CONTENT-BASED DIGITAL VIDEO PROCESSING

J. CHEN

PhD

2009

CONTENT-BASED DIGITAL VIDEO PROCESSING

Digital Videos Segmentation, Retrieval and Interpretation

Juan CHEN

Submitted for the degree of Doctor of Philosophy

Department of Computing

University of Bradford

2009

Juan Chen

Content-Based Digital Video Processing

Keywords: shot boundary detection, video copy detection, video annotation

ABSTRACT

Recent research approaches in semantics based video content analysis require shot boundary detection as the first step to divide video sequences into sections. Furthermore, with the advances in networking and computing capability, efficient retrieval of multimedia data has become an important issue. Content-based retrieval technologies have been widely implemented to protect intellectual property rights (IPR). In addition, automatic recognition of highlights from videos is a fundamental and challenging problem for content-based indexing and retrieval applications.

In this thesis, a paradigm is proposed to segment, retrieve and interpret digital videos. Five algorithms are presented to solve the video segmentation task. Firstly, a simple shot cut detection algorithm is designed for real-time implementation. Secondly, a systematic method is proposed for shot detection using content-based rules and FSM (finite state machine). Thirdly, the shot detection is implemented using local and global indicators. Fourthly, a context awareness approach is proposed to detect shot boundaries. Fifthly, a fuzzy logic method is implemented for shot detection. Furthermore, a novel analysis approach is presented for the detection of video copies. It is robust to complicated distortions and capable of locating the copy of segments inside original videos. Then,

objects and events are extracted from MPEG Sequences for Video Highlights Indexing and Retrieval. Finally, a human fighting detection algorithm is proposed for movie annotation.

ACKNOWLEDGEMENTS

I would like to give my deepest thanks and best wishes to my dearest parents Mr. Ya Xiong Chen and Mrs. Bi Hua Wu. Their love and support is the strongest influence for me to do the research work and finish my PhD study. Also, I would like to thank my beloved cousin Mr. Ge Wu for his wise suggestions to help me make correct decisions during my PhD study.

I would like to give my special thanks to my primary supervisor Prof. Jianmin Jiang. His sponsorship gives me this precious opportunity to be a PhD student. His profound knowledge and wisdom inspires me to make progress in my research work. His patience and kindness helps me to overcome the difficulties in the PhD study.

Many thanks are given to my second supervisor, Dr. Stan S. Ipson for his kind help in my study. His preciseness and excellence of using the English language inspires me a lot in the paper writing.

Also, I would like to thank my colleagues and friends. Their kind help and encouragement supports me in a difficult time.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Background

There has been an explosive growth of digital media applications in the past two decades caused by the rapid developments in Internet and multimedia techniques. Among these applications, digital video processing technology has become a hot topic owing to its flexibility and huge commercial potential. Some applications in digital video processing technology are summarized as follows.

(1) Video indexing: For browsing, searching, and manipulating video documents, an index describing the video content is required. It forms the crux for applications like digital libraries storing multimedia data, or filtering systems which automatically identify relevant video documents based on a user profile. To cater for these diverse applications, the indexes should be rich and as complete as possible.

(2) Visual surveillance: Visual surveillance in dynamic scenes attempts to detect, recognize and track certain objects from image sequences, and more generally to understand and describe object behaviours. The aim is to develop intelligent visual surveillance to replace the traditional passive video surveillance that is proving ineffective as the number of cameras exceeds the capability of human operators to monitor them.

(3) Video data mining: In visual media an object may appear in different imaging conditions (different camera angles, zoom positions, or lighting conditions) and may also be occluded. All these variations make visual data mining more challenging compared to text data mining.

(4) Video summarization: Video summarization aims to generate a series of visual contents for users to browse and understand efficiently the whole story of a video. From the perspective of the video summarization, presenting video content in the form of a structuralized and systematic view is important. Video content as well as articles are highly organized, and usually this content can be outlined according to the people involved, where the events took place, the things they related to and when they happened.

(5) Video copy detection: With the exponential growth of social media, there exist huge numbers of near-duplicate web videos, ranging from simple re-formatting to complex mixtures of different editing effects. In addition to the abundant video content, the social web provides rich sets of context information associated with web videos, such as thumbnail image, time duration and so on. At the same time, the popularity of Web 2.0 makes demands for timely response to user queries.

Although much research effort has been devoted to the development of digital video processing technologies, there are still many research problems related to the effective and accurate implementation of applications such as video segmentation, video content retrieval and video interpretation, which are discussed in detail in the next section.

1.2 Problems

It is well known that shot boundary detection is a major and important task for all content based video processing, analysis and applications. In the past decades, many shot boundary detection algorithms and techniques have been reported in the literature. However, there is still some work remaining unsolved. First, even the latest formal studies have not been advanced enough to cover the recent development of shot boundary detection (SBD) techniques. Second, the previous work, though it identified various specific core techniques, did not evaluate the calculation methods for content discontinuity.

Content-based video copy detection is one of the main approaches to IPR (intellectual property rights) protection, which searches the extracted signatures in an indexed database. The primary advantage of content-based copy detection is the fact that copies are detectable without previously embedded marks or the availability of the original material. On the other hand, retrieval efficiency is a key issue in the applications of multimedia search. Redundant copies from the search results need to be identified and removed in order to improve the efficiency and effectiveness of multimedia browsing. Video copy retrieval has been successfully applied to many content-based video processing and applications such as media tracking, video indexing, etc. Most of the video copy detection algorithms reported so far primarily focuse on simple changes introduced by different encoding parameters or simple editing effects, such as letter-box and frame rate changes. Little research has been carried out to address the combination of various distortions.

For extracting useful information from massive multimedia data sources, conventional text-based methods rely on adequate and accurate annotation of associated

contents. Due to the lack of flexibility and robustness in such annotations, a content-based approach to analysis, indexing and retrieval of media data has attracted much attention. Owing to its flexible nature and huge commercial potential, content-based approaches have been usefully applied in many applications such as digital library, video on demand, telemedicine, etc. Among these applications, a fundamental task is how to extract semantics and meaningful highlights from videos. This is highly desired as it could help to automate the annotation and abstract general video content for fast browsing and searching of whole videos. Unfortunately, this problem is still far from being completely solved as far as real applications are concerned.

1.3 Aims and Objectives

In shot boundary detection, my objective is to: (i) extract new features to facilitate the robust detection of gradual transition; (ii) construct new models to achieve a good trade-off between the detection accuracy and computation speed; (iii) apply human knowledge for the algorithm design. I aim to improve the shot boundary detection in five aspects. Firstly, a fast and simple shot cut detection algorithm needs to be designed, which operates directly in the compressed domain and is suitable for real-time implementation. Secondly, a fast and systematic method needs to be investigated for shot boundary detection in the compressed domain using content-based rules and finite state machine (FSM). Thirdly, shot boundary detection in MPEG videos can be implemented using local and global indicators. Fourthly, a context awareness approach can be studied with multiple feature indicators and multiple thresholds to detect shot boundaries. Fifthly, a fuzzy logic method of feature representation can be examined for shot boundary detection.

In video copy detection, my objective is to propose a new algorithm which is robust to a range of complicated distortions, and capable of locating the copy of segments inside original videos. Specific aims for video copy detection are summarized as follows. Firstly, a dedicated video distortion analysis can be implemented for input videos, which ensures the accurate detection of the complicated distortions query videos may undergo. Secondly, simple signatures can be extracted for the benefit of time and space efficiency, and a frame mask needs to be generated adaptively to reduce video temporal redundancy. Thirdly, a progressive matching process can be implemented and the property of linear regression can be utilized to find video copies.

In video interpretation, my objective is to extract semantics for video highlight and detect new events from video sequences. For video highlight extraction, my aims are summarized as follows. Firstly, compressed domain features can be extracted such as luminance, chrominance and motion. Then, the input video needs to be segmented into shots using the knowledge-supported rules. Zoom-in highlights can be determined by integrating the estimation of camera motion and the segmentation of skin patches.

In addition, for event detection from video sequences, my aim is to investigate a trajectory based approach for human fighting detection. The Kanade-Lucas-Tomasi (KLT) tracking record can be utilized to build the trajectory of points of interest and global motion estimator can be constructed based on the trajectory features. Then, camera motion can be measured and classified. Furthermore, the behaviours of points of interest can be measured by fuzzy scores and labelled by a voting scheme. Finally, the main movement and direction of points of interest with the same label can be used to detect human fighting.

1.4 Layout of the Thesis

This thesis consists of six chapters, the remainder of which are organized as follows:

Chapter 2 Literature Review

A literature review is presented of the state-of-art techniques and algorithms for video segmentation, video retrieval and video interpretation. Specifically, existing methods of shot boundary detection, content-based video copy detection and human activity recognition are discussed and summarized.

Chapter 3 Video Segmentation

Shot boundary detection is the fundamental task in content-based analysis, indexing and retrieval of videos, as it helps to provide a hierarchical structure for videos and enables the extraction of meaningful highlights from such a structure. As a result, this topic has continuously attracted extensive attention, which was also one of the motivations for the well-known TREC Video Retrieval Evaluation (TRECVID) activity, providing objective samples as a common platform for SBD and other video processing tasks. Five algorithms are presented to solve the video segmentation task.

Chapter 4 Video Retrieval

My strategy for video copy detection is implemented in three major procedures, including video distortion analysis, signature extraction and video matching. Firstly, a dedicated video distortion analysis is implemented for input videos, so that different kinds of complex editing effect are usefully tackled. Secondly, key frames are scaled down to extract simple signatures, and the frame mask is generated adaptively to reduce video temporal redundancy. Thirdly, linear regression and Support Vector Machines (SVM) are

utilized in the matching and decision making process to guarantee robust copy retrieval results.

Chapter 5 Video Interpretation

It is a fundamental problem to extract semantic events and highlights from video sequences for the interpretation of video content. Two approaches are proposed to address this problem. A knowledge-supported approach is presented to segment videos and extract semantic highlights, and a trajectory-based method is proposed to detect human fighting for movie annotation.

Chapter 6 Conclusion

In Chapter Six, the research is summarized and the contributions of the thesis are highlighted. Furthermore, some future plans are proposed to improve the current work.

Chapter 2

Literature Review

2.1 Introduction

This chapter mainly presents a literature review of the state-of-art techniques and algorithms for video segmentation, video retrieval and video interpretation. Specifically, existing methods on shot boundary detection, content-based video copy detection and knowledge-supported video highlights extraction are discussed and summarized. This chapter is organized into three parts. Firstly, existing shot boundary detection methods in both pixel domain and compressed domain are summarized in a survey of video segmentation. Secondly, video retrieval systems and content-based video matching methods are analyzed in a survey of video retrieval. Thirdly, camera motion estimation, skin detection techniques and human activity recognition are reviewed in a survey of video interpretation.

2.2 Survey of the Video Segmentation

It is well recognized that shot boundary detection is a major and important task for all content based video processing, analysis and applications. Recent research trends in semantics based video content analysis [1, 2] requires shot boundary detection as the first step to divide video sequences into sections maintaining a certain level of visual consistency so semantics can be extracted within content consistent sections [3-6].

For the past decades, numerous algorithms and techniques have been reported in the literature for shot boundary detections. Recently, such efforts are coordinated by TRECVID evaluations organized as an annual event since the year of 2001 [7-10]. A brief highlight of the existing work is given below.

The platforms for shot boundary detection are classified into two kinds: the pixel domain and the compressed domain. In the compressed domain, the algorithms proposed in [11-15] exploited the MPEG properties and its embedded motion estimation and compensation scheme for shot cut detection. In the pixel domain, shot cut detection is mainly conducted by detecting the major differences between adjacent frames. Representative techniques include histogram comparisons and edge difference examinations etc. [3, 16-20].

In order to achieve effective and efficient motion estimation and compensation inside digital videos, MPEG arranges video sequences into groups of pictures (GoP). The structure of such arrangement is illustrated in Figure-2.1. The MPEG video structure has the feature that inside each GoP there exist two B-frames between every pair of I-P frames or P-P frames. For the convenience of discussion, I refer the first B-frame as B_f (front-B) and the second B-frame as B_r (rear-B). As a result, shot cut detection can be designed in terms of these front-B and rear-B frames.

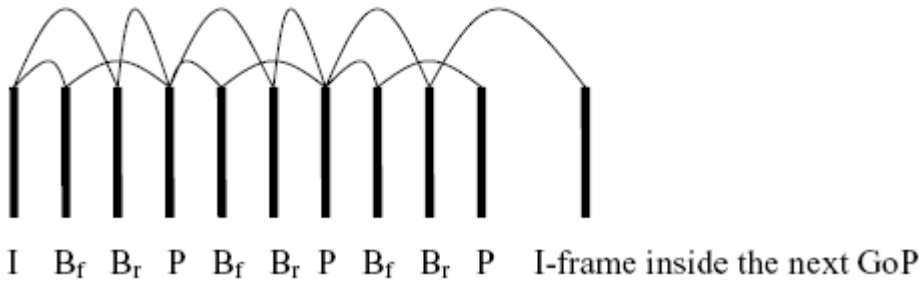


Figure 2.1 Illustration of MPEG video structure for motion estimation and compensation.

Fernando et al. [11] proposed an algorithm to detect shot cuts by exploiting the motion vectors which are available inside MPEG compressed videos. Lelescu [12] modelled video sequences as stochastic processes, in which changes of characteristics or parameters were exploited to detect scene changes. Kobla et al. [13] provided a detailed analysis of MPEG compressed domain, based on which shot cut detection was proposed via exploiting the MPEG motion estimation and compensation scheme. An earlier attempt for shot cut detection in the compressed domain was reported in [14], where Meng et al. used motion vectors and I, P-frame DC images to detect shot cuts. In [15], Pei and Chou proposed a simpler macro-block (MB) type based scene change detection algorithm for both abrupt and gradual scene changes. This algorithm differs from all others as only the type of MB is monitored to exploit the motion estimation and compensation scheme in MPEG and no motion vector or prediction status is analyzed. Therefore, such a scheme has the advantage of extremely low computing costs. Specifically, the scheme firstly counts the number of MBs in intra-coding mode inside P-frames. Whenever the number of intra-coded MBs is above a pre-defined threshold, two separate operations for scene change detection are activated for detecting abrupt changes and gradual changes respectively. The detection of abrupt changes is based on one of the three motion estimation and compensation forms in MPEG videos as illustrated in Figure-2.2. The detection of gradual changes is based on two conditions. One is that a significant number of MBs inside P-frames are intra-coded, indicating significant change of content and the other is that a dominant number of MBs inside B-frames are interpolative motion compensated. While the scheme is simple, with low computing cost, detailed analysis reveals that there exist a range of weaknesses, which include: (i) the scene change

detection is dependent on four fixed and pre-defined thresholds; (ii) abrupt change detection and gradual scene change detection are two separate operations; (iii) the performance is low in terms of the precision of detecting scene changes. Therefore, a new algorithm is proposed in Section 3.2 to improve the method [15].

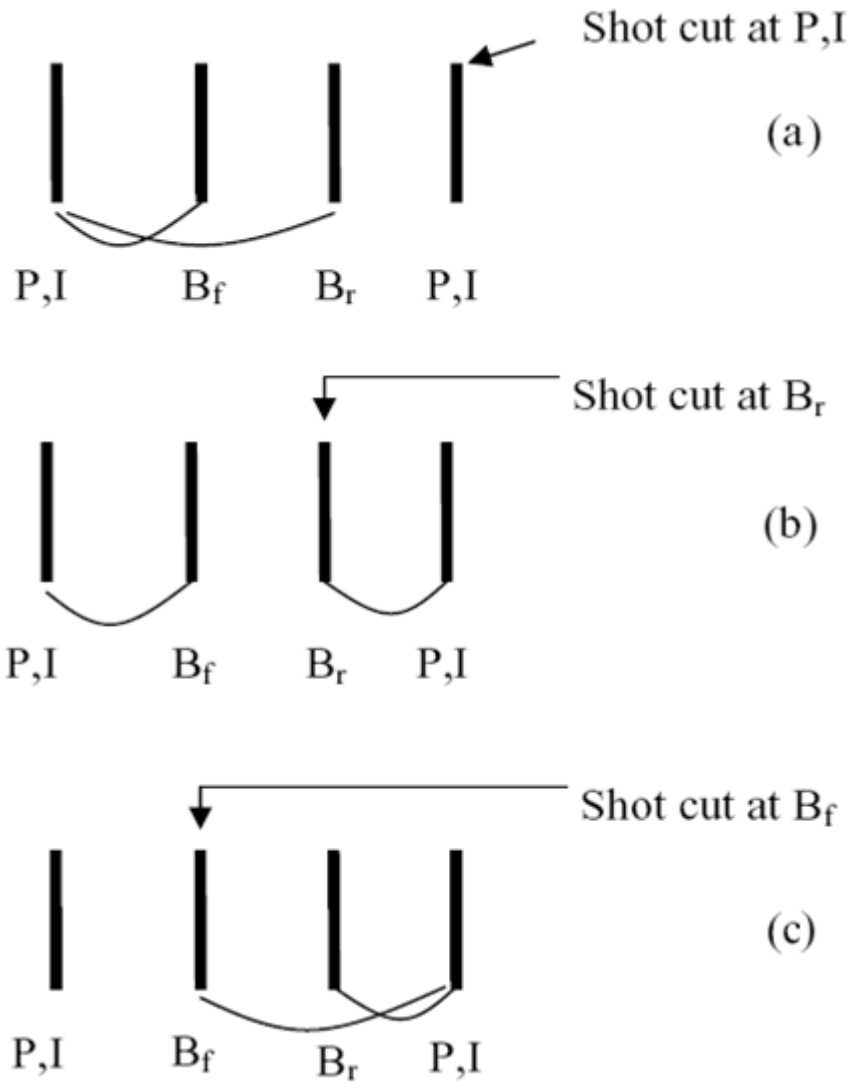


Figure 2.2 Illustration of abrupt shot cut detection.

In the approach proposed by Fang et al. [16], colour histogram intersection, motion compensation, texture change and edge variances are integrated in a fuzzy logic framework for the temporal segmentation of videos. However, the proposed fuzzy rules are designed without the integration of any domain knowledge. Yuan et al. [7] introduce a graph partition model to construct features for the SVM classification of shot boundaries, which requires massive training and the complex fusion of SVM classification results. Bescos et al. [21] map the inter-frame distance values onto a multidimensional space and selected a set of thresholds for shot change detection, the major weakness of which is the requirement for many manually defined thresholds.

In [17-19], colour intensity histogram features between consecutive frames are extracted to find the frame similarity on the hypothesis that frames in one shot will have similar histograms in the colour domain. Although this approach is less sensitive to motion, it has difficulty in detecting gradual transitions in similar colour shots. Edge-based methods can be found in [20], where the spatial distribution of exiting and entering edge pixels in successive frames is considered in a term called edge change ratio. However, it is found that edge features are sensitive to camera motions such as zoom, pan and tilt.

Additionally, some fast algorithms are proposed in [13, 14, 22], which focus on performing the shot boundary detection directly in the compressed domain. In general, these approaches use features like macro-block type information, discrete cosine transform (DCT) coefficients and motion vectors to characterize shot boundaries. However, the shot boundary detection is disturbed by special edit effects. Grana and Cucchiara [8] develop a linear transition model to identify the shot transition centre and length. Their proposed iterative algorithm measures the linear behaviour of shot transitions by minimizing an

error function. However, the performance of the model suffers from camera and object motions. Urhan et al. [23] present a hard-cut detection system based on modified phase correlation with application to archived films. Video frames are spatially sub-sampled for phase correlation and the generated peaks are detected by double thresholding, i.e. using global and local thresholds. A false detection is carried out using the mean and variance test in uniformly coloured frames. The main drawback of this approach is that the computational load of the frequency-domain processing is high and hence the processing speed is very low.

Boccignone et al. [24] partition a video into shots based on a foveated representation of the video. Their proposed method computes a consistency measure for the foveation sequences and Bayesian inference is adopted to detect the change of consistency. Due to the requirement of computing visuo-motor traces, the method is computing intensive. Cooper et al. [9] represent the local temporal structure of shot transitions using the pair-wise inter-frame similarity derived from YUV colour histograms. A discriminative feature selection process is performed based on mutual information for the K-Nearest Neighbours (KNN) classification of video shots. This method needs an offline feature selection so it is inefficient for fast applications.

Cao and Cai [10] employ a multi-class support vector machine (SVM) classifier to differentiate video frames into three categories: abrupt change, gradual change and non-change. The macro-block information from all frames within a temporal window is primarily exploited to construct the feature vector. The major disadvantage of their proposed algorithm is its lack of robustness as indicated in the experimental results. Another compressed domain method, introduced in [25], extracts DC images from videos

and an intensity variance sequence is generated to find “U” shape intervals for shot detection via an adaptive resonance theory 2 (ART2) neural network. In practice, however, such “U” shape intensity variance sequences often becomes indistinct due to motion, light changes, and error propagation caused by inaccurate feature extraction.

Matsumoto et al. [26] propose a 2-stage data fusion approach with an SVM technique to decide whether a boundary exists or not within a given video sequence. In [27], a temporal multi-resolution analysis is introduced to classify shot boundaries using SVM. A classification method based on rough sets and fuzzy c-means clustering for feature reduction and rule generation is proposed in [28] to deal with shot detection. In [29], frame features based colour histograms are clustered into shot change clusters, suspected shot change clusters and no change clusters using a fuzzy c-means method. Then, shot change frames are identified from the shot change clusters and suspected shot change clusters. In [30], a Fuzzy c-means algorithm is used to detect video-shot boundaries for the segmentation step.

2.3 Survey of the Video Retrieval

With the advances in high-performance networking and improvements in computing capability, efficient retrieval of multimedia data has become an important issue. Content-based retrieval technologies have been widely implemented to protect intellectual property rights (IPR) [31-35]. Watermarking and content-based copy detection are the main approaches for IPR protection. Watermarking inserts the identification of a document prior to distribution, while content-based copy detection searches the extracted signatures in an indexed database [36-39]. The primary advantage of content-based copy detection over watermarking is the fact that copies are detectable without previously embedded mark or

the availability of the original material. On the other hand, retrieval efficiency is a key issue in applications of multimedia search. Redundant copies from the search result need to be identified and removed in order to improve the efficiency and effectiveness in multimedia browsing. Video copy retrieval has been successfully applied to many content-based video processing and applications such as media tracking [40], video indexing [41], etc.

Video retrieval has been extensively studied, and several prototype systems have been developed [41-49]. Recently, a video analysis and retrieval system named InsightVideo [50], was proposed to integrate video content hierarchy, hierarchical browsing and retrieval for efficient video access, in which key frame extraction, shot grouping, scene detection and pair wise scene clustering are applied to construct the video content hierarchy. The video similarity evaluation at different levels enables progressive video retrieval. The weakness of this system, however, lies in the fact that too many thresholds are needed and they are determined by empirical studies. As a result, content hierarchy constructed in such a way could be sensitive to changes of video content. Shao et al [51] report a novel approach towards efficient high-dimensional Batch nearest neighbour (BNN) search called dynamic query ordering (DQO) for advanced optimizations of both I/O and CPU costs. A comprehensive overview of the retrieval topic can be found in [52-55].

Several shot-based similarity matching approaches have been proposed for query-by-clip applications [56-60]. In [5], an attention-driven shot matching method is proposed, and a concept called focus of attention (FOA) has been introduced. It combines spatial and motion attention maps adaptively to form an overall attention map, from which the FOA is

detected for a given frame. Hence, the shot matching method is designed to match frames with emphasis on attentive regions. However, the FOA detection method has a limitation on the use of the threshold in the object segmentation process. A fast coarse-to-fine video retrieval scheme using shot-level spatio-temporal statistics is addressed in [61]. In the coarse search stage, the shot-level motion and colour distribution is computed as spatio-temporal features for shot matching. An extra step is required to refine the search result by using the local colour features extracted from the key frames inside the query shots. The performance of these shot-based methods depends heavily on the accuracy of shot detection. As known to all, the detection of gradual transitions is still challenging and unresolved. Therefore, shot-based approaches are vulnerable to the inaccurate detection of shot boundary transitions.

Rothganger et al. [62] proposed the representation of multiple rigid objects in 3D models, and apply the constructed models for shot matching. By using multiple cameras, the work reported establishes groups of affine-covariant scene patches and segments each scene into its rigid components in a three dimensional space. The major disadvantage of this approach is its implementation time. The total processing time for a typical shot is approximately 90 minutes [62]. Approaches based on points of interest [63-65] were proposed recently and good results are reported in term of robustness to geometry changes such as cropping, shifting and change of ratios. In [66], a new approximate similarity search technique is proposed, in which the probabilistic selection of the feature space is based on the distribution of the features distortion. Chiu et al. [67] defines the problem of video copy detection as a partial matching problem in a probabilistic model and transforms it into a shortest-path problem in a matching graph. To reduce the computation costs, the

beginning and ending part of query videos are directly selected as the candidate segments to compare with target videos. However, the framework would fail if the query video is generated by incorporating irrelevant scenes, especially those added at its beginning and ending parts. In [68], a sequential search approach is proposed to identify video copies. The drawback of this method is that it ignores the effect of many other changes, such as frame rate change inside the query videos, and the search process is time-consuming.

Most of the video copy detection algorithms reported so far focus primarily on simple changes introduced by different encoding parameters or simple editing effects, such as letter-box and frame rate changes. Little research, however, has been carried out to address the combination of various distortions. Hence, in Chapter 4 a new algorithm is proposed for video copy detection, in which combination of complicated distortion is taken into consideration when those copied parts of query videos are determined. The advantage of my proposed algorithm lies in the fact that it is robust to a range of complicated distortions, and yet is capable of locating the copy of segments inside original videos.

2.4 Survey of the Video Interpretation

It is found that camera motion usually represents the focus and interest of human beings on the video content. Therefore, the estimation of camera motion is used to facilitate effective video parsing [69] and browsing [70]. Some existing work on camera motion estimation is discussed and summarized below.

The direction change and magnitude difference of motion vectors are calculated for the estimation of camera motion in [69]. In the approach proposed by Kobla et al. [13], a histogram of motion direction is constructed to facilitate the detection of camera pan and

tilt. The determination of camera zoom is based on the focus of contraction and expansion. Ewerth et al. [71] studied the distinction between camera translation and rotation in the MPEG domain. By the investigation of motion vectors, outliers are removed and the appropriate ones are chosen for camera motion estimation. Tan et al. [70] estimated the camera zoom in (out), pan and tilt based on the project camera model.

The human skin occupies a consistent range in the colour space and this is the primary principle for skin detection algorithms. The segmented skin patches provide efficient visual clues for the detection of human objects. Some representative approaches on skin detection are reviewed below.

A universal colour model is proposed in [72] and the skin pixels are segmented by thresholding the Mahalanobis distance. Phung et al. [73] addressed the problem of skin detection in terms of colour representation, colour quantization and classifier selection. It shows that the luminance is necessary for the colour representation. In addition, the selection on quantization level and classifier are compared and discussed. In the approach proposed by Jones et al. [74], the construction of skin colour models on a large data set is described. The performance of histogram and mixture models is compared. Kakumanu et al. [75] provided a review on skin modelling and classification, as well as the illumination adaptation techniques, which are applied to improve the performance of skin detection under environmental changes. A Markov model is used in [76] to predict evolution for real-time skin segmentation in video sequences.

The automatic recognition of human activities [77-81] is a fundamental step for the annotation of video content. Thus, it has a wide range of applications for video

surveillance [82-86]. Some state-of-art algorithms on human activity recognition are summarized below.

Lv et al. [79] proposed a statistical learning method to detect human activities in videos. The dynamic programming-based training algorithm complying with Neyman-Pearson criterion is used to select the optimal classifiers. In the approach presented by Ribeiro et al. [80], data distributions are learned by a Bayesian classifier with the likelihood functions of Gaussian mixtures. Human activities of daily living are recognized by a switching hidden semi-Markov model in [81]. Both the inherent hierarchical organization and the typical duration of human activities are exploited to build the model. Oliver et al. [87] introduced a layered hidden Markov model to recognize human activity in an office environment.

A combination of category components is proposed in [88] to represent human activities such as walking, running and fighting. The proposed method is flexible for new activities added into the system. This method utilized the changes of bounding boxes, which embraced human objects, as the main feature for activity recognition. However, this feature is unavailable in the real-world video sequences. Hence, general features are extracted in the proposed algorithm in Section 5.2 for movie annotation.

2.5 Summary

A literature review of video segmentation, video retrieval and video interpretation is presented in this chapter. The state-of-art algorithms are discussed and summarized in the survey. Specifically, algorithms and methods for shot boundary detection, video retrieval systems, content-based copy detection, camera motion estimation, skin detection and human activity recognition are introduced and analyzed in the literature review.

Chapter 3

Video Segmentation

3.1 Introduction

Shot boundary detection (SBD) plays important roles in many video applications. A fast and simple shot cut detection algorithm is proposed in Section 3.2, which operates directly in the compressed domain and is suitable for real-time implementation. The proposed algorithm exploits the existing MPEG techniques by examining the prediction status for each macro-block inside B frames and P frames. As a result, locating both abrupt and dissolved shot cuts is achieved by a sequence of comparison tests, and thus no feature extraction or histogram differentiation is needed. Although the description of the algorithm is primarily based on MPEG-1 and MPEG-2 streams, the scheme can be readily extended to other video compression standards such as MPEG-4 and H.264 by following the principles of monitoring: (i) balance between forward prediction and backward prediction; (ii) boundaries among P, B and I frames. Extensive experiments demonstrate that the proposed algorithm outperforms similar existing algorithm, providing a useful technique for fast and on-line video content processing.

A fast and systematic method is presented for shot boundary detection in the compressed domain using content-based rules and finite state machine (FSM) in Section 3.3. Firstly, several feature indicators are acquired from DC images in MPEG videos including luminance, colour, edge, prediction error and inter-frame difference as well as

motion. Then, several content-based rules are utilized to detect abrupt cuts. Thirdly, boundaries of gradual transitions are determined by a coarse to fine procedure with a pre-processing module and an FSM. The results from experiments using publicly available sequences from TRECVID have showed that the proposed algorithm outperforms the representative existing algorithms in both precision and recall rates.

In Section 3.4, my contribution for the submission to TRECVID 2007 on the shot boundary detection task is summarized as: (i) Novel features are extracted from the compressed domain and the feature selection is carried out using Adaboost; (ii) Typical gradual transitions including fade in (out), dissolve and wipes, as well as a new type of gradual transition called “others” are detected. (iii) The MPEG decoding scheme is thoroughly studied so that the whole shot detection system is embedded in the compressed domain of the standard MPEG2 decoder to achieve the real-time efficiency.

In Section 3.5, some recent work on a hierarchical content-aware approach to design multiple test conditions for shot cut detection is described. This is organized into a multiple phase decision tree for abrupt cut detection and an FSM for dissolve detection. In comparison with existing approaches reported in the literature, the proposed algorithm is characterized by two categories of content difference features and testing. While the first category indicates the content changes that are directly used for shot cut detection, the second category indicates the contexts under which the content change occurs. As a result, indications of frame differences are tested with context awareness to make the detection of shot cuts adaptive to both content and context changes. Evaluations announced by TRECVID 2007 indicate that, among all submissions, the proposed algorithm achieved the

fifth best results for gradual transition detection, the fourth best results for gradual transition frame accuracy and sixth best results for abrupt shot cut detection.

Unlike most approaches reported in the literature, the proposed algorithm in Section 3.6 is characterized by using a fuzzy logic method for feature representation of shot detection. Firstly, novel features are extracted in the compressed domain. Secondly, a fuzzy logic method is used to implement feature representation. Thirdly, multiple Support Vector Machines (SVMs) are constructed for further verification using features generated from the fuzzification process in step two. The proposed algorithm combines the merits of the generalization ability of the SVM and the comprehensibility of the fuzzy logic. I have carried out extensive experiments using the data from TRECVID 2007. The proposed algorithm achieved very good detection results compared with TRECVID 2007 algorithms and its run time is 4 times faster than the video play time.

3.2 Real-time Shot Cut Detection in Compressed Domain

To improve the scheme in [15], I propose to: (i) introduce an adaptive mechanism so that comparison tests are adaptive to the outcome of motion estimation and compensation for each B frame; (ii) control all comparison tests by using only three parameters; (iii) combine abrupt change detection and gradual change detection into an integrated shot cut detection algorithm.

3.2.1 The Proposed Algorithm Design

Following the spirit of the work reported in [15], the MPEG motion estimation and compensation technique is exploited to detect shot cuts by monitoring the number of predicted macro-blocks inside each P or B frame. For the convenience of description and presentation, a range of essential variables is defined as follows:

N_i, the number of intra-coded blocks inside P-frames;

N, the total number of blocks inside each video frame;

N_{fb}, the number of both forward and backward predicted macro-blocks inside each B-frame;

N_b, the number of backward predicted macro-blocks inside each B-frame;

N_f, the number of forward predicted macro-blocks inside each B-frame.

A global view of the overall proposed algorithm is shown in Figure 3.1. Given the input video sequence, the number of intra-coded blocks inside each P-frame is monitored via the following comparison test:

$$\frac{N_i}{N} > T \,. \tag{3.1}$$

where T stands for a threshold, which is to be determined empirically. The essence of the ratio between N_i and N calculated in (3.1) determines whether this tested P-frame is content correlated to its reference frame or not. When the ratio is smaller than the threshold, it indicates that most of the macro-blocks can be motion compensated by its reference frame, and hence a significant extent of correlation between this P-frame and its reference frame can be established. Therefore, it is not likely that there exists any shot cut around this P-frame. As a result, the next P-frame should be examined. Otherwise, if the condition represented by (3.1) is satisfied, it should indicate that most of the macro-blocks are not well compensated by the reference frame, and hence it is likely that there exists a shot cut in the neighbourhood of this P-frame. As a result, further confirmation of such a shot cut needs to be tested by examining its subsequent B-frames to find out: (i) whether such shot cut is abrupt or gradual; (ii) the exact location of the shot cut.

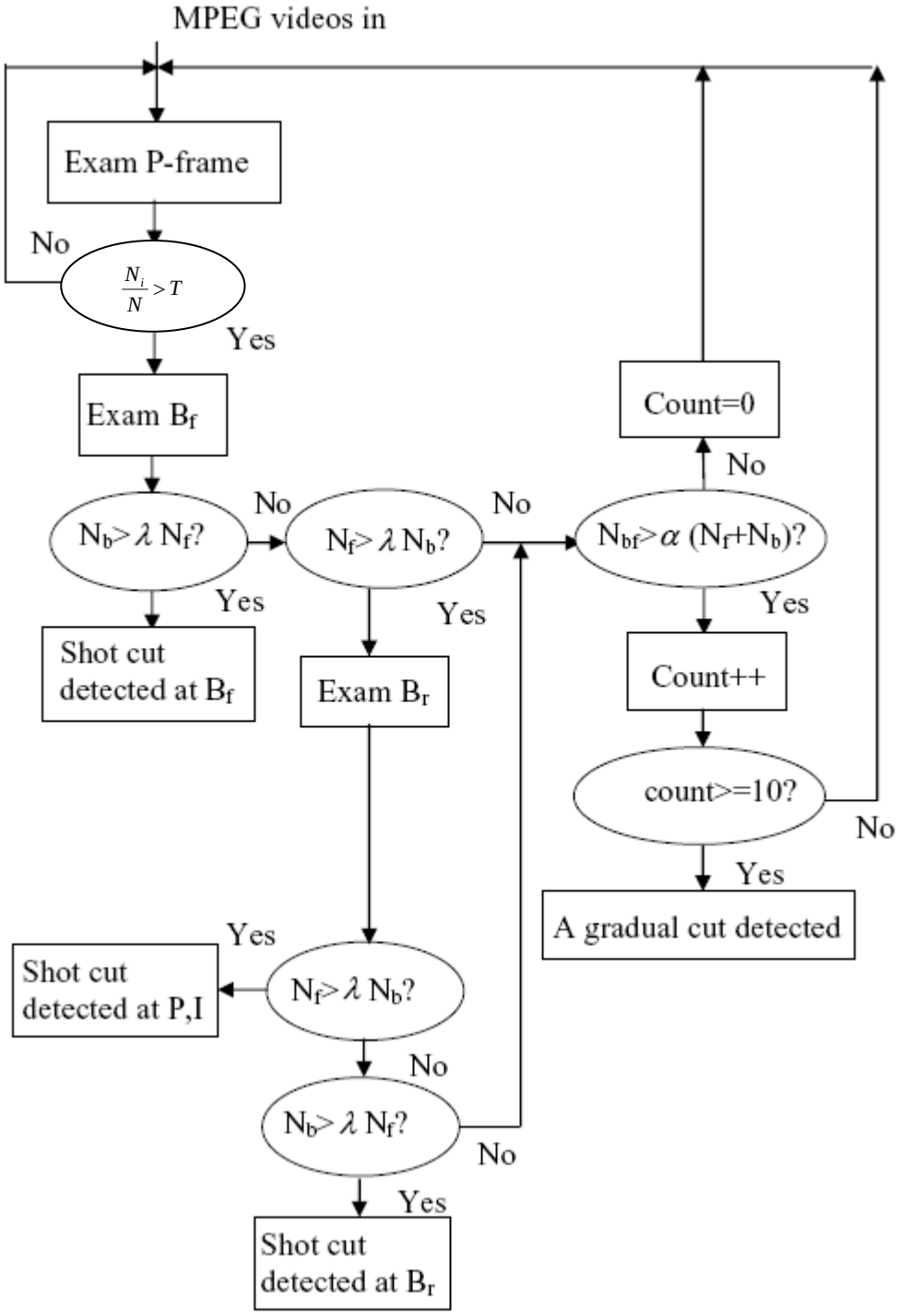


Figure 3.1 Overview of the proposed shot cut detection algorithm

According to MPEG specifications, all B-frames have three possibilities for their specific process of motion estimation and compensation, which include: (i) forward prediction, (ii) backward prediction and (iii) bi-directional prediction. To confirm a shot

cut and its location, it is needed to determine for each B frame whether its content is more correlated to its preceding reference frame (P or I) or subsequent reference frame (P or I) by monitoring its prediction status, i.e. forward, backward, or bidirectional prediction. A B-frame is more content correlated to its preceding reference frame if there exists an overwhelming number of forward predicted macro-blocks. In other words, MPEG motion estimation and compensation is more balanced towards forward prediction. This corresponds to the illustration in part (a) of Figure 2.2. Equally, a B-frame will be more content correlated to its subsequent reference frame if more macro-blocks are backward predicted, which corresponds to the illustration shown in Part (c) of Figure 2.2. Therefore, to test whether the B-frame is more content correlated to its preceding reference frame, the following test is proposed:

$$N_f > \lambda \cdot N_b . \tag{3.2}$$

where λ is a parameter controlling the balance between the backward prediction and forward prediction. Similarly, the balance towards backward reference frame can be detected via:

$$N_b > \lambda \cdot N_f . \tag{3.3}$$

Correspondingly, the abrupt shot cut detection can be designed into the following two steps:

Step-1: By using equation (3.1), examine every P-frame inside the video sequence to see if the shot cut detection process should be activated or not. For positive test of (3.1), go to step-2. Otherwise, carry on with step-1 to examine the next P-frame;

Step-2: Following satisfaction of (3.1), examine B_f and B_r to detect an abrupt shot cut according to one of the three situations: (i) if both B_f and B_r have more forward predicted blocks (satisfaction of equation (3.2)), an abrupt shot cut is detected between B_r and its subsequent P or I frame (see part (a) of Figure 2.2); (ii) if both B_f and B_r have more backward predicted blocks (satisfaction of equation (3.3)), an abrupt shot cut is detected between B_f and its preceding P or I frame (see part (c) of Figure 2.2); (iii) if B_f has more forward predicted blocks yet B_r has more backward predicted blocks, an abrupt shot cut is detected between B_f and B_r (see part (b) of Figure 2.2).

If none of the conditions given in Step-2 is detected, it is needed to check if there exists a possible gradual shot cut by examining the B-frames with the following test

$$N_{bf} > \alpha \cdot \left(N_f + N_b\right). \tag{3.4}$$

where α is another parameter indicating the dominance of bi-directional prediction of MBs inside B-frames.

The above test exploits the fact that gradual shot cuts incur gradual content change and such gradual change support bidirectional prediction inside B-frames such as fade-ins and fade-outs. To detect the possible gradual shot cuts out of those candidates satisfying (3.4), following the work in [16], a consecutive number of such candidate frames are examined. In general, the duration of most gradual shot boundaries is more than 1 second, which means that the duration of such changes is around 30 frames. To this end, a simple counting process is adopted to monitor the number of consecutive B frames that satisfy the condition given in (3.4). As illustrated in Figure 3.1, whenever the consecutive number of candidate B-frames is greater than 10, a gradual cut is detected. This value of 10 is

determined on both theoretical and empirical bases, where the duration of 30 frames corresponds to around 20 B frames (see Figure 2.1), leading to around 10 P frames. The empirical investigation also verifies that consecutive satisfaction of (3.4) for 10 times is an appropriate indicator for a gradual shot cut in consideration of general trend on most duration of gradual changes. For other compression schemes, this value needs to be adjusted according to a specific ratio between P frames and B frames.

3.2.2 Experimental Results

To evaluate the performance of the proposed shot cut detection algorithm, a test set with three groups of video sequences is prepared, including news (12 video clips), movies (8 video clips) and cartoons (7 video clips). The test set contains a total of 403 abrupt shot cuts and 87 dissolved shot cuts lasting around 90 minutes. The selected test sequences are complex with extensive graphical effects. Videos were captured at a rate of 30 frames/s and resolution of 640×480 pixels.

For benchmarking purposes, the algorithm reported in [15] is selected as a representation of the existing techniques to provide a comparison for the evaluation of the proposed algorithm. It is understood that there exists extensive research work on shot cut detections, with numerous algorithms reported in both pixel domain and compressed domain. However, this selection is justified by applying the principle that any benchmark selected should maintain fair comparisons with comparable computing cost and algorithm complexity. To provide some further information about the proposed algorithm, the work in [16] is used as the second benchmark, which is an unfair comparison since this algorithm has much higher complexity and computing cost.

To measure the performances on shot cut detection for the evaluated algorithms, the recall and precision rates [16, 89, 90] are used following the common practice. By the analysis, the three parameters (T, λ, α), are in the range of (0%,100%), (1,5) and (1,5), respectively. (5%, 3.6, 2.5) are used in the experiments, which was determined empirically. Such an empirical approach has the same nature as all those threshold-based techniques reported in the pixel domain [16, 89, 90], where a certain range of flexibility exists. As indicated by equation (3.1), higher values of T increases the reliability of shot cut detection but misses more shot cuts. Similarly, equations (3.2) and (3.4) also indicate that, while higher values of λ or α reduce the false positive detection rate, it could increase the number of shot cuts being missed.

All the experimental results are summarized in Table 3.1, where the number in the first row specifies the total number of shot cuts inside the video sequences, and the pair of values inside the brackets are recall and precision. For the convenience of visual inspection, samples of detected shot boundaries are shown in Figure 3.2, where each row of frames corresponds to one category of the video clips.

Table 3.1 Summary of experimental results for shot detection

Video Sequences		**News (93)**	**Movies(138)**	**Cartoons(172)**	**Total(403)**
Proposed method	**Abrupt**	(87%,82%)	(88%,85%)	(84%,90%)	(86%,85%)
	Dissolved	(72%,52%)	(53%,68%)	(63%,73%)	(63%,64%)
Benchmark-1	**Abrupt**	(38%,64%)	(31%,78%)	(32%,39%)	(34%,68%)
	Dissolved	(35%,46%)	(28%,50%)	(36%,58%)	(33%,51%)
Benchmark-2	**Abrupt**	(96%,94%)	(100%,100%)	(98%,100%)	(98%,98%)
	Dissolved	(87%,73%)	(81%,77%)	(79%,80%)	(82%,77%)

(a): Shot boundary samples in news.

(b): Shot boundary samples in movies.

(c): Shot boundary samples in cartoons.

Figure 3.2 Frame sample illustration of the test video clips

From Table 3.1, it can be seen that the proposed algorithm outperforms the benchmark-1 in terms of both recall and precision rates for abrupt and dissolved shot cut detections. Specific comparisons between the proposed and benchmark-1 can be summarized as: (i) for abrupt shot cut detection, the proposed algorithm achieves on average a 86% recall rate and a 85% precision rate, yet the benchmark achieves 34% recall rate and 68% precision rate; (ii) for dissolved shot cut detection, the proposed algorithm delivers 63% recall and 64% precision rate, yet the benchmark delivers 33% recall and 51% precision; (iii) both the techniques work better for abrupt shot cut detection than for dissolved shot cut detection.

Compared with benchmark-2, the proposed algorithm is about 12% inferior on all measurements. Considering the fact that benchmark-2 has much more complicated operations, such as multiple feature extraction (colour histograms, motion compensation

and textures) and fuzzy inferences [16], the proposed algorithm possesses unique advantages in terms of real-time applications and compressed domain operation. Further comparisons in practical terms are given below.

As the work is prompted by an EU funded FP-6 integrated project, the figures shown in Table 3.1 can be further interpreted in practical applications to reveal its general performance. In this integrated project, a video processing tool is to be developed to extract series of semantic features, like close-up of human objects, given input videos of various types (documentary, sporting, news, films etc.). The first step in applying such a tool is to cut the input video sequence into sections, where consistent spatial content can be identified and thus semantics can be extracted within each section. In this circumstance, both benchmark-2 and the proposed algorithm are applied to the video processing system. It was discovered that both algorithms meet the needs for semantics feature extraction and no noticeable difference was found during the comparative tests. The further observations explain the reason that: (i) although some cuts were missed by the proposed algorithm, the visual content still maintains certain level of consistency, such as both shots follow the same human objects but with different background (different corners of the same room etc.). To this end, the missed cuts do not produce any noticeable negative impact upon semantics feature extraction; (ii) for those false positive cuts detected, it was noticed that the boundary visual content does present significant differences, and thus such false alarms do not produce any noticeable negative impacts either.

While the proposed algorithm is primarily based on MPEG-1 or MPEG-2 schemes, it is readily extendable to other video compression standards such as MPEG-4 and H.264. The major implications are two fold. One is the variable block sized motion estimation and

compensation and the other is variable allocation of B-frames and P-frames. For the issue of variable block size, the proposed algorithm will still work without any significant change, since the balance between forward-prediction and backward-prediction can still be monitored by testing the number of predicted blocks. Further consideration may be required by looking into the size of each predicted block, which can be implemented via introducing a table of fixed weighting factors. Regarding the issue of variable allocation of B-frames and P-frames, detailed analysis of their corresponding modes of motion estimation and compensation is required. However, the same principle can still be applied to such analysis in the sense that only three boundaries need to be examined. That is: (i) the boundary between the front P-frame or I-frame and all other B-frames; (ii) the boundary between B-frames and (iii) the boundary between the last P-frame or I-frame and all other B-frames. This is illustrated in Figure 3.3.

Finally, to provide a wider evaluation of the proposed algorithm, comparisons are made with the motion-vector based scene change detection algorithm [14]. This algorithm is very different from the proposed. The major differences can be highlighted by the fact that: (i) Meng's algorithm monitors motion vectors, yet the proposed algorithm monitors the MB types; (ii) while Meng's work could be interpreted as monitoring MB type through motion vectors, no consideration is given for bi-directional prediction; (iii) the proposed algorithm goes one step further by monitoring the MB types to detect the correlation of B frames, i.e., more correlated to front reference frame or more correlated to rear reference frame (see Figure 2.2). (iv) Meng's algorithm requires the variances of DC image for I and P frames; (v) Meng's work relies on distances among suspected scene changed frames to

detect the scene changes, yet the proposed algorithm uses the content adaptive thresholds to determine how close the B frames are correlated to their front or rear reference frames.

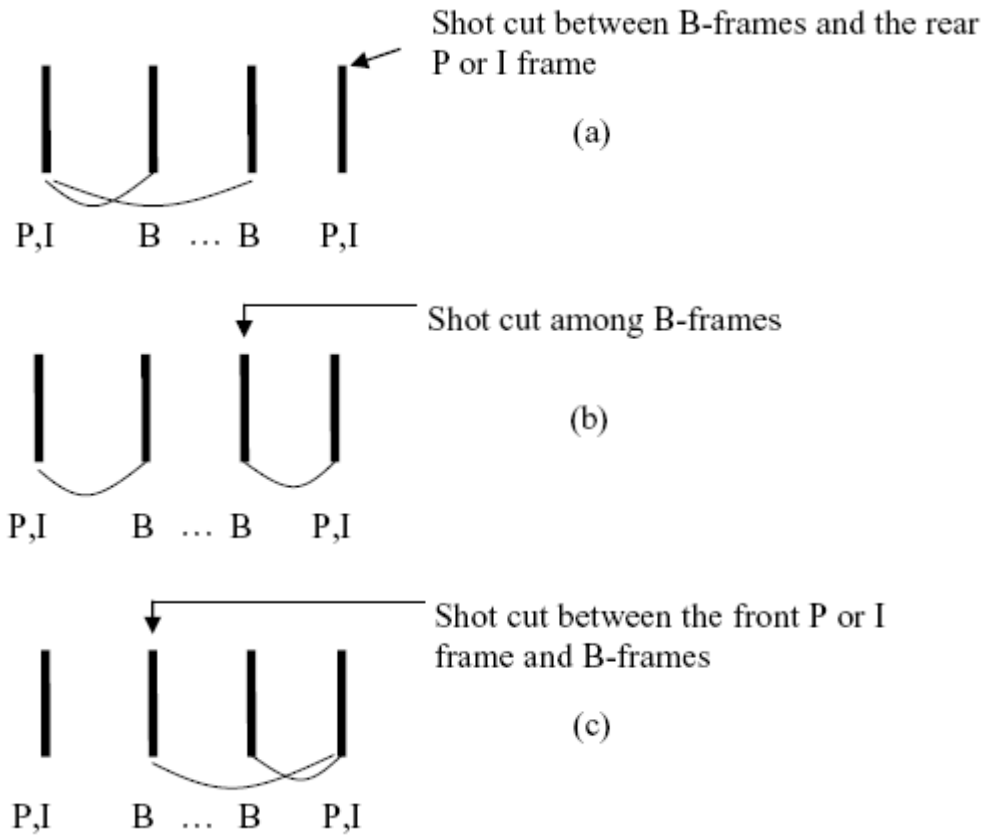


Figure 3.3 Analysis of shot cut modes in variable video streams

To highlight the comparison in terms of exploiting MPEG motion estimation and compensation scheme, this algorithm is implemented based on the three ratios of R_p, R_b and R_f to detect their peaks and then decide the scene changes by using the distance criteria $T_{rejection}$ (default one GOP) [14]. The experimental results are shown in Table 3.2. From these results, it can be seen that the proposed algorithm outperforms Meng's work in terms of abrupt shot cut detection. While Meng's recall rates are close to the proposed algorithm, its precision rates are significantly lower due to a large number of false alarms.

As suggested by the reviewer, Meng's algorithm on dissolved cut detection is not implemented since it requires variance of DC images, which cannot be implemented within the designated deadline. In addition, such variance-based parabolic detection remains the same as conventional techniques in pixel domain, which is much more computing intensive than the proposed.

Table 3.2 Comparative experiments for abrupt cut detection

Video Sequences	The proposed	Meng's algorithm
News (93)	(87%,82%)	(85%,63%)
Movies(138)	(88%,85%)	(74%,38%)
Cartoons(172)	(84%,90%)	(81%,47%)
Total(403)	(86%,85%)	(83%,78%)

3.3 Compressed-domain Shot Boundary Detection using Finite State Machine and Content-based Rules

In this section, I propose techniques to efficiently detect shot boundaries via content based indicators for cuts and a finite state machine (FSM) for gradual transitions. This algorithm uses YCbCr components, DCT coefficients, and motion vectors extracted directly from the compressed domain to form the features. Its efficiency comes from compressed domain processing where only partial decoding is required, avoiding the computationally expensive inverse discrete cosine transform (IDCT).

3.3.1 Algorithm Design

In this Section, the algorithm design is presented including feature extraction from the compressed domain and detection of cuts and gradual transitions using a rule-based approach and FSM, respectively. The rules and FSM states are derived by observations on more than 50000 frames containing several hundreds of shots in various types. Technical details are given in the following sections.

3.3.1.1 Compressed Domain Feature Extraction

Given an input MPEG video sequence, {IPBBP … IBBP…}, I use the technique proposed in [91] to extract the DC image from DC coefficients. Please note that according to the definition of DCT in (3.5), it is easily found that the first DC coefficient *F(0,0)* is 8 times the average intensity of the 8×8 block.

$$F(u,v) = \frac{C(u)}{2}\frac{C(v)}{2}\sum_{y=0}^{7}\sum_{x=0}^{7} f(x,y)\cos(\frac{2x+1}{16}\pi u)\cos(\frac{2y+1}{16}\pi v). \qquad (3.5)$$

where $C(0) = 2^{-1/2}$ and for other *u* or *v*, $C(u) = C(v) = 1$. In the following I discuss how to extract several feature indicators including luminance, colour, motion, edge, prediction error and inter-frame difference.

Luminance indicator: the normalized intensity difference between the n^{th} frame and the $(n-1)^{th}$ frame is represented as:

$$D_n = (4MN \times 255)^{-1} \sum_{i=1}^{2M} \sum_{j=1}^{2N} \left| Y_n(i, j) - Y_{n-1}(i, j) \right|. \tag{3.6}$$

where, *M* and *N* are the number of macro-blocks in the frame on the vertical and horizontal direction, respectively. $Y_n(i, j)$ is the luminance value of block (i, j).

Colour indicator: the colour dissimilarity based on histogram correlation between the n^{th} frame and the $(n-1)^{th}$ frame is determined as follows:

$$p_n = 1 - \frac{\sum_{i=1}^{8} \sum_{j=1}^{8} H_n(i, j) \cdot H_{n-1}(i, j)}{\sqrt{\sum_{i=1}^{8} \sum_{j=1}^{8} H_n(i, j)^2 \cdot \sum_{i=1}^{8} \sum_{j=1}^{8} H_{n-1}(i, j)^2}}. \tag{3.7}$$

where, $H_n(i, j)$ is the two dimensional 8×8 colour histogram in the n^{th} DC image, from which Cb and Cr are both placed into 8 bins.

Motion indicator: the normalized motion magnitude extracted from motion vectors $V_x(i, j)$ and $V_y(i, j)$ in the n^{th} frame is represented as:

$$mag_n = (10K_n)^{-1} \max(\sum \left| V_x(i, j) \right|, \sum \left| V_y(i, j) \right|). \tag{3.8}$$

where K_n is the number of inter-coded macro-blocks in the n^{th} frame. Here, the value 10 is used for normalization.

Edge indicator: the compressed domain block-edge ratio difference between the n^{th} frame and the $(n-1)^{th}$ frame is defined as follows:

$$edge_n = (4MN)^{-1}(\max(\sum|H_v(n)-H_v(n-1)|,\sum|H_h(n)-H_h(n-1)|)). \quad (3.9)$$

where, $H_v(n)$ and $H_h(n)$ are respectively the numbers of vertical block-edges and horizontal block-edges (defined in [92]) in the n^{th} frame.

Prediction error indicator: the normalized prediction error of the n^{th} frame is given as:

$$err_n = (816C_n)^{-1}\sum_{i=1}^{C_n}|DC(i)|. \quad (3.10)$$

where C_n is the number of non-intra coded 8×8 blocks. The value 816 is used for normalization. $DC(i)$ is the DC coefficient of the prediction error of the Y component.

The mean of prediction errors from the n^{th} frame to the m^{th} frame is defined as:

$$mean\_err(n,m) = \frac{1}{m-n+1}\sum_{i=n}^{m} err_i. \quad (3.11)$$

Inter-frame difference indicator: the normalized YUV histogram intersection between the n^{th} frame and the $(n-d)^{th}$ frame is defined as follows:

$$diff(n,d) = 1-\frac{1}{3}\left(\frac{1}{N_y}\sum_{i=1}^{32}\min(H_Y^n(i),H_Y^{n-d}(i))+\frac{1}{N_u}\sum_{i=1}^{32}\min(H_U^n(i),H_U^{n-d}(i))+\frac{1}{N_v}\sum_{i=1}^{32}\min(H_V^n(i),H_V^{n-d}(i))\right). \quad (3.12)$$

where, $d=9$ is a re-sample step. H_Y^n, H_U^n and H_V^n represent the histograms of the Y, Cb and Cr components of the n^{th} DC image, which are quantized into 32 bins. N_y, N_u and N_v are the corresponding numbers of 8×8 blocks.

3.3.1.2 Cut Detection

To improve the method in [93], the basic cut detector is proposed to reduce the massive false alarms caused by feature peaks at the presence of motions and sudden light changes. Hence, I add a sliding window to choose the cut candidate and the improved method is presented in Figure 3.4. The size of the sliding window varies from 5 frames to 11 frames. In the experiment, it is chosen as 7 frames based on the trade-off of computation load and detection accuracy. Here, D_n and p_n are respectively the luminance and colour indicator of the n^{th} element in the sliding window. β_1, χ_1, δ_1 and ε_1 are the parameters for the elements comparison inside the sliding window and they were chosen as 0.6, 0.08, 0.5, and 0.05, respectively, in the experiments.

IF $D_4 \times \beta_1 > \max(D_3, D_5)$ and $p_4 \times \beta_1 > \max(p_3, p_5)$

THEN a cut candidate is detected;

ELSE IF $p_4 > \chi_1$ **THEN** a cut candidate is detected;

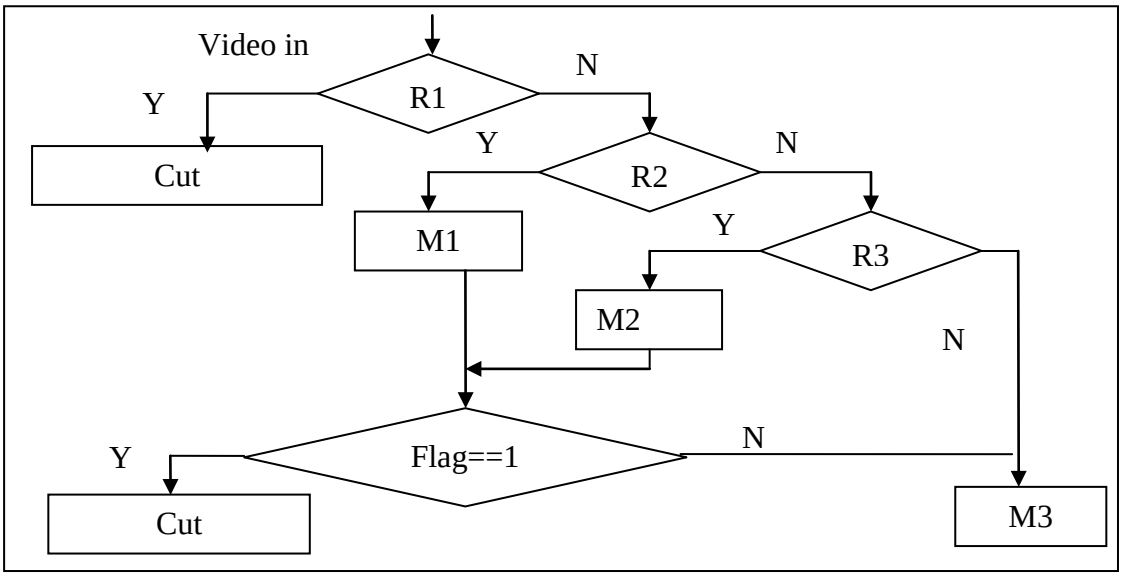
ELSE IF $D_4 \times \delta_1 > \max(D_3, D_5)$ or $D_4 - \varepsilon_1 > \max(D_3, D_5)$

THEN a cut candidate is detected.

Figure 3.4 The basic cut detector

Based on the following observation, edge and motion indicators are applied to design the improved cut detector for robustness. In some cases, a shot cut occurs with indistinct luminance and colour indicators but the edge indicator is dominant in the sliding window. The Edge indicator facilitates the detection of this kind of shot cut missed by the basic cut detector. In other cases, the luminance indicator is dominant due to object or camera movement, so the luminance indicator needs to be verified by a threshold when the

corresponding motion indicator is large enough and triggered by another threshold. Hence, frames with large values of motion indicator but small values of luminance indicator are excluded as false alarms from the cut candidates. Flowcharts of the improved cut detector are given in Figure 3.5.



R1(rule1)	Luminance indicator is larger than 0.3	
R2(rule2)	Luminance indicator is larger than 0.11	
R3(rule3)	Luminance indicator is larger than 0.08	
M1(module1)	Step1	IF luminance indicator is the local peak in the sliding window, THEN go to step2; ELSE flag=0;
	Step2	IF the luminance mean of the other elements in the sliding window is no more than 45 percent of the current luminance peak, THEN go to step3; ELSE flag=0;
	Step3	IF motion indicator is larger than 0.25 and luminance indicator is smaller than 0.2, THEN flag=0; ELSE go to step4;
	Step4	IF luminance indicator and colour indicator satisfy the rules of the basic cut detector, THEN flag=1; ELSE flag=0;
M2(module2)	Step1	IF luminance indicator is the local peak in the sliding window, THEN go to step2; ELSE flag=0;
	Step2	IF the luminance mean of the other elements in the sliding window is no more than 20 percent of the current luminance peak, THEN go to step3; ELSE flag=0;
	Step3	IF motion indicator is larger than 0.25 and luminance indicator is smaller than 0.2, THEN flag=0; ELSE go to step4;
	Step4	IF luminance indicator and colour indicator satisfy the rules of the basic cut detector, THEN flag=1; ELSE flag=0;

M3(module3)	Step1	IF colour indicator is the local peak in the sliding window and its value is larger than 0.9, THEN a cut is detected; ELSE go to step2.
	Step2	IF edge indicator is the local peak in the sliding window and its value is larger than 0.4, THEN a cut is detected; ELSE no cut is detected.

Figure 3.5 The improved cut detector

In the improved cut detector, the luminance indicator is classified into four levels, where the first level is the highest. If the luminance indicator achieves the first level, a cut is detected. Otherwise, if the luminance indicator achieves either the second or the third level, module1 or module2 is triggered respectively for further verification. It is noted that module1 and module2 have similar procedures but employ different values of the percentage parameter for the luminance comparison in the sliding window. This design is based on the observation that when the level of luminance indicator is higher, the luminance peak is more dominant compared to other elements in the sliding window. In both module1 and module2, a sliding window of 7 frames is applied for the luminance peak location and the dominance comparison. Then, if the motion is large enough and triggered by the threshold, the false alarms featured with small values of luminance are excluded. These false alarms are usually caused by object or camera movement. Otherwise, the basic cut detector is applied for further verification. If there is no cut detected so far or the luminance indicator is in the fourth level, module3 is triggered. Module3 detects shot cuts with indistinct luminance change but dominant edge or colour indicators in a sliding window.

3.3.1.3 Gradual Transition Detection

Many existing methods [14, 93] use the intensity variance curve (IVC), which roughly shows a parabolic (‘U’ type) shape when a gradual transition occurs, to locate the shot boundary of the gradual transition. However, IVC can be unsmooth due to motion or

camera flashes which make the 'U' shape indistinct. The shot boundaries detected from the identified parabolic shape of the IVC then become inaccurate. In addition, error propagation caused by misdetection of the previous parabolic shape is unavoidable, and this can affect the further verification process based on the boundaries which have already been located. The twin comparison method proposed in [94] is likely to cause false positives due to camera or object motion, and long gradual transitions may be truncated frequently, which is caused by editing effects.

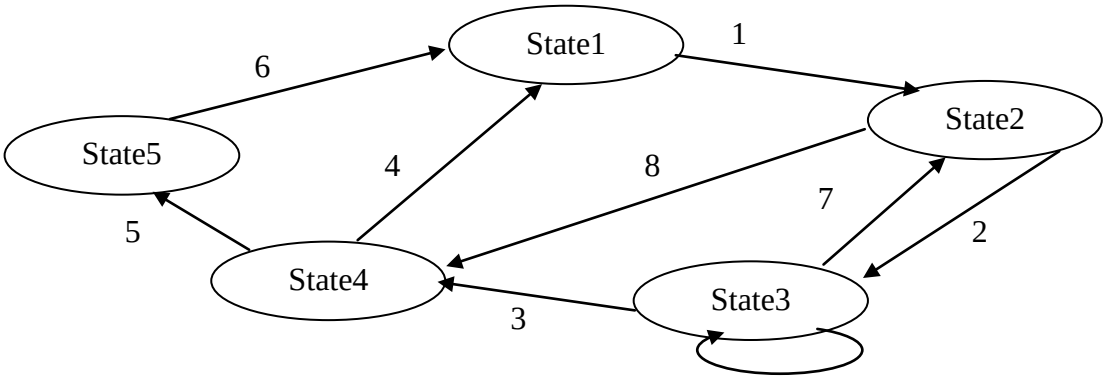
To overcome such shortcomings, the detector proceeds in two steps. Firstly, a pre-processing module acts as a filter to select the promising gradual transition candidate frames after the cut detection for efficiency. Secondly, the FSM is applied to locate the boundaries of the gradual transition candidates and further verify the candidates in two aspects: (i) the colour indicator is employed to detect gradual transitions between shots in distinct colour; (ii) the prediction error indicator is used to determine gradual transitions between shots in similar colour.

By observation, I find that the values of prediction error indicator and the inter-frame difference indicator of gradual transition frames are higher than those of normal frames. In the pre-processing module, if either of these two values is lower than some threshold, there is no gradual transition detected. False alarms caused by motion are filtered out by excluding frames with large values of motion indicator but small values of inter-frame difference indicator. Considering the above principles, the rule based pre-processing module was designed as follows:

Rule1 $IF \quad diff_n < 0.1 \,||\, err_n < 0.015, \quad THEN \quad err_n = 0;$

Rule2 $IF \quad mag_n > 0.2 \,\&\, diff_n < 0.25, \quad THEN \quad err_n = 0;$

where $diff_n$ is the inter-frame difference, err_n is the prediction error, and mag_n is the motion indicator of the n^{th} frame. For frames satisfying either of the two rules, the corresponding prediction error indicators are set to be zero. Then, candidate frames with non-zero prediction error are in clusters, and this helps the FSM to group candidate frames for the location of shot boundaries by reducing the computation cost.



1	IF $err_n > 0 \,\&\, err_{n-1} == 0,$ THEN start-frame=n
2	IF $err_n == 0$
3	IF $err_m == 0,$ m=n+1…n+9, THEN end-frame=n-1
4	IF $p_{n,m} > 0.25,$ n=start-frame, m=end-frame, THEN a gradual transition is detected;
5	IF $p_{n,m} \le 0.25,$ n=start-frame, m=end-frame
6	IF $mean\_err(n,m) > 0.03,$ n=start-frame, m=end-frame THEN a gradual transition is detected, ELSE there is no gradual transition;
7	IF the end-frame is not found
8	IF the length of the gradual transition candidate is longer than 100 frames

Figure 3.6 FSM for gradual transition detection

The algorithm for the location of the shot boundaries and the further verification of the gradual transition candidates is integrated by an FSM. There are five states in the FSM: state 1 is the initial state, when there is no shot boundary; state 2 indicates that the start frame of a gradual transition is found; state 3 indicates the verification of the end frame of the gradual transition; state 4 indicates the verification of the gradual transition with

distinct colour change; state 5 indicates the verification of the gradual transition with indistinct colour change. Details of state transitions in the FSM are shown in Figure 3.6.

When the detection process begins, if the prediction error indicator of the current frame is above zero and that of the previous frame is zero, then the current frame is marked as the start of the gradual transition, and the FSM enters state 2. In state 2, once the prediction error indicator of the n^{th} frame is zero, the FSM enters state 3 to wait for the following 9 frames to verify the end of the gradual transition; or if the length of the gradual transition candidate is longer than 100 frames, the FSM enters state 4. In state 3, if the prediction error indicators of the following 9 frames are all zero, then the $(n-1)^{th}$ frame is marked as the end-frame of the gradual transition, and the FSM enters state 4 for further verification; otherwise, the FSM returns to state 2. In state 4, the colour dissimilarity between the start frame and the end frame is calculated, if the colour dissimilarity is above some threshold, a gradual transition is detected, and the FSM returns to state 1; otherwise, the FSM enters state 5 to verify the shot boundaries between shots in similar colour. In state 5, the mean of the prediction errors from the start frame to the end frame of the gradual transition candidate is calculated, if it is above some threshold, a gradual transition is detected, and the FSM returns to state 1.

3.3.2 Experimental Results

To evaluate the proposed algorithm, extensive experiments were carried out on a number of test video clips from the well-known TREC Video Retrieval Evaluation (TRECVID) activity which is organized by National Institute of Standards and Technology (NIST) annually [95]. Currently the data used are the MPEG-1 format video clips available on the web from the competition in 2001. Accordingly, the results from this year are also used as

benchmarks in the experiments for comparisons. The resolution of the test sequences is 352×240 pixels.

The computing environment used includes: (i) a PC with 1.73GHz CPU, 512MB memory and windows XP operating system; (ii) Microsoft VC++ 6.0 programming platform. To compare the performances of the proposed algorithm with the algorithm from the TRECVID competition team, the following measurements were adopted for quantitative evaluations [96, 97]:

$$\mathrm{Re}\,call = N_C (N_C + N_M)^{-1}, \tag{3.13}$$

$$\mathrm{Pr}\,ecision = N_C (N_C + N_F)^{-1} \tag{3.14}$$

$$F1 = 2 \cdot \mathrm{Re}\,call \cdot \mathrm{Pr}\,ecision (\mathrm{Re}\,call + \mathrm{Pr}\,ecision)^{-1}. \tag{3.15}$$

where N_c is the number of correctly detected shot boundaries, N_M is the number of missed shot boundaries, and N_F is the number of falsely detected shot boundaries. The higher these ratios are, the better the performance.

Table 3.3 Description of the test video sequences

Video name	Number of frames	Cut count	Gradual transition count
Bor08	50569	375	153
Bor03	48451	226	11
Anni005	11364	38	27
Anni009	12307	38	65
In total	122691	677	256

Table 3.3 gives the total numbers of frames, cuts and gradual transitions within each video clip. Table 3.4 and 3.5 illustrate the experimental results from the proposed algorithm and those among the best performing from TRECVID 2001 including teams from Microsoft research, Tsinghua University and the University of Florida [98]. Table

3.6 illustrates the weighted average recall, precision and F1 of cuts and gradual transitions, where the weighted average is calculated via the number of cut count in each sequence.

Table 3.4 Evaluation results of cut: the proposed algorithm V.S. the best results from TRECVID 2001

Video name	Best results from TRECVID 2001			Proposed algorithm		
	Recall	**Precision**	**F1**	**Recall**	**Precision**	**F1**
Bor08	0.965	0.882	0.922	0.899	0.988	0.941
Bor03	0.938	0.954	0.946	0.952	0.989	0.970
Anni005	0.973	0.948	0.960	1	0.844	0.915
Anni009	0.842	0.888	0.864	0.949	0.925	0.937
Average	0.930	0.918	0.923	0.950	0.937	0.941

Table 3.5 Evaluation results of gradual transition: the proposed algorithm V.S. the best results from TRECVID 2001

Video name	Best results from TRECVID 2001			Proposed algorithm		
	Recall	**Precision**	**F1**	**Recall**	**Precision**	**F1**
Bor08	0.758	0.816	0.786	0.940	0.707	0.807
Bor03	0.818	0.281	0.418	1	0.360	0.529
Anni005	0.666	0.782	0.719	0.963	0.900	0.930
Anni009	0.507	0.733	0.599	0.813	0.825	0.819
Average	0.687	0.653	0.631	0.929	0.698	0.771

Table 3.6 Weighted average evaluation results: the proposed algorithm V.S. the best results from TRECVID 2001

Weighted average		Best results from TRECVID 2001	Proposed Algorithm
Cut	**Recall**	0.9495	0.9252
	Precision	0.9101	0.9767
	F1	0.9289	0.9490
Gradual transition	**Recall**	0.6871	0.9128
	Precision	0.7684	0.7424
	F1	0.7156	0.8111

Generally speaking, average recall rate and precision rate provide a reasonable measurement of the overall performance, but it is difficult to find a good trade-off between these two measurements as normally one is higher than the other. Consequently, F1 seems more suitable in such an evaluation. By observing the experimental results, it can be seen that the proposed algorithm outperforms those reported in the TRECVID competition teams in terms of the combined measurement of average recall and precision rate, i.e. F1 measurement.

3.4 Shot Boundary Detection in MPEG Videos using Local and Global Indicators

In this section, my contribution for the submission to TRECVID 2007 on the shot boundary detection task is summarized as: (i) Novel features are extracted from the compressed domain and the feature selection is carried out using Adaboost; (ii) Typical gradual transitions including fade in (out), dissolve and wipes, as well as a new type of gradual transition called "others" are detected. (iii) The MPEG decoding scheme is thoroughly studied so that the whole shot detection system is embedded in the compressed domain of the standard MPEG2 decoder to achieve the real-time efficiency.

3.4.1 Feature Extraction

Given the input of an MPEG video, DC images are extracted for the corresponding frames in the video sequence. In the i^{th} DC image, the n^{th} Y, Cb and Cr components are denoted as Y_i^n, Cb_i^n and Cr_i^n, respectively. Hence, a DC-difference image D between the i^{th} frame and the j^{th} frame is calculated as follows:

$$D_{i,j}^n = \frac{1}{3}\left(\left|Y_i^n - Y_j^n\right| + \left|Cb_i^n - Cb_j^n\right| + \left|Cr_i^n - Cr_j^n\right|\right) \quad (3.16)$$

Furthermore, the mean and standard deviation of D are calculated and denoted as $\mu_{i,j}$ and $\sigma_{i,j}$ respectively. In order to identify the activity of the pixels in D, a classification scheme is implemented as follows:

$$L_{i,j}^n = \begin{cases} 0 \ \ (static), \ \ if \ \ \ 0 \le D_{i,j}^n < T_s \\ 1 \ \ (inactive), \ \ if \ \ \ T_s \le D_{i,j}^n < T_a \\ 2 \ \ (active), \ \ if \ \ \ T_a \le D_{i,j}^n \end{cases} \quad (3.17)$$

$$T_s = \mu_{i,j} - \sigma_{i,j} \tag{3.18}$$

$$T_a = \mu_{i,j} + \sigma_{i,j} \tag{3.19}$$

Therefore, the percentages of active and inactive pixels in D are calculated as R_A and R_I respectively. Since thresholds T_s and T_a for the classification of pixel activity are adaptive to the video content, the percentage R_A and R_I are insensitive to the disturbing factors for shot cut detection, such as camera motion, object movement and sudden light changes.

In the i^{th} DC image, the MPEG motion compensation error err_i is defined in (3.20). Here, N_i is the total number of inter-coded blocks in the i^{th} frame and $DC(j)$ is the j^{th} DC coefficient from the *B, P* frames. In the *I* frames, err_i are copied from those of the latest output *B* frames because there are no inter-coded macro-blocks in *I* frames. In addition, the definition of energy E_i is given in (3.21), where N_y is the total number of pixels in the i^{th} DC image.

$$err_i = \frac{1}{N_i}\sum_{j=1}^{N_i} DC(j) \tag{3.20}$$

$$E_i = \frac{1}{N_y}\sum_{n=1}^{N_y}\left(Y_i^n\right)^2 \tag{3.21}$$

3.4.2 Feature Validation and Shot Cut Detection

AdaBoost [99] is utilized to verify the discrimination power of the selected features. The extracted features $\mu_{i,j}, \sigma_{i,j}, R_A, R_I$ and err_i constitute a five dimensional feature set S_1. In addition, some existing features are extracted to constitute a six dimensional feature set S_2, which includes luminance, colour, motion magnitude, edge and inter-frame differences. A

sliding window of the chosen features is employed to measure the temporal differences for shot cuts. There are two choices for the size of the sliding window, 3 frames W_3 and 11 frames W_{11}. Hence, the feature set are divided into three groups: $(S_1 + S_2) \times W_{11}$, $S_1 \times W_{11}$ and $S_1 \times W_3$. In the AdaBoost training stage, a five-fold cross-validation scheme is utilized, based on the test videos from TRECVID in 2006 and 2005 with manually labelled shot boundaries. The training results are presented in Table 3.7. Here, the first test uses 11 features combining S_1 and S_2 with a sliding window of 11 frames. The second test uses features S_1 with a sliding window of 11 frames. The third test uses features S_1 with a sliding window of 3 frames. From the detection rates, it is found that the third choice of feature selection produces good performances. In addition, it has the advantage of low feature dimensions.

Table 3.7 Performance comparison using AdaBoost based cross validation on the data from TRECVID in 2005 and 2006

Experiments	**Test 1**	**Test 2**	**Test 3**
Feature dimension	11×11	5×11	5×3
Average recall	0.9709	0.9718	0.9716
Average precision	0.9702	0.9706	0.9705
Description	$(S_1 + S_2) \times W_{11}$	$S_1 \times W_{11}$	$S_1 \times W_3$

By modelling the visual features of the shot cuts into five kinds, cut detection is implemented in a decision-making process. For further details please refer to the paper [P1] in appendix.

3.4.3 Detecting Gradual Transitions

In this section, the detection of typical gradual transitions including fade in (out), dissolve and wipes, as well as a new type of gradual transition called “others” are described.

3.4.3.1 Detecting Dissolves and Wipes

One of the spatial features widely used for dissolve detection is the intensity variance curve (IVC), which often has an approximately parabolic ('U' type) shape when a dissolve occurs. However, the U shape IVC is often corrupted in reality due to motion, camera flash, and many other factors. Consequently, it is difficult in practice to capture such transition processes. Looking for an alternative feature, presenting a stronger indication of dissolves, a range of possibilities was tested in the experiments, such as luminance, colour and edge features. MPEG motion compensation error err_i was found to be the best choice. In comparison with IVC, the MPEG motion compensation error indicator presents two advantages: (i) it can be readily extracted from MPEG compressed domain; (ii) it presents a sequence of peaks during dissolve transitions and thus can be exploited to detect dissolves.

To optimize the exploitation of the MPEG motion compensation error indicator, a two-step procedure is proposed for dissolve detection, where the first step is to pre-process the video sequence and remove those unlikely to be dissolve candidates, and the second step is to apply a cluster procedure to detect and verify the dissolves.

During a dissolve, frames within a dissolve actually present large inter-frame differences as well as MPEG motion compensation errors. Therefore, the following pre-processing is defined to remove those frames that are not likely to be dissolve candidates:

$$if \qquad \mu_{i,i+d} < T_\mu \,||\, err_i < T_e \qquad then \quad err_i = 0 \tag{3.22}$$

Here, T_μ and T_e are thresholds. In the experiment, both $\mu_{i,i+d}$ and err_i were scaled in the range from 0 to 1. The thresholds T_μ and T_e are determined by applying the principle of

maximum entropy to the labelled dissolves and normal frames from the training data. In the experiment, T_μ and T_e are set as 0.14 and 0.02, respectively.

Following the pre-processing, the remaining operation is focused on those frames with non-zero err_i values to detect candidates for dissolves under the principle that dissolves present a sequence of peak values in err_i. A cluster procedure is proposed to group these frames, the details of which are as follows:

- Input G_i: candidate frames with non-zero err_i values in a circular buffer.
- Output: dissolve candidates
- Procedure:

  Step 1: Select the frame F_i in G_i, with smallest frame number as the seed for cluster C_i, and subtract F_i from G_i. If the last frame in the buffer is reached, wait for buffer to fill, until the whole video sequence is processed.

  Step 2: If there exists another candidate frame F_j in the rightwards neighbourhood of d frames, absorb frame F_j in cluster C_i. Subtract F_j from G_i.

  Step 3: Iteratively execute step 3, until there are no candidate frames in the rightwards neighbourhood of d frames. As a result, cluster C_i constitutes a dissolve candidate. Go to step 1.

In the cluster procedure, d represents the scale of the neighbourhood and it is set as 5 in the experiment. This is because typical dissolves are longer than 5 frames. The use of a circular buffer to monitor the frames to be clustered enables the real-time detection of dissolves. Given the beginning and ending frames of the dissolve candidate as F_b and F_e,

respectively, a temporal verification is carried out by a thresholding process as follows. This represents the fact that longer dissolves have larger temporal differences.

$$if \quad \mu_{b,e} > \frac{(e-b)}{d} \cdot T_{\mu}, \quad then \quad dissolve \quad found \qquad \textbf{(3.23)}$$

In the experiment, it was found that the proposed scheme for dissolve detection is also effective for the detection of typical wipes.

3.4.3.2 Detecting Fades

A typical fade combines with three different stages of transition: fade out, a serial of stable monochrome frames and fade in. In the temporal intensity domain, the fade transition is featured with either of the shapes illustrated in the figure 3.7. In the temporal variance domain, the fade transition is featured with the shape illustrated in figure 3.8. Hence, the variance of the pixel intensity in the DC image is a reliable indicator for the detection of fades. Fade out is modelled as a variance decreasing process while fade in is modelled as a variance increasing process. Therefore, the proposed knowledge-based rules are described as follows to facilitate the detection of fades. A fade is detected when the test for all the six rules are positive.

- Rule 1: if $V_i - V_{i+d} \geq 0$ then candidate fade out frames are found. Here, V_i is the variance of the i^{th} frame. The frame step d is introduced for robustness, since the variance curve of fade out may not be strictly decreasing due to the effect of noise. In the experiment, d is set as 2.
- Rule 2: if Rule 1 is satisfied for a number of n frames, a fade out candidate is claimed.

- Rule 3: if Rule 2 is satisfied, a sequence of stable monochrome frames need to be identified. If $V_i < T_m$, candidate monochrome frames are found.
- Rule 4: if Rule 3 is satisfied for a number of n frames, a stable stage is claimed.
- Rule 5: if Rule 4 is satisfied, the fade in process needs to be monitored. If $V_{i+d} - V_i \geq 0$ then candidate fade in frames are found.
- Rule 6: if Rule 5 is satisfied for a number of n frames, a fade in candidate is claimed.
- Rule 7: if Rule 6 is satisfied, a temporal verification is carried out. The averaged variance of the monochrome frames is calculated as V_m. The variance of the first frame of fade out is denoted as V_b. The variance of the last frame of fade in is represented as V_e. If $\frac{V_m}{V_b} < T_v \,\&\, \frac{V_m}{V_e} < T_v$, a fade is claimed.

In the experiment, the variance is scaled in the range from 0 to 1. n is a measurement of the length of fade and it is determined as 5, according to the minimum length of fade in real cases. T_m controls the variance of monochrome frames. By the visual observations on training frames, it is acceptable that the variance of monochrome frames is under 5%. T_v represents the temporal difference of fade, it is decided as 20% by the empirical study on the training data.

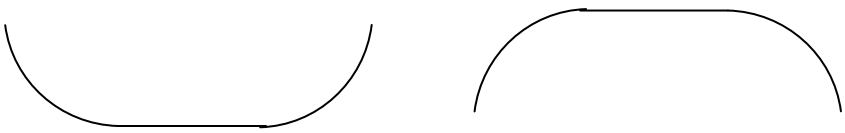


Figure 3.7 The shapes of fades in the Temporal domain of intensity

Figure 3.8 The shape of fades in the Temporal domain of variance

3.4.3.3 Detecting “others”

A new type of gradual transition is defined as “others” by TRECVID in the challenge of difficult cases in practical video sequences. The category “other” is a kind of combined shot boundaries. In the case of “other”, there is a shot cut at the i^{th} frame and another shot cut at the j^{th} frame, with a sequence of monochrome frames in between. Since the two shot cuts are quite near each other and there is no meaningful video content in between, it can be seen as a whole transition process. In the temporal domain of $\mu_{i,i+1}$, this process is featured with the shape illustrated in Figure 3.9. It can be seen that the beginning and ending points are like impulses while the middle parts are almost static. Based on the detection of monochrome frames in section 3.4.3.2 and the detection of shot cuts in section 3.4.2, “others” are identified by the fusion of previous detection results.

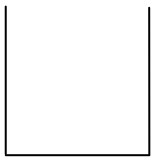


Figure 3.9 The shape of fades in the Temporal domain of intensity.

3.4.4 Evaluation Result

In the shot boundary detection task of TRECVID 2007, 35 teams participated and 15 teams finished the task with 128 runs for submission. The submission was evaluated by TRECVID and its performance is summarized as follows [95]:

- In terms of the cut detection, the submission is ranked as number 1.
- In terms of gradual transition detection, the submission is ranked as number 6.
- In terms of gradual transition frame accuracy, the submission is ranked number 7.
- In terms of overall evaluation, the submission is ranked as number 3.

3.5 A Hierarchical Content-Aware Approach for Shot Cut Detection in Compressed Domain

In this section, a content-aware approach is proposed, with multiple feature indicators and multiple thresholds to detect shot boundaries. The proposed approach extracts all features in MPEG compressed domain, and organizes all operations into a decision tree for cut detection and a finite state machine (FSM) for dissolve detection.

3.5.1 Content-aware Construction of Multiple Content Difference Indicators with Multiple Thresholds

Given an MPEG compressed video input, a DC image sequence $\{Y_1, Y_2, ...Y_n, ...Y_{end}\}$ is extracted from the corresponding video frames. If the original video frame size is $W \times H$, the DC image will have the size of $\frac{W}{8} \times \frac{H}{8}$. Around the current DC frame Y_n, which is to be examined for shot cut detection, I define a shifting window with 11 neighbouring DC frames to test all the features extracted and determine whether there exists a cut or not between the frame Y_{n-1} and the frame Y_n. The window function is defined as:

$$w(n-m) = \begin{cases} 1 & -5 \le m \le 5 \\ 0 & else \end{cases}. \qquad \textbf{(3.24)}$$

where $n \in [1, end]$.

In other words, the proposed shot cut detection is essentially carried out inside the window set $\{Y_{n-5}, Y_{n-4}, Y_{n-3}, Y_{n-2}, Y_{n-1}, Y_n, Y_{n+1}, Y_{n+2}, Y_{n+3}, Y_{n+4}, Y_{n+5}\}$.

To detect shot cuts, most of the existing work reported in the literature measure the content difference between the current frame Y_n and its preceding frame Y_{n-1}, and then

apply a threshold to decide whether the difference measured is large enough to justify a cut detection [21, 23, 100]. Many algorithms have been developed and reported on how to decide such a threshold, which can be briefly summarized as: (i) statistics based approaches such as Bayesian rules, maximum likelihood based probabilistic modelling etc. [8, 24]; (ii) empirical studies [21, 23]; (iii) machine learning approaches such as SVM, neural networks etc to bypass the threshold and make decisions based on the training and learning process [7, 9, 10]. The fundamental issue here, however, is that such a measured neighbourhood frame difference indicator alone would not provide sufficient information for shot cut detection. In practice, many cuts do not necessarily generate sufficiently high peak values for this neighbourhood frame difference. Such a situation makes it impossible to apply one threshold and get all the cuts detected. As an example, Figure 3.10 shows the values of such neighbourhood frame differences at the locations of all the shot cuts for the video sequence 20051205-185800-PHOENIX-GOODMORNCN-CHN from TRECVID 2005, in which all the cuts are indicated by a negative peak value of -0.05. As can be seen, to ensure that all the cuts are detected, the threshold value should be as low as 0.025 since the value of neighbourhood frame difference at the 27^{th} cut (close to frame-1400) is around 0.029. If I apply this value as the threshold, however, many false positives would have been generated. On the other hand, to ensure that no false positive is generated, the threshold value should be set as high as 0.175 (see the non-cut peak value around frame-1250). Yet if I use this value as the threshold, I can only manage to detect 5 shot cuts, generating many false negatives.

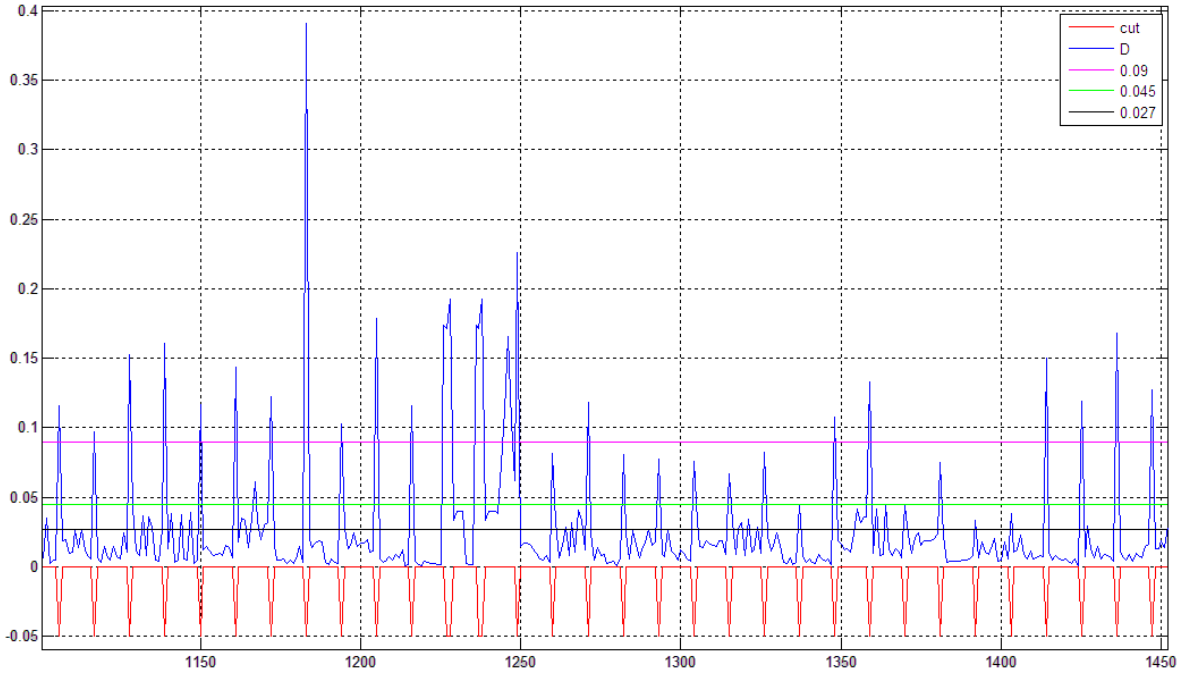


Figure 3.10 Luminance neighbourhood frame difference indicator (*D*) with respect to ground truth cuts

The above empirical analysis reveals that: (i) I need to apply multiple thresholds and examine multiple indicators in order to ensure that true cuts are differentiated from those non-cuts; (ii) to optimize the application of multiple thresholds, I need to be aware of the contexts under which the frame difference indicators are generated, and thus different contexts should have different thresholds applied.

To this end, three frame difference indicators are proposed, including: (i) neighbourhood frame difference indicators to measure the content difference between the current frame Y_n and its preceding frame Y_{n-1}; (ii) inter-frame difference indicators to measure the content difference between the current frame Y_n to be examined and its preceding 9^{th} frame Y_{n-9}; and finally (iii) comparative frame difference indicator to measure the difference of all the indicators inside the shifting window as defined in (3.24). While the first frame difference indicator follows the same principle adopted by all

existing work [3, 4, 7, 16-22, 25, 89-93, 101] that, if there exist a cut at the current frame, there should exists some content difference between the two neighbouring frames, the second frame difference indicator is mainly used to verify such a possible cut by examining the difference between the current frame and a frame some distance away [21, 102] to overcome the false positives caused by factors other than cuts, such as motion, camera movement, or editing effect etc. Selection of the preceding 9^{th} frame is just one of the reasonable choices which could be made, and any other neighbouring frame will be sufficient to serve the same purpose, as long as it is located within the same shot as Y_{n-1}. The third frame difference indicator is to compare the indicators within a shifting window to test the consistency and remove the false negatives for cases where some cuts may present small content differences.

To measure the three frame difference indicators, the proposal is to extract four features to construct the neighbourhood frame difference indicator and fully exploit MPEG compression techniques to enable shot cut detection to be carried out in the compressed domain rather than the pixel domain. These four features are luminance, colour, edge, and motion, the details of which are as follows.

Based on the work described in [103], a normalized luminance feature is proposed to make it convenient for evaluation of all the features in a systematic and unified way. Such a normalized luminance difference between the n^{th} frame and the $(n-1)^{th}$ frame is represented as:

$$D_n = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left|y_n(i,j) - y_{n-1}(i,j)\right|}{M \times N \times 255}. \qquad \textbf{(3.25)}$$

where M and N are the number of 8×8 blocks inside video frames along the vertical direction and horizontal direction, respectively, $y_n(i,j)$ is the DC luminance value of the block positioned at (i,j) inside the n^{th} DC frame, and $D_n \in [0,1]$.

To extract the colour feature in compressed domain, a normalized colour histogram correlation between the n^{th} frame and the $(n-1)^{th}$ frame is used, defined as:

$$p_n = 1 - \frac{\sum_{i=1}^{8}\sum_{j=1}^{8} H_n(i,j)\cdot H_{n-1}(i,j)}{\sqrt{\sum_{i=1}^{8}\sum_{j=1}^{8} H_n(i,j)^2 \cdot \sum_{i=1}^{8}\sum_{j=1}^{8} H_{n-1}(i,j)^2}}. \qquad (\mathbf{3.26})$$

where $H_n(i,j)$ is the two dimensional 8×8 colour histogram in the n^{th} DC image, from which the components of U and V are both placed into 8 bins respectively. To produce the histogram $H_n(i,j)$, all the DC values inside the U and V blocks are quantized into 8 levels and thus the histogram would have 64 bins altogether.

Considering the work on block-based edge detection directly carried out in the compressed domain [104], I propose the following edge ratio difference between the n^{th} frame and the $(n-1)^{th}$ frame as defined below:

$$\gamma_n = \frac{1}{M\times N}\left(\max\left(\left|N_v^e(n) - N_v^e(n-1)\right|, \left|N_h^e(n) - N_h^e(n-1)\right|\right)\right). \qquad (\mathbf{3.27})$$

where $N_v^e(n)$ and $N_h^e(n)$ are the number of vertical block-edges and horizontal block-edges, respectively in the n^{th} frame.

As MPEG makes the motion information available in the compressed domain, I extract a normalized motion feature based on the MPEG motion vector $\overrightarrow{M_n} = \left(V_x(i,j), V_y(i,j)\right)$ in the n^{th} frame as:

$$\left|\overrightarrow{M}\right|_n = \max\left(\sum_{i,j}\left|\frac{V_x(i,j)}{T_{vx}}\right|, \sum_{i,j}\left|\frac{V_y(i,j)}{T_{vy}}\right|\right). \quad \textbf{(3.28)}$$

where (T_{vx}, T_{vy}) is the maximum allowable motion vector designed by MPEG.

Consequently, the four features defined in (3.25-3.28) can be applied as content features, which can be directly used to indicate the content difference and thus detect the shot cuts, or context features, which can be used to indicate the contexts of the content changes. For example, the luminance and colour can be readily used as content features since both of them are primarily used to represent the visual information in all image generation processes (such as TV, cameras, printing etc.). Yet motion and edges can be used as context features since both of them mainly reflect the activities inside the captured visual scenes. In this way, shot cut detection can be made adaptive to the context changes as well as content changes. When motion is high, for example, it indicates that proportional content difference is caused by motion rather than by cuts, and thus the threshold should be moved higher. To this end, a training video set has assembled, drafted from the TRECVID test sequences in 2001 and 2005, and carrying out empirical studies by extracting the neighbourhood frame difference indicators for all the four features to determine the multiple thresholds.

By analyzing the value of neighbourhood frame difference indicators in correspondence with the ground truth cuts inside the training set, I am able to determine two threshold values for the content features, where one is the lowest possible threshold, under which no cut could possibly exist, and the other is the highest possible threshold, above which a maximum number of false positives are eliminated. For major content

features such as luminance, one or two medium thresholds are needed to fine tune the decision process, which can be determined as:

$$T_M = \begin{cases} \dfrac{T_H}{2} & \textit{if one medium threshold is needed} \\ T_{M1} = \dfrac{2T_H}{3}; T_{M2} = \dfrac{T_{M1} - T_L}{2} & \textit{if two thresholds are required} \end{cases}. \quad \textbf{(3.29)}$$

where T_L and T_H are the lower threshold and higher threshold respectively, which are determined via empirical approaches from the training video sequences, T_M is the medium threshold if only one medium threshold is needed, and T_{M1} and T_{M2} are the lower medium threshold and higher medium threshold respectively for cases where two medium thresholds are needed.

As an indicator for content activities rather than content itself, edge is used as a context feature. Hence, a certain threshold is observed and determined for the neighbourhood frame difference indicator of edge, at all locations of the ground truth cuts inside the training sequences, which serves as a necessary condition for cut detection.

The Motion feature is also used as a context feature in the proposed algorithm, for which the main purpose is to determine whether the extracted peak value of neighbourhood frame difference, such as D_n and p_n, is caused by motion or cuts. As a result, I need to determine a lower threshold, under which the incurred motion is too small to cause any concern. Similarly, a higher threshold also needs to be determined, above which all motion indicators should cause concerns. Furthermore, a medium threshold can be determined as given in (3.29) in order to ensure that motion indicators could be further classified into different regions to provide sufficient contexts and discriminating power.

The inter-frame difference indicator is designed to check whether there exists any significant difference between the frame under examination and its preceding 9th frame. This is essentially a verification test for shot cut detection. For the current frame Y_n, for example, any peak value of its neighbourhood frame difference indicator should be verified by another peak value of its inter-frame difference indicator between Y_n and Y_{n-9}. This follows the principle that the existence of a cut should cause significant content change between Y_n, the first frame of the next shot, and all the frames inside the current shot, including Y_{n-1}, Y_{n-2}, ... Y_{n-9} …. In contrast, if the peak value of neighbourhood frame difference at Y_n is accompanied by another peak value of inter-frame difference at Y_{n-1}, it is likely that such content change is caused by factors other than cuts.

To measure the inter-frame difference in the current DC frame Y_n, a normalized histogram is introduced, with 32 bins for all YUV components. By representing such normalized YUV histograms as $H_{Yn} = \left(h_{Yn}^1, h_{Yn}^2, ...h_{Yn}^{32}\right)$, $H_{Un} = \left(h_{Un}^1, h_{Un}^2, ...h_{Un}^{32}\right)$, $H_{Vn} = \left(h_{Vn}^1, h_{Vn}^2, ...h_{Vn}^{32}\right)$ respectively, the inter-frame difference indicator can be defined as:

$$\Delta_n = 1 - \frac{1}{3}\left(\frac{1}{N_y}\sum_{i=1}^{32} \min\left(h_{Yn}^i, h_{Y(n-9)}^i\right) + \frac{1}{N_u}\sum_{i=1}^{32} \min\left(h_{Un}^i, h_{U(n-9)}^i\right) + \frac{1}{N_v}\sum_{i=1}^{32} \min\left(h_{Vn}^i, h_{V(n-9)}^i\right) \right). \quad \textbf{(3.30)}$$

where N_y, N_u and N_v are the corresponding number of 8×8 blocks for Y, U and V components respectively.

Similarly, to ensure that all cuts can be differentiated by a peak value of the inter-frame difference indicator, a lower threshold value is determined by considering all the peaks with respect to the ground truth cuts inside the training set. To ensure that all peaks do not generate any false positive, a higher threshold value is determined. Based on these

two thresholds, more contexts can be identified to fine tune the cut detection by determining two medium thresholds as described in (3.29). Details of all the multiple threshold values for the two frame difference indicators are summarized in Table 3.8.

Table 3.8 Summary of multiple thresholds for neighbourhood frame difference and inter-frame difference indicators

Content feature	Lower threshold (T_L)	Higher threshold (T_H)	Medium threshold (T_M)
D_n (luminance)	0.027	0.09	0.045
p_n (colour)	0.025	0.09	-
γ_n (edge)	0.02	-	-
$\left\|\overrightarrow{M_n}\right\|$ (motion)	0.06	0.2	0.1
Δ_n (inter-frame difference)	0.14	0.3	(0.17, 0.2)

Table 3.9 Comparative frame difference indicators

Comparative frame difference indicators	Definitions	
Comparative luminance higher peak	$\hat{D}_n = \begin{cases} 1 & if\ D_n w(n) > \alpha \max\limits_{m\in[-5,5],m\neq 0}\left(D_{n-m} w(n-m)\right) \\ 0 & else \end{cases}$	(3.31)
Comparative luminance lower peak	$\hat{D}_n = \begin{cases} 1 & if\ D_n w(n) > \alpha \max\limits_{m\in[-3,3],m\neq 0}\left(D_{n-m} w(n-m)\right) \\ 0 & else \end{cases}$	(3.32)
Comparative colour peak	$\hat{p}_n = \begin{cases} 1 & if\ p_n w(n) > \alpha \max\limits_{m\in[-5,5],m\neq 0}\left(p_{n-m} w(n-m)\right) \\ 0 & else \end{cases}$	(3.33)
Comparative edge peak	$\hat{\gamma}_n = \begin{cases} 1 & if\ \gamma_n w(n) > \beta \max\limits_{m\in[-5,5],m\neq 0}\left(\gamma_{n-m} w(n-m)\right) \\ 0 & else \end{cases}$	(3.34)
Comparative inter-frame peak	$\hat{\Delta}_n = \begin{cases} 1 & if\ \Delta_n w(n) > \beta \max\limits_{m\in[-5,5],m\neq 0}\left(\Delta_{n-m} w(n-m)\right) \\ 0 & else \end{cases}$	(3.35)

Generally, if there exists a cut between frames Y_n and Y_{n-1}, all the frames inside the shifting window would be classified into two different shots, where the first shot contains frames $Y_{n-m} w(n-m)$, $m \in [1, 5]$, and the second shot contains frames $Y_{n-m} w(n-m)$, $m \in [-5, 0]$. Consequently, the concept of comparative frame difference indicators is introduced to make sure that suspicious candidates can be further confirmed

and screened whether they are true positives or not. Details of their definitions for frame Y_n are summarized in Table 3.9.

The two parameters, α and β, are used to determine the minimum peak value to allow the extracted feature to be regarded as a comparative peak inside the window. Their values are normally within the range of [1,5]. In this algorithm, $\alpha = 2.5$ and $\beta = 1.8$ are selected. Apparently, equations (3.31-3.35) indicate that the larger the values of α and β selected, the higher the comparative peaks are required to confirm the cut detection.

3.5.2 Cut Detection

With the proposed multiple frame difference indicators and thresholds, the shot cut detection is designed in a coarse-to-fine manner with three phases. The first phase is an initial shot cut detection, in which the aim is filtering through all suspicious candidates that could be cuts. Following that, two further phases process its outputs. While the second phase is to process all those cut candidates to remove as many as possible of the false positives, the third phase is designed to process those non-cut candidates to remove as many as possible of the false negatives.

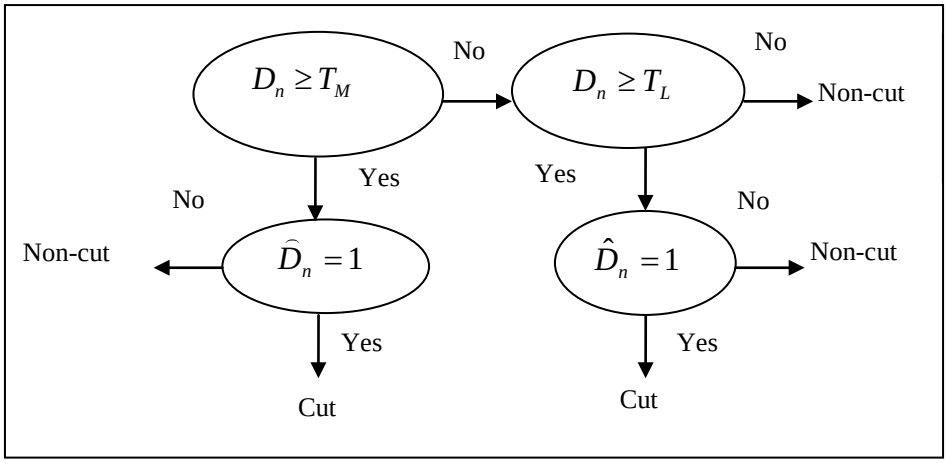


Figure 3.11 Decision tree for initial shot detection

Given the current DC frame Y_n, its luminance content feature D_n is primarily used for the first phase cut detection. Since the major aim of this first phase is to detect as many cuts as possible, only the lower threshold and the medium threshold in Table 3.8 are used to process the luminance neighbourhood frame difference D_n. The entire detection process is summarized into a decision tree as shown in Figure 3.11.

As can be seen, the decision is made by examining D_n against both medium threshold and lower threshold supported by consistency tests via the comparative frame difference indicators defined in (3.31) and (3.32).

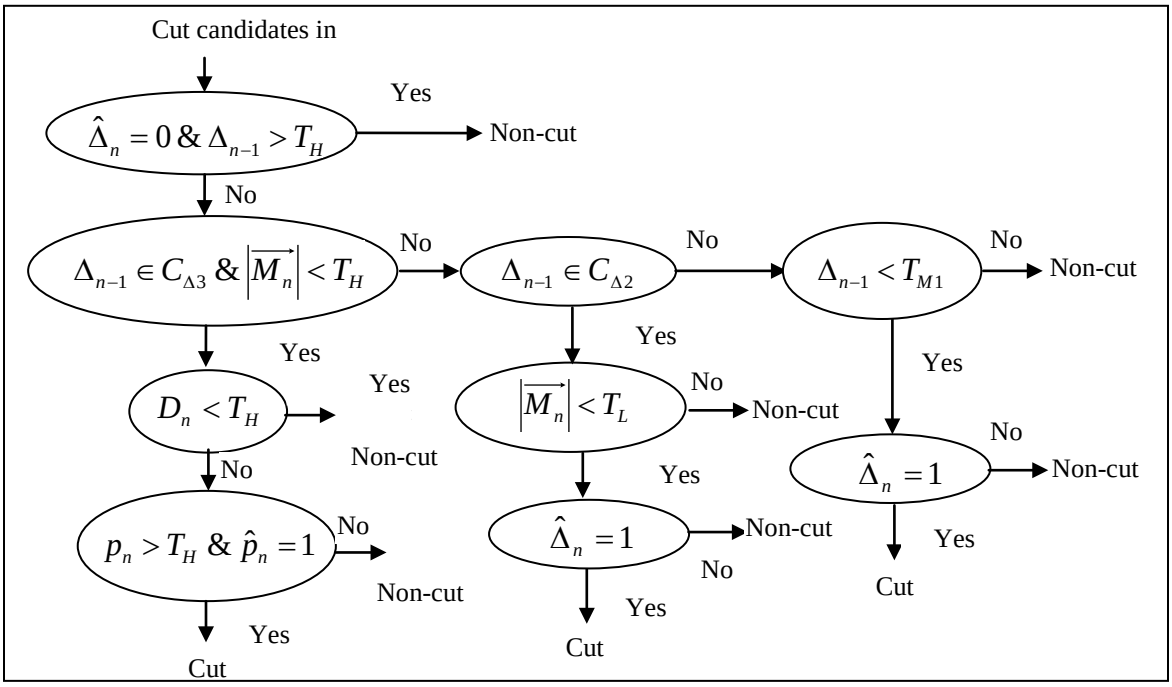


Figure 3.12 The second phase: decision tree for removal of false positives

In the second phase, the primary aim is to remove false positives by applying the principle that, if a peak value detected at Y_n in the initial phase is accompanied by another inter-frame difference peak at Y_{n-1}, the peak difference detected in the first phase at Y_n is likely caused by factors other than a cut. As a result, the inter-frame difference indicators

play leading roles in the second phase detection, and the entire process is structured into another decision tree as illustrated in Figure 3.12.

As seen, satisfaction of the first condition $\hat{\Delta}_n = 0 \,\&\, \Delta_{n-1} > T_H$ establishes that the peak frame difference detected in the first phase is not caused by a true cut, since it is accompanied by a high inter-frame difference between Y_{n-1} and Y_{n-10}, yet there exists no comparative peak at Y_n. As a result, the input cut candidate is detected as a false positive.

Non-satisfaction of the first condition leads to further examination of Δ_{n-1} across all the remaining regions divided by its multiple thresholds as shown in Table 3.8. Figure 3.12 shows that the remaining tests of Δ_{n-1} are arranged in terms of $C_{\Delta 3} = [T_{M2}, T_H)$, $C_{\Delta 2} = [T_{M1}, T_{M2})$, and $\Delta_{n-1} < T_{M1}$, respectively. Since all these regions have different strengths in indicating the inter-frame difference at Y_{n-1}, I need to use other features to indicate its contexts and complete the false positive detection.

Furthermore, in the test: $\Delta_{n-1} \in C_{\Delta 3} \,\&\, \left|\overrightarrow{M_n}\right| < T_H$, $\left|\overrightarrow{M_n}\right| < T_H$ is a context condition to improve the strength of $\Delta_{n-1} \in C_{\Delta 3}$. In other words, if $\Delta_{n-1} \in C_{\Delta 3}$ is true and meanwhile the motion feature is less than the higher threshold, indicating that motion is not sufficient to cause a peak value between Y_n and Y_{n-1}, it is likely that the initially detected cut could still be a false positive. This is verified by $D_n < T_H$, as it indicates that the peak in the neighbourhood frame difference detected in the first phase and is not strong enough to justify that this is a true cut.

The test $p_n > T_H$ & $\hat{p}_n = 1$ along the decision tree in Figure 3.12 is the strongest indication for a colour peak at Y_n, which is used to maintain that the input cut candidate is indeed a cut. Otherwise, it is detected as a false positive.

A positive test on $\Delta_{n-1} \in C_{\Delta 2}$ indicates that a peak value is still detected at Y_{n-1}, suggesting that the input cut candidate could be a false positive. Since such a peak is relatively weak in comparison with $\Delta_{n-1} \in C_{\Delta 3}$ and $\Delta_{n-1} > T_H$, a stronger condition $\left|\overrightarrow{M_n}\right| < T_L$ is required to make sure that the motion extracted at Y_n is definitely unable to make any contribution to the high frame difference. Therefore, its positive test suggests that the initially detected cut candidate is likely to be a true cut. To verify this judgment, another strong condition $\hat{\Delta}_n = 1$ is tested to indicate that the inter-frame difference indicator at Y_n is the largest in comparison with all inter-frame difference indicators inside the shifting window, which overrules $\Delta_{n-1} \in C_{\Delta 2}$, and hence the input cut candidate should be decided as a true positive.

Along the decision tree in Figure 3.12, the final test $\Delta_{n-1} < T_{M1}$ indicates that the frame difference between Y_{n-1} and Y_{n-10} is relatively small, suggesting that the candidate cut detected in the first phase is likely to be a true positive, which is verified by $\hat{\Delta}_n = 1$.

To remove false negatives in the third phase, it is proposed to use the strongest possible test for a cut. If the input non-cut candidate satisfies such a strongest possible condition, it is likely that the input is a false negative. The entire decision tree for the third phase false negative detection is illustrated in Figure 3.13 where the strongest test is defined as:

$$\left(D_n > T_H \,||\, \hat{D}_n = 1\right) \& \; p_n > T_L \,\&\, \gamma_n > T_L \,\&\, \hat{\Delta}_n = 1 \& \; \hat{r}_n = 1 \& \; \hat{p}_n = 1. \qquad \textbf{(3.36)}$$

As can be seen, the condition is to test that, if all content features including neighbourhood frame difference and comparative inter-frame difference for colour, luminance, and edges are high, it is likely that the non-cut candidate could be a false negative. To verify such a possibility, two context tests are carried out in Figure 3.13. Firstly, if the comparative inter-frame difference at its preceding frame Y_{n-1} is relatively low by testing $\Delta_{n-1} < T_L$, this confirms that all the peaks indicated by (3.36) are not accompanied by any inter-frame difference peak at Y_{n-1}, and thus the non-cut candidate should be declared a false negative and hence detected as a cut instead. Otherwise, I further test the context: $\left|\overrightarrow{M_n}\right| < T_M$ to see if all the peaks indicated by (3.36) are caused by motion or not. If the test is negative, indicating that motion is sufficiently strong to account for all the peaks at Y_n, the input non-cut candidate is maintained as a non-cut.

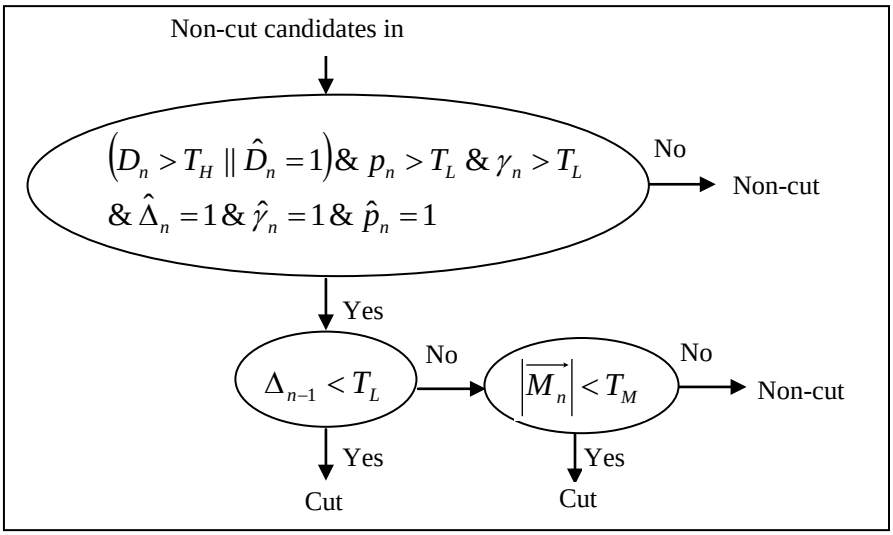


Figure 3.13 The third phase: decision tree for removal of false negatives

3.5.3 Dissolve Detection

Many existing methods for dissolved cut detection [14] use the intensity variance curve (IVC), which shows a roughly parabolic ('U' type) shape when a gradual transition occurs, to locate the shot boundary of the gradual transition in dissolves. The principle behind it is that a gradual transition incurs gradual increase and gradual decrease inside the intensity variance. However, the U shape inside IVC is often corrupted in reality due to motion, camera flash and many other factors. Consequently, it is difficult in practice to capture such transition processes, or in other words, the transition process is not sufficiently clear to be captured by the intensity variance curve. As a result, the detected shot boundaries based on such an ambiguous parabolic shape of IVC become inaccurate. In addition, misdetection of such parabolic shapes could cause error propagation, producing negative impact upon detection of other dissolves. Figure 3.14 illustrates an example of IVC for the video sequence 20051227-125800-CNN-LIVEFROM-ENG from TRECVID 2005, from which it can be seen that the parabolic shape is not sufficiently clear and thus making it difficult to detect dissolves accurately in many practical cases. Looking for an alternative feature, presenting a stronger indication of dissolves, a range of possibilities were tested and a new feature, MPEG motion compensation error indicator is proposed, which is defined as follows:

$$err_n = \frac{1}{C_n \times \sigma} \sum_{i=1}^{C_n} \left| DC(i) \right|. \qquad (\mathbf{3.37})$$

where C_n is the number of inter-coded blocks, and σ is the threshold applied by MPEG to decide whether a block should be inter-coded or not.

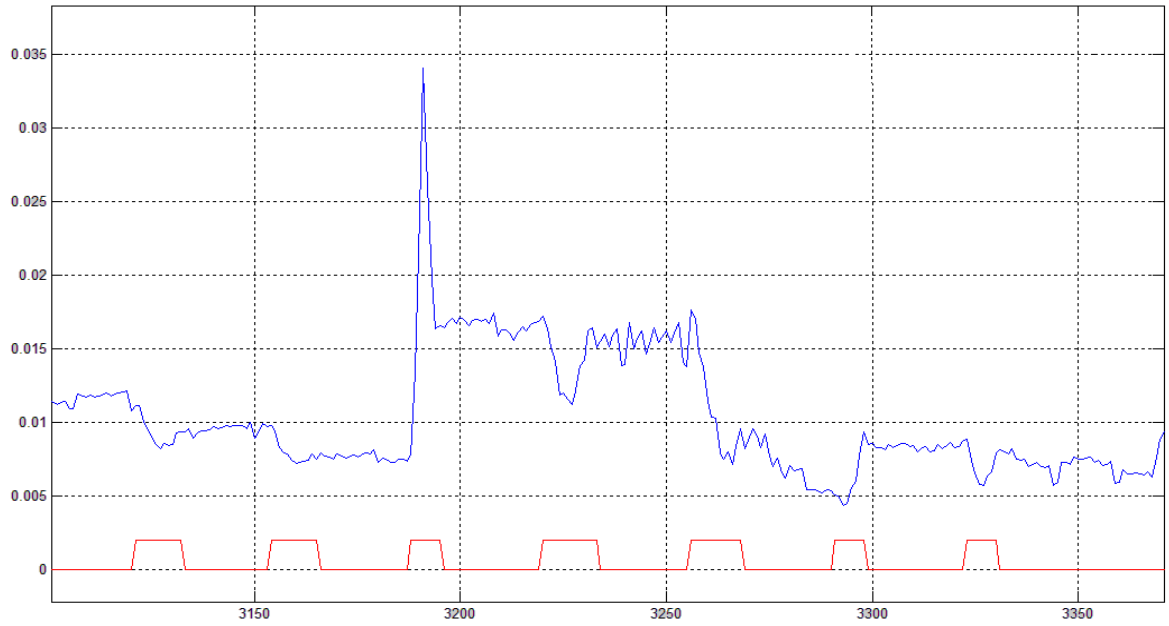


Figure 3.14 Example illustration of IVC with respect to dissolves

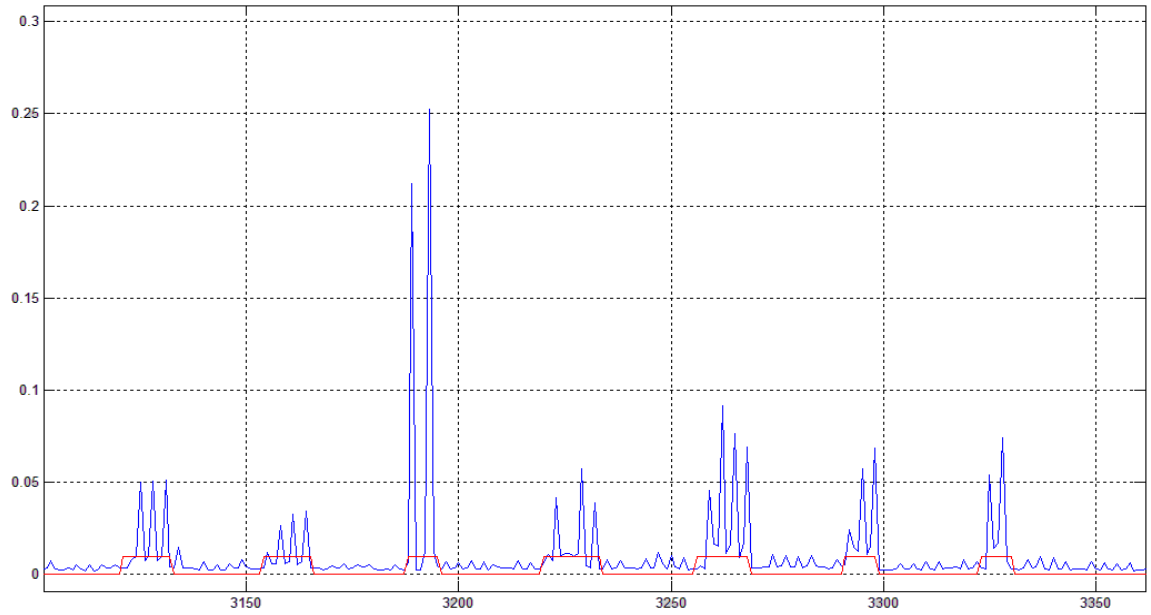


Figure 3.15 Illustration of MPEG motion compensation error graph on dissolves

In comparison with IVC, the MPEG motion compensation error indicator presents two advantages: (i) it can be readily extracted from MPEG compressed domain; (ii) it presents a sequence of peaks during dissolve transitions and thus can be exploited to detect dissolves. Figure 3.15 presents a graph of the MPEG motion compensation error for the

same example video illustrated in Figure 3.14, from which it is seen that a sequence of peaks is present in every dissolve. While such peaks may not indicate the increase and the decrease transition for dissolves, their starting and ending locations would certainly be helpful for detection of boundaries inside the dissolves.

To optimize the exploitation of the MPEG motion compensation error indicator, a two-step procedure is proposed for dissolve detection, where the first step is to pre-process the video sequence and remove those unlikely to be dissolve candidates, and the second step is to apply a finite state machine (FSM) to detect and verify the dissolves. In comparison with existing work in FSM [105], the proposed FSM features: (i) dissolve candidates are detected by monitoring the MPEG motion compensation errors in DC values; (ii) the detected dissolve candidates are further verified by testing the colour correlation between the beginning and the ending frame of the detected dissolve.

During the gradual transition, frames within a dissolve actually present large inter-frame difference as well as MPEG motion compensation errors. Therefore, the following pre-processing is defined to remove those frames that are not likely to be a dissolve candidate:

$$\begin{aligned} &\textit{if} \quad \Delta_n < T_L \,||\, err_n < T \quad \textit{then} \quad err_n = 0 \;; \\ &\textit{else} \quad \textit{if} \quad \left|\overrightarrow{M_n}\right| > T_H \,\&\, \Delta_n < T_H \quad \textit{then} \quad err_n = 0 \;. \end{aligned} \tag{3.38}$$

where T_L and T_H are the lower and higher thresholds as defined in Table 3.8, and T is a new threshold introduced for err_n, which is determined as 0.015 via empirical studies as discussed in the previous sections.

Essentially, the pre-processing forces the value of err_n to be zero when either the inter-frame difference indicator or the MPEG motion compensation error indicator is sufficiently small. Under the context of strong motion indicated by $\left|\overrightarrow{M_n}\right| > T_H$, however, the threshold for the inter-frame difference indicator needs to be increased in order to reduce the negative effect brought in by the strong motion.

Following the pre-processing, the remaining operation is focused on those frames with non-zero err_n values to detect candidates for dissolves under the principle that dissolves present a sequence of peak values in err_n. To this end, an FSM with five states is proposed to complete the detection, details of which are summarized in Table 3.10 and Figure 3.16.

Table 3.10 FSM states description

State-ID	State Description
State_0	Initial state
State_1	Detection state for the beginning frame of a possible dissolve candidate
State_2	Detection state for the ending frame of a possible dissolve candidate
State_3	Verification state
State_4	Dissolve detected state

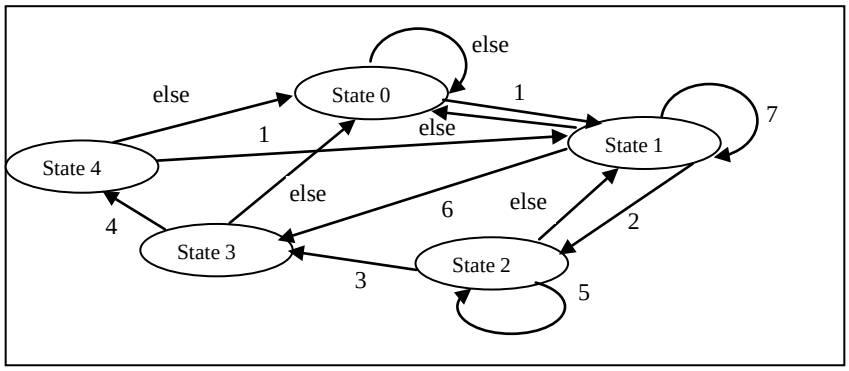


Figure 3.16 Structure of FSM for dissolve detection

As seen in Figure 3.16, seven conditions are defined to complete the inter-state transition inside the FSM. Details of their definitions are given in Table 3.11, where *B*

and E stand for the beginning frame and the ending frame of the dissolve candidate, and $p_{B,E}$ is their colour correlation which is defined in (3.26) by replacing *n-1* and *n* with *B* and *E*. In addition, three fixed length values, $L_{\min}$, $L_{\max}$ and $L_{transition}$ are used to control i in recording the length of the dissolve candidate. $L_{\min}$ and $L_{\max}$ controls the minimum and the maximum length of all dissolve candidates. After the pre-processing, most of the frames within the dissolve present non-zero values of err_n and these frames are closer to each other compared to the neighbouring normal frames. Hence, $L_{transition}$ is utilized as a distance measurement for grouping these candidate frames of a dissolve together. In this algorithm, the parameters are set as follows: $L_{\min} = 5$, $L_{\max} = 100$ and $L_{transition} = 10$.

Table 3.11 Definition of conditions for inter-state transition in FSM

Condition Number	Condition Definition
1	$if\left(err_n > 0 \,\&\, err_{n-1} = 0\right)$
2	$if\left(err_{n+i} = 0 \,\&\, i \geq L_{\min}\right)$
3	$if\left(err_{E+10} = 0\right)$
4	$if\left(p_{B,E} > T_H \,\|\| \frac{1}{E-B+1}\sum_{n=B}^{E} err_n > 2T\right)$
5	$if\left(err_{E+i} = 0 \,\&\, i < L_{transition}\right)$
6	$if\left(err_{n+i} \neq 0 \,\&\, i \geq L_{\max}\right)$
7	$if\left(err_{n+i} \neq 0 \,\&\, i < L_{\max}\right)$

Given the current frame Y_n, its pre-processed MPEG compensation error indicator and its preceding error indicator are examined (condition-1). If the condition 1 is satisfied, the initial state transition is to state_1, where the current frame is taken as the starting frame of a possible dissolve candidate $(B = n)$, and the length of the possible dissolve candidate is recorded by the variable i, while further frames are taken in to see whether there should be further transitions from state_1.

As seen in Figure 3.16, state_1 has three conditions to be tested, i.e. 2, 6 and 7. When condition 2 is satisfied, indicating that the dissolve candidate has reached its end $(E = n + i)$, and thus the transition should be to state_2. Condition 6 sets up a maximum length for dissolves to be detected. In other words, if the dissolve candidate has recorded $L_{\max}$ frames where all MPEG compensation error indicators are non zeros, the state_1should go straight to state_3 for its verification. Satisfaction of condition 7 will keep it in the state_1 without any transition since the maximum length for dissolves is not reached. Finally, if none of the three conditions are satisfied, the state has to go back to the initial state.

To detect that $(E = n + i)$ is the ending frame of the dissolve candidate, I require that $L_{transition}$ continuous frames with $err_{n+i} = 0$ (specified by condition 5) should be recorded. If condition 3 is satisfied, indicating that $L_{transition}$ continuous frames with $err_{n+i} = 0$ have been recorded, the state transfers to state_3 for verification of the dissolve candidate. Otherwise, the state will transfer back to state_1 as indicated in Figure 3.16, and the corresponding frame will be taken as part of the dissolve candidate.

At state_3, verification is done by two examinations as indicated by condition 4. One is to examine the colour correlation between the beginning frame and the ending frame of the dissolve candidate, and the other is to examine the average of the compensation errors for all the frames inside the dissolve candidate. If the candidate is a true dissolve, there should exist some colour dissimilarity as defined in (3.26), or alternatively, the average of the compensation errors is larger than a threshold. Satisfaction of this condition leads to state_4, where a dissolve is detected. Otherwise, the state has to be transferred back to the initial state.

3.5.4 Experimental Results

To evaluate the proposed algorithm, extensive experiments were carried out on a number of test video clips from the TRECVID activity, which is organized by National Institute of Standards and Technology (NIST) annually [95]. I also submitted my detection results as one of the 9 runs to participate in the shot boundary detection task in TRECVID 2007, and therefore, the experimental results reported here are mainly for the TRECVID 2007 test data set, in which the resolution of the test sequences is 352×240 pixels. Table 3.12 provides a summary description of all test video sequences, including the total number of frames, cuts and gradual transitions within each video clip.

Table 3.12 Description of the video sequences in the test set

Video name	Number of frames	Cut count	Gradual transition count
BG_2408	35892	103	18
BG_9401	50049	89	3
BG_11362	16416	104	4
BG_14213	83115	107	60
BG_34901	34389	225	15
BG_35050	36999	100	2
BG_35187	29025	135	23
BG_36028	44991	87	0
BG_36182	29610	96	13
BG_36506	15210	77	6
BG_36537	50004	259	30
BG_36628	56564	196	6
BG_37359	28908	165	5
BG_37417	23004	84	4
BG_37822	21960	120	9
BG_37879	29019	95	4
BG_38150	52650	215	4
In total	637805	2257	206

The computing environment used for software implementation of the proposed algorithm includes: (i) a PC with 1.73GHz CPU, 512MB memory and windows XP operating system; (ii) Microsoft VC++ 6.0 programming platform. The performances of the proposed algorithm are measured by recall rate, precision rate and F1 rate as defined by TRECVID.

Table 3.13 presents the experimental results using the proposed algorithm for all the 17 sequences as evaluated and announced by TRECVID 2007, where the recall and precision rate figures are listed in four groups, including overall, cuts, gradual transitions and gradual transition boundary frame accuracy, in accordance with the submission requirement specified by TRECVID 2007 organizers. While the two groups, cuts and gradual transitions, directly relate to the performances on cut detection and gradual transition detection respectively, the overall recall and precision rates are worked out according to the proportion of cuts and gradual transitions inside each video sequence. The group, gradual transition boundary frame accuracy, is used by TRECVID 2007 to measure the accuracy of boundary detection for gradual transitions, and the figure specifies the percentage of detected frames that overlap with the ground truth.

From Table 3.13, it can be seen that the proposed algorithm provides some excellent experimental results as well as some poor results depending on the nature of the video sequence content. The performance on cut detection is overwhelmingly better than gradual transitions. Specifically, it is noticed that the proposed algorithm produced zero detection rates for three video sequences, among which one indicates an excellent performance for the video sequence *BG_36028* but poor performances for the other two sequences, *BG_36182 and BG_36628*. Further analysis reveals that the poor performance of the proposed algorithm is largely due to the missed type defined by TRECVID 2007 as “others”, which are neither standard dissolve gradual transition nor cuts. Yet in my proposed algorithm, no techniques have been designed to target such special type of shot boundaries. Figure 3.17 illustrates two examples of such “others”, from which it is seen that part-(a) is very close to a cut since only a small part of the picture goes through the

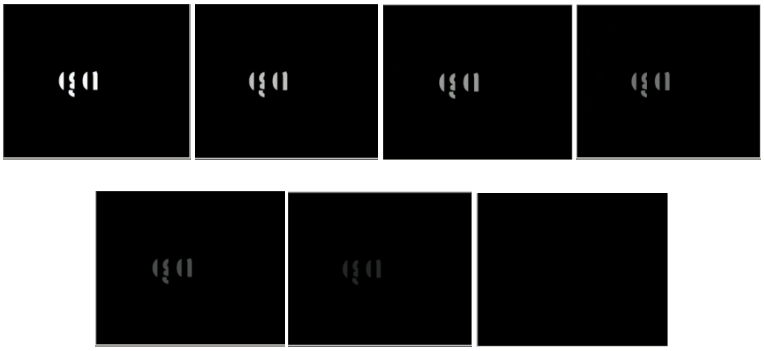
wipe transition inside the second frame and then the third frame is entirely different. In part-(b) of Figure 3.15, the transition part is again very small, only involving a few white letters inside the middle of the picture. As the proposed algorithm is designed primarily for those standard cuts and dissolves, the performance on these kinds of "others" are poor and more dedicated detection techniques are required in the future work.

Table 3.13 Recall and precision results for 17 sequences

Video name	Overall		Cut	
	Recall	**Precision**	**Recall**	**Precision**
BG_2408	0.958	0.943	0.961	0.970
BG_9401	0.934	0.843	0.943	0.988
BG_11362	0.888	0.653	0.913	0.655
BG_14213	0.874	0.797	0.971	0.954
BG_34901	0.895	0.884	0.928	0.976
BG_35050	0.950	0.941	0.96	1.0
BG_35187	0.835	0.874	0.911	0.891
BG_36028	0.977	0.867	0.977	0.904
BG_36182	0.779	0.944	0.885	1.0
BG_36506	0.807	0.893	0.831	0.969
BG_36537	0.878	0.927	0.915	0.995
BG_36628	0.905	0.915	0.933	0.989
BG_37359	0.941	0.958	0.945	1.0
BG_37417	0.863	0.873	0.880	0.913
BG_37822	0.891	0.898	0.9	0.955
BG_37879	0.757	0.903	0.768	1.0
BG_38150	0.922	0.935	0.934	1.0

Video name	Gradual transition		Gradual transition frame-based	
	Recall	**Precision**	**Recall**	**Precision**
BG_2408	0.944	0.809	0.8958	0.4841
BG_9401	0.666	0.117	0.9091	0.7692
BG_11362	0.25	0.5	0.5094	1.0
BG_14213	0.7	0.567	0.8281	0.8124
BG_34901	0.4	0.206	0.7692	0.2564
BG_35050	0.5	0.142	0.6800	1.0
BG_35187	0.391	0.692	0.8209	0.9643
BG_36028	0	0	0	0
BG_36182	0	0	0	0
BG_36506	0.5	0.333	0.8012	0.8808
BG_36537	0.566	0.472	0.5395	0.6287
BG_36628	0	0	0	0
BG_37359	0.8	0.363	0.9423	0.5568
BG_37417	0.5	0.333	0.7647	1.0
BG_37822	0.777	0.466	0.7160	0.9206
BG_37879	0.5	0.2	1.0	0.7624
BG_38150	0.25	0.066	0.7500	0.0612

Part-(a) First example of “others”



Part-(b) Second example of “others”

Figure 3.17 Illustration of “others”

Table 3.14 lists the average experimental results produced by all 15 participating teams, and Table 3.15 lists the F1 measurement of all the participating teams’ performances as announced in TRECVID 2007, where my submission is denoted by the letter M and only the results produced by the proposed algorithm are included. As indicated by F1 measurements, my proposed algorithm achieved the 5th best overall performance, the 5th best performance for gradual transition detection, the 4th best performance for gradual transition boundary frame accuracy, and the 6th best performance for cut detection. It is worthy of mention that the differences among the top 6 participating teams is small, the results of which are summarized in Table 3.16.

Table 3.14 Recall and precision results for all teams in TRECVID 2007

Team name	Overall				Cut			
	Recall		Precision		Recall		Precision	
A	0.7435	(14)	0.9505	(2)	0.8035	(13)	0.9540	(5)
B	0.9419	(3)	0.9506	(1)	0.9718	(2)	0.9639	(2)
C	0.9220	(4)	0.8210	(9)	0.9614	(5)	0.9591	(4)
D	0.8879	(9)	0.9120	(5)	0.9689	(4)	0.9120	(10)
E	0.8360	(12)	0.7392	(12)	0.8840	(12)	0.8094	(12)
F	0.8801	(10)	0.4995	(15)	0.9064	(11)	0.7429	(14)
G	0.8797	(11)	0.8157	(10)	0.9201	(9)	0.9346	(7)
H	0.7514	(13)	0.8646	(8)	0.7663	(15)	0.8885	(11)
I	0.9018	(6)	0.8726	(7)	0.9288	(7)	0.9337	(8)
J	0.9036	(5)	0.9284	(4)	0.9304	(6)	0.9620	(3)
K	0.9509	(2)	0.9328	(3)	0.9692	(3)	0.9763	(1)
L	0.9005	(7)	0.6171	(13)	0.9276	(8)	0.9154	(9)
M	**0.8890**	**(8)**	**0.8870**	**(6)**	**0.9200**	**(10)**	**0.9520**	**(6)**
N	0.9525	(1)	0.6080	(14)	0.9778	(1)	0.7248	(15)
O	0.7108	(15)	0.7514	(11)	0.7758	(14)	0.7514	(13)

Team name	Gradual transition				Gradual transition Frame-based			
	Recall		Precision		Recall		Precision	
A	0.0900	(13)	0.6785	(2)	0.5365	(11)	0.9315	(1)
B	0.6142	(3)	0.7579	(1)	0.7312	(4)	0.9279	(2)
C	0.4909	(10)	0.2555	(8)	0.6628	(6)	0.7927	(10)
D	0	(14)	0	(14)	0	(14)	0	(14)
E	0.3124	(12)	0.2455	(9)	0.5811	(9)	0.8105	(8)
F	0.5923	(6)	0.0937	(13)	0.6540	(8)	0.6056	(13)
G	0.4378	(11)	0.2233	(10)	0.5390	(10)	0.8510	(6)
H	0.5873	(8)	0.6458	(3)	0.6584	(7)	0.8687	(5)
I	0.5922	(7)	0.4008	(6)	0.4010	(12)	0.8013	(9)
J	0.6118	(4)	0.6298	(4)	0.6824	(5)	0.8894	(4)
K	0.7504	(1)	0.6036	(5)	0.7755	(2)	0.8382	(7)
L	0.6014	(5)	0.1332	(12)	0.7490	(3)	0.7292	(11)
M	**0.5530**	**(9)**	**0.3940**	**(7)**	**0.7920**	**(1)**	**0.7180**	**(12)**
N	0.6747	(2)	0.1972	(11)	0.2649	(13)	0.9096	(3)
O	0	(14)	0	(14)	0	(14)	0	(14)

**The number in brackets is the rank of the corresponding value from each team.*

Table 3.15 F1-measure results for all teams

Team name	Overall F1		Cut F1		Gradual transition F1		Gradual transition frame-based F1	
A	0.8343	(9)	0.8723	(10)	0.1563	(12)	0.6724	(8)
B	0.9460	(1)	0.9677	(2)	0.6602	(1)	0.8177	(1)
C	0.8651	(7)	0.9601	(3)	0.2725	(8)	0.7180	(7)
D	0.8975	(4)	0.9372	(5)	0	(14)	0	(14)
E	0.7815	(10)	0.8431	(11)	0.2287	(10)	0.6402	(10)
F	0.6280	(15)	0.8158	(13)	0.1462	(13)	0.5971	(11)
G	0.8455	(8)	0.9272	(8)	0.2818	(7)	0.6593	(9)
H	0.7500	(11)	0.7671	(14)	0.5769	(4)	0.7481	(5)
I	0.8847	(6)	0.9307	(7)	0.4532	(6)	0.5344	(12)
J	0.9148	(3)	0.9456	(4)	0.6038	(3)	0.7722	(3)
K	0.9414	(2)	0.9727	(1)	0.6577	(2)	0.8056	(2)
L	0.7230	(13)	0.9190	(9)	0.1768	(11)	0.7384	(6)
M	**0.8880**	**(5)**	**0.9357**	**(6)**	**0.4602**	**(5)**	**0.7532**	**(4)**
N	0.7261	(12)	0.8241	(12)	0.2696	(9)	0.4102	(13)
O	0.6773	(14)	0.7073	(15)	0	(14)	0	(14)
Our Rank	5		6		5		4	

**The number in brackets is the rank of the corresponding F1 value from each team.*

Table 3.16 Top 6 F1 results in TRECVID 2007

Overall F1	Cut F1	Gradual transition F1	Gradual transition Frame-based F1
0.9460(B)	0.9727(K)	0.6602(B)	0.8177(B)
0.9414(K)	0.9677(B)	0.6577(K)	0.8056(K)
0.9148(J)	0.9601(C)	0.6038(J)	0.7722(J)
0.8975(D)	0.9456(J)	0.5769(H)	**0.7532(M)**
0.8880(M)	0.9372(D)	**0.4602(M)**	0.7481(H)
0.8847(I)	**0.9357(M)**	0.4532(I)	0.7384(L)

**The letter in brackets is the name of the corresponding team.*

Table 3.17 Mean runtime of participants in TRECVID 2007

Team name	Runtime (seconds)	Ratio to real-time	Rank
A	10586.3	0.4150	9
B	7325.5	0.2871	8
C	4157.9	0.1630	3
D	17540	0.6875	12
E	611200.2	23.9572	15
F	15637.9	0.6130	11
G	3615.1	0.1417	2
H	96948.8	3.8001	14
I	5517.9	0.2163	6
J	1686.5	0.0661	1
K	12974.7	0.5086	10
L	7319.7	0.2869	7
M	**5357.6**	**0.2100**	**5**
N	4223.1	0.1656	4
O	42397.3	1.6618	13
Real-time of all videos playing		25512.2	
Our performance	4 times faster than video real-time playing		Rank 5

Table 3.17 presents the runtime of the proposed algorithm and TRECVID 2007 participants. As indicated by the ratio of runtime to real-time video playing, my algorithm is 4 times faster than real-time video playing.

3.5.5 Conclusions

In Section 3.5, I describe a hierarchical content-aware algorithm with multiple content difference indicators and thresholds for shot cut detection. While this area has been well researched over the past decades, the proposed algorithm has made three contributions. Firstly, a hierarchical content-aware approach with multiple content difference indicators and multiple thresholds is proposed to deal with cuts and dissolves in practical cases, which are much more complicated than theoretically described or expected. The application of each individual threshold is controlled by multiple context indicators extracted in compressed domain. Secondly, the entire detection process is organized by decision trees as well as FSM to achieve operation efficiency and effectiveness in its performances. Thirdly, a coarse-to-fine procedure is introduced with pre-processing and post-processing modules to reduce the computation cost and increasing its detection reliability. Extensive experiments demonstrate that the proposed algorithm achieves promising results and performances for a well-known but complicated test set (TRECVID 2007), where video sequences present a wide range of cuts and gradual transitions under various circumstances and mixed scene changes.

While empirical study on training video sequences is required to determine the multiple thresholds, the specific process is much simpler than the training and learning process for machine learning approaches such as SVM [7]. As illustrated in Table 3.8, such empirical study only needs to determine the lower threshold and higher threshold,

where the principle is very clear that above T_L, the corresponding frame difference indicator should enable all cuts to be detected, and above T_H, no false positives should exist. Following that, medium thresholds are calculated via equation (3.29), providing more fine-tuned scopes for detecting cuts under complicated contexts. Similar to machine learning approaches, the proposed algorithm is also sensitive to training video sequences in special cases, such as a shot change with very small content difference not occurred inside the training sequences. However, such special cases are rare in practice and often require dedicated attention as illustrated in Figure 3.17. Under this circumstance, machine learning approaches will require no exception simply because learning from what happened inside the training video sequences is primarily required for all machine learning approaches to detect shot changes. To this end, the robustness of the proposed algorithm remains the same as those machine learning approaches. Due to the nature of context-aware with multiple difference indicator and multiple thresholds, however, the proposed algorithm could be more robust than machine learning approaches. This can be established on the grounds that, while one content difference indicator remains small, other content difference indicators could be high as identified by multi-level thresholds. Such advantages are illustrated in Figure 3.11 to Figure 3.13.

Finally, the proposed algorithm is not strong enough for the detection of “others”. It usually includes very complicated transitions, such as part of the picture incurring gradual transition, and picture-in-picture transition. The extraction of novel features and the use of relevant knowledge-supported rules would improve the detection performance of “others” in the future work.

3.6 A Fuzzy Logic Method of Feature Representation for Shot Boundary Detection

In this section, all features are extracted in the compressed domain exploiting MPEG compression techniques and features are organized into a fuzzy logic representation. Then a uniform framework based on SVM is implemented for cut and gradual transition detection.

3.6.1 Feature Selection

To correctly locate shot boundary, the first requirement is to select effective features. Another criterion is that the feature extraction should be robust to disturbing factors, such as lighting changes and motion. I choose four kinds of features as the measurement for shot detection. The first feature is inter-frame difference, which measures the difference between frames in a local context. The second feature is motion, which calculates camera and object movements. The third feature is luminance energy, which detects luminance changes between frames, i.e. flashing light. The fourth feature is motion compensation error (MCE), which is a unique feature for gradual transition detection. All the features are examined in a sliding window containing the DC components of 12 consecutive images, i.e. $\{f_{n-6}, f_{n-5},..., f_n,..., f_{n+4}, f_{n+5}\}$.

3.6.1.1 Inter-frame Difference

Given a DC image in the YUV domain, a normalized histogram with 32 bins is constructed. The inter-frame difference of the histogram is used to measure the temporal content change. By representing such normalized Y, U and V histograms as:

$H_{Yn} = \left(h_{Yn}^1, h_{Yn}^2, ...h_{Yn}^{32}\right)$, $H_{Un} = \left(h_{Un}^1, h_{Un}^2, ...h_{Un}^{32}\right)$, $H_{Vn} = \left(h_{Vn}^1, h_{Vn}^2, ...h_{Vn}^{32}\right)$ respectively, an inter-frame difference between frame f_n and frame f_m can be defined as:

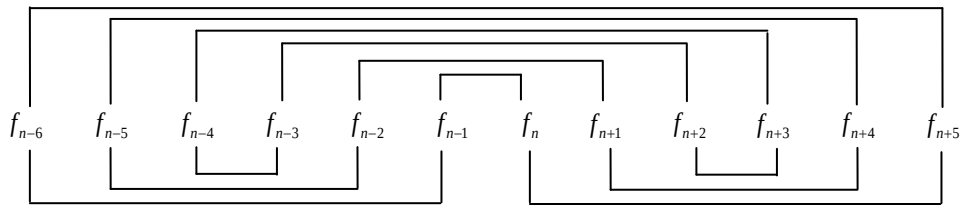
$$\Delta(n,m) = 1 - \frac{1}{3}\left(\frac{1}{N_y}\sum_{i=1}^{32}\min\left(h_{Yn}^i, h_{Ym}^i\right) + \frac{1}{N_u}\sum_{i=1}^{32}\min\left(h_{Un}^i, h_{Um}^i\right) + \frac{1}{N_v}\sum_{i=1}^{32}\min\left(h_{Vn}^i, h_{Vm}^i\right)\right)\cdot \quad (3.39)$$

where N_y, N_u and N_v are the corresponding numbers of 8×8 blocks for Y, U and V components respectively.

Given a shot cut between frame f_{n-1} and frame f_n, the frames $\{f_{n-6},..., f_{n-1}\}$ belong to the previous shot, while the frames $\{f_n,..., f_{n+5}\}$ belong to the subsequent shot. Therefore, the inter-frame differences could be classified into two groups, the differences inside shots and differences between shots. The inter-frame differences inside the previous and subsequent shots are measured by the vectors $\Delta_p = \{\Delta(n-4,n-3), \Delta(n-5,n-2), \Delta(n-6,n-1)\}$ and $\Delta_s = \{\Delta(n+2,n+3), \Delta(n+1,n+4), \Delta(n,n+5)\}$ respectively. In addition, the inter-frame difference between two shots is measured by the vector $\Delta_t = \{\Delta(n-i,n+i-1), i = 1...6\}$. This is illustrated in the following figure.

The inter-frame Difference between two shots:

$$\Delta_t = \{\Delta(n-1,n),..., \Delta(n-4,n+3),..., \Delta(n-6,n+5)\}$$



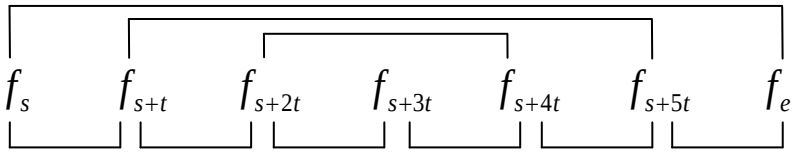
The inter-frame difference inside the previous shot: $\Delta_p = \{\Delta(n-4,n-3), \Delta(n-5,n-2), \Delta(n-6,n-1)\}$

The inter-frame difference inside the subsequent shot: $\Delta_s = \{\Delta(n+2,n+3), \Delta(n+1,n+4), \Delta(n,n+5)\}$

Figure 3.18 Three types of inter-frame differences for cut detection

On the other hand, given the gradual transition candidate with a serial of frames $\{f_s,..., f_i,..., f_e\}$, the whole transition is divided into 6 segments with equal length, which is taken as the re-sample rate t. This segmentation is based on the fact that the minimum length of a gradual transition is 6 frames. Hence, the gradual transition candidate is represented by the re-sampled frames $\{f_s, f_{s+t}, f_{s+2t}, f_{s+3t}, f_{s+4t}, f_{s+5t}, f_e\}$. Therefore, the inter-frame difference for a gradual transition candidate is measured by the neighbouring inter-frame difference vector $\Delta_n = \{\Delta(s, s+t),..., \Delta(s+i\cdot t, s+(i+1)\cdot t),..., \Delta(s+5t, e), i=1...4\}$ and the expansion inter-frame difference vector $\Delta_e = \{\Delta(s+2t, s+4t), \Delta(s+t, s+5t), \Delta(s, e)\}$. This is illustrated in the below figure.

The expansion inter-frame difference: $\Delta_e = \{\Delta(s+2t, s+4t), \Delta(s+t, s+5t), \Delta(s, e)\}$



The neighbouring inter-frame difference:

$\Delta_n = \{\Delta(s, s+t),..., \Delta(s+3t, s+4t),..., \Delta(s+5t, e)\}$

Figure 3.19 Two types of inter-frame differences for gradual transition detection

3.6.1.2 Motion

As MPEG video has motion information available in the compressed domain, I extract a normalized motion feature based on the MPEG motion vector $(V_x(i,j), V_y(i,j))$ in the n^{th} image as:

$$M_x(n) = \frac{1}{N_{inter}} \sum_{i,j} \left| \frac{V_x(i,j)}{T_{vx}} \right| . \quad (3.40)$$

$$M_y(n) = \frac{1}{N_{inter}} \sum_{i,j} \left| \frac{V_y(i,j)}{T_{vy}} \right| . \quad (3.41)$$

where (T_{vx}, T_{vy}) is the maximum allowable motion vector designated by MPEG. N_{inter} is the number of inter-coded macro blocks. For an I frame, the motion information is copied from the latest P frame to be output.

3.6.1.3 Luminance Energy

Luminance is a basic and reliable visual feature for checking lighting change in video shots. The luminance energy of the n^{th} image is defined in (3.42) as:

$$\eta(n) = \frac{1}{4MN \times 255 \times 255} \sum_{i=1}^{2M} \sum_{j=1}^{2N} y_{i,j}(n)^2 . \quad (3.42)$$

Here, M and N are the number of macro blocks in a frame on the vertical and horizontal direction, respectively. $y_{i,j}(n)$ is the luminance value of the pixel (i, j) in the n^{th} DC image.

3.6.1.4 Motion Compensation Error

One of the spatial features widely used for gradual transition detection is the intensity variance curve (IVC), which often has an approximately parabolic ('U' type) shape when a gradual transition occurs. However, the U shape IVC is often corrupted in reality due to motion, camera flash and many other factors. Consequently, it is difficult in practice to capture such transition processes. Looking for an alternative feature, presenting a stronger indication of gradual transitions, I tested a range of possibilities and propose a new feature, MPEG motion compensation error (MCE), which is defined as follows:

$$\varepsilon(n) = \frac{1}{C_n \times \kappa} \sum_{i=1}^{C_n} |DC(i)| \cdot \quad (3.43)$$

Here C_n is the number of inter-coded blocks and κ is the threshold applied in MPEG video to decide whether a block should be inter-coded or not. In comparison with IVC, MCE possesses two advantages: (i) it can be readily extracted from MPEG compressed domain; (ii) it presents a sequence of peaks during gradual transitions and thus can be exploited to detect gradual transitions.

3.6.2 Feature Representation using Fuzzy Logic

Fuzzy logic is introduced to represent the relationships of the feature elements inside a sliding window of 12 frames. The features include inter-frame difference, motion, luminance energy and motion compensation error, which are defined as linguistic variables. Three fuzzy sets, *small*, *medium* and *large* are defined for each linguistic variable and I choose a Gaussian membership function for each fuzzy set. The relation between the feature elements inside the sliding window can be represented by the fuzzy operator AND. Hence, the fuzzification of the feature is represented as follows, where $\mu \in (\mu_S, \mu_M, \mu_L)$ indicates the membership functions of *small*, *medium* and *large*.

- The fuzzification of the inter-frame difference in the previous shot:

  $$\tilde{\Delta}_p = \mu(\Delta(n-4, n-3)) AND \mu(\Delta(n-5, n-2)) AND \mu(\Delta(n-6, n-1))$$

- The fuzzification of the inter-frame difference in the subsequent shot:

  $$\tilde{\Delta}_s = \mu(\Delta(n+2, n+3)) AND \mu(\Delta(n+1, n+4)) AND \mu(\Delta(n, n+5))$$

- The fuzzification of the inter-frame difference between two shots:

  $$\tilde{\Delta}_t = \mu(\Delta(n-1, n)) AND ... AND \mu(\Delta(n-6, n+5))$$

- The fuzzification of the neighbouring inter-frame difference in gradual transition:

$$\tilde{\Delta}_n = \mu(\Delta(s, s+t)) AND ..., \mu(\Delta(s+3t, s+4t)), ... AND \mu(\Delta(s+5t, e))$$

- The fuzzification of the expansion inter-frame difference in gradual transition:

$$\tilde{\Delta}_e = \mu(\Delta(s+2t, s+4t)) AND \mu(\Delta(s+t, s+5t)) AND \mu(\Delta(s, e))$$

- The fuzzification of motion in the x direction:

$$\tilde{M}_x = \mu(M_x(n-6)) AND ... AND \mu(M_x(n+5))$$

- The fuzzification of motion in the y direction:

$$\tilde{M}_y = \mu(M_y(n-6)) AND ... AND \mu(M_y(n+5))$$

- The fuzzification of luminance energy:

$$\tilde{\eta} = \mu(\eta(n-6)) AND ... AND \mu(\eta(n+5))$$

- The fuzzification of motion compensation error:

$$\tilde{\varepsilon} = \mu(\varepsilon(n-6)) AND ... AND \mu(\varepsilon(n+5))$$

3.6.3 Cut Detection

SVM is utilized to determine the shot cuts using feature vectors generated from the feature fuzzification process. The chosen feature vector is $\{\tilde{\Delta}_p, \tilde{\Delta}_s, \tilde{\Delta}_t, \tilde{M}_x, \tilde{M}_y, \tilde{\eta}\}$. The feature vectors labelled as cut or non-cut form the training set for SVM. During the training process, a parallel grid search based on cross-validation is applied to optimize penalty factor C and kernel parameter γ for an SVM model with radial basis function (RBF) kernel. Since the cross-validation procedure solves the over fitting problem, pairs of (C, γ) are tried and the one with the best cross-validation accuracy is selected. The strong generalization ability of the SVM facilitates accurate cut detection.

3.6.4 Gradual Transition Detection

The gradual transition detection contains three phases: (i) gradual transition candidate frames are selected by the SVM; (ii) the selected frames are grouped by a cluster scheme; (iii) the temporal difference of a gradual transition candidate is verified by SVM.

In the first phase, SVM is utilized to determine the candidate frames within a gradual transition using the feature vector $\{\tilde{M}_x, \tilde{M}_y, \tilde{\eta}, \tilde{\varepsilon}\}$, generated from the feature fuzzification process. Here, the motion and luminance features are combined to verify the peak values of motion compensation error. Because in the real cases, the movement of large objects like human beings and the change of lightning condition will cause peak values of motion compensation error other than a true gradual transition.

The second operation is to group the candidate frames selected by phase one into gradual transition candidates for further verification. We propose a cluster procedure to complete the grouping, details of which are given below.

- Input: candidate frames G_i in a circular buffer.
- Output: gradual transition candidates
- Procedure:

  Step 1: Select the candidate frame F_i in G_i, with smallest frame number as the seed for cluster C_i, and subtract F_i from G_i. If the last frame in the buffer is reached, wait for buffer to fill until the whole video sequence is processed.

  Step 2: If there exists another candidate frame F_j in the rightwards neighbourhood of n_H frames, absorb frame F_j in cluster C_i. Subtract F_j from G_i.

Step 3: Iteratively execute step 3, until there are no candidate frames in the rightwards neighbourhood of n_H frames. As a result, cluster C_i constitutes a gradual transition candidate. Go to step 1.

In the cluster procedure, n_H represents the scale of the neighbourhood. Using a multi-resolution approach, it is chosen from the set (5,10,15,20) in the experiments. The use of circular buffer to monitor the frames to be clustered enables the real-time detection of gradual transitions.

According to the re-sample scheme proposed in Section 3.6.1.1, seven re-sampled motion elements for both x and y direction, and seven re-sampled luminance elements are obtained within a gradual transition candidate. Furthermore, the fuzzified re-sampled motion and luminance features are denoted as $\left\{\tilde{M}_x^s, \tilde{M}_y^s, \tilde{\eta}^s\right\}$. For the process of temporal verification of gradual transition detection, I use $\left\{\tilde{M}_x^s, \tilde{M}_y^s, \tilde{\eta}^s, \tilde{\Delta}_n, \tilde{\Delta}_e\right\}$ as the vector for making the SVM decision.

3.6.5 Experimental Results

To evaluate the proposed algorithm, we carried out extensive experiments on a number of test video clips from the TREC Video Retrieval Evaluation (TRECVID) activity which is organized annually by National Institute of Standards and Technology (NIST). The experimental results reported here are mainly for the TRECVID07 test data set, in which the resolution of the test sequences is 352×240 pixels. The whole video set consists of 637805 frames from 17 videos, including 2257 cuts and 206 gradual transitions.

Table 3.18 illustrates the performance of the current algorithm compared with those from participants in TRECVID07. The recall, precision and F1 rate are listed for

both cut and gradual transition. The time ratio of runtime to video play time is also compared. It shows that the current algorithm achieves very competitive results in terms of the detection rate. Also, the runtime of the current algorithm is 4 times faster than the video play time, which ensures the real time shot detection.

Table 3.18 Performance of the proposed algorithm compared to teams from TRECVID 2007

Team	Cut			Gradual transition			Time ratio
	Recall	Precision	F1	Recall	Precision	F1	
A	0.80	0.95	0.87	0.09	0.68	0.16	0.42
B	0.97	0.96	0.96	0.61	0.76	0.68	0.29
C	0.96	0.96	0.96	0.49	0.26	0.34	0.16
D	0.97	0.91	0.94	0.00	0.00	0.00	0.69
E	0.88	0.81	0.84	0.31	0.25	0.28	23.96
F	0.91	0.74	0.82	0.59	0.09	0.16	0.61
G	0.92	0.93	0.92	0.44	0.22	0.29	0.14
H	0.77	0.89	0.83	0.59	0.65	0.62	3.80
I	0.93	0.93	0.93	0.59	0.40	0.48	0.22
J	0.93	0.96	0.94	0.61	0.63	0.62	0.07
K	0.97	0.98	0.97	0.75	0.60	0.67	0.51
L	0.93	0.92	0.92	0.60	0.13	0.21	0.29
M	0.92	0.95	0.93	0.55	0.39	0.46	0.21
N	0.98	0.72	0.83	0.67	0.20	0.31	0.17
O	0.78	0.75	0.76	0.00	0.00	0.00	1.66
Current	0.98	0.98	0.98	0.64	0.84	0.73	0.21

3.7 Summary

In Section 3.2, we described a simple and fast algorithm for detection of both abrupt shot cuts and dissolved shot cuts. The proposed algorithm works in compressed domain via exploiting existing MPEG motion estimation and compensation mechanisms. While the proposed algorithm can save a lot of computation cost, extensive experiments support that the proposed algorithm also achieves superior performances over the existing counterpart. In summary, the feature and the advantage of the proposed algorithm can be highlighted as: (i) an integrated technique for both abrupt shot cut and dissolved shot cut detection; (ii) directly operates in compressed domain and thus suitable for real-time implementation;

and (iii) only two thresholds and two parameters are required for shot cut detection and yet the detection mechanism is made adaptive to the input video content; (iv) such technique will provide a range of useful applications for on-line video content analysis, processing and management. Specific examples include real-time scene change analysis for MPEG compressed video streams, on-line annotation of video sequences and shot-based video content retrievals.

In Section 3.3, I propose techniques for shot boundary detection with two main contributions. Firstly, content-based rules via multiple indicators are acquired from compressed domain to detect cuts. Secondly, a coarse-to-fine procedure is introduced with a pre-processing module to lower the computation cost and a FSM to verify gradual transitions and locate their boundaries. Extensive experiments have demonstrated that the proposed algorithm is highly efficient and yields quite promising results in terms of recall rate and precision rate.

In Section 3.4, my contribution for the submission to TRECVID 2007 on the shot boundary detection task is summarized as: (i) Novel features are extracted from the compressed domain and the feature selection is carried out using Adaboost. (ii) Typical gradual transition including fade in (out), dissolve and wipes, as well as a new type of gradual transition called “others” are detected. (iii) The MPEG decoding scheme is thoroughly studied so that the whole shot detection system is embedded in the compressed domain of the standard MPEG2 decoder to achieve the real-time efficiency.

In Section 3.5, a context awareness algorithm with multiple thresholds and features for shot cut detection is described. While this area is well researched for the past decades, the proposed algorithm has made two contributions. Firstly, given the fact that presence of

cuts and gradual transitions in practical cases is much more complicated than theoretically described or expected, I applied multiple thresholds to deal with various and different cases, and application of each individual threshold is controlled and indicated by multiple context features, all of which are extracted in MPEG compressed domain. Secondly, the complicated controlling process and detection is organized by decision trees as well as FSM to achieve operation efficiency as well as effectiveness in its performances. Thirdly, a coarse-to-fine procedure is introduced with pre-processing and post-processing modules to reduce the computation cost and increasing its detection reliability. Extensive experiments demonstrate that the proposed algorithm achieves promising results and performances in a test set, where video sequences present a wide range of cuts and gradual transitions under various circumstances and mixed scene changes. The proposed algorithm presents certain level of weakness in dealing with special type of shot cuts, such as part of the picture incurs gradual transition, picture-in-picture transition, etc, which requires dedicated techniques and tailored designs for their correct detection.

In Section 3.6, a fuzzy logic method of feature representation for shot detection is proposed. By carrying out extensive experiments, it is found that the proposed algorithm achieves very competitive results compared to other state-of-art algorithms.

Chapter 4

Video Retrieval

4.1 Introduction

In this chapter, a novel analysis approach for the detection of video copies subject to complicated distortions is presented. The main contribution of the proposed method lies in three aspects. Firstly, a dedicated video distortion analysis is implemented for input videos, which ensures the accurate detection of the complicated distortions query videos may undergo. Secondly, simple signatures are extracted for the benefit of time and space efficiency and a frame mask is generated adaptively to reduce video temporal redundancy. Thirdly, a progressive matching process is implemented and the property of linear regression is utilized to find video copies. Extensive experiments were carried out using data from the TRECKVID 2008 content-based video copy detection task. The proposed video copy detection framework is very effective, and robust against spatial and temporal variations, in comparison with other state-of-art algorithms.

4.2 A Novel Analysis Approach towards the Detection of Video Copies under Complicated Distortions

In comparison with all existing approaches, a new algorithm is proposed in this chapter for video copy detection, in which combinations of complicated distortions are taken into consideration when those copied parts of query videos are determined. The advantage of the proposed algorithm lies in the fact that it is robust to a range of complicated distortions,

and yet capable of locating the copied segments inside original videos. Most of the video copy detection algorithms reported so far focus primarily on simple changes introduced by different encoding parameters or simple editing effects, such as letter-box and frame rate changes. Little research, however, has been carried out to address the combination of various distortions.

4.2.1 Video Copy Detection Strategy

By analyzing the causes and properties of video distortions, they are classified into five major categories, including geometrical editing, extra information insertion, video in video, video quality decline and temporal sequence editing. Geometrical editing is a video distortion that changes the geometry of the original frame. Shift, letter box and crop are examples of geometrical editing. Extra information insertion is a video distortion that inserts text or patterns into the original frame. Video in video is a video distortion that displays a foreground video over a background video. Video quality decline is a video distortion that changes encoding parameters or introduces disturbances. Encoding parameters include resolution, contrast, bit rate, gamma, etc. Disturbances are mainly noise or blur. Temporal sequence editing is a video distortion that changes the frame rate or the display order of original videos. Hence, my copy detection process is implemented in two major stages to handle the video distortions. The first one is a video distortion analysis stage. The second one is a signature extraction and matching stage. The overview of video copy detection strategy is shown in figure 4.1.

In the video distortion analysis stage, the first three video distortions, including geometrical editing, extra information insertion and video in video, are analyzed and processed. Firstly, key frames are extracted to save computation time and space for

processing video distortions. Secondly, geometrical editing is analyzed and processed using a cluster scheme. Thirdly, extra information insertion is detected via the difference frame calculation. Finally, video in video is determined by an edge search method.

In the signature extraction and matching stage, the last two video distortions, video quality decline and temporal sequence editing are tackled. Firstly, each key frame is scaled down for the sake of signature representation in terms of compression and robustness. Secondly, the corresponding frame mask is generated adaptively to filter the extra insertions. Thirdly, the input video is classified into video groups. Fourthly, query and target signatures are matched via my video group level and frame level comparison. Fifthly, linear regression is applied to the matched frame sequence. Finally, SVM is utilized to make the final decision based on the matching similarity and linear regression properties.

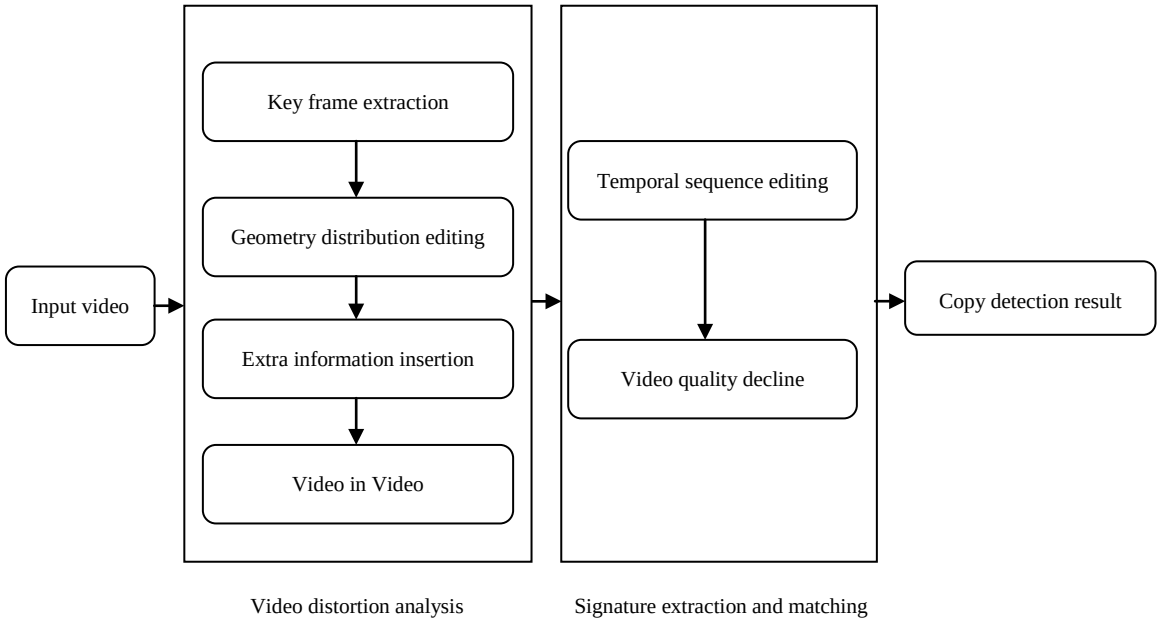


Figure 4.1 Overview of video copy detection strategy.

In order to maintain consistency between the query and target videos, extra information insertion and video in video are not carried out for the target videos in the video distortion analysis stage. As a result, frame masks are not generated for target videos.

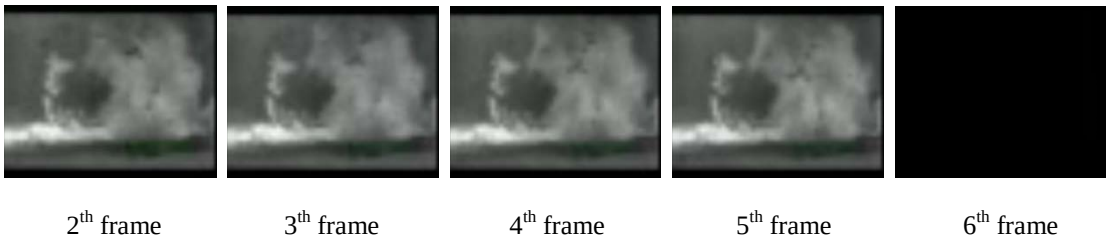
4.2.2 Video Distortion Analysis

In order to save computation time and space for processing video distortions, key frames are extracted from input videos. Key frames should meet two requirements. First, key frames do not contain gap frames, i.e. monochrome frames, especially black or white frames that are inserted to separate video sequences, causing a visual gap effect. Second, key frames do not contain temporally redundant frames, i.e. static frames or colour bar frames that last for a long time. Examples of gap frames and temporally redundant frames are shown in Figure 4.2. Gap frames are characterised by uniform pixel intensities, while temporally redundant frames are characterised by very small frame differences. Hence, the normalized standard deviation and frame difference of pixel intensity are calculated as follows.

$$R(t) = \frac{1}{255}\sqrt{\frac{1}{m^P \cdot n^P}\sum_{k=1,l=1}^{m^P,n^P}\left(I_{k,l}^t - \frac{1}{m^P \cdot n^P}\sum_{i=1,j=1}^{m^P,n^P} I_{i,j}^t\right)^2}, \quad \textbf{(4.1)}$$

$$d^F(t,s) = \frac{1}{255 \cdot m^P \cdot n^P}\sum_{i=1,j=1}^{m^P,n^P}\left|I_{i,j}^t - I_{i,j}^{t+s}\right|. \quad \textbf{(4.2)}$$

where $I_{i,j}^t$ is the intensity of pixel $(i,\ j)$ in the t^{th} frame and m^P and n^P are the number of pixels on the vertical and horizontal directions respectively. $R(t)$ measures the pixel change rate and varies in the range of $(0,\ 1)$. $d^F(t,s)$ measures the frame difference between the t^{th} frame and the $(t+s)^{th}$ frame and also varies in the range of $(0,\ 1)$.



2th frame 3th frame 4th frame 5th frame 6th frame

7th frame 8th frame 9th frame 10th frame 11th frame

2.mpg gap frames

631th frame 632th frame 633th frame 634th frame 635th frame

636th frame 637th frame 638th frame 639th frame 640th frame

154.mpg colour bar frames

1350th frame 1351th frame 1352th frame 1353th frame 1354th frame

1355th frame 1356th frame 1357th frame 1358th frame 1359th frame

365.mpg static frames

Figure 4.2 Examples of gap frames and temporally redundant frames.

Based on a visual test of a large number of video frames, a universal threshold T_u is set at 5%, which represents the visual perception of no change. Hence, if the values of $d^F(t,s)$ and $R(t)$ are lower than T_u, temporally redundant and gap frames, respectively, are detected.

Hence, the non gap frames and non temporally redundant frames constitute a meaning frame sequence of input video. Furthermore, key frames are searched based on meaningful frame sequence. The search process always begins with the latest detected key frame. The only exception is that, if it is the first time to screen the input meaningful frame sequence, the search process begins with the first meaningful frame. Given the latest detected key frame t, the subsequent frame $(t+s)$, is determined as follows:

$$K(t,s)=\begin{cases}1 & d^F(t,s)>th_k \\ 0 & d^F(t,s)\le th_k\end{cases}. \tag{4.3}$$

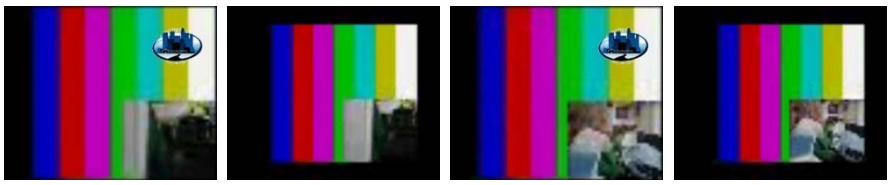
where $K(t,s)$ is a flag indicating that the subsequent frame $(t+s)$ is determined to be a key frame when it is 1. Otherwise, it is determined to be a normal frame. The value of the threshold th_k controls the sampling rate of key frames, which was set as 10% in the experiment.

The proposed processing of video distortions is based on the spatial position of area of interest (AOI) in key frames, which locates the margin introduced by the geometrical change. Hence, points of interest in key frames are extracted using the Harris detector [106]. Suppose that $x^A_{\min}$ and $x^A_{\max}$ are the lower and upper limits, respectively, of the AOI location in the vertical direction. $y^A_{\min}$ and $y^A_{\max}$ are those in the horizontal direction. The AOI is the 2-D area enclosed by $\left(x^A_{\min}, x^A_{\max}\right)$ and $\left(y^A_{\min}, y^A_{\max}\right)$. It is likely that

noise exists in video sequences, especially in low quality video scenes. So it will not be reliable to operate directly on the location of AOI in each key frame. In addition, the change of geometry may be of different kinds within one video sequence, so the location of the AOI may vary from frame to frame. An example is shown in Figure 4.3. Therefore, it is necessary to group the key frames having similar locations of AOI together, and obtain the averaged location of AOI. The simplest strategy is to implement a two-class cluster in a temporal unit. The current cluster centre is obtained by averaging the location of AOI from key frames existing in the same cluster. Furthermore, the difference of the AOI location between the t^{th} key frame and the u^{th} cluster centre is calculated as follows.

$$d^A(t,u) = \frac{1}{2\cdot w + 2\cdot h}\left(\left|x^A_{\min}(t) - x^A_{\min}(u)\right| + \left|x^A_{\max}(t) - x^A_{\max}(u)\right| + \left|y^A_{\min}(t) - y^A_{\min}(u)\right| + \left|y^A_{\max}(t) - y^A_{\max}(u)\right|\right). \quad (4.4)$$

where h and w are the height and width of the original frame, respectively. In the experiment, u is up to 2.



264th frame 270th frame 2211th frame 2219th frame

Sample frames from video 121.mpg

600th frame 1080th frame 2590th frame 3491th frame

Sample frames from video 268.mpg

Figure 4.3 Examples of different geometrical editing methods within one video.

Given the t^{th} key frame, I can test whether it has a different location of AOI, by thresholding the d^A value with T_u. If the d^A value is less than T_u, it is determined to have a similar location of AOI. Firstly, I set the first key frame as the seed of the first cluster, and the first temporal unit starts. Secondly, the second key frame is screened by calculating $d^A(2,1)$. If it is determined to have a different location for AOI, it is set as the seed of the second cluster. Otherwise, it belongs to the first cluster and the second cluster remains empty. Thirdly, the third key frame is compared with the existing cluster centres. $d^A(3,1)$ is calculated when it is compared to the first cluster centre. If the second cluster is not empty, $d^A(3,2)$ is also calculated. With a smaller difference in AOI location (d^A value), if the third key frame is determined to have a similar AOI location, then it belongs to the corresponding cluster. Otherwise, it is set as the seed of a new cluster. If the current cluster number exceeds two, a new temporal unit starts. The subsequent key frames are processed similarly using the above criteria.

Once a cluster is determined, the final cluster centre is obtained by averaging the location of AOI from all key frames in the cluster. Then, the location of AOI from the key frames is replaced with the location of AOI from the cluster centre. Furthermore, the boundary areas outside the location of AOI are removed. In this way, the margin effect introduced by the change of geometry is minimized.

Extra information inserted areas are often characterised by static pixels. In a specified cluster, a difference image I_D of every two neighbouring frames is obtained. All the difference images are averaged to get a new image denoted as I_A. In the image I_A,

the spatial positions of static pixels are located by thresholding the pixel intensity with T_u. If the intensity value is less than T_u, a static pixel is found.

Video in video is a complicated transformation where a foreground video is combined with a background video. The video copy could be either the foreground video or the background video. The key issue to handle video in video is to locate the foreground video. The insertion of the foreground video occurs mainly in five positions, the top right, top left, bottom right, bottom left and centre of frame. Its scale usually varies from 50 percent to 20 percent of the original size. The proposed strategy for the detection of video in video is implemented in two steps. Firstly, the foreground video candidates are located using the edge information. Secondly, the final decision of foreground video is decided among candidates.

Vertical and horizontal discontinuities are present at the boundaries between the foreground video and background video. These are characterised by lines of edge points. Hence, the solution is to locate the lines with maximum edge points. The Sobel edge detector [107] is implemented to find horizontal and vertical edge points for its computation efficiency.

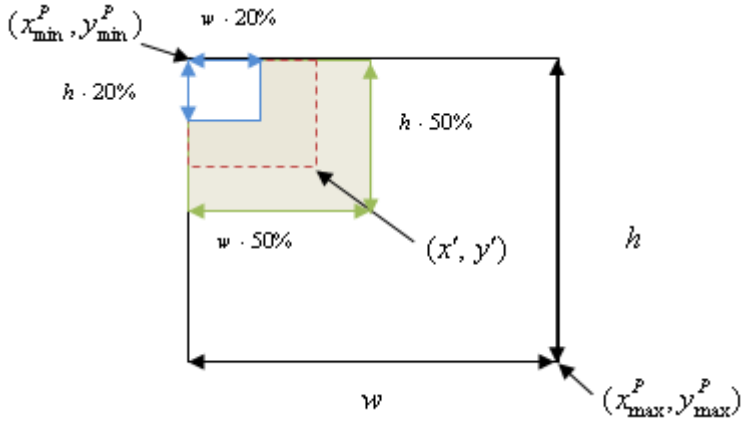
If pixel (x, y) is a vertical edge point, its value is set 1 in the vertical edge buffer $E_x(x, y)$. Otherwise, its value is set 0 in $E_x(x, y)$. Similarly, the corresponding value is set in the horizontal edge buffer $E_y(x, y)$. In addition, if pixel (x, y) is a static pixel, its value in $E_x(x, y)$ is set as 0 and the same for $E_y(x, y)$.

Hence, vertical and horizontal boundary lines could be located by finding the lines with the maximum accumulation value from $E_x(x, y)$ and $E_y(x, y)$. For example, to

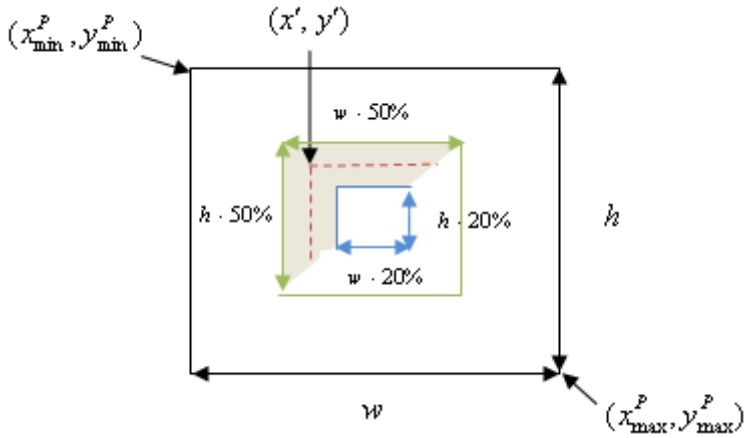
locate a foreground video at the top left, the search of the maximum accumulation value is defined as following:

$$G = \underset{\substack{x' \in (x^P_{\min}+h\cdot 20\%,\, x^P_{\min}+h\cdot 50\%),\\ y' \in (y^P_{\min}+w\cdot 20\%,\, y^P_{\min}+w\cdot 50\%)}}{\arg\max} \left(\sum_{x=x^P_{\min}}^{x'} \sum_{y=y^P_{\min}}^{y'} E_x(x,y) + E_y(x,y) \right) \cdot \frac{1}{w+h} \ . \qquad (4.5)$$

where $(x^P_{\min}, x^P_{\max})$ and $(y^P_{\min}, y^P_{\max})$ is the range of pixels on vertical and horizontal direction, after the removal of margins. G is a normalized value in the range from 0 to 1, which measures the percentage of edge points in a frame. The location of foreground video candidate at the top right, bottom left and bottom right of frame picture is similarly defined. As for the location of foreground video candidate at the centre, only the top and left boundary lines are searched to save computation load, since the left and right boundary lines are symmetrical, as are the top and bottom boundary line. The location process is illustrated in Figure 4.4.



Foreground video on the top left



Foreground video on the centre

Boundary lines are searched and located in the shadow areas. ------ *is the line being examined.*

Figure 4.4 Examples of the locations for video in video.

After the location process, five candidates for a foreground video are generated at the five positions (top left, top right, bottom left, bottom right and centre). The candidate whose G value, is the maximum and is larger than 15%, is chosen as the foreground video. The remaining part of the frame picture is determined to be the background video.

4.2.3 Signature Extraction and Matching

In the frame scaling down process, each key frame is classified into non-overlapping equal sized blocks, with 9 blocks in the vertical direction and 11 blocks in the horizontal direction. Hence, there are 99 pixels in the scaled down frames. The pixel intensities in the scaled down frame equal the averaged intensity values of the corresponding blocks. In order to measure the discrimination ability of a pixel, its significance is calculated as the percentage of non-static pixels in the corresponding block. If video in video is detected, this process applies to both the foreground and background video respectively.

An appropriate number of pixels need to be selected based on the significance values. Furthermore, the principle of maximum entropy is utilized to find the threshold for

the pixel selection. Assume there are f_s pixels whose significances have the value $s(s \in [0,1])$. Given a threshold, say Th_s, the probability for the non-selected pixels $P_n(s)$ and selected pixels $P_e(s)$ can be defined as (4.6) and (4.7), respectively.

$$P_n(s) = \frac{f_s}{\sum_{h=0}^{Th_s} f_h}, \quad 0 \le s \le Th_s, \tag{4.6}$$

$$P_e(s) = \frac{f_s}{\sum_{h>Th_s}^{1} f_h}, \quad Th_s < s \le 1. \tag{4.7}$$

where $\sum_{h=0}^{Th_s} f_h$ give the total number of pixels with significance in the range of $[0, Th_s]$. The entropies for these two classes are then given by (4.8) and (4.9).

$$H_n(Th_s) = -\sum_{s=0}^{Th_s} P_n(s) \log P_n(s), \tag{4.8}$$

$$H_e(Th_s) = -\sum_{s>Th_s}^{1} P_e(s) \log P_e(s). \tag{4.9}$$

The optimal threshold T_c for classification has to satisfy the following criterion function:

$$H(T_c) = \max_{Th_s=0,\ldots,1} \{H_n(Th_s) + H_e(Th_s)\}. \tag{4.10}$$

As a result, the frame mask is generated by marking the selected pixels as ones and the unselected pixels as zeros. This excludes the video temporal redundancy from my signature extraction.

The similarity between query video and target video could be observed in two aspects: temporal similarity between video groups and spatial similarity between frames.

Based on the extracted signatures, the query and target video matching strategy is implemented in four steps. Firstly, both target video and query video are classified into video groups. Secondly, the two-stage matching procedure is implemented on the video group level and frame level. Thirdly, linear regression is applied to the mapping sequence of frames. Finally, a decision is made using a SVM based on the properties of the linear regression result.

Motion information is utilized to classify query and target videos into video groups. Hence, motion activity between neighbouring scaled down frames, defined in (4.11), is calculated as the motion information measurement.

$$\left|\vec{M}\right| = \frac{1}{\hat{n}} \sum_{k=1}^{\hat{n}} \left| \hat{I}_k^t - \hat{I}_k^{t+1} \right|. \tag{4.11}$$

where $\hat{I}_k^t$ is the intensity value of the k^{th} pixel in the t^{th} scaled down frame and $\hat{n}$ is the total number of pixels in a scaled down frame. The value of $\left|\vec{M}\right|$ is classified by two levels into: low motion, medium motion and high motion. My scheme for the motion level classification is given by:

$$\hat{M} = \begin{cases} 0 \quad (low \quad motion), \quad if \quad 0 \le \left|\vec{M}\right| < M_l \\ 1 \quad (medium \quad motion), \quad if \quad M_l \le \left|\vec{M}\right| < M_h \\ 2 \quad (high \quad motion), \quad if \quad M_h \le \left|\vec{M}\right| \end{cases}. \tag{4.12}$$

The definition of M_l and M_h depends on the application. However, a simple way to define them automatically is proposed based on the training frames, assuming that most observations have medium motion. An array is generated containing the motion

information for all frames, and its mean μ_M and standard deviation σ_M are computed. Thresholds M_l and M_h are determined according to:

$$\begin{aligned} M_l &= \mu_M - k \cdot \sigma_M \\ M_h &= \mu_M + k \cdot \sigma_M \end{aligned}. \tag{4.13}$$

where k controls the spread of medium motion. In the experiment, k was set as 2. Then, consecutive frames with the same motion level are clustered as a video group.

In order to compare the similarity between query and target video groups, representative frame and representative frame mask are generated from the video group. The representative frame is obtained by averaging pixel intensities from all scaled down frames inside the same video group. The representative frame mask is generated by averaging the values from frame masks of the query video group. The value 0 in the representative frame mask indicated that the pixel is temporally redundant information. Hence, if the value in the representative frame mask is 0, the corresponding pixel intensity is set to -1 in both representative frames from query video and target video. The intensity value -1 indicates it is a pixel with temporal redundancy and will not be taken into consideration for subsequent processing. This setting has the advantage of not affecting the rank order relation among other pixels with useful information. Rank order of pixels is generated using the quick sort method [108]. The similarity between query and target video group is measured using (4.14).

$$\beta = 1 - \frac{1}{99 \times 99} \sum_{k=1}^{99} \left| R_k^q - R_k^t \right|. \tag{4.14}$$

where R_k^q and R_k^t are the rank order of the k^{th} pixel in frames being examined from query and target videos. In this case, the frames are query and target representative frames. For a

query video group, its matched target video group is defined as the one with the maximum value of β. Furthermore, rank orders of pixels are calculated from scaled down frames inside matched query and target video groups. For a scaled down frame inside a query video group, its matched target frame is defined as the one with maximum value of β. All the matched frame pairs constitute the mapping sequence of query and target video groups. This two-stage matching procedure ensures the robustness of the framework to temporal editing changes, such as video copies generated from segments of target videos and frame dropping in the query video.

The mapping sequence of frame pairs is examined by linear regression. This is implemented by fitting the curve represented by (4.15) to verify the order property of query and target frame numbers from the mapping sequence.

$$y = kx + b\,. \tag{4.15}$$

where k is slope of the line and b is the y-intercept. The ideal fitting curve of frame numbers is $y = x$. The increasing tendency of frame number' order is characterised by the positive sign of the slope k, despite the frame rate change in query video. The y-intercept b reflects the transfer error for linear fitting. The residual sum defined in (4.16) reflects the accumulative error of linear fitting.

$$\chi = \sum_{k=1}^{n} (N_t^k - N_f^k)^2\,. \tag{4.16}$$

where, for the k^{th} query frame, N_t^k is the frame number of its matched target frame and N_f^k is the frame number obtained by fitting. n is the total number of matched frame pairs.

The linearity of fitting data defined in (4.17) reflects the non-overlapped mapping from query frames to target frames.

$$\delta = 1 - \chi \Big/ \sum_{k=1}^{n} (N_t^k - \overline{N}_t)^2 . \quad (4.17)$$

where $\overline{N}_t$ is the mean value of frame numbers from target frames. Moreover, $\overline{\beta}$, the overall similarity value, generated by averaging similarity values from all matched frame pairs, reflects the spatial similarity between target video and query video.

The final decision is made using SVM based on the five features, including slope k, the y-intercept b, residual sum χ, linearity δ and overall frame similarity $\overline{\beta}$. The evaluation contribution of the features is summarized in table 4.1.

Table 4.1 The evaluation contribution of the features

Feature	Description	Evaluation contribution
k	Slope	Reflecting the increasing tendency of frame numbers' order
b	The y-intercept	Reflecting the transferring error of linear fitting
χ	Residual sum	Reflecting the accumulative error of linear fitting
δ	Linearity	Reflecting the non-overlapped mapping from query frames to target frames
$\overline{\beta}$	Overall frame similarity	Reflecting the spatial similarity between target video and query video.

4.2.4 Experimental Results

Extensive experiments were carried out to evaluate the proposed algorithm using the data set from TRECVID [95] 2008 content-based copy detection task. The whole data set includes 218.88 hours of target videos (in total 18,723,596 frames) and 2010 query video clips that are generated by 10 types of transformation from the original videos. The 10 types of the transformation are described in table 4.2. The video format is 352×288 pixels. The experiment using the proposed algorithm was implemented on a PC with windows XP operating system, Intel Core 2 CPU, 6400 @ 2.13GHz and 1.99 GB of RAM.

Table 4.2 Description of transformation types

Transformation type	Sub-transformation	Description
T1: Cam Cording		Change of angle
T2: Picture in picture Type 1		The original video is inserted in a front of a background video
T3: Insertions of pattern		Random pattern inserted
T4: Strong re-encoding		Change of resolution and bit-rate
T5: Gamma		Change of gamma
T6 and T7: decrease in quality	Blur	Blur added
	Gamma	Change of gamma
T6: 3 sub-transformations are randomly selected and combined	Frame dropping	Change of frequency
	Contrast	Change of contrast
	Compression	Change of resolution and bit-rate
T7: 5 sub-transformations are randomly selected and combined	Ratio	Letterbox
	Noise	White noise
T8 and T9: post production transformation	Crop	Pixels covered
	Shift	In the vertical and horizontal direction
	Contrast	Change of contrast
T8: 3 sub-transformations are randomly selected and combined	Caption	Text insertion
	Flip	Vertical mirroring
	Insertion of pattern	Random pattern inserted
T9: 5 sub-transformations are randomly selected and combined	Picture in picture type 2	The original video is in the background
T10: Combination of everything		5 transformations among all the transformations described above (T1-T9) are randomly selected and applied

Two state-of-art algorithms from [67] and [68] were also implemented on the same data set for bench marking purposes. Table 4.3 lists all the experimental results using the proposed algorithm in comparison with the two benchmarks [67], [68]. From the experimental results, it can be seen that the proposed algorithm achieves satisfactory performances in terms of recall, precision and F1 rate, which are as following:

$$\begin{aligned} recall &= \frac{N_{TP}}{N_{TP} + N_{FN}} \\ precision &= \frac{N_{TP}}{N_{TP} + N_{FP}} \\ F1 &= \frac{2 \times recall \times precision}{recall + precision} \end{aligned} \quad . \tag{4.18}$$

where N_{TP} is the number of true positives, that video copies are correctly detected by the algorithms. N_{FP} is the number of false positives that video copies are incorrectly detected, while N_{FN} is the number of false negatives that video copies are missed by algorithms.

Table 4.3 Experimental results of method from [67], [68] and the proposed algorithm

Transfor-mation type	**Shortest path search** [67]			**Sequential search** [68]			**Proposed algorithm**		
	Recall	**Precision**	**F1**	**Recall**	**Precision**	**F1**	**Recall**	**Precision**	**F1**
T1	0.0597	0.0143	0.0230	0.4552	0.4266	0.4404	0.8833	0.8346	0.8583
T2	0.0299	0.0111	0.0162	0.1940	0.1926	0.1933	0.9237	0.9646	0.9437
T3	0.0746	0.0291	0.0418	0.3358	0.3782	0.3557	0.9592	0.8785	0.9171
T4	0.0746	0.0195	0.0310	0.5373	0.4832	0.5088	0.9845	1.0000	0.9922
T5	0.0896	0.0243	0.0382	0.5373	0.4675	0.5000	1.0000	0.9524	0.9756
T6	0.0821	0.0251	0.0385	0.5373	0.5496	0.5434	0.9672	0.9752	0.9712
T7	0.0970	0.0255	0.0404	0.4403	0.5566	0.4917	0.9217	0.9815	0.9507
T8	0.0448	0.0117	0.0185	0.2090	0.2593	0.2314	0.8283	0.8723	0.8497
T9	0.0299	0.0136	0.0186	0.1866	0.2252	0.2041	0.7973	0.8082	0.8027
T10	0.0373	0.0127	0.0189	0.2313	0.2696	0.2490	0.8506	0.8506	0.8506
Total	0.0619	0.0188	0.0288	0.3664	0.3863	0.3761	0.9196	0.9187	0.9192

The method from [67] uses shortest path search model to find the video copy. In the pre-processing stage, the candidate segments from query clips are sampled from the head and tail parts of video. This limitation hinders the detection of the mixed video copy, where the video copy is only part of query clip instead of whole query clip. This is the main reason causing the low detection performance of the method from [67], since many true video copies are skipped in the pre-processing stage. In addition, the method from [68] utilizes sequential search for the detection of video copy. This method ignores frame rate changes in query clips and is time-consuming compared to the proposed method and the method from [67]. Both methods from [67] and [68] use simple features, i.e. block rank order as the signatures for video copy search. There is no analysis of input videos, so different kinds of editing effects are not taken into consideration.

The processing time of the methods from [67], [68] and the proposed method are measured and summarized in table 4.4. The time for feature extraction is measured by the

mean time of extracting signatures from all query clips. The time for the search process is measured by the mean time spent on comparing one query with one target clip. In the feature extraction process, the proposed algorithm analyzed the different kinds of video distortions. This is why it takes longer time in the proposed algorithm for the feature extraction. In the search process, the proposed algorithm used the two-step matching which makes the search efficient and effective. By a trade-off of the feature extraction and search process, the run-speed of proposed algorithm is in the middle level.

Table 4.4 Processing time of method from [67], [68] and the proposed method

Time (seconds)	**Shortest path search** [67]	**Sequential search**[68]	**Proposed method**
Feature extraction	11.8633	11.8519	32.2069
Search process	0.3592	37.6325	2.5173
In total	12.2225	49.4844	34.7242

4.3 Summary

The proposed strategy for video copy detection is implemented in three major procedures, including video distortion analysis, signature extraction and query and target video matching. Firstly, a dedicated video distortion analysis is implemented for input videos, so that different kinds of complex editing effect are well tackled. Secondly, key frames are scaled down to extract simple signatures, and the frame mask is generated adaptively to reduce video temporal redundancy. Thirdly, linear regression and SVM are utilized in the matching and decision making process to guarantee a robust copy retrieval result.

Chapter 5

Video Interpretation

5.1 Introduction

It is a fundamental requirement to extract semantic events and highlights from video sequences for the interpretation of video content. In this chapter, two approaches are proposed to address this problem. In Section 5.2, a knowledge-supported approach is presented to segment videos and extract semantic highlights. Firstly, luminance, chrominance and motion features are extracted directly in the compressed domain. Secondly, video shots are segmented based on knowledge-supported rules. Thirdly, camera motion is estimated using motion vectors and skin patches are segmented in the video frames. Finally, zoom-in human objects are extracted as semantic highlights.

In Section 5.3, a trajectory-based method is proposed to detect human fighting for movie annotation. Firstly, the trajectory of points of interest is constructed based on KLT tracker. Secondly, global motion is estimated to get high motion frames using the trajectory record and camera motion is classified via the tracking information. Thirdly, the behaviour of points of interest is labelled by a voting scheme. Hence, the main movement and direction of the same kind of points of interest are calculated for the detection of human fighting.

5.2 Knowledge-Supported Segmentation and Semantic Contents Extraction from MPEG Videos

In the proposed algorithm, the compressed streams of MPEG videos are used as the input data. In the MPEG scheme, the embedded motion estimation and compensation of macro-blocks could be further exploited for the shot detection and camera motion estimation. Each macro-block consists of 16×16 pixels, which is further divided into four 8×8 sub-blocks for application of the DCT and IDCT. In most video sequences, the format 4:2:0 chromatic is applied, which indicates that there are four Y components, one Cb component and one Cr component in a macro-block.

5.2.1 Feature Extraction

Given an MPEG compressed video input, a DC image sequence is extracted from its corresponding video frames. If the original video frame size is $W \times H$, the DC image will have the size $\frac{W}{8}\times\frac{H}{8}$. The total number of Y components in each DC image is represented by N_y and $N_y = \frac{W}{8}\times\frac{H}{8}$. According to the 4:2:0 chromatic format, the total numbers of Cb and Cr components both equal $\frac{1}{4}N_y$, which also equals the total number of macro-blocks in each frame. At the i^{th} DC image, the Y, Cb and Cr component are denoted as Y_i, Cb_i and Cr_i. Hence, the luminance difference between the i^{th} and j^{th} frame is defined as

$$D_y(i,j) = \frac{1}{N_y}\sum_{n=1}^{N_y}\left|Y_i^n - Y_j^n\right|. \qquad (5.1)$$

Similarly, the chrominance differences for the Cb and Cr components are defined as $D_{Cb}(i,j)$ and $D_{Cr}(i,j)$, respectively. In the i^{th} DC image, given the MPEG motion vector components $V_x^n(i)$ and $V_y^n(i)$ from the n^{th} macro-block in the vertical and horizontal direction respectively, the motion strength M_i^n is defined as follows:

$$M_i^n = \left|V_x^n(i)\right| + \left|V_y^n(i)\right|, \tag{5.2}$$

In B frames, the motion strength of the macro-block which is bi-directional coded is calculated by choosing the larger one from the forward and backward motion strengths. Hence, the mean M_i^μ and standard deviation M_i^σ of the motion strength in the i^{th} DC image are defined as follows:

$$M_i^\mu = \frac{1}{0.25N_y}\sum_{n=1}^{0.25N_y} M_i^n, \tag{5.3}$$

$$M_i^\sigma = \sqrt{\sum_{n=1}^{0.25N_y}\left(M_i^n - M_i^\mu\right)^2}, \tag{5.4}$$

The macro-blocks with sound values of motion strength M_i^n are chosen by the following voting scheme to estimate the overall motion magnitude for the current DC image.

$$\beta_n = \begin{cases} 1 & if \quad M_i^n \geq M_i^\mu - M_i^\sigma \\ 0 & otherwise \end{cases}, \tag{5.5}$$

Here, β_n is the voting score for the n^{th} macro-block. If β_n is one, the corresponding motion strength is counted for the calculation of motion magnitude. Otherwise, it is excluded as an outlier. Furthermore, the overall motion magnitude $\widehat{M}_i$ is calculated based on the selected macro-blocks as follows:

$$\widehat{M}_i = \frac{1}{\alpha_i} \sum_{n=1}^{0.25N_y} \beta_n M_i^n, \tag{5.6}$$

$$\lambda_i = \frac{\alpha_i}{0.25N_y}. \tag{5.7}$$

where α_i is the total number of selected macro-blocks and λ_i indicates the selection rate. In I frames, the motion magnitudes are copied from those of the latest output B frames since there are no inter-code macro-blocks in I frames.

5.2.2 Shot Detection

From observations on more than 50000 frames containing several hundreds of shots, it was recognized that the disturbing factors for shot detection include large object movement, camera motion and sudden light change. Hence, the proposed knowledge-supported rules are summarized as follows for the detection of both cuts and typical gradual transitions.

- Rule 1: if $D_y(i-d,i) > T_y \,||\, D_{Cb}(i-d,i) > T_{Cb} \,||\, D_{Cr}(i-d,i) > T_{Cr}$, a potential shot change is claimed. Here, T_y, T_{Cb} and T_{Cr} are the thresholds for the difference of Y, Cb and Cr components between the i^{th} and $(i-d)^{th}$ frames.
- Rule 2: if Rule 1 is not satisfied, the screened frame is stepped forward by d frames to examine the $(i+d)^{th}$ frame. Then, Rule 1 is repeated until all the frames in the video are addressed.
- Rule 3: if Rule 1 is satisfied, further examination is carried out to decide whether the potential shot change is a cut or gradual transition. If

$D_y(i_0, i_0+1) > T_y \,||\, D_{Cb}(i_0, i_0+1) > T_{Cb} \,||\, D_{Cr}(i_0, i_0+1) > T_{Cr}$, a cut candidate is determined. Here, the i_0^{th} frame is any frame between the i^{th} and $(i-d)^{th}$ frames.

- Rule 4: if Rule 3 is not satisfied, the accurate boundary of the gradual transition needs to be decided. If $D_y(i\_, i) > T_y \,||\, D_{Cb}(i\_, i) > T_{Cb} \,||\, D_{Cr}(i\_, i) > T_{Cr}$, a gradual transition is found. Here, $i\_$ is initialized as $(i-d)$ and updated by decreasing d each time until the test is positive or the length limit for gradual transition is reached. If the test is positive, a gradual transition is found between the $i\_^{th}$ and i^{th} frames.
- Rule 5: if Rule 3 is satisfied, a verification test is carried out to avoid the effect of flashing light. If $D_y(i_0 - t, i_0 + t) > T_y \,||\, D_{Cb}(i_0 - t, i_0 + t) > T_{Cb} \,||\, D_{Cr}(i_0 - t, i_0 + t) > T_{Cr}$, a cut is determined. Here, t equals to $0.5d$.

The threshold T_y is determined by the primary constant C_y, which measures the difference of the Y component, and the selection rate λ_i. Their relationship is presented as follows:

$$T_y = C_y \cdot \lambda_i \tag{5.8}$$

Therefore, T_y is adaptive to the overall motion magnitude. In the experiment, D_y, D_{Cb} and D_{Cr} are scaled in the range from 0 to 1. C_y and d are set as 0.09 and 6, respectively. T_{Cb} and T_{Cr} are similarly determined.

5.2.3 Camera Motion Estimation

In the algorithm proposed by Tan et al. [70], the projective camera model with 6 parameters is summarized in (5.9) and (5.10). The parameter p_1 is an indicator for the detection of camera zoom.

- If the value of p_1 is larger than one, camera zoom-in is detected.
- If the value of p_1 is smaller than one, camera zoom-out is detected.
- Otherwise, there is no camera zoom.

$$x_i^n = \frac{p_1 \cdot x_{i-1}^n + p_2 \cdot y_{i-1}^n + p_3}{p_5 \cdot x_{i-1}^n + p_6 \cdot y_{i-1}^n + 1}. \tag{5.9}$$

$$y_i^n = \frac{-p_2 \cdot x_{i-1}^n + p_1 \cdot y_{i-1}^n + p_4}{p_5 \cdot x_{i-1}^n + p_6 \cdot y_{i-1}^n + 1}. \tag{5.10}$$

Here, $\left(x_i^n, y_i^n\right)$ is the spatial position of the n^{th} motion vector in the i^{th} frame.

In the experiment, it was found that the proposed method in [70] is sensitive to noise and produced lots of false positives. Hence, the smoothness condition [109] is introduced to filter out the motion vectors from the outlier macro-blocks.

5.2.4 Skin Detection

The proposed method for skin detection is mainly based on the posterior probability in YCbCr colour space. In the training stage, given the pixels $\{X, X = (Y, Cb, Cr)\}$ labelled as skin, the histogram of their YCbCr values is constructed and denoted as $H_s(X)$. Similarly, given the pixels labelled as non-skin, the corresponding histogram of YCbCr is obtained and represented as $H_{\bar{s}}(X)$. Hence, the likelihood function of the observed

YCbCr of the pixel being skin is represented as $P(X \mid S)$. Similarly, the likelihood function of the observed YCbCr of the pixel being non-skin is denoted as $P(X \mid \overline{S})$. Furthermore, the likelihood function $P(X \mid S)$ and $P(X \mid \overline{S})$ are determined as following:

$$P(X \mid S) = H_s(X) \tag{5.11}$$

$$P(X \mid \overline{S}) = H_{\overline{s}}(X) \tag{5.12}$$

The histogram of the YCbCr values of all training pixels is obtained and denoted as $H(X)$. Among all the labelled pixels, the ratio of skin pixels and non-skin pixels are calculated as $R(S)$ and $R(\overline{S})$. Therefore, the prior probabilities of the YCbCr values, skin and non-skin are determined as follows:

$$P(X) = H(X) \tag{5.13}$$

$$P(S) = R(S) \tag{5.14}$$

$$P(\overline{S}) = R(\overline{S}) \tag{5.15}$$

The classifier for the skin and non-skin pixel is proposed as follows according to the Bayesian rule and maximum posterior principle.

$$C^* = \arg\max_{C \in (S, \overline{S})} P(C \mid X) \tag{5.16}$$

$$P(S \mid X) = \frac{P(X \mid S) \cdot P(S)}{P(X)} \tag{5.17}$$

$$P(\overline{S} \mid X) = \frac{P(X \mid \overline{S}) \cdot P(\overline{S})}{P(X)} \tag{5.18}$$

5.2.5 Video Highlights Extraction

The proposed video highlight extraction is implemented in the following steps:

- Shot boundaries are determined for both cuts and gradual transitions.
- Camera zoom-in is estimated inside each detected video shot.
- The skin areas are located and human objects are found based on the skin areas.
- The highlight frames featured with camera zoom-in and human objects are extracted.

5.2.6 Results

The description of the test videos is shown in table 5.1. There are 6 test videos with 68225 frames and 227 shots, including films, sports and news. The experimental results are illustrated in table 5.2.

Table 5.1 Description of the test video sequences

Sequence	(a)	(b)	(c)	(d)	(e)	(f)	**Sum**
Category	Film	Film	Sports	Sports	News	News	N/A
Frames	13281	6900	9605	22162	7554	8727	68225
Shots	57	29	31	76	21	23	227
Highlights	3	2	3	5	2	3	18

Table 5.2 Performance on shot detection and highlights extraction

Sequence		(a)	(b)	(c)	(d)	(e)	(f)	**Average**
Shot Detection	**Precision**	83.3%	80.2%	84.9%	84.2%	81.1%	79.1%	81.7%
	Recall	91.5%	92.0%	93.8%	92.4%	90.2%	87.6%	91.2%
Highlights extraction	**Precision**	67.7%	67.7%	75.0%	80.0%	100.0%	75.0%	77.6%
	Recall	67.7%	100.0%	100.0%	80.0%	100.0%	100.0%	91.3%

5.3 Trajectory Based Human Fighting Detection for Movie Annotation

In the approach presented by Lin et al. [88], since the background in the video sequence is fixed, the segmentation of human object can easily be carried out. All the human objects are located by bounding boxes, so the spatial positions and sizes of human beings are known. However, in real-world movies, the background is changing due to camera movements and the locations of human objects are very difficult to determine. Hence, a trajectory based approach is presented to address the problem of human activity recognition in general movies. The proposed algorithm for human fighting detection is carried out in four steps. Firstly, shot boundary detection is implemented for the robust tracking of points of interest via the Kanade-Lucas-Tomasi (KLT) algorithm and the tracking record is used to build the trajectory of points of interest. Secondly, global motion is estimated based on the trajectory features. Thirdly, camera motion is measured and classified into zoom, pan, tilt and mixed motion via the movements of points of interest. Fourthly, the behaviours of points of interest are measured by fuzzy scores and labelled as background points, active body points and inactive body points by a voting scheme. Hence, the main movement and direction of the three kinds of points of interest are used to detect human fighting by SVM.

5.3.1 Trajectory Building for Points of Interest

There exist various approaches towards points of interest tracking. Generally, points of interest are selected to measure the temporal differences between consecutive frames. Linking strategies are applied to find the correspondences for points of interest and minimize the cost functions. One of the most popular approaches is the Kanade-Lucas-

Tomasi (KLT) tracker. The KLT tracker is based on the early work of Lucas and Kanade [110], which is fully developed by Tomasi and Kanade [111] and clearly explained in the paper by Shi and Tomasi [112]. Firstly, points of interest are located by examining the spatial intensity gradient of frames. Then, points of interest are tracked by using a Newton-Raphson iteration method. The KLT tracker is reliable for the real-time tracking of points of interest.

The majority of points of interest lose track at the presence of shot cuts due to the dramatic change of video contents. Hence, the shot cut detection [113] is implemented to ensure the robust tracking of points of interest in subsequent frames. At the presence of gradual transitions of shot changes, the lost track rate of points of interest is tolerable so the detection of gradual transitions is not implemented on the balance of computation cost and tracking accuracy. In the technique implementation, the KLT tracker is restarted at the first frame of each shot. In the experiment, 120 points of interest were extracted at the first frame of every shot and tracked in the subsequent frames. However, due to strong movements or occlusions, some points of interest are inevitably lost track of. Therefore, new points of interest with high texture scores are extracted to replace those lost and maintain the fixed number of points of interest. Based on the tracking record of points of interest, three decisions can be made to build the trajectory of points of interest:

- Matching only in the future: start of a new trajectory
- Matching only in the past: end of a trajectory
- Matching in the future and in the past: add the point to an existing trajectory

After the trajectory building process, the following parameters for the i^{th} trajectory are calculated:

- Time code of the beginning and end: $[tc_{in}^{i}, tc_{out}^{i}]$
- Distance of the spatial movement between neighbouring frames: $\{d_l^i, l \in (tc_{in}^{i}+1, tc_{out}^{i})\}$

Hence, the life period L^i and the averaged distance D^i are defined as follows to describe the length and movement of the i^{th} trajectory.

$$L^i = tc_{out}^{i} - tc_{in}^{i} + 1. \tag{5.19}$$

$$D^i = \frac{1}{tc_{out}^{i} - tc_{in}^{i}} \sum_{l=tc_{in}^{i}+1}^{tc_{out}^{i}} d_l^i \tag{5.20}$$

Furthermore, the i^{th} trajectory is defined as follows:

$$T^i = \{L^i, D^i, s^i\}, s^i \in \{n_b, ..., n_l, ..., n_e\} \tag{5.21}$$

Here, S is the set of frames whose time codes are covered by the i^{th} trajectory. n_b and n_b are the first and last frame in the frame set S. There are 120 points of interest in the frame n_l, so there are 120 trajectories across it.

5.3.2 Global Motion Estimator

In a movie clip, fighting frames always contain high motions due to strong body movements or quick camera movements. Therefore, high motion is a very straightforward indicator for human fighting. From observations, high motion frames are featured by a high lost track rate for points of interest, small values of life period L^i and large values of averaged distance D^i. Based on the trajectory definition, in the n^{th} frame, the histogram of the life period L^i and averaged distance D^i are calculated as $H_n(L^i)$ and $H_n(D^i)$. In

the experiment, the two histograms contain 10 bins each. The lost track rate for points of interest in the n^{th} frame, P_n is also measured. Hence, the feature vector $\{H_n(L^i), H_n(D^i), P_n\}$ is formed to represent the global motion in the n^{th} frame and SVM is utilized to classify the high motion frames and ordinary frames.

In the training set, the movie frames with high motions are labelled as 1, while the rest frames are labelled as -1. Then, the labelled feature vectors are trained by SVM in an off-line process and the global motion estimator is obtained through the training. In the experiment, LIBSVM [114] was implemented to do a "grid-search" for the best parameters C and γ of RBF kernel function using cross-validation. Here, C is the penalty parameter of error and γ is the parameter of kernel width. Basically, pairs of (C, γ) are tried and the one with the best cross-validation accuracy is picked. It is found that trying exponentially growing sequences of C and γ is a practical method to identify good parameters (for example, $C = 2^{-5}, 2^{-3}, ..., 2^{15}, \gamma = 2^{-15}, 2^{-13}, ..., 2^{3}$). Furthermore, the grid-search is parallelized because each (C, γ) is independent. In the data prediction stage, ordinary frames are filtered out through the global motion estimator.

5.3.3 Camera Motion Classification

Camera movement is the major disturbing factor in the detection of human fighting. Based on the 6-parameter projective camera model proposed in [70], which is repeated in (5.22-5.23), camera movement such as zoom-in (out), pan and tilt are estimated by parameters p_1, p_3 and p_4 determined in (5.24-5.25). The spatial positions of the k^{th} point of interest in the $(n-1)^{th}$ and n^{th} frames are represented as $\{(x_{n-1}^k, y_{n-1}^k), (x_n^k, y_n^k)\}$. Hence,

the camera motion classification is conducted based on the spatial positions of the points of interest.

$$x_n^k = \frac{p_1 \cdot x_{n-1}^k + p_2 \cdot y_{n-1}^k + p_3}{p_5 \cdot x_{n-1}^k + p_6 \cdot y_{n-1}^k + 1}. \tag{5.22}$$

$$y_n^k = \frac{-p_2 \cdot x_{n-1}^k + p_1 \cdot y_{n-1}^k + p_4}{p_5 \cdot x_{n-1}^k + p_6 \cdot y_{n-1}^k + 1}. \tag{5.23}$$

$$p_1 = \frac{\sum_{k=1}^{N} \left(w_n^k - \overline{w}_n\right)^T \left(w_{n-1}^k - \overline{w}_{n-1}\right)}{\sum_{k=1}^{N} \left\| w_{n-1}^k - \overline{w}_{n-1} \right\|^2}. \tag{5.24}$$

$$\begin{pmatrix} p_3 \\ p_4 \end{pmatrix} = \frac{\overline{w}_n}{p_1} - \overline{w}_{n-1} = \frac{1}{p_1} \begin{pmatrix} \overline{x}_n^k \\ \overline{y}_n^k \end{pmatrix} - \begin{pmatrix} \overline{x}_{n-1}^k \\ \overline{y}_{n-1}^k \end{pmatrix} \tag{5.25}$$

where $w_n^k = \left(x_n^k, y_n^k\right)^T$, $w_{n-1}^k = \left(x_{n-1}^k, y_{n-1}^k\right)^T$, $\overline{w}_n = \frac{1}{N}\sum_{k=1}^{N} w_n^k = \begin{pmatrix} \overline{x}_n^k \\ \overline{y}_n^k \end{pmatrix}$ and $\overline{w}_{n-1} = \frac{1}{N}\sum_{k=1}^{N} w_{n-1}^k = \begin{pmatrix} \overline{x}_{n-1}^k \\ \overline{y}_{n-1}^k \end{pmatrix}$. Here, N is the total number of tracked points of interest.

Camera motions are estimated using the values of p_1, p_3 and p_4. The decision making process is summarized in the following table. Here, a parameter T_c is used for robust thresholding and it is set to 0.05 in the experiment.

Table 5.3 Determination of camera motion type

Camera motion type	Condition
Zoom in	$p_1 > 1 + T_c$
Zoom out	$p_1 < 1 - T_c$
No zoom	$1 - T_c \le p_1 \le 1 + T_c$
Pan	$\lvert p_3 \rvert > 10T_c \wedge \lvert p_4 \rvert < 10T_c$
Tilt	$\lvert p_3 \rvert < 10T_c \wedge \lvert p_4 \rvert > 10T_c$
Pan & tilt	$\lvert p_3 \rvert > 10T_c \wedge \lvert p_4 \rvert > 10T_c$
No pan or tilt	$\lvert p_3 \rvert \le 10T_c \wedge \lvert p_4 \rvert \le 10T_c$

5.3.4 Behaviour Labelling of Points of Interest

The behaviour of the k^{th} point of interest in the n^{th} frame at the spatial position (x_n^k, y_n^k), can be analyzed by the crossing trajectory T^i. Hence, the bin value (in the range from 0 to 9) of D^i in $H_n(D^i)$ is defined as *movement* of the k^{th} point of interest. Similarly, the bin value of L^i in $H_n(L^i)$ is defined as *length* of the k^{th} point of interest. In order to measure their relative quantities, three fuzzy sets *small*, *medium* and *large* are established for the linguistic variables *length* and *movement*, respectively. The membership functions of *small* and *large* are chosen as triangle, and the membership function of *medium* is chosen as trapezoid. This is illustrated in figures 5.1 and 5.2. Given the numerical values of L^i and D^i, the degrees of membership of *length* are calculated as μ_S^L, μ_M^L and μ_L^L. Similarly, the degrees of membership of *movement* are calculated as μ_S^M, μ_M^M and μ_L^M.

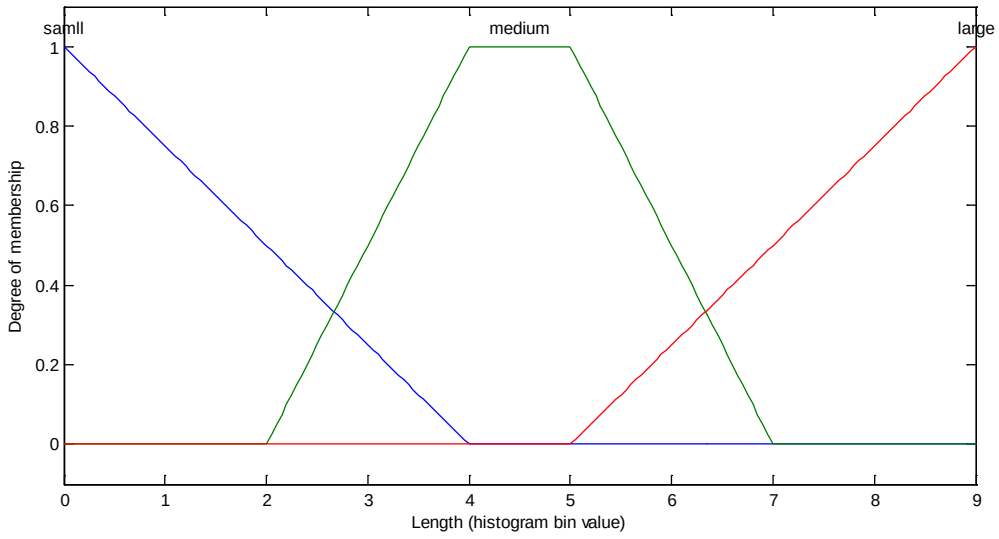


Figure 5.1 Fuzzy sets of *Length*

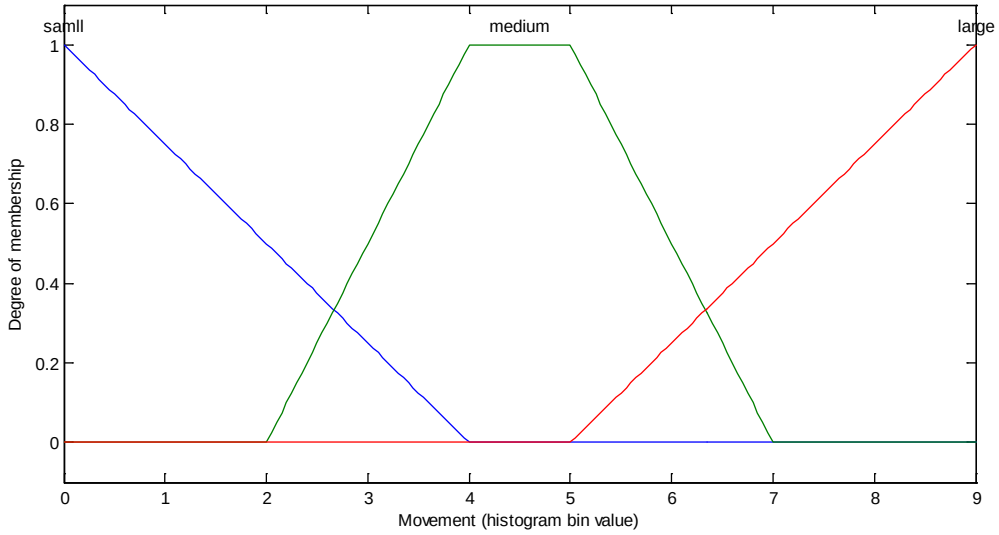


Figure 5.2 Fuzzy sets of *Movement*

Based on the behaviour trends of points of interest, they are classified into three kinds: background points, active body points and inactive body points. The background points are persistent and motionless and have *small* values of *movement* and *large* values of *length*. The active body points are moving fast and usually disappear quickly due to occlusion. They have *large* values of *movement* and *small* values of *length*. The inactive body points in essence can be seen as the transition points in between, which have *medium* values of *movement* and *medium* values of *length*. Their characteristics and the fuzzy score calculation are defined in table 5.4.

Table 5.4 Behaviour labelling of points of interest

Types of interest points	Movement	Length	Fuzzy score	
Background points	Small	large	$\mu^B = \mu_S^M \cdot \mu_L^L$	(5.26)
Active body points	large	Small	$\mu^A = \mu_L^M \cdot \mu_S^L$	(5.27)
Inactive body points	Medium	Medium	$\mu^I = \mu_M^M \cdot \mu_M^L$	(5.28)

Furthermore, a voting scheme is implemented for labelling the behaviour of the interest points as following.

$$t^* = \arg \max_{t \in (B,A,I)} (\mu^t) . \quad (5.29)$$

Here, t^* is the label that maximizes the fuzzy score of the corresponding behaviour of the interest point.

After the point labelling process, the percentage of the background points, active body points and inactive body points are calculated as P_n^B, P_n^A and P_n^I. The movement and direction of the k^{th} point of interest from the $(n-1)^{th}$ to n^{th} frames are measured as M_n^k and R_n^k. The averaged movement and direction of the background points, active body points and inactive body points are obtained as (M_n^B, R_n^B), (M_n^A, R_n^A) and (M_n^I, R_n^I). In the meanwhile, the percentage of the number of tracked active body points is calculated as P_n^T. Finally, the feature vector $\{P_n^B, P_n^A, P_n^I, P_n^T, M_n^B, R_n^B, M_n^A, R_n^A, M_n^I, R_n^I\}$ is classified by a specific SVM according to the camera motion type for the human fighting detection. Hence, multiple SVMs are trained to detect human fighting in terms of different camera motion types.

5.3.5 Experimental Results

To evaluate the proposed algorithm, extensive experiments were carried out on a number of fighting movie clips. The resolution of the test sequences is 320×240 pixels. The computing environment used included: (i) a PC with 1.73GHz CPU, 512MB memory and windows XP operating system; (ii) Microsoft VC++ 6.0 programming platform. To evaluate the performance of the proposed algorithm, the following measurements are adopted for quantitative evaluations [96, 97]:

$$\mathrm{Re}\,call = N_C (N_C + N_M)^{-1}, \quad (5.30)$$

$$\Pr ecision = N_C (N_C + N_F)^{-1} \tag{5.31}$$

$$F1 = 2 \cdot \mathrm{Re}\, call \cdot \Pr ecision (\mathrm{Re}\, call + \Pr ecision)^{-1}. \tag{5.32}$$

where N_c is the number of correctly detected shot boundaries, N_M is the number of missed shot boundaries, and N_F is the number of falsely detected shot boundaries. The higher these ratios are, the better the performance. The detection results are shown in table 5.5.

Table 5.5 Human fighting detection results under various camera motions

Camera motion type	Precision	Recall	F1
No camera motion	0.9251	0.9311	0.9281
Pan	1	1	1
Tilt	0.9259	1	0.9615
Zoom	0.8333	0.6383	0.7229
Zoom & Pan	0.8333	0.5882	0.6897
Zoom & Tilt	0.6667	1	0.8
Overall	0.9033	0.9013	0.9023

5.4 Summary

A knowledge-supported approach is presented to segment videos and extract semantic highlights. Compressed domain features are extracted including luminance, chrominance and motion. Then, the input video is segmented into shots using knowledge-supported rules. Zoom-in highlights are determined by the estimation of camera motion and the segmentation of skin patches.

In addition, a trajectory based approach is proposed to detect human fighting for movie annotation. The KLT tracking record is utilized to build the trajectories of points of interest and a global motion estimator is constructed based on the trajectory features. Then, camera motion is measured and classified. Furthermore, the behaviours of the points of interest are measured by fuzzy scores and labelled by a voting scheme. Finally, the main

movement and direction of points of interest with the same label are used to detect human fighting.

Chapter 6

Conclusion

6.1 Thesis conclusion

Digital video processing is a wide research topic including various techniques and algorithms for video segmentation, video retrieval and video interpretation. Segmenting video streams is a fundamental problem which has found many multimedia applications. Properly segmented streams provide points of access that facilitate browsing and retrieval. The core research in content-based video retrieval is developing technologies that automatically parse video, audio and text in order to identify meaningful composition structure and to extract and represent content attributes of any video sources. The goal of video interpretation is to develop a systematic methodology for the design, implementation and integration of cognitive vision systems for recognizing scenarios involved in a scene depicted by a video sequence.

In this thesis, paradigms for the video segmentation, retrieval and interpretation are proposed to process and manage the digital videos. Five algorithms are proposed to address the video segmentation problem focusing on the detection of video shot boundary. Content-based video copy detection is an application of video retrieval to protect intellectual property rights. A new algorithm is proposed for video copy detection, in which combination of complicated distortion is taken into consideration when those copying parts of query videos are determined. For the video interpretation task, techniques

are proposed to solve this problem using knowledge supported extraction of semantics, and employing compressed-domain processing for efficiency. Statistical skin detection is applied for human object detection and the improved detection of camera motions is also implemented. High-level semantics like video highlights are then automatically extracted via low-level analysis in the detection of human objects and camera motion events. In addition, a trajectory based approach is proposed to detect human fighting for movie annotation.

6.2 Thesis contribution

The main contribution of the thesis is concluded in terms of eight aspects and stated as follows.

(1) A simple and fast algorithm is described for detection of both abrupt shot cuts and dissolved shot cuts. The proposed algorithm works in the compressed domain and exploits existing MPEG motion estimation and compensation mechanisms. While the proposed algorithm saves a lot of computational cost compared with some of its existing counterparts, extensive experiments verify that the proposed algorithm also achieves superior performances over the existing counterparts. In summary, the features and the advantages of the proposed algorithm can be highlighted as: (i) an integrated technique for both abrupt shot cut and dissolved shot cut detection; (ii) operating in the compressed domain and thus suitable for real-time implementation; (iii) only two thresholds and two parameters are required for shot cut detection and yet the detection mechanism is made adaptive to the input video content; (iv) such technique will provide a range of useful applications for on-line video content analysis, processing, and management. Specific examples include real-time scene

change analysis for MPEG compressed video streams, on-line annotation of video sequences, and shot-based video content retrievals.

(2) Techniques are proposed for shot boundary detection with two main contributions. Firstly, content-based rules via multiple indicators acquired from the compressed domain to detect abrupt cuts are presented. Secondly, a coarse-to-fine procedure is introduced with a pre-processing module to lower the computational cost and a FSM to verify gradual transitions and locate their boundaries. Extensive experiments have demonstrated that the proposed algorithm is highly efficient and yields quite promising detection results in terms of recall and precision rates.

(3) My contribution for the submission to TRECVID 2007 on the shot boundary detection task is summarized as: (i) Novel features are extracted from the compressed domain and the feature selection is carried out using Adaboost. (ii) Typical gradual transition including fade in (out), dissolve and wipes, as well as a new type of gradual transition called “others” is detected. (iii) The MPEG decoding scheme is well studied so that the whole shot detection system is embedded in the compressed domain of the standard MPEG2 decoder to achieve the real-time efficiency.

(4) A context awareness algorithm with multiple thresholds and features for shot cut detection is described. While this area is well researched for the past decades, the proposed algorithm has made two contributions. Firstly, given the fact that the presence of cuts and gradual transitions in practical cases is much more complicated than theoretically described or expected, I applied multiple thresholds to deal with various different cases. The application of each individual threshold is

controlled and indicated by multiple context features, all of which are extracted in the MPEG compressed domain. Secondly, the complicated controlling process and detection is organized by decision trees as well as FSM to achieve operation efficiency as well as effectiveness in its performances. Thirdly, a coarse-to-fine procedure is introduced with pre-processing and post-processing modules to reduce the computational cost and increase its detection reliability. Extensive experiments demonstrate that the proposed algorithm achieves promising results and performances in a test set, where video sequences present a wide range of cuts and gradual transitions under various circumstances and mixed scene changes.

(5) A fuzzy logic method of feature representation for shot detection is proposed. By carrying out extensive experiments, it was found that the proposed algorithm achieves very competitive results compared to other state-of-art algorithms.

(6) The proposed strategy for video copy detection is implemented in three major procedures, including video distortion analysis, signature extraction and query and target video matching. Firstly, a dedicated video distortion analysis is implemented for input videos, so that different kinds of complex editing effect are well tackled. Secondly, key frames are scaled down to extract simple signatures, and the frame mask is generated adaptively to reduce video temporal redundancy. Thirdly, linear regression and SVM are utilized in the matching and decision making process to guarantee the robust copy retrieval result.

(7) A knowledge-supported approach is presented to segment videos and extract semantic highlights. Firstly, luminance, chrominance and motion features are extracted directly in the compressed domain. Secondly, video shots are segmented

based on the knowledge-supported rules. Thirdly, camera motion is estimated using motion vectors and skin patches are segmented in the video frames. Finally, zoom-in human objects are extracted as semantic highlights.

(8) The proposed algorithm for human fighting detection is carried out in four steps. Firstly, shot boundary detection is implemented for the robust tracking of points of interest via KLT algorithm and the tracking record is used to build the trajectory of points of interest. Secondly, global motion is estimated based on the trajectory features. Thirdly, camera motion is measured and classified into zoom, pan, tilt and mixed motion via the movements of points of interest. Fourthly, the behaviours of points of interest are measured by fuzzy scores and labelled as background points, active body points and inactive body points by a voting scheme. Hence, the main movement and direction of the three kinds of points of interest are used to detect human fighting by SVM.

6.3 Future work

In the near future, the proposed paradigms would be improved in terms of three aspects discussed as follows.

(1) Better detection result of shot boundary could be obtained by solving the challenging problem of gradual transition detection. High level features and semantic components would be further studied and extracted within the video sequences, which could facilitate the accurate detection of various types of gradual transitions.

(2) More machine learning algorithms would be surveyed and examined for the classification of the content-based video copy detection task. The mapping

methods would be studied to better formulate the relationship between the original video and the video copy.

(3) Semantic-based video events would be further studied. The performance of human fighting detection could be improved by analyzing the relationship between movie content and camera motion. More meaningful video events such as sport highlights would be detected and classified by extracting novel features and content-based rules within the video sequences.

REFERENCE

[1] J. Hoey and J. J. Little, “Value-Directed Human Behavior Analysis from Video Using Partially Observable Markov Decision Processes,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 7, pp. 1118-1132, 2007.

[2] K. C. Yang, C. C. Guest, K. E. Maleh et al., “Perceptual Temporal Quality Metric for Compressed Video,” IEEE Transactions on Multimedia, vol. 9, no. 7, pp. 1528-1535, 2007.

[3] C. Cotsaces, N. Nikolaidis and I. Pitas, “Video Shot Detection and Condensed Representation: A Review,” IEEE Signal Processing Magazine, vol. 23, no. 2, pp. 28-37, 2006.

[4] U. Gargi, R. Kasturi and S. H. Strayer, “Performance Characterization of Video-shot-change Detection Methods,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 10, no. 1, pp. 1-13, 2000.

[5] S. Li and M. C. Lee, “An Efficient Spatiotemporal Attention Model and Its Application to Shot Matching,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 10, pp. 1383-1387, 2007.

[6] Z. Rasheed and M. Shah, “Detection and representation of scenes in videos,” IEEE Transactions on Multimedia, vol. 7, no. 6, pp. 1097-1105, 2005.

[7] J. Yuan, H. Wang, L. Xiao et al., “A Formal Study of Shot Boundary Detection,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 17, no. 2, pp. 168-186, 2007.

[8] C. Grana and R. Cucchiara, “Linear transition detection as a unified shot detection approach,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 17, no. 4, pp. 483-489, 2007.

[9] M. Cooper, T. Liu and E. Rieffel, “Video Segmentation via Temporal Pattern Classification,” IEEE Transactions on Multimedia, vol. 9, no. 3, pp. 610-618, 2007.

[10] J. Cao and A. Cai, “A robust shot transition detection method based on support vector machine in compressed domain,” Pattern Recognition Letters, vol. 28, no. 12, pp. 1534-1540, 2007.

[11] W. A. C. Fernando, C. N. Canagarajah and D. R. Bull, “Video Segmentation and Classification for Content Based Storage and Retrieval Using Motion Vectors,” SPIE Conference on Storage and Retrieval for Image and Video Databases, vol. 3656, pp. 687-698, 1999.

[12] D. Lelescu and D. Schonfeld, “Statistical sequential analysis for real-time video scene change detection on compressed multimedia bitstream,” IEEE Transactions on Multimedia, vol. 5, no. 1, pp. 106-117, 2003.

[13] V. Kobla, D. Doermann and K. Lin, “Archiving, Indexing and Retrieval of Video in the Compressed Domain,” SPIE Conference on Multimedia Storage and Archiving Systems, pp. 78-89, 1996.

[14] J. Meng, Y. Juan and S. Chang, “Scene Change Detection in a MPEG Compressed Video Sequence,” SPIE Conference on Digital Video Compression: Algorithms and Technologies, pp. 14-25, 1995.

[15] S. C. Pei and Y. Z. Chou, “Novel error concealment method with adaptive prediction to the abrupt and gradual scene changes,” IEEE transactions on multimedia, vol. 6, no. 1, pp. 158-173, 2004.

[16] H. Fang, J. Jiang and Y. Feng, “A fuzzy logic approach for detection of video shot boundaries,” Pattern Recognition, vol. 39, no. 11, pp. 2092-2100, 2006.

[17] A. Nagasaka and Y. Tanaka, “Automatic Video Indexing and Full-video Search for Object Appearances,” Proc. IFIP 2nd Working Conf. Visual Database Systems, pp. 113-127, 1992.

[18] M. J. Swain and D. H. Ballard, “Color Indexing,” International Journal of Computer Vision, vol. 7, no. 1, pp. 11-32, 1993.

[19] J. Hafner, H. S. Sawhney, W. Equitz et al., “Efficient Color Histogram Indexing for Quadratic Form Distance Functions,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 7, pp. 729-736, 1995.

[20] R. Lienhart, “Reliable Transition Detection in Videos: A Survey and Practitioner’s Guide,” International Journal of Image and Graphics, vol. 1, no. 3, pp. 469-486, 2001.

[21] J. Bescos, G. Cisneros, J. M. Martinez et al., “A Unified Model for Techniques on Video Shot Transition Detection,” IEEE transactions on multimedia, vol. 7, no. 2, pp. 293-307, 2005.

[22] S. C. Pei and Y. Z. Chou, “Efficient MPEG Compressed Video Analysis Using Macroblock Type Information,” IEEE Transactions on Multimedia, vol. 1, no. 4, pp. 321-333, 1999.

[23] Q. Urhan, M. K. Gullu and S. Erturk, “Modified phase-correlation based robust hard-cut detection with application to archive film,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 16, no. 6, pp. 753-770, 2006.

[24] G. Boccignone, A. Chianese, V. Moscato et al., “Foveated shot detection for video segmentation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 15, no. 3, pp. 365-377, 2005.

[25] M. H. Lee, H. W.Yoo and D. S. Jang, “Video Scene Change Detection Using Neural Network: Improved ART2,” Expert Systems with Applications, vol. 31, no. 1, pp. 13-25, 2006.

[26] K. Matsumoto, M. Naito, K. Hoashi et al., “SVM-based shot boundary detection with a novel feature,” IEEE International Conference on Multimedia and Expo, vol. 1-5, pp. 1837-1840, 2006.

[27] T. S. Chua, H. M. Feng and A. Chandrashekhara, “A unified framework for shot boundary detection via active learning,” IEEE International Conference on Acoustics, Speech and Signal Processing, vol. II, pp. 845-848, 2003.

[28] X. B. Gao, B. Han and H. B. Ji, “A shot boundary detection method for news video based on rough sets and fuzzy clustering,” 2nd International Conference on Image Analysis and Recognition, vol. 3656, pp. 231-238, 2005.

[29] C. C. Lo and S. J. Wang, “Video segmentation using a histogram-based fuzzy c-means clustering algorithm,” 10th IEEE International Conference on Fuzzy Systems, vol. 1-3, pp. 920-923, 2001.

[30] X. B. Gao and X. Tang, “Unsupervised video-shot segmentation and model-free, anchorperson detection for news video story parsing,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 12, no. 9, pp. 765-776, 2002.

[31] S. A. Berrani, L. Amsaleg and P. Gros, “Robust content-based image searches for copyright protection,” Proc. ACM Int. Workshop on Multimedia Databases, pp. 70-77, 2003.

[32] A. Joly, C. Frelicot and O. Buisson, “Robust content-based video copy identification in a large reference database,” Second international conference on image and video retrieval, vol. 2728, pp. 414-424, 2003.

[33] Y. Ke, R. Sukthankar and L. Huston, “Efficient near-duplicate detection and sub-image retrieval,” Proceedings of the 12th annual ACM international conference on Multimedia, pp. 869-876, 2004.

[34] E. Chang, J. Wang, C. Li et al., “Rime-a replicated image detector for the world-wide web,” in Proc. SPIE Symp. Voice, Video and Data Communications, pp. 58-67, 1998.

[35] A. Hampapur and R. Bolle, “Comparison of sequence matching techniques for video copy detection,” in Proc. Conf. Storage and Retrieval for Media Databases, pp. 194-201, 2002.

[36] C. Kim, “Ordinal measure of DCT coefficients for image correspondence and its application to copy detection,” in Proc. SPIE Storage and Retrieval for Media Databases, pp. 199-210, 2003.

[37] C. Kim, “Content-based image copy detection,” Signal Processing: Image Communication, vol. 18, no. 3, pp. 169-184, 2003.

[38] R. Mohan, “Video sequence matching,” in Proc. Int. Conf. Audio, Speech and Signal Processing (ICASSP), vol. 6, pp. 3697-3700, 1998.

[39] A. Hampapur and R. M. Bolle, “Comparison of distance measures for video copy detection,” in Proc. IEEE Int. Conf. Multimedia and Expo (ICME), pp. 188-192, 2001.

[40] A. Hampapur and R. M. Bolle, “Feature based indexing for media tracking,” in Proc. IEEE Int. Conf. Multimedia and Expo (ICME), pp. 67-70, 2000.

[41] Y. A. Aslandogan and C. T. Yu, “Techniques and systems for image and video retrieval,” IEEE Transactions on Knowledge and Data Engineering, vol. 11, no. 1, pp. 56-63, 1999.

[42] A. Hampapur, A. Gupta, B. Horowitz et al., “Virage video engine,” in Proc. SPIE, vol. 3022, pp. 188-198, 1997.

[43] H. Wactlar, “Informedia-Search and summarization in the video medium,” Proceedings of Imagina Conference, pp. 1-10, 2000.

[44] S. F. Chang, W. Chen, H. J. Meng et al., “VideoQ: an automated content based video search system using visual cues,” In Proceedings of ACM Multimedia, pp. 313-324, 1997.

[45] A. Pentland, R. Picard and S. Sclaroff, “Photobook: content-based manipulation of image databases,” International Journal of Computer Vision, vol. 18, no. 3, pp. 233-254, 1996.

[46] J. R. Smith and S. F. Chang, “VisualSEEk: a fully automated content-based image query system,” in Proc. ACM Multimedia Conf., pp. 87-98, 1996.

[47] V. E. Ogle and M. Stonebraker, “Chabot: Retrieval from a relational database of image,” Computer, vol. 28, no. 9, pp. 40-48, 1995.

[48] C. Faloutsos, W. Equitz, M. Flickner et al., “Efficient and effective querying by image content,” Journal of Intelligent Information Systems, vol. 3, no. 3-4, pp. 231-262, 1994.

[49] H. Jiang and A. Elmagarmid, “WVTDB-A semantic content-based video database system on the world wide web,” IEEE Transactions on Knowledge and Data Engineering, vol. 10, no. 6, pp. 947-966, 1998.

[50] X. Zhu, A. K. Elmagarmid, X. Xue et al., “InsightVideo: Toward Hierarchical Video Content Organization for Efficient Browsing, Summarization and Retrieval,” IEEE Transactions on Multimedia, vol. 7, no. 4, pp. 648-666, 2005.

[51] J. Shao, Z. Huang, H. T. Shen et al., “Batch Nearest Neighbor Search for Video Retrieval,” IEEE Transactions on Multimedia, vol. 10, no. 3, pp. 409-420, 2008.

[52] N. Boujemaa, J. Fauqueur and V. Gouet, “What’s beyond query by example?” Springer Verlag, pp. 1-19, 2003.

[53] A. K. Elmagarmid, H. Jiang, A. A. Helal et al., “Video Database Systems: Issues, Products and Applications,” Springer, vol. 5, pp. 1-192, 1997.

[54] A. W. M. Smeulders, M. Worring, S. Santini et al., “Content-based image retrieval at the end of the early years,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pp. 1349-1380, 2000.

[55] K. Vu, K. A. Hua and W. Tavanapong, “Image retrieval based on regions of interest,” IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 4, pp. 1045-1049, 2003.

[56] X. Liu, Y. Zhuang and Y. Pan, “A new approach to retrieve video by example video clip,” Proceedings of the seventh ACM international conference on Multimedia, pp. 41-44, 1999.

[57] M. R. Naphade, M. M. Yeung and B. L. Yeo, “A novel scheme for fast and efficient video sequence matching using compact signatures,” in Proc. SPIE Conf. Storage Retrieval Media Databases, vol. 3972, pp. 564-572, 2000.

[58] Y. Wu, Y. Zhuang and Y. Pan, “Content-based video similarity model,” Proceedings of the eighth ACM international conference on Multimedia, pp. 465-467, 2000.

[59] L. Chen and T. S. Chua, “A match and tiling approach to content-based image retrieval,” IEEE International Conference on Multimedia and Expo, pp. 301-304, 2001.

[60] Y. X. Peng, C. W. Ngo, Q. J. Dong et al., “Video clip retrieval by maximal matching and optimal matching in graph theory,” IEEE International Conference on Multimedia and Expo, pp. 317-320, 2003.

[61] Y. H. Ho, C. W. Lin, J. F. Chen et al., “Fast Coarse-to-Fine Video Retrieval Using Shot-Level Spatio-Temporal Statistics,” IEEE Transactions on Circuits and Systems for Video Technology Transactions Letters, vol. 16, no. 5, pp. 642-648, 2006.

[62] F. Rothganger, S. Lazebnik, C. Schmid et al., “Segmenting, Modeling and Matching Video Clips Containing Multiple Moving Objects,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 3, pp. 477-491, 2007.

[63] J. L. To, L. Chen, A. Joly et al., “Video copy detection: a comparative study,” Proceedings of the 6th ACM International Conference on Image and Video Retrieval, pp. 371-378, 2007.

[64] S. Poullot, O. Buisson and M. Crucianu, “Z-grid-based probabilistic retrieval for scaling up content-based copy detection,” Proceedings of the 6th ACM International Conference on Image and Video Retrieval, pp. 348-355, 2007.

[65] J. L. To, O. Buisson, V. G. Brunet et al., “Robust voting algorithm based on labels of behavior for video copy detection,” Proceedings of the 14th annual ACM international conference on Multimedia, pp. 835-844, 2006.

[66] A. Joly, O. Buisson and C. Frélicot, “Content-Based Copy Retrieval Using Distortion-Based Probabilistic Similarity Search,” IEEE Transactions on Multimedia, vol. 9, no. 2, pp. 293-306, 2007.

[67] C. Y. Chiu, C. S. Chen and L. F. Chien, “A Framework for Handling Spatiotemporal Variations in Video Copy Detection,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 3, pp. 412-417, 2008.

[68] C. Kim and B. Vasudev, “Spatiotemporal Sequence Matching for Efficient Video Copy Detection,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 15, no. 1, pp. 127-132, 2005.

[69] H. J. Zhang, S. Y. Low and S. W. Smoliar, “Video parsing and browsing using compressed data,” Multimedia Tools and Applications, vol. 1, no. 1, pp. 89-111, 1995.

[70] Y. P. Tan, D. D. Saur, S. R. Kulkarni et al., “Rapid estimation of camera motion from compressed video with application to video annotation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 10, no. 1, pp. 133-146, 2000.

[71] R. Ewerth, M. Schwalb, P. Tessmann et al., “Estimation of arbitrary camera motion in MPEG videos,” Proceedings of the 17th International Pattern Recognition, pp. 512-515, 2004.

[72] N. Habili, C. C. Lim and A. Moini, “Segmentation of the face and hands in sign language video sequences using colour and motion cues,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 14, no. 8, pp. 1086-1097, 2004.

[73] S. L. Phung, A. Bouzerdoum and D. Chai, “Skin segmentation using colour pixel classification: analysis and comparison,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 27, no. 1, pp. 148-154, 2005.

[74] M. J. Jones and J. M. Rehg, “Statistical colour models with application to skin detection,” International Journal of Computer Vision, vol. 46, no. 1, pp. 81-96, 2002.

[75] P. Kakumanu, S. Makrogiannis and N. Bourbakis, “A survey of skin-colour modelling and detection methods,” Pattern Recognition, vol. 40, no. 3, pp. 1106-1122, 2007.

[76] L. Sigal, S. Sclaroff and V. Athitsos, “Skin colour-based video segmentation under time-varying illumination,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 26, no. 7, pp. 862-877, 2004.

[77] A. Datta, M. Shah and N. Lobo, “Person-on-person violence detection in video data,” Proceedings of the 16th International Conference on Pattern Recognition, vol. 1, pp. 433-438, 2002.

[78] J. Snoek, J. Hoey, L. Stewart et al., “Automated detection of unusual events on stairs,” Image and Vision Computing, vol. 27, no. 1-2, pp. 153-166, 2009.

[79] F. Lv, J. Kang, R. Nevatia et al., “Automatic tracking and labeling of human activities in a video sequence,” Proceedings of the 6th IEEE International Workshop on Performance Evaluation of Tracking and Surveillance, pp. 33-40, 2004.

[80] P. C. Ribeiro and J. S. Victor, “Human activity recognition from video: Modeling, feature selection and classification architecture,” in Proc. Int. Workshop Human Activity Recognition and Modeling, pp. 61-70, 2005.

[81] T. V. Duong, H. H. Bui, D. Q. Phung et al., “Activity recognition and abnormality detection with the switching hidden semi-Markov model,” IEEE International Conference on Computer Vision and Pattern Recognition, pp. 838-845, 2005.

[82] D. Chen, J. Yang and H. Wactlar, “Towards automatic analysis of social interaction patterns in a nursing home environment from video,” Proceedings of the 6th ACM SIGMM international workshop on Multimedia information retrieval, pp. 283-290, 2004.

[83] D. Hall, L. Eubanks, L. S. Meyyazhagan et al., “Evaluation of covert video surveillance in the diagnosis of Munchausen syndrome by proxy: Lessons from 41 cases,” Pediatrics, vol. 105, no. 6, pp. 1305-1312, 2000.

[84] D. Ayers and M. Shah, “Monitoring human behavior from video taken in an office environment,” Image and Vision Computing, vol. 19, no. 12, pp. 833-846, 2001.

[85] L. Z. Manor and M. Irani, “Event-based analysis of video,” Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 123-130, 2001.

[86] H. Zhong, J. Shi and M. Visontai, “Detecting unusual activity in video,” IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 2, pp. 819-826, 2004.

[87] N. Oliver, E. Horvitz and A. Garg, “Layered representations for human activity recognition,” in Proc. IEEE Conf. Multimodal Interfaces, pp. 3-8, 2002.

[88] W. Y. Lin, M. T. Sun, R. Poovendran et al., “Activity recognition using a combination of category components and local models for video surveillance,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 18, no. 8, pp. 1128-139, 2008.

[89] J. Bescos, “Real-time Shot Change Detection Over Online MPEG-2 Video,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 14, no. 4, pp. 475-484, 2004.

[90] H. Koumaras, G. Gardikis, G. Xilouris et al., “Shot boundary detection without threshold parameters,” Journal of Electronic Imaging, vol. 15, no. 2, pp. 0205031-0205033, 2006.

[91] B. L. Yeo and B. Liu, “Rapid Scene Analysis on Compressed Video,” IEEE Transaction on Circuits and Systems for Video Technology, vol. 5, no. 6, pp. 533-544, 1995.

[92] K. J. Qiu, J. Jiang, G. Xiao et al., “DCT-Domain Image Retrieval via Block-Edge-Patterns,” International Conference on Image Analysis and Recognition, pp. 673-684, 2006.

[93] M. Sugano, K. Hoashi, K. Matsumoto et al., “Shot Boundary Determination on MPEG Compressed Domain and Story Segmentation Experiments for TRECVID 2003,” Available on-line: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2003.

[94] H. J. Zhang, A. Kankanhalli and S. W. Smoliar, “Automatic Partitioning of Full-motion Video,” ACM Multimedia Systems, vol. 1, no. 1, pp. 10-28, 1993.

[95] “TRECVID,” Available on-line: http://www-nlpir.nist.gov/projects/trecvid/.

[96] S. Porter, M. Mirmehdi and B. Thosmas, “Temporal Video Segmentation and Classification of Edit Effects,” Image and Vision Computing, vol. 21, no. 13-14, pp. 1097-1106, 2003.

[97] T. Y. Liu, K. T. Lo, X. D. Zhang et al., “A New Cut Detection Algorithm with Constant False-alarm Ratio for Video Segmentation,” Journal of Visual Communication and Image Representation, vol. 15, no. 2, pp. 132-144, 2004.

[98] Y. F. Ma, J. Sheng, Y. Chen et al., "MSR-Asia at TREC-10 Video Track: Shot Boundary Detection Task," Available on-line: http://trec.nist.gov//pubs/trec10/index_track.html#video.

[99] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of online learning and an application to boosting," Journal of Computer and System Sciences, vol. 55, no. 1, pp. 119-139, 1997.

[100] S. Lefèvre and N. Vincent, "Efficient and robust shot change detection," Journal of Real-Time Image Processing, vol. 2, no. 1, pp. 23-34, 2007.

[101] R. M. Ford, C. Robson, D. Temple et al., "Metrics for Shot Boundary Detection in Digital Video Sequences," Multimedia Systems, vol. 8, no. 1, pp. 37-46, 2000.

[102] A. Hanjalic, "Shot boundary detection: unraveled and resolved?" IEEE Transactions on Circuits and Systems for Video Technology, vol. 12, no. 2, pp. 90-105, 2002.

[103] K. Matsumoto, M. Sugano, M. Naito et al., "Shot Boundary Detection and Low-Level Feature Extraction Experiments for TRECVID 2005," Available online: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2005.

[104] K. Qiu, J. Jiang and G. Xiao, "An edge based content descriptor for content based image and video indexing," Lecture Notes in Computer Science, Image Analysis and Recognition, Springer, vol. 4141, no. 1, pp. 673-684, 2006.

[105] Z. Liu, D. Gibbon, E. Zavesky et al., "AT&T RESEARCH AT TRECVID 2006," Available online: http://www-nlpir.nist.gov/projects/tvpubs/tv.pubs.org.html#2006.

[106] C. Harris and M. J. Stephens, "A combined corner and edge detector," Alvey Vision Conference, pp. 147-152, 1988.

[107] I. Sobel and G. Feldman, “A 3x3 Isotropic Gradient Operator for Image Processing,” presented at a talk at the Stanford Artificial Project, 1968.

[108] C. A. R. Hoare, “Algorithm 64: Quicksort,” Communications of the ACM, vol. 4, no. 7, pp. 321-322, 1961.

[109] A. Dante and M. Brookes, “Precise real-time outlier removal from motion vector fields for 3D reconstruction,” Proceedings of International Conference on Image Processing, pp. 393-396, 2003.

[110] B. D. Lucas and T. Kanade, “An Iterative Image Registration Technique with an Application to Stereo Vision,” International Joint Conference on Artificial Intelligence, pp. 674-679, 1981.

[111] C. Tomasi and T. Kanade, “Detection and Tracking of Point Features,” Carnegie Mellon University Technical Report CMU-CS-91-132, 1991.

[112] J. Shi and C. Tomasi, “Good Features to Track,” IEEE Conference on Computer Vision and Pattern Recognition, pp. 593-600, 1994.

[113] J. Chen, J. Ren and J. Jiang, “Compressed-Domain Shot Boundary Detection using Finite State Machine and Content-based Rules,” Asia-Pacific Workshop on Visual Information Processing, pp. 137-142, 2007.

[114] C. C. Chang and C. J. Lin, “LIBSVM: a library for support vector machines,” Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[115] Z. Cernekova, I. Pitas and C. Nikou, “Information theory-based shot cut/fade detection and video summarization,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 16, no. 1, pp. 82-91, 2006.

[116] Y. T. Chen and C. S. Chen, “Fast human detection using a novel boosted cascading structure with meta stages,” IEEE Transactions on Image Processing, vol. 17, no. 8, pp. 1452-1464, 2008.

[117] S. Y. Chien, S. Y. Ma and L. G. Chen, “Efficient Moving Object Segmentation Algorithm Using Background Registration Technique,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 12, no. 7, pp. 577-586, 2002.

[118] L. Y. Duan, M. Xu, Q. Tian et al., “A unified framework for semantic shot classification in sports video,” IEEE Transactions on Multimedia, vol. 7, no. 6, pp. 1066-1083, 2005.

[119] C. L. Huang and B. Y. Liao, “A robust scene-change detection method for video segmentation,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 11, no. 12, pp. 1281-1288, 2001.

[120] R. A. Joyce and B. Liu, “Temporal segmentation of video using frame and histogram space,” IEEE Transactions on Multimedia, vol. 8, no. 1, pp. 130-40, 2006.

[121] B. Leibe, K. Schindler, N. Cornelis et al., “Coupled object detection and tracking from static cameras and moving vehicles,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 30, no. 10, pp. 1683-1698, 2008.

[122] C. C. Lien, C. Y. Hong and Y. T. Fu, “Object-based accumulated motion feature for the compressed domain human action analysis,” Proceedings of the 9th Joint Conference on Information Sciences, pp. 1-4, 2006.

[123] C. Liu, P. Chung and Y. Chung, “Human home behaviour interpretation from video stream,” in Proc. IEEE Int. Conf. Networking, Sensing and Control, vol. 1, pp. 192-197, 2004.

[124] H. Lu and Y. P. Tan, “An effective post-refinement method for shot boundary detection,” IEEE Transactions on Circuits and Systems for Video Technology, vol. 15, no. 11, pp. 1407-1421, 2005.

[125] P. Viola, M. J. Jones and D. Snow, “Detecting pedestrians using patterns of motion and appearance,” International Journal of Computer Vision, vol. 63, no. 2, pp. 153-161, 2005.

APPENDIX------Author's Contributions

P1. Jinchang Ren, Jianmin Jiang and **Juan Chen**, "Shot Boundary Detection in MPEG Videos using Local and Global Indicators", Accepted to IEEE Transactions on Circuits and Systems for Video Technology.

P2. Jinchang Ren, Jianmin Jiang, **Juan Chen** and Stan S. Ipson "Extracting Objects and Events from MPEG Sequences for Video Highlights, Indexing and Retrieval", Accepted to Journal of Multimedia.

P3. Jinchang Ren, **Juan Chen**, Jianmin Jiang and Stan S. Ipson, "Knowledge-Supported Segmentation and Semantic Contents Extraction from MPEG Videos for Highlight-Based Annotation, Indexing and Retrieval", Lecture Notes in Computer Science, the Fourth International Conference on Intelligent Computing, pp. 258-265, 2008.

P4. **Juan Chen** and Jianmin Jiang, "University of Bradford at TRECVID 2008: Content Based Copy Detection Task", Video Retrieval Evaluation Online Proceedings 2008.

P5. Jianmin Jiang , Zhengmin Li , Gugoqiang Xiao and **Juan Chen**, "Real-Time Shot-Cut Detection in a Compressed Domain", Journal of Electronic Imaging, SPIE, vol. 16, no. 4, 2007.

P6. **Juan Chen**, Jinchang Ren and Jianmin Jiang, "Compressed-Domain Shot Boundary Detection using Finite State Machine and Content-based Rules", Asia-Pacific Workshop on Visual Information Processing 2007, pp. 137-142, 2007.

P7. Jinchang Ren, Jianmin Jiang and **Juan Chen**, "Determination of Shot Boundary in MPEG Videos for TRECVID 2007", Video Retrieval Evaluation Online Proceedings 2007.

P8. **Juan Chen**, Jianmin Jiang and Jinchang Ren, "A Context Awareness Approach towards Detection of Video Shot Boundaries in Compressed Domain", Journal paper to be submitted.

P9. **Juan Chen**, Stan S. Ipson and Jianmin Jiang "A Fuzzy Logic Method of Feature Representation for Shot Boundary Detection", Conference paper submitted to 2009 IEEE International Conference on Image Processing.

P10. **Juan Chen** and Jianmin Jiang, "A Novel Analysis Approach towards the Detection of Video Copies under Complicated Distortions", Journal paper to be submitted.

P11. **Juan Chen** and Jianmin Jiang, "Trajectory-Based Human Fighting Detection for Movie Annotation", Conference paper to be submitted.