video summarization community has agreed that any summarization method should seek to eliminate redundancy and present the video in a compact form that allows for maximum user retention and comprehension. Additionally, it should briefly and concisely present the contents of the original video [118]. However, these criteria are subjective and user-dependent. Gong and Liu [51] have defined summarization performance in terms of a redundancy metric. The performance of the summarization is characterized by the amount of redundant information produced in the summary. This method provides a novel first approach in developing a performance evaluation criterion for summarization however; it only focuses on the outputs of the summarization and not on attempting to determine the important information in the video sequence.

Our results fusion research for shot segmentation shows that improved performance can be achieved by utilizing results fusion, utilizing multiple classifiers or representations to improve performance and reliability over using a single classifier

or representation. Chapter 4 describes our method for shot segmentation based on results fusion and Chapter 5 illustrates the performance of the various implemented results fusion strategies. Our suggested method for video summarization is an extension of the results fusion based approaches used for shot segmentation. As with the results fusion segmentation methods, this solution can be generalized to work for a wide variety of content without the input of manually collected structural knowledge.

6.1 Unstructured Video

Commercial video sequences are characterized as having manual shot transitions and editing effects. Some of these effects include straight cuts, fades, dissolves, wipes, and dissolves. Most segmentation and summarization methods rely on these manually edited effects to cue their algorithms. These methods make assumptions about the structure of the video and exploit them in their classification. However, raw and unstructured video sequences have few, if any, manual edit effects. These videos are characterized as having long continuous sequences with no structure or meta-data to facilitate access [65].

Fundamentally, the input to any summarization algorithm is unstructured video. As illustrated in Figure 6.1, summarization follows segmentation and is assumed to be working on continuous shots, be they short intervals between rapid cuts or long sequences of raw camera footage. A third operation, *abstraction* is often applied to the output of summarization to further group content across shot boundaries so as to further reduce redundancy. This chapter, however, is concerned only with summarization.

Access to this type of video has been usually facilitated by linear navigation through the entire video sequence. This type of access is suitable for viewing, however it is cumbersome for retrieval or for quick browsing of the content. Beyond the cuts that make up commercial video content, this type of video also includes documentary films, home video, surveillance video, and unmanned aerial vehicle (UAV) video. Figure 6.2 shows a typical UAV video sequence. Summarizing this type of

Figure 6.2: UAV Video Sequence

video involves developing structure where no structure is present in the underlying video sequence. Once structure has been determined, it can be processed via a myriad of summarization techniques. Our novel results fusion summarization strategy incorporates structured and unstructured video.

6.2 Features

As shot segmentation depends on the extraction of key features from the video sequence to determine possible shot boundaries, summarization also involves extracting important features from within each shot. After a video has been segmented into shots, it is assumed that the each shot is characterized by similar content. However, temporally long and complex shots have important information that must be captured. Additionally, unstructured video can be thought of as one long continuous shot, and the important information contained in these sequences must be captured.

It has been shown that when developing video shot segmentation strategies that utilize multiple features, improved results can be obtained when features are chosen that supplement one another [53]. The same criteria can be applied to video summarization. In a situation in which extracting one feature may be difficult or unreliable, extracting another feature may be more appropriate. For example, in

the UAV video sequence shown in Figure 6.2 no color information is available. Any summarization technique that relies on the color content of the video sequence will fail. In this situation it would be important for the summarization method to utilize another feature.

Our results fusion summarization strategy uses a global histogram module, local histogram module, and motion module to select relevant keyframes. The result of each individual feature extraction module is used to form a feature vector, and the new feature vector is fed to the results fusion engine to make a final decision as illustrated in Figure 6.3. Global and local histograms were used in our results fusion shot segmentation method. The importance, strengths, and weaknesses of these features were described in Section 4.3. Where global and local histograms are good at analyzing color sequences in video for summarization and keyframe extraction, they fail in the presence of camera motion. Global and local histograms can be used to detect features for summarization when the camera is not moving or has moved very little. New subjects walking into the camera field of view or an explosion in an image sequence are examples of situations in which these methods could uncover important features in the video. Camera motion analysis can be used to extract features when the camera is constantly moving. Figure 6.2 shows a typical UAV video sequence.

6.3 Camera Motion

Motion is a key aspect in most video shots [14]. Motion in video sequences can be either characterized as originating from the camera or objects in the shot. Many researchers have experimented with characterizing motion in video sequences [12, 13, 37, 68, 71, 2, 142, 144, 156]. Camera motion can be characterized as panning, tilting, zooming, booming, dollying, or tracking [16]. Camera panning refers to the camera rotating around its vertical axis. Tilting refers to the camera rotating about its horizontal axis. Zooming refers to a stationary camera adjusting its focal length either to concentrate on a specific area of interest or to get an overall view of a scene. Booming refers to the camera moving up and down as if it were on a physical crane.

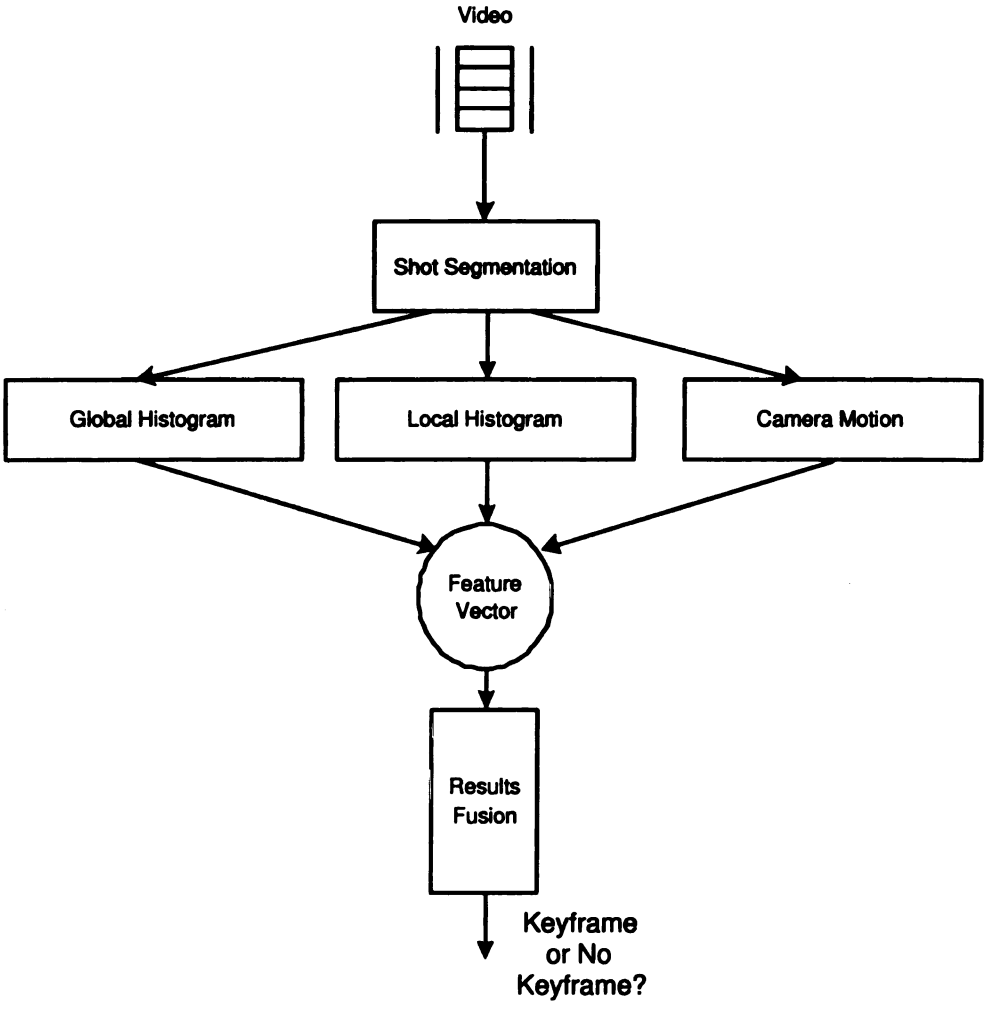

Figure 6.3: Results Fusion for Video Summarization

Dollying can be characterized as a camera moving in and out of a scene, with the movement parallel to the camera lens axis. Tracking refers to the camera moving perpendicular to the camera lens axis. Each one of these motion types allows one to gain information as to the important aspects of the contents of a shot. Often, camera motion reflects the intentions of the director which allows one to gain some semantic understanding of the video sequence.

Local motion within an image sequence can be described using *motion vectors*. A motion vector is an indication of where in a frame the content from the frame came from in the previous (or earlier) image. As an example, a video sequence consisting of a stationary camera photographing a train moving to the right would have motion vectors at each pixel or block that indicate that the content at that location came from the left in the previous image (the content moved to the right, so it came from the left). Each one of the standard camera motions exhibit specific patterns in the motion vector fields between successive frames in a video sequence. Figure 6.4 illustrates the motion vectors between successive frames in a panning and tilting video sequence. The majority of the vectors indicate content came from to the left and down, so it is moving up and to the right. This indicates the camera is moving to the left and down. Some motion vectors in this image are missing because there was no underlying content to perform analysis on (moving fixed colors are not distinguishable). Other vectors seem to move in unexpected directions. This is due to errors in the motion analysis process. Motion analysis is a complex topic and subject to local errors.

In order to detect camera motion operations, motion vectors can be extracted from frame sequences by various techniques. Many of the techniques rely on the edge information of frames to locate the best motion vectors in combination with either block tracking or optical flow-based algorithms.

6.4 Motion-based keyframe extraction

We have developed a novel based video summarization method based on camera motion. This method is not sufficient for video summarization alone because it is not

Figure 6.4: Motion Vectors from a Panning Video Sequence

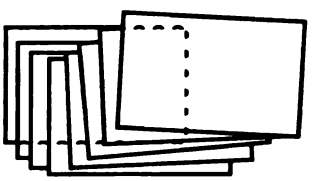

Figure 6.5: Camera motion over a scene

effective in modeling content changes that do not involve camera motion. However, the principles and features extracted from this method are utilized in our approach to results fusion for video summarization.

Our novel motion-based keyframe extraction technique can accurately select keyframes with significant global motion between frames in raw unstructured video sequences. Imagine a camera moving over a fixed scene. The camera will take a path somewhat like that illustrated in Figure 6.5. As the camera moves, the new image overlaps the previous images by varying amounts. Given a starting reference frame (keyframe), $1 - overlap$ represents the amount of de-occlusion; the amount of new content uncovered by the camera motion. All computations are done on adjacent pairs of frames. The previous frame and current frame are compared using a block matching algorithm commonly used for MPEG motion vector computation. However,

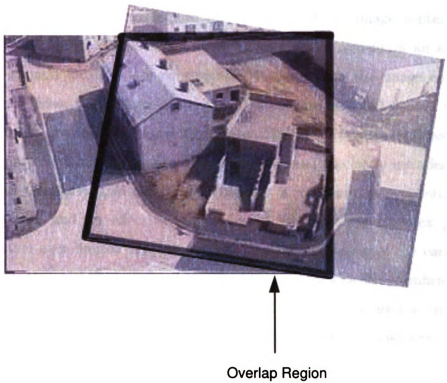

Figure 6.6: Possible keyframe overlap.

motion vectors are not selected arbitrarily. A set of motion vectors is selected based on presence of edge detail as indicated by a statistical edge detection algorithm.

The result of the block matching algorithm is a set of displacements and their associated block centers in the current frame. This computation is based on searching the previous image for matching blocks. The system attempts to deduce where content in the previous frame moved. These displacements are converted to (x, y) to (u, v) correspondences and used to compute a least-square estimate of the affine transformation from the previous image to the current image. The affine transformation is then used to warp all recent keyframe locations to correspond with the current motion. The warped keyframes are tested for percentage overlap with the current frame. When the overlap drops below a preset threshold, a new keyframe is selected. Figure 6.6 illustrates the overlap of two possible keyframes. In order to characterize the background motion we implemented a global two-dimensional parametric model. The goal of this model is to approximate the motion of the

camera platform relative to the scene. Several models are possible. An exact reproduction of reality would require 3D modeling of the image contents, which is typically not practical and is not likely to be of great benefit in an application where the camera location is a considerable distance from the image content. In unstructured video, the background typically approximates a 2D image. Given the choice of using a two-dimensional transformation, the available options included affine and projective modeling. The typical camera motions clearly includes rotation, scaling, and translation, so an affine solution is a minimum parameterization for this application and an effective approximation of the more complex projective transformation over limited time sequences. The large distance of the camera from the image content and the long focal length of the camera lens also significantly limit the value of a projective model. The affine model is commonly used to characterize motion in video sequences [12, 13, 37, 68, 71, 142]. The affine model was considered because of its resilience to noisy and sparse motion vector conditions [71]. The affine model is expressed as:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a_2 & a_3 \\ a_5 & a_6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} a_1 \\ a_4 \end{bmatrix} \tag{6.1}$$

It is assumed that the affine model is characterizing the underlying motion flow of the background from one image to the next (or within a sequence of images). In this equation, $[x, y]^T$ denotes pixels in the previous frame and $[u, v]^T$ represent position pixels in the current frame. The motion parameters $(a_1, a_2, a_3, a_4, a_5, a_6)$ can be estimated by a linear least square estimation. For a set of motion displacements, Equation 6.1 can be expressed as:

$$\begin{bmatrix} u_1 & u_2 & \dots & u_n \\ v_1 & v_2 & \dots & v_n \end{bmatrix} \approx \begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \end{bmatrix} \begin{bmatrix} 1 & 1 & \dots & 1 \\ x_1 & x_2 & .. & x_n \\ y_1 & y_2 & .. & y_n \end{bmatrix} \tag{6.2}$$

Equation 6.1 is the homogenous representation of the transformation of a set of points. For a large set of points, the equation is over determined and cannot be

directly solved. Instead, we seek a solution that will minimize the least squares solution to the over determined system of linear equations. The least-square problem can be expressed as:

$$\min_x ||b - AX|| \tag{6.3}$$

Given two input image frames, a robust statistical based edge detection algorithm adapted from Kundu [77] is used to compute the edge maps of both frames. Motion vectors are computed from the edge maps based on a block correspondence algorithm proposed by Kundu [76]. A block correspondence algorithm was chosen for this application in order to decrease the computation requirements relative to methods such as optical flow and because only sections of the image with significant edge detail will be examined for correspondences. Block correspondence is clearly not a good choice when a significant rotation component is evident. However, for most video the rotation rate is physically constrained so that the method remains effective.

The Kundu method computes motion estimation by decomposing the pixels within each block into three categories, quasi-constant regions, dominant edge regions, and textured regions. The goal is to match blocks of similar textures. Motion vectors are obtained by minimizing a cost function measuring the mismatch between a block and each predictor candidate. This method utilizes a cost function based on perceptual factors rather than simple squared-error differences. The cost function consists of two elements $D = D_1 + D_2$. The two components of a cost function are a contrast measure of the difference image, D_1, and a measure of the edge alignment, D_2 [1]. D_1 is defined as:

$$D_1 = \sum_i 0.416 \log(|1 + d_i|) \tag{6.4}$$

d_i is the residual when the two blocks are subtracted. The summation is over the pixel values in the block. This method accounts for the logarithmic nature of the human visual system. The value has been normalized using the constant 0.416 so that the measurement for a single pixel will range from 0 to 1, assuming a pixel range

[1] Kundu describes D_2 as a "brightness" function and incorporates texture alignment into the equation. Our solution does not utilize texel alignment due to the wide variety of possible incorrect alignments.

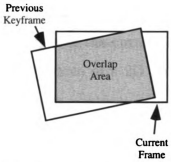

Figure 6.7: Example of polygon overlap

of 255.

The second component of the function D_2 is set to 1 when the pixels of an edge do not align between the two images and 0 otherwise. This classification penalizes the alignment of a detected edge with a location that is not an edge.

In order to increase efficiency and performance, motion vectors are not computed for each possible block between images, instead they are only for a finite set of locations where edge content is evident. Figure 6.4 is an example set of computed motion vectors and illustrates the grouping of vectors where complex content is located.

Based on the computed motion vectors, the affine parameters are estimated using linear least-square decomposition. The previous keyframe image corner coordinates are continuously warped using the estimated affine parameters. It is not necessary to actually warp the previous keyframe or even keep it around. All that is necessary is a warping of the polygon that the keyframe represents. As an example, if a keyframe is selected from a 352 by 240 image sequence, the coordinates of the keyframe corners are $(0, 240)$, $(352, 240)$, $(352, 0)$, $(0, 0)$, assuming a counter-clockwise presentation order. If the affine alignment to the next frame is computed as $(a_1, a_2, a_3, a_4, a_5, a_6)$=$(-14.05,\ 1.04,\ 0.056,\ 1.74,\ -0.0093,\ 0.965)$, the keyframe corners would be warped to $(-0.758, 233.2)$, $(366.9, 229.9)$, $(353.6, -1.54)$, $(-14.1, 1.74)$. The overlap of the keyframe and the current image is simply the overlap of the two polygons. Figure 6.7 illustrates the overlap of a previous keyframe and the current frame. A pair-wise comparison of the warped keyframe image and the current candidate frame based on Sutherland-Hodgeman polygon clipping

is used to determine if a new keyframe is chosen. The Southerland-Hodgeman algorithm clips against the four edges of the current candidate frame in succession. Equation 6.5 is the equation for the area of a 2D polygon after clipping, assuming $N-1$ vertices and $(x_1, y_1) = (x_N, y_N)$.

$$A = \frac{1}{2}\sum_{i=1}^{N} x_i y_{i+1} - x_{i+1} y_i \tag{6.5}$$

If the amount of overlap is less that a supplied threshold, the current candidate frame is chosen as the next keyframe. Figure 6.8 depicts the interface to the motion-based keyframe extraction algorithm. The computed overlap amount of this motion-based keyframe extraction algorithm is an important feature that is used in our results fusion approach for video summarization.

6.5 Results Fusion

Section 4.1 detailed the various levels of results fusion. From our analysis, it was concluded that the best levels to achieve optimal shot segmentation performance was at the *measurement and feature extraction levels.* As a result, we extend our ideas gleaned from the shot segmentation analysis to video summarization and attempt to achieve maximum video summarization performance at these levels. The abstract and output levels provide little information with respect to the decision making process. With only the output decisions or the output frames of the various summarization methods available, there is only a limited amount of fusion strategies that can be employed. Fusion at the measurement and feature extraction levels provide the most information available to make decisions regarding a best summarization. These levels attempt fusing algorithm thresholds, scores, and confidence measures and offer a wide variety of combination methods.

Our extension of results fusion for video summarization operates at the feature extraction level. Feature extraction level fusion concatenates features extracted from multiple classifiers to form a new synthesized higher-dimensional feature vector. The

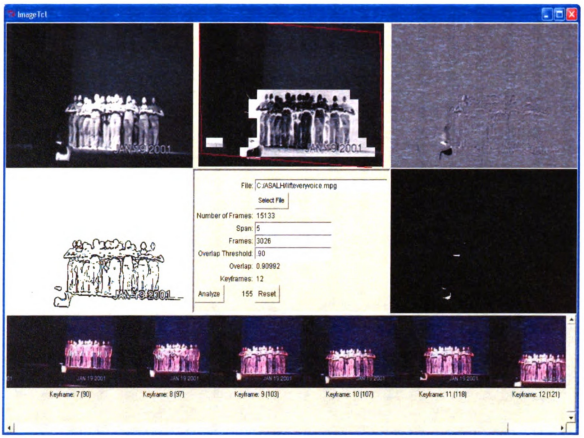

Figure 6.8: Motion-based keyframe extraction interface

premise behind feature extraction level fusion is that the new synthesized vector will be more discriminating than each single modality feature. The features extracted for fusion are global histograms, local histograms, and camera motion or sequence overlaps and are used to create the new higher-dimensional feature vector. These features are not chosen arbitrarily, but are based on their overall performance exhibited by a history of research in the video shot summarization community. This feature vector is then submitted to the results fusion engine for classification. The results fusion engine could be comprised of decision trees, neural networks, or support vector machines to make the final decision. In Section 5 each of these fusion methods was shown to significantly increase performance over using a unimodal method.

The fusion-based shot segmentation algorithms fuse together the pair-wise frame comparisons of global histograms, local histograms, the edge change ratio, and DCT coefficients. Video summarization does not utilize successive frame comparisons, but comparisons between the current frame under analysis and the previous extracted keyframe. Once a keyframe has been extracted, the new keyframe is compared to subsequent frames until another keyframe has been selected or until the end of the shot has been reached. To utilize results fusion for video summarization the global histogram, local histogram, and camera motion or sequence overlap comparison values between the current frame and the previous keyframe are fed to the results fusion engine. When a new keyframe has been chosen based on the characteristics of the features, the new keyframe will be used to make further comparisons. This process is continued until the end of the shot and is done for all shots in the video sequence. The video data used to train the results fusion-based summarization algorithms would consists of numerous sample shot sequences from a wide variety of video classes.

The proposed approach to results fusion for video summarization can be applied with a variety of features. The approach is highly extensible and can incorporate features from other areas, such as text and speech processing to improve fusion results.

6.6 Summary

This chapter demonstrated how results fusion methodologies could be applied to video summarization. In Chapter 5 we showed that improved performance could be obtained by combining the strengths of multiple modalities using feature level results fusion on a wide variety of video classes. Just as the important features for video segmentation were fused and classified via a results fusion engine, the important features used for video summarization can be classified and fused in the same manner. Global histograms, local histograms, and camera motion are all important features that can capture information contained in a video shot sequences of various classes of video. One of the strengths of any summarization method should be its generalizability, or its ability to adapt to the video under analysis. When these modalities are used alone, in certain situations their results can be unreliable. However, a results fusion implementation can create a more reliable and robust technique.

Chapter 7

Summary

This thesis contributes to the overall research in digital video in seven distinct ways. The contributions in this thesis are:

- A Decision Tree and Ruleset based results fusion engine for shot segmentation and summarization that improves performance over using a single algorithm.
- Neural Network results fusion for shot segmentation and summarization that improves performance over using a single algorithm.
- Support Vector Machine-based results fusion for shot and video summarization that fuses key features from video and determines a best segmentation with extensions to summarization that improves performance over using a single algorithm.
- Feature extraction and analysis for results fusion.
- Experimental validation on a large and varied video test suite.
- Adaptability to detect shot boundaries when receiving unreliable data from one or more modalities.
- A novel keyframe-based video summarization technique based on camera motion.

The strength of the results fusion-based methods are their generalizability. Any general solution must work for a wide variety of content without the input of manually collected structural knowledge. When the type of video is known a priori, any category of techniques can be employed with relative success. However, when the type of video to be analyzed is unknown, certain assumptions cannot be made about the video and the best choice of algorithms cannot be predetermined, nor can the appropriate parameterization of the algorithms be made (setting thresholds and intervals for example). Our results fusion-based approaches have shown to be able to generalize well to include new data. The fusion of multiple methods allows these strategies to produce good results in spite of receiving unreliable information from inferior algorithms. This thesis has developed fusion-based shot segmentation algorithms based on decision trees, rulesets, neural networks, and support vector machines that fuse together several key features and perform shot segmentation based on the characteristics of those features. These key features include color, texture, motion, and compressed image features. From experimentation, it was shown that each individual method itself may not provide superb performance, however the fusion of these methods has produced a reliable and quality result. The novel results fusion-based approaches were tested on over twenty-four baseline methods. Moreover, the results from the experimental analysis show that all the fusion-based systems improved performance over the best performing single modality system. When comparing the composite precision and recall values against the best performing single modality algorithm, the results fusion-based strategies performed as follows: The decision tree method increased precision 2% and recall 11%, the ruleset method increased precision 4% and recall 11%, the neural network method increased precision 4% and increased recall 12%, and the SVM method increased precision and recall 8%.

Additionally, our novel results fusion-based methods outperformed two existing combination strategies in the video shot segmentation community, majority voting and Boolean logic [15, 151]. When compared to the best performing single modality algorithm, the majority voting method increases precision 21% and the dynamic voting method increases precision 50%. However, this increased precision is at the

expense of recall. The static voting method reduces recall by 44% and the dynamic voting method decreases recall 30%. Voting methods suffer when unreliable estimates are fed into the system by low performing algorithms, which causes the system to produce numerous missed detections resulting in decreased recall.

When compared to the best performing single modality algorithm, both the adaptive and static Boolean logic methods decrease precision 9%. The Boolean logic static method also decreases recall 1% and the dynamic Boolean logic method increases performance 1%. In this algorithm, the color histogram is always given extra weight in the final shot boundary decision. When the color histogram algorithm is unreliable, it produces errors throughout the entire system. The results fusion-based methods developed in this thesis have been shown to overcome errors due to unreliable algorithms.

The results fusion method is extendable and can be applied to video summarization. After the video is segmented, the extracted shots can be analyzed to eliminate redundancy and to extract important information from those shots. In Chapter 5 we showed that improved performance could be obtained by combining the strengths of multiple modalities using feature level results fusion on a wide variety of video classes. Just as the important features for video segmentation were fused and classified via a results fusion engine, the important features used for video summarization can be classified and fused in the same manner. Key features such as global histograms, local histograms, and camera and object motion can be fused to increase summarization performance and accuracy. Additionally, this thesis has presented a novel keyframe extraction algorithm based on camera motion. Motion is a key aspect in most video shots [14]. Often, camera motion reflects the intentions of the director, which allows one to gain some semantic understanding of the video sequence. This keyframe extraction algorithm has been used to summarize structured an unstructured video sequences. It can be used with any results fusion based summarization technique to improve summarization reliability and performance.

7.1 Future Work

In this thesis, we primarily focused on image processing techniques to improve video shot detection and summarization. As stated above our approach to results fusion shot segmentation can be extended to include more features from other domains. Text and speech processing techniques have shown to be able to increase query performance and reliability in video retrieval systems [141]. Results fusion could be applied to video retrieval systems and browsing systems to increase performance.

Additionally, confidence measure could be implemented into the results fusion architecture. If one modality was highly confident that a shot boundary exists, its confidence measure could influence the outcome of the final decision. Also, if one modality was confident that a shot boundary did not exist, its confidence measure could influence the decision as well. Bengio, et al. experimented with using confidence measures for multi-modal biometric systems using SVMs. The incorporation of confidence measures into a results fusion shot segmentation method could further increase performance.

APPENDICES

Appendix A

Appendix A

The purpose of this section is to describe the contents of the video corpus in detail. We have created a video corpus of a wide variety of content. The classes of video include motion pictures, TV sitcoms, cartoons, and music videos. The collection consists of over 8 hours of video.

A.1 Television Programs

This class of video consists of 2 one hour episodes of the television program *24*. The two episodes that were used in our evaluation and testing are *24: 12am to 1am* and *24: 1am to 2am*. These videos were labeled *24 A* and *24 B* respectively. Video *24 A* consisted of 76560 total frames with 564 cuts. Video *24 B* consisted of 76140 total frames with 700 cuts. The majority of the shot transitions in this video class are hard cuts. False positive detections usually occur due to gradual transitions, object motions, and lighting effects. Additionally, missed detections occur because of local shot transitions. Local shot transitions occur when a frame is split into separate windows with each window representing a different shot. Figure A.1 illustrates a local shot change effect.

Figure A.1: *24:12am to 1am* local shot effect

A.2 Movies

This class of videos includes the movie *Blade 2* and the first 50 minutes of *The Royal Tenenbaums*. *Blade 2* consisted of 197550 total frames with 2301 cuts. The *Blade 2* video was split into two separate videos, *Blade 2 A* and *Blade 2 B*. This movie can be characterized as having many fast action sequences resulting in quick shot transitions and camera changes. Additionally, this movie had numerous dark scenes that can lead to missed detections. Moreover, this movie contains many manually edited lighting effects that can lead to missed detections. Figure A.2 illustrates a sample lighting effect in the *Blade 2* movie.

The Royal Tenenbaums consists of 91290 total frames with 479 cuts. This video was shot with a 16 : 9 aspect ratio, but was encoded with a 4 : 3 aspect ratio. As a result, all of the shot transitions and changes occur with the center region of each frame. This video has numerous dissolve and fades effects that can lead to missed detections. Figure A.3 illustrates a sample shot sequences from the *The Royal Tenenbaums* movie sequence.

A.3 Cartoons

This class of videos included five episodes of *The Family Guy*. The five episodes of *The Family Guy* that were used are *The Family Guy: Brian Does Hollywood*, *The Family Guy: Da Boom*, *The Family Guy: Fifteen Minutes of Shame*, *The Family Guy: Lethal Weapons*, and *The Family Guy: The Thin White Line*. These videos were labeled *The Family Guy A-E*, respectively. *The Family Guy A* consisted of

Figure A.2: *Blade 2* action sequence

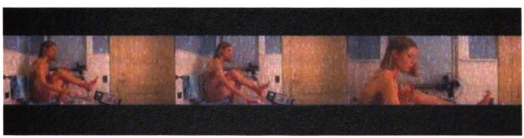

Figure A.3: *The Royal Tenenbaums* shot sequence

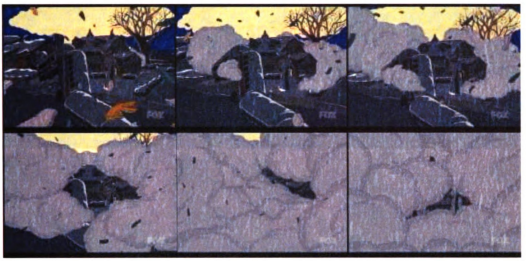

Figure A.4: *The Family Guy: Da Boom* explosion sequence

39270 total frames with 261 cuts. *The Family Guy B* consisted of 38970 total frames with 326 cuts. *The Family Guy C* consisted of 39240 total frames with 246 cuts. *The Family Guy D* consisted of 39270 total frames with 260 cuts. *The Family Guy E* consisted of 38940 total frames with 310 cuts. These videos can be characterized as having numerous manually edited transitions effects, such as dissolves and fades. Additionally, explosions and object motions can lead to false detections. Figure A.4 illustrates and explosion in *The Family Guy: Da Boom* video sequence.

A.4 Music Videos

This class of videos included eight music videos by artists Destinys Child, Jay Z, Mya, Michael and Janet Jackson, and R. Kelly. The Destinys Child videos that was used for testing and evaluation were *Destinys Child: Bills*, *Destinys Child: Bootylicious*, and *Destinys Child: Jumpin.* These videos are labeled *Destinys Child: A-C* respectively. *Destinys Child A* consisted of 7380 total frames with 233 cuts. *Destinys Child B* consisted of 6600 total frames with 239 cuts. *Destinys Child C* consisted of 5940 total frames with 208 cuts. The R Kelly videos that were used are *R Kelly: Feelin* and *R Kelly: If.* These videos are labeled *R Kelly A* and *R Kelly B* respectively. The Jay Z video that was used was *Jay Z: Love You.* This video was labeled *Jay Z* and consisted

Figure A.5: *R Kelly: If* gradual transition sequence

of 7890 total frames with 208 cuts. The Mya video that was used was titled *Mya: Best of Me* and consisted of 7200 total frames with 111 cuts. The Michael and Janet Jackson video evaluated was titled *Michael and Janet Jackson: Scream.* This video consisted of 8670 total frames with 250 cuts. These videos can be characterized as having numerous gradual transitions effects, manually edited effects, and short shot durations. Short shot durations can lead to missed detections when using adaptive methods that analyze frames within specific window sizes. If the window size is larger than the shortest shot length, the shot could be missed. In addition, manually edited effects can lead to false detections. Figure A.5 illustrates a gradual transition effect in the *R Kelly: If* music video sequence.

Bibliography

[1] F. Arman, A. Hsu, and M. Chiu. Image processing on compressed data for large video databases. In *ACM Multimedia*, pages 267–272, 1993.

[2] Serge Ayer and Harpreet S. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL encoding. In *ICCV*, pages 777–, 1995.

[3] A. Bagga, H. Jianying, J. Zhong, and G. Ramesh. Multi-source combined-media video tracking for summarization. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, pages 818–821, 2002.

[4] J. L. Barron, D. J. Fleet, and S. S. Beauchemin. Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77, 1994.

[5] Brian T. Bartell, Garrison W. Cottrell, and Richard K. Belew. Automatic combination of multiple ranked retrieval systems. In *Research and Development in Information Retrieval*, pages 173–181, 1994.

[6] Brian T. Bartell, Garrison W. Cottrell, and Richard K. Belew. Automatic combination of multiple ranked retrieval systems. In *Research and Development in Information Retrieval*, pages 173–181, 1994.

[7] Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. *Machine Learning*, 36(1-2):105–139, 1999.

[8] Mandis Beigi, Ana B. Benitez, and Shih-Fu Chang. Metaseek: A content-based metasearch engine for images. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 118–128, 1998.

[9] S. Ben-Yacoub. Multi-modal data fusion for person authentication using SVM. In *Proc. Second International Conference on Audio and Video-based Biometric Person Authentication*, pages 25–30, 1999.

[10] John S. Boreczky and Lawrence A. Rowe. Comparison of video shot boundary detection techniques. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 170–179, 1996.

[11] J.S. Boreczky and L.D. Wilcox. A hidden markov model framework for video segmentation using audio and image features. In *Proc. Int. Conf. Acoustics, Speech, and Signal Processing*, pages 3741–3744, 1998.

[12] G.D. Borshukov, G. Bozdagi, Y. Altunbasak, and A.M. Tekalp. Motion segmentation by multistage affine classification. *IP*, 6(11):1591–1594, November 1997.

[13] P. Bouthemy, M. Gelgon, and F. Ganansia. A unified approach to shot change detection and camera motion characterization. *CirSysVideo*, 9(7):1030, October 1999.

[14] Patrick Bouthemy, Marc Gelgon, and Fabrice Ganansia. A unified approach to shot change detection and camera motion characterization. Technical Report RR-3304, INRIA, 1997.

[15] P. Browne. Evaluating and combining digital video shot boundary detection algorithms. In *Irish Machine Vision and Image Processing Conference (IMVIP'2000)*, 2000.

[16] R. Brunelli, O. Mich, and C. Modena. A survey on video indexing. Technical Report 9612-06, IRST Technical Report, 1996.

[17] Roberto Brunelli and Daniele Falavigna. Person identification using multiple cues. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(10):955–966, 1995.

[18] E. Bruno and D. Pellerin. Video shot detection based on linear prediction of motion. In *IEEE International Conference on Multimedia and Expo (ICME 2002)*, 2002.

[19] K. Chan, T. Lee, P. Sample, M. Goldbaum, R. Weinreb, and T. Sejnowski. Comparison of machine learning and traditional classifiers in glaucoma diagnosis. *IEEE Trans. Biomed. Engr.*, pages 963–974, 2002.

[20] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[21] Shih-Fu Chang, John R. Smith, Mandis Beigi, and Ana Benitez. Visual information retrieval from large distributed online repositories. *Communications of the ACM*, 40(12):63–71, 1997.

[22] S.B. Cho. Neural-network classifiers for recognizing totally unconstrained handwritten numerals. In *IEEE Transactions on Neural Networks*, pages 43–53, 1997.

[23] M. Christel and A. Hauptmann. Adjustable filmstrips and skims as abstractions for a digital video library. In *IEEE Proc. ADL*, pages 98–104, 1999.

[24] Michael G. Christel, Michael A. Smith, C. Roy Taylor, and David B. Winkler. Evolving video skims into useful multimedia abstractions. In *CHI*, pages 171–178, 1998.

[25] Michael G. Christel, David B. Winkler, and C. Roy Taylor. Multimedia abstractions for a digital video library. In *ACM DL*, pages 21–29, 1997.

[26] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[27] F. Cremer, W. de Jong, and K. Schutte. Feature-level sensor-fusion of polarimetric ir sensor and gpr for landmine detection. In *2nd InternationalWorkshop on Advanced Ground Penetrating Radar (IWAGPR)*, 2003.

[28] A. Dailianas, R. Allen, and P. England. Comparison of automatic video segmentation algorithms. In *SPIE Photonics West*, pages 2–16, 1995.

[29] Daniel DeMenthon, Vikrant Kobla, and David S. Doermann. Video summarization by curve simplification. In *ACM Multimedia*, pages 211–218, 1998.

[30] N. Dimitrova, L. Agnihotri, and G. Wei. Video classification based on hmm using text and faces. In *European Conference on Signal Processing*, 2000.

[31] John Dixon and Charles Owen. Fast client-server video summarization for continuous capture (poster session). In *ACM Multimedia*, 2001.

[32] A. Doulamis, N. Doulamis, and S. Kollias. Efficient video summarization based on a fuzzy video content representation. In *ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, pages 301–304, 2000.

[33] A. Doulamis, N. Doulamis, and K. Ntalianis. An optimal interpolation-based scheme for video summarization. In *Multimedia and Expo, 2002. Proceedings. 2002 IEEE International Conference on*, pages 297–300, 2002.

[34] Daniel Dreilinger and Adele E. Howe. Experiences with selecting search engines using metasearch. *ACM Transactions on Information Systems*, 15(3):195–222, 1997.

[35] Mark Drew, Ze-Nian Li, and Xiang Zhong. Video dissolve and wipe detection via spatio-temporal images of chromatic histogram differences. In *ICIP*, 2000.

[36] J. Drish. Obtaining calibrated probability estimates from support vector machines. Seminar on Learning Algorithms, University of California, San Diego. June 2001 (http://www-cse.ucsd.edu/users/jdrish/papers.html).

[37] F. DuFaux and F. Moscheni. Background mosaicing for low bit rate video coding. In *ICIP96*, page 16P6, 1996.

[38] R. Dugad, K. Ratakonda, and N. Ahuja. Robust video shot change detection. In *IEEE Workshop on Multimedia Signal Processing*, 1998.

[39] S. Dumais, J. Platt, D. Heckman, and M. Sahami. Inductive learning algorithms and representations for text categorization. *Proceedings of the International Conference on Information and Knowledge Management.*, pages 148–155, 1998.

[40] Paul England, Robert B. Allen, Mark Sullivan, Andrew Heybey, Mike Bianchi, and Apostolos Dailianas. I/browse: The bellcore video library toolkit. In *Storage and Retrieval for Image and Video Databases (SPIE)*, pages 254–264, 1996.

[41] D. Farin and W. Effelsberg. Robust clustering-based video-summarization with integration of domain-knowledge. In *Multimedia and Expo, 2002. Proceedings. 2002 IEEE International Conference on*, pages 89–92, 2002.

[42] A. Ferman and A. Tekalp. Muliscale content extraction and representation for video indexing. In *Proc. SPIE Multimedia Storage and Archiving Systems II*, pages 23–31, 1997.

[43] A. M. Ferman and A. Murat Tekalp. Efficient filtering and clustering for temporal video segmentation and visual summarization. *Journal of Visual Communication and Image Representation*, 2419, 1998.

[44] Y. Freud, R. Iyer, R. Schapire, and Y. Singer. An efficient boosting algorithm for combining preferences. In *International Conference on Machine Learning.*, 1998.

[45] N. Friedman, D. Geiger, and M. Goldszmidt. Bayesian network classifiers. In *Machine Learning*, pages 131–163, 1997.

[46] K. Fujimura, K. Honda, and K. Uehara. Automatic video summarization by using color and utterance informatiom. In *Multimedia and Expo, 2002. Proceedings. 2002 IEEE International Conference on*, pages 49–52, 2002.

[47] U. Gargi, R. Kasturi, and S. Antani. Performance characterization and comparison of video indexing algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, 1998.

[48] D. Genoud, F. Bimbot, G. Gravier, and G. Chollet. Combining methods to improve speaker verification decision. In *Proc. ICSLP '96*, volume 3, pages 1756–1759, Philadelphia, PA, 1996.

[49] Eric J. Glover, Steve Lawrence, William P. Birmingham, and C. Lee Giles. Architecture of a metasearch engine that supports user information needs. In *Eighth International Conference on Information and Knowledge Management (CIKM'99)*, pages 210–216, Kansas City, MO, November 1999. ACM Press.

[50] Y. Gong and X. Liu. Video shot segmentation and classification. In *International Conference on Pattern Recognition*, 2000.

[51] Y. Gong and X. Liu. Video summarization with minimal visual content redundancies. In *Image Processing, 2001. Proceedings. 2001 International Conference on*, pages 362–365, 2001.

[52] Luis Gravano, Chen-Chuan K. Chang, Héctor García-Molina, and Andreas Paepcke. STARTS: Stanford proposal for Internet meta-searching. In *Proc. of the 1997 ACM SIGMOD International Conference On Management of Data*, pages 207–218, 1997.

[53] B. Günsel, A. Müfit Ferman, and A. Murat Tekalp. Temporal video segmentation using unsupervised clustering and semantic object tracking. *SPIE Journal of Electronic Imaging*, pages 592–604, 1998.

[54] M.T. Hagan and M.B. Menhaj. Training feedforward networks with the marquardt algorithm. In *IEEE Transactions on Neural Networks*, pages 989–993, 1994.

[55] A. Hanjalic and H. Zhang. An integrated scheme for automated video abstraction based on unsupervised clustervalidity analysis. In *IEEE Trans. Circuits Syst.*, 1999.

[56] Alexander G. Hauptmann and Michael J. Witbrock. Story segmentation and detection of commercials in broadcast news video. In *Advances in Digital Libraries*, pages 168–179, 1998.

[57] D. Heckerman. A tutorial on learning with bayesian networks. Technical report, Microsoft Research, Redmond, Washington, 1995. Revised June 96.

[58] David Heckerman, Dan Geiger, and David Maxwell Chickering. Learning bayesian networks: The combination of knowledge and statistical data. In *KDD Workshop*, pages 85–96, 1994.

[59] T. Ho, J. Hull, and S. Srihari. Decision combination in multiple classifier systems. In *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, pages 66–75, 1994.

[60] Adele E. Howe and Daniel Dreilinger. SAVVYSEARCH: A metasearch engine that learns which search engines to query. *AI Magazine*, 18(2):19–25, 1997.

[61] David A. Hull, Jan O. Pedersen, and Hinrich Schütze. Method combination for document filtering. In Hans-Peter Frei, Donna Harman, Peter Schäuble, and Ross Wilkinson, editors, *Proceedings of SIGIR-96, 19th ACM International Conference on Research and Development in Information Retrieval*, pages 279–288, Zürich, CH, 1996. ACM Press, New York, US.

[62] J.J. Hull, S.N. Srihari, E. Cohen, C.L. Kuan, P. Cullen, and P. Palumbo. A blackboard-based approach to handwritten zip code recognition. In *Proc. US Postal Service Adv. Tech Conf.*, pages 1018–1032, 1988.

[63] M. Irani and P. Anandan. Video indexing based on mosaic representation. In *IEEE Trans. on PAMI*, pages 905–921, 1998.

[64] ISO-IEC. Mpeg standard draft iso-iec/jtc1 sc29, 1991.

[65] Giridharan Ranganathan Iyengar. *Characterization of Unstructured Video.* PhD thesis, MASSACHUSETTS INSTITUTE OF TECHNOLOGY, 1999.

[66] A.K. Jain and R.C. Dubes. *Algorithms for Clustering Data*, page 1988. Prentice Hall, 1988.

[67] Anil K. Jain and Arun Ross. Learning user-specific parameters in a multibiometric system. In *Proc. of International Conference on Image Processing (ICIP)*, September 2002.

[68] H. Jozawa, K. Kamikura, A. Sagata, H. Kotera, and H. Watanabe. Two-stage motion compensation using adaptive global mc and local affine mc. *CirSys Video*, 7(1):75–85, February 1997.

[69] Sung-Bae Jun, Kyoungro Yoon, and Hee-Youn Lee. Dissolve transition detection algorithm using spatio-temporal distribution of MPEG macro-block types (poster session). In *ACM Multimedia*, pages 391–394, 2000.

[70] J. Katzer, M. McGill, J. Tessier, W. Frakes, and P. Dasgupta. A study of the overlap among document representations. In *Information Technology: Research and Development*, 1982.

[71] J.G. Kim, H.S. Chang, J. Kim, and H.M. Kim. Efficient camera motion characterization for mpeg video indexing. In *ICME00*, page TP11, 2000.

[72] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 226–239, 1998.

[73] Anita Komlodi and Gary Marchioini. Keyframe preview techniques for video browsing. In *ACM International Conference on Digital Libraries*, pages 118–125, 1998.

[74] Irena Koprinska and Sergio Carrato. Temporal video segmentation: A survey. citeseer.nj.nec.com/378900.html.

[75] Irena Koprinska and Sergio Carrato. Hybrid rule-based/neural approach for segmentation of mpeg compressed video. In *Multimedia Tools and Applications*, 2002.

[76] A. Kundu. Motion estimation by image content matching and application to video processing. In *ICASSP96*, 1996.

[77] Amlan Kundu. Robust edge detection. *Computer Vision and Pattern Recognition*, pages 11–18, 1989.

[78] Martin Law. *An Introduction to Support Vector Machines.* Michigan State University, 2003. Powerpoint Class Slides from CSE 802 Pattern Recognition.

[79] Steve Lawrence and C. Lee Giles. Context and page analysis for improved web search. *IEEE Internet Computing*, 2(4):38–46, 1998.

[80] V. Lecce, G. Dimauro, A. Dimauro, S. Impedova, G. Pirlo, and A. Salzo. Classifier combination: The role of a-priori knowledge. In *7th International Workshop on Frontiers in Handwriting Recognition*, 2000.

[81] Jong-Hak Lee. Analyses of multiple evidence combination. In *Research and Development in Information Retrieval*, pages 267–276, 1997.

[82] Joon Ho Lee. Combining multiple evidence from different properties of weighting schemes. In *Research and Development in Information Retrieval*, pages 180–188, 1995.

[83] Joon Ho Lee. Combining multiple evidence from different relevant feedback networks. In *Database Systems for Advanced Applications*, pages 421–430, 1997.

[84] Zhibin Lei, Wu Chou, Jialin Zhong, and Chin-Hui Lee. Video segmentation using spatial and temporal statistical analysis method. In *IEEE International Conference on Multimedia and Expo (ICME 2000)*, 2000.

[85] Ying Li, Tong Zhang, and Daniel Tretter. An overview of video abstraction techniques. Technical report, Hewlitt Packard Labs, 2001.

[86] R. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Storage and Retrieval for Image and Video Databases VII*, volume 3656, January 1999.

[87] Rainer Lienhart. Reliable transition detection in videos: A survey and practitioner's guide. *International Journal of Image and Graphics*, 1(3):469–486, 2001.

[88] Rainer Lienhart, Silvia Pfeiffer, and Wolfgang Effelsberg. Video abstracting. *Communications of the ACM*, 40(12):54–62, 1997.

[89] Chih-Jen Lin. *Can Support Vector Machine be a Major Classification Method*, January 2003. Talk at Max Planck Institue.

[90] W. Lin and A. Hauptman. News video classification using svm-based multimodal classifiers and combination strategies. In *Proceedings of ACM Multimedia 2002*, 2002.

[91] W. Lin and A. Hauptmann. News video classification using svm-based multimodal classifiers and combination strategies. In *Proceeedings of ACM Multimedia*, pages 1–6, 2002.

[92] H. Liu and G. Zick. Automatic determination of scence changes in mpeg compressed video. In *Proc. ICASS-IEEE Int. Symp. Circuits and Systems*, pages 764–767, 1995.

[93] H. Liu and G. Zick. Scene decomposition of mpeg compressed video. In *Digital Video Compression: Algorithms and Technologies*, 1995.

[94] R. Losee. Comparing boolean and probabilistic information retrieval systems across queries and disciplines. *Journal of the American Society for Information Science*, pages 143–156, 1997.

[95] G. Lupatini, C. Saraceno, and R Leonardi. Scene break detection: A comparison. In *Proceedings of the RIDE'98 Eighth International Workshop on Research Issues In Data Engineering*, pages 34–41, Orlando, Florida, 1998.

[96] S. Mann and R. Picard. Video orbits: characterizing the coordinate transformation between two images using the projective group. Technical report, MIT Media Lab, Perceptual Computing, 1995.

[97] Steve Mann and Rosalind W. Picard. Virtual bellows: Constructing high quality stills from video. In *ICIP (1)*, pages 363–367, 1994.

[98] Jianhao Meng, Yujen Juan, and Shih-Fu Chang. Scene change detection in a mpeg compressed video sequence. In *SPIE Symposium on Electronic Imaging: Science And Technology*, 1995.

[99] Weiyi Meng, Clement Yu, and King-Lup Liu. Building efficient and effective metasearch engines. citeseer.nj.nec.com/376145.html.

[100] A. Miene, A. Dammeyer, Th. Hermes, and O. Herzog. Advanced and adaptive shot boundary detection. In *ECDL WS Generalized Documents*, 2001.

[101] S.K. Murthy, S. Kasif, and S. Salzberg. Automatic partitioning of full-motion video. *Journal of Artificial Intelligence Research*, 2(1):1–32, 1994.

[102] Akio Nagasaka and Yuzuru Tanaka. *Visual Database Systems II*, chapter Automatic Video Indexing and Full-Video Search for Object Appearances, pages 113–127. Elsevier Science Publishers B. V., North-Holland, 1992.

[103] M. Naphade, R. Mehrotra, A. M. Ferman, J. Warnick, T. S. Huang, and A. M. Tekalp. A high performance shot boundary detection algorithm using multiple cues. In *IEEE International Conference on Image Processing*, pages 884–887, 1998.

[104] C. W. Ngo, T. C. Pong, and R. T. Chin. Detection of gradual transitions through temporal slice analysis. In *IEEE Computer Vision and Pattern Recognition*, pages 36–41, 1999.

[105] N. O'Connor, C. Czirjek, S. Deasy, S. Marlow, N. Murphy, and A. Smeaton. News story segmentation in the fischlar video indexing system. In *ICIP01*, page Summarizing Video, 2001.

[106] N. O'Connor, S. Marlow, N. Murphy, A. Smeaton, P. Browne, S.Deasy, H. Lee, and K. McDonald. Fschlr: an on-line system for indexing and browsing of broadcast television content. In *Proceedings of ICASSP 2001*, 2001.

[107] N. Omoigui, L. He, A. Gupta, J. Grudin, and E. Sanocki. Time compression: System concerns, usuage, and benefits. In *Proc. of the ACM Conference on human Computer Interaction*, 1999.

[108] E. Osuna, R. Freund, and F. Girosi. Training support vector machines:an application to face detection. *CVPR'97*, 1997.

[109] C. OToole, A. Smeaton, N. Murphy, and S. Marlow. Evaluation of automatic shot boundary detection on a large video test suite. *Challenge of Image Retrieval*, 1999.

[110] N. Patel and I. Sethi. Compressed video processing for cut detection. In *IEE Proceedings - Vision, Image and Signal Processing*, 1996.

[111] N. Patel and I. Sethi. Video shot detection and characterization for video databases. In *Pattern Recognition*, pages 583–592, 1997.

[112] Silvia Pfeiffer, Rainer Lienhart, Stephan Fischer, and Wolfgang Effelsberg. Abstracting digital movies automatically. Technical Report TR-96-005, Hewlitt Packard Labs, 1 1996.

[113] Massimiliano Pontil and Alessandro Verri. Support vector machines for 3d object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(6):637–646, 1998.

[114] R.J. Quinlan. *C4.5: Programs for Machine Learning, The Morgan Kaufmann Series in Machine Learning*, chapter Constructing Decision Trees, page 1992. Morgan Kaufmann Publishers, 1992.

[115] L. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. In *Proceedings of the IEEE*, pages 257–285, February 1989.

[116] M.D. Richard and R.P. Lippmann. Neural network classifiers estimate bayesian a posteriori probabilities. In *Neural Computation*, pages 461–483, 1991.

[117] Arun Ross, Anil K. Jain, and Jian-Zhong Qian. Information fusion in biometrics. *Lecture Notes in Computer Science*, 2091:354–??, 2001.

[118] D. Russell. A design pattern-based video summarization technique: moving from low-level signals to high-level structure. In *System Sciences, 2000. Proceedings of the 33rd Annual Hawaii International Conference on*, pages 1137–1141, 2000.

[119] B. Sabata and M. Goldszmidt. Fusion of multiple cues for video segmentation. In *Proceedings of the International Conference on Information Fusion*, 1999.

[120] Bernhard Scholkopf. *Support Vector Learning.* PhD thesis, Berlin University, 1997.

[121] P. B. W. Schwering, B. A. Baertlein, S. P. van den Broek, and F. Cremer. Evaluation methodologies for comparison of fusion algorithms in land mine detection. In *SPIE Vol. 4742, Detection and Remediation Technologies for Mines and Minelike Targets VII*, 2002.

[122] B. Shahraray and D. Gibbon. Automatic generation of pictorial transcripts of video programs. In *Proc. SPIE Multimedia Computing and Networking*, pages 512–518, 1995.

[123] Joseph A. Shaw and Edward A. Fox. Combination of multiple searches. In *Text REtrieval Conference*, pages 0–, 1994.

[124] Jianbo Shi and Carlo Tomasi. Good features to track. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, Seattle, June 1994.

[125] Romildo Silva, Jonas Gomes, and Luiz Velho. Segmentation of video sequences using volumetric image processing. In *EG Multimedia Workshop*, 1999.

[126] M. Smith and T. Kanade. Video skimming for quick browsing based on audio and image characterization. Technical Report CMU-CS-95-186R, Carnegie Mellon University, 1996.

[127] M. Smith and T. Kanade. Video skimming and characterization through the combination of image and language understanding techniques. In *Proc. Computer Vision and Pattern Recognition Conf.*, pages 775–781, 1997.

[128] Anthony Stefanidis, Panos Partsinevelos, Peggy Agouris, and Peter Doucette. Summarizing video datasets in the spatiotemporal domain. In *DEXA Workshop*, pages 906–912, 2000.

[129] M. Sugano, Y. Nakajima, and H. Yanagihara. Automated mpeg audio-video summarization and description. In *Image Processing. 2002. Proceedings. 2002 International Conference on*, 2002.

[130] A. Tan and D. Gilbert. An empirical comparison of supervised machine learning techniques in bioinformatics. In *Proceedings of the First Asia Pacific Bioinformatics Conference*, 2003.

[131] Y. Tanomura, A. Akutsu, K. Otsuji, , and T. Sadakata. Videomap and video spaceicon: Tools for anatomizing video content. In *Proceedings of INTERCHI*, pages 131–136, 1993.

[132] C. Taskiran and E. Delp. Video scene change detection using the generalized trace. In *IEEE Int'l Conference on Acoustic, Speech and Signal Processing*, pages 2961–2964, Seattle, May 1998.

[133] D. M. J. Tax, R. P. W. Duin, and M. van Breukelen. Comparison between product and mean classifier combination rules. In *Proceedings of the Workshop on Statistical Pattern Recognition.*, 1997.

[134] L. Teodosio and W. Bender. Salient stills from video. In *Proceedings of the ACM Multimedia 1993 Conference*, 1993.

[135] S. Uchihashi and J. Foote. Summarizing video using a shot importance measure and frame-packing algorithm. In *Proceedings of ICASSP'99*, pages 3041–3044, 1999.

[136] Hirotada Ueda, Takafumi Miyatake, Shigeo Sumino, and Akio Nagasaka. Automatic structure visualization for video editing. In *INTERCHI '93 , ACM Press*, pages 137–141, 1993.

[137] V. Vapnik. *"The Nature Of statistical Learning Theory"*. "Springer-Verlag", "1995".

[138] N. Vasconcelos and A. Lippman. Towards semantically meaningful feature spaces for the characterization of video content. In *IEEE ICIP*, pages 25–29, 1997.

[139] Patrick Verlinde, Gerard Chollet, and Marc Acheroy. Multi-modal identity verification using expert fusion. *Information Fusion*, 1(1):17–33, 2000.

[140] H. Wactlar and M. Christel. Lessons learned from the creation and deployment of a terabyte digital video library. *Computer*, 32(2):66–73, February 1999.

[141] H.D. Wactlar, T. Kanade, M.A. Smith, and S.M. Stevens. Intelligent access to digital video: Informedia project. *Computer*, 29(5), May 1996.

[142] J.Y.A. Wang and E.H. Adelson. Representing moving images with layers. *IP*, 3(5):625–638, September 1994.

[143] I.H. Witten and E. Frank. *Data Mining: Practical machine Learning Tools and Techniques with Java Implementations*, chapter Implementations: Real Machine Learning Schemes, page 2000. Morgan Kaufmann Publishers, 2000.

[144] W. Wolf. Key frame selection by motion analysis. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 1228 –1231, Atlanta, GA, 1996.

[145] L. Xu, A. Krzyzak, and C. Suen. Methods of combining multiple classfiers and their applications to handwriting recognition. In *IEEE Transactions on Systems, MAN, and Cybernetics*, 1992.

[146] S. Yacoub, Y. Abdeljaoued, and E. Mayoraz. Fusion of face and speech data for person identity verification. Technical Report IDIAP-RR 99-03, Dalle Molle Institue for Perceptual Artifical Intelligence, January 1999.

[147] G. Yannakakis, J. Levine, J. Hallam, and M. Papageorgiou. Performance, robustness and effort cost comparison of machine learning mechanisms in flatland. In *11th Mediterranean Conference on Control and Automation*, 2003.

[148] B. Yeo and B. Liu. Rapid scene analysis on compressed video. In *IEEE Transactions on Circuit and Systems for Video Technology*, 1993.

[149] X. Yuan, X. Yuan, B. Buckles, and J. Zhang. A comparison study of decision tree and svm to classify gene sequence. *Workshop on Bioinformatics in conjunction with ICDE 2003*, 2003.

[150] Y. Yusoff, W. Christmas, and J. Kittler. Video shot cut detection using adaptive thresholding. In *British Machine Vision Conference*, 2000.

[151] Yusseri Yusoff, Josef Kittler, and William J. Christmas. Combining multiple experts for classifying shot changes in video sequences. In *ICMCS, Vol. 2*, pages 700–704, 1999.

[152] R. Zabith, J. Miller, and K. Mai. A feature-based algorithm for detecting and classifying scene breaks. In *ACM Multimedia Proceedings*, pages 189–200, 1995.

[153] H. Zhang, C. Low, and S. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, pages 89–111, 1995.

[154] H. J. Zhang, C. Y. Low, Y.H. Gong, and S. W. Smoliar. Video parsing using compressed data. In *Proc. SPIE Conf. Image and Video Processing II*, pages 142–149, 1994.

[155] HongJiang Zhang. *The Handbook of Multimedia Computing*, chapter Content-Based Video Browsing and Retrieval, pages 255–280. CRC Press, Boca Raton, FL, 1999.

[156] HongJiang Zhang, Atreye Kankanhalli, and Stephen W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1(1):10–28, 1993.

[157] HongJiang Zhang, Chien Yong Low, Stephen W. Smoliar, and Di Zhong. Video parsing, retrieval and browsing: An integrated and content-based solution. In *ACM Multimedia*, pages 15–24, 1995.

[158] D. Zhong and S. Chang. Video shot detection combining multiple visual features. Technical report, Columbia University ADVENT Technical Report, 2000.

[159] Y. Zhuang, Y.Rui, T. Huang, and S. Mehrotra. Adaptive key frame extraction using unsupervised clustering. In *Proc. of Intl. Conf. on Image Processing*, pages 886–870, 1998.

[160] Y. Zuev and S. Ivanon. The voting as a way to increase decision reliability. In *Foundations of Information/Decision Fusion with Applications to Engineering Problems*, pages 206–210, 1996.