Optimisation of video detection algorithms for use in advertisement detection applications

M.G. de Klerk

orcid.org 0000-0002-2053-4486

Thesis submitted in fulfilment of the requirements for the degree *Doctor of Philosophy in Computer and Electronic Engineering* at the North-West University

Promoter: Prof W.C. Venter

Graduation May 2018

Student number: 20555466

Declaration

I, Martinus Gerhardus de Klerk, hereby declare that the thesis entitled "Optimisation of video detection algorithms for use in advertisement detection applications" is my own original work and has not already been submitted to any other university or institution for examination.

M.G. de Klerk

Student number: 20555466

Signed on the 20th day of November 2017 at Potchefstroom.

Acknowledgements

First and foremost, I would like to thank our Heavenly Father for not only the opportunity, but the strength and patience to pursue this Ph.D. When I started my academic career, I never even contemplated post-graduate studies for a masters, even less so to pursue a Ph.D. Through all the ups and downs over all the years, Your mercy and grace was my embrace.

Father, thank you for parents, Christo and Henna, who did not just tell me how to live, but provided a living example thereof. Parents whose unfaltering love and support has been ever-present throughout the years. Money might be able to buy a lot of things, but a loving family, my family, is priceless! Father, I would also like to thank you for my sister, Erika. Sus, thank you for always being the optimistic dreamer who can always find something to be happy about and be there for me!

Lord, I am truly blessed by the friends I made along the way. In life there are many people that come and go, but sometimes you are lucky enough to meet a few that become like family. Father, thank you for providing me with a mentor and a friend whom has been ever-present since my early days in high school. Mnr GP, thank you for helping bring out the best in me, for teaching me that 79.5% is not a distinction - if you want 80%, you will have to work for it. It is a blessing to have someone that will push you when needed, and support you throughout the journey.

While there were many friends along this journey, a few of them played a larger role than they will ever know. Father, thank You for blessing me with a brother, a brother not by something as trivial as blood, but by choice. Rudi, thank you for being my brother through the good times and the bad. Father, there are a special group of friends, my support crew, that I would like to thank You for. Countless late nights at the office was made bearable, actually enjoyable with these friends. While there were many, I would like to thank You in particular for sending Rikus, Angelique, Melvin, Leenta, Henri, Gert, Jan and Arno my way. Thank you all for your friendship and support!

Father, thank You for blessing me with a supervisor, mentor and friend, Prof Venter. Someone once said - "A supervisor can make or break a student". Luckily I am blessed with a supervisor who not only supervised and guided me, but supported me in my quest for knowledge. Thank

you Prof always going the extra mile for me. Thank you for assisting me with funding as well Prof!

Lord, I would like to ask You to bless these people the same way that they have been a blessing to me!

Amen

Abstract

The focus of this thesis is on the optimisation of the video detection process to detect advertisements in streaming digital media. While the process of video detection has been around for many years, it is very limited in its scope of application. These limitations pertain not only to the types of video, but in particular the execution time thereof. For a video detection technique to be effectively utilised in an advertisement detection environment, it must be able to perform *real-time* analysis.

A proposed method was found in literature which addressed some of the core functionality required for such an application. This method was however still extremely limited with regard to its scope of application and devoid of scientific justification for the parameters used therein.

The work presented in this thesis aim to address these limitations by not only expanding the scope of operation of the detection algorithm, but to provide scientific justification for the techniques and parameters utilised therein.

One of these core components is the video segmentation algorithm, realised by employing the Jensen-Shannon divergence. While the Jensen-Shannon divergence is commonly seen as an information metric, it poses an uncanny ability to help detect video shot boundaries. By analysing the Jensen-Shannon divergence in this context, a profound insight into the technique and its associated parameters was derived. In doing so, a wider variety of videos can be detected with better accuracy and precision.

Since the video segmentation aspect of the investigation provided a means to increase the speed by reliably reducing the data to be processed, the subsequent core module tasked with identifying the video was evaluated. The identification of an unknown video is done by extracting and comparing a unique, yet robust, digital video fingerprint against a known database. Due to the infinite variance in possible advertisements, a unique video fingerprinting algorithm was employed, consisting of unique hash descriptors derived from prominent keypoints found within each video.

While each of these core modules have been individually optimised, the main contribution of this thesis is the integration of these modules to create a video detection *ecosystem* incorporating the unique underlying dependencies between these modules. Lastly, the optimisation of the

integrated video detection *ecosystem*, provides a means of detecting a multitude of different videos, while adhering to the *real-time* processing requirements with scientific justification for the techniques and parameters employed to accomplish the task.

Keywords: *video detection, boundary detection, video segmentation, hash generation, Jensen-Shannon divergence*

Contents

List of Figures

List of Tables

List of Acronyms

BRISK Binary Robust Invariant Scalable Keypoints

DSTV Digital Satellite television

FPS Frames Per Second

H-SIFT Hexagonal Scale-invariant Feature Transform

JSD Jensen-Shannon Divergence

JSDSDPI Jensen-Shannon Divergence Standard Deviation of Pixel Intensities

SABC South African Broadcasting Corporation

SBD Shot Boundary Detection

SDPI Standard Deviation of Pixel Intensities

SIFT Scale-invariant Feature Transform

SURF Speeded-Up Robust Features

vfID Video Frame ID

Chapter 1

Introduction

"Creative without strategy is called 'art'. Creative with strategy is called 'advertising'" —Jef I. Richards
This chapter provides an introduction to the creative strategy behind the advertisement detection process.

1.1 Marketing and advertisements

The American Marketing Association defines marketing as the activity, set of institutions, and processes for creating, communicating, delivering, and exchanging offerings that have value for customers, clients, partners, and society at large [1].

The most common methods of marketing consist of printed media, radio broadcasts as well as television advertisements. Although printed media has been around for ages, it is limited to text and images to represent a product or service. The physical medium of printed media restricts the audience to only geographical locations where the media is disseminated.

In contrast to printed media, radio broadcasts tend to disperse information much easier over large areas, but lack the ability to convey rich media e.g. image and text pertaining to the advertisement.

Luckily with the advent of modern technology, it became possible to disseminate information containing a graphical and audio component in the form of television advertisements.

Although television broadcasting is the most effective of the marketing techniques, it does however come with overhead, namely the cost thereof.

There is an old saying: *"Life has a golden rule - he who has the gold, makes the rules"*. This is a timeless truth, which is one of the underlying motivators in our economy. Companies are incurring great expenses in order to promote their products. With the advent of the digital revolution, this process has been made easier and can reach multitudes of people, but it comes with a substantial price tag, especially for television advertising.

The South African Broadcasting Corporation (SABC) is the state television broadcasting agent in South Africa which has the widest population reach. This is due to the availability as a *standalone* service with the purchase of a TV licence as well as the default bundling in packages such as MultiChoice's Digital Satellite television (DSTV). This makes the SABC a good choice for advertisers in the South African market.

While the population is enjoying new product advertisement just before the prime-time news, not many of them comprehend the costs involved for that advertisement. Advertisement costs are based not only on the duration thereof, but also on the time of day that they are aired. The cost is dramatically escalated if they are aired during large events such as national sport, etc. To give but a small example of these costs, the advertisement costs for SABC for the week of 4 - 10 October 2016 are tabulated in Table 1.1 [2].

Table 1.1: SABC Advertising Costs

CHANNEL	TIME	DURATION	Mon.	Tue.	Wed.	Thur.	Fri.	Sat.	Sun.
SABC 1	18:59	60sec	R78000	R78000	R78000	R78000	R78000	R37200	R44400
SABC 2	18:29	60sec	R25200	R25200	R25200	R25200	R25200	R20400	R19200
SABC 2	19:29	60sec	R72000	R72000	R72000	R72000	R72000		R37200
SABC 3	18:29	60sec	R27600	R27600	R27600	R27600	R27600	R26400	R28800

As is apparent in the aforementioned table, the cost of advertisements are excessive, in certain instances equating to R1300 per second.

In order to ensure a positive return on investment, various companies enlist the help of media monitoring companies to ensure that their television advertisements were aired in the correct time slot and for the correct duration. Although this might seem like a trivial task which might

be done by singular person watching television, it is far from it.

A need exists for a detection technique by which multiple broadcasting streams can be monitored and advertisements tracked artificially. Although there are multiple video detection techniques available, the majority of them were developed from a research perspective and are not particularly suited for commercial applications requiring *real-time* video detection as described later in Section 1.2.1.

1.2 Legacy implementation

One such a detection technique was proposed by Moolman et al. in [3]. This proposed technique, hereafter referred to as the legacy implementation, provided a concept whereby video detection could be accomplished in *real-time*. However, the legacy implementation was developed from an academic perspective as a proof of concept and was found lacking for commercial applications. As a proof of concept, the legacy implementation was created to function on only a small set of videos while there are various parameters used in the algorithm without scientific justification or optimisation.

The key idea proposed by the legacy implementation pertains to the segmented manner in which a video is analysed. Rather than analysing each and every chronological frame within the video, the video segmentation technique is implemented to reduce the number of frames to consist of only diverse collection of frames located in the various shots from which the video is comprised. By reducing the number of frames to be analysed by the detection algorithm it allows the technique to execute in *real-time*. While this segmented approach is able to greatly reduce the detection times, it however has a crucial limitation: if the segmentation is not 100% reproducible, then all subsequent detection techniques will be voided. The legacy implementation has a particular problem with video segmentation where gradual transitions are encountered. These gradual transitions are discussed later in Section 2.2.1 and for a particular use-case in Section 5.5.

1.2.1 Real-time

A video detection technique is required capable of detecting video in a sequential stream. By minimising the detection times, the algorithm can be implemented to monitor multiple broadcasting channels simultaneously. The main requirement of these algorithms is to adhere

to the real-time requirements.

The concept of real-time is a fairly relative one. In this context the term does not refer to the validity of time, but rather to a relational expression. Gambier stated that in a real-time system, the correctness of a result does not only depend on the logical correctness of the calculation, but also upon the time at which the result is made available [4]. For the purpose of this investigation, the term real-time will be defined as the analysis time domain where the time required for all computations t, is equal to or preferably less than the actual playback duration $t_{duration}$ of the video being analysed

$$t_{calculations} \leq t_{duration}. \tag{1.1}$$

Another constraint imposed on the analysis techniques pertaining to the streaming media aspect, is that the analysis technique will only have historic data available to perform the analysis. This becomes a notable factor when calculating the metrics such as the threshold, as some traditional techniques rely on a global threshold calculated from the whole composite video which is now unavailable.

1.3 Optimising the video detection process

The co-founder of the Microsoft Corporation, Dr. Gordon E. Moore, tried to predict the future in 1965. He did this by formulating a clause, now commonly known as Moore's Law, which states that processors will shrink in size and the components within them double every two years [5]. A little less known law that is directly affected by Moore's Law, is called Nathan's Law [6]. Nathan P. Myhrvold's laws state that:

1. Software has gas like properties since it expands to fit the container that it is in.
2. Software exhibits rapid initial growth which is ultimately limited by Moore's Law.
3. Software growth makes Moore's Law possible since improved hardware is required to run the software.
4. Software is only limited by human ambition and expectation.

Over the years the technology sector trends have followed this prediction with the creation of faster and faster processors. This trend has been feeding the ever-insatiable need for speed

by allowing the implementation of more complex and computationally intensive algorithms by means of a *brute-force* approach due to the abundance of processing power. This *brute-force* approach is however starting to become problematic.

Recent trends have indicated that we might be approaching the end of Moore's law [7], [8]. This inflection point has a profound impact on the aforementioned complex algorithms. Since the *brute force* approach is no longer a viable one, but the need for speed is ever-present, something has to be done. The only logical way forward is to optimise the algorithms that are to be executed.

One of the key concepts in the legacy implementation is the implementation of a shot boundary detector. By reducing the number of frames to analyse, the video detection algorithm is able to execute faster and adhere to *real-time* constraints. If Moore's Law was still applicable, the extra processing power would only contribute up to a certain point. This is due to the fact that increasing the number of frames to be analysed beyond a certain point, does not necessarily contribute to the accuracy of the algorithm.

In optimising a process, a process model is generally used by a numerical procedure to compute the optimal solution [9]. This elegant quantitative optimisation approach is however not ideally suited for the optimisation of the video recognition algorithm. This can be attributed to the unbounded nature of the input variable, i.e. the input videos. This implies that the input parameter can be viewed as a qualitative data metric when considering videos in the digital advertisement domain.

In order to overcome the boundless input problem hindering the optimisation process, a different approach is implemented. By creating a subset of generally representative input videos, the input videos become a bounded variable, which can then be analysed. This approach is known as the quantification of qualitative data, which is regarded as indicative of a quantitative research approach [10]. A detailed description of the test data will be provided in Section 2.6.

1.4 Research problem

It is evident from the previous sections that there is a great need for optimisation detection speed without relying on the use of brute-force processing. This begs the all important

question:

> *What is the optimal method to detect advertisements in real-time video streams?*

This is in effect a loaded question which addresses the research problem which can be broken down into incremental segments:

- How can digital video advertisements be detected?
- What is the optimal detection technique with regards to speed and accuracy?
- Will this optimal technique satisfy the real-time requirements?
- Is this detection method applicable to various different video types?

While the legacy implementation provides an answer to the *how* aspect of video advertisement detection, there is still a lot of research required as to the optimal parameters of the various video detection components. This breakdown of the research problem allows a clear high-level roadmap for the research methodology.

1.5 Research methodology

The term *Video Detection* is an ambiguous term which in this context will be used as a collective term for the process by which a video (in particular an advertisement) can be recognized and identified within a stream.

This collective nature hence requires analysis and optimisation of the various underlying techniques. This will be accomplished by:

- An overview of the most relevant literature pertaining to video detection and its various underlying techniques;
- Analysing and optimising of the core modules:
  - Video segmentation;
  - Video fingerprinting;
- Integration of optimized modules into an optimised detection technique.

1.5.1 Literature overview

A literature survey will be done to identify and fully quantify the various aspects of the video detection process. The information obtained thereof serves as a background for the subsequent sub-modules which will be implemented and optimised.

1.5.2 Core module optimisation

The modular nature of the video detection process allows for individual assessment and optimisation of each core module.

The video segmentation is of particular interest since it greatly reduces the number of frames to analyse and fingerprint. However since only the frames that are identified by the video segmentation algorithm is passed to the fingerprinting algorithm, any errors in video segmentation greatly reduce the probability of identifying the video.

1.5.3 Optimised module integration

Due to the linear nature of the video detection process, it should exhibit commutative properties. Thus once each module has been optimised, they are integrated into a singular application for video detection. This integrated application should inherit the optimised characteristics of the various sub modules.

1.6 Research contributions

The legacy implementation as eluded to in Section 1.2 was in essence a *proof-of-concept*, devoid of scientific parameter justification and any optimisations. During the cause of the research presented in this thesis, these shortfalls are addressed, and the functionality of the advertisement detection system is expanded to function with larger subsets of videos.

The main research contributions arising from this thesis include:

Providing new insight into the Jensen-Shannon Divergence for shot boundary detection

A comprehensive sensitivity analysis of the Jensen-Shannon Divergence provides a deeper insight to the inner workings of the entropy-based video segmentation technique following a scientific parameter sensitivity analysis.

Integration of multiple techniques to create a robust video detection technique (Combining existing methods/designs to achieve a new method.)

By combining data metrics as commonly encountered in the network and informatics industry, the Jensen-Shannon Divergence boundary detection technique was incorporated with the Scale-invariant Feature Transform (SIFT)-based video fingerprinting technique in order to reduce fingerprinting costs while uniquely identify digital videos. This premise was suggested by Moolman et al. in the legacy implementation, but was investigated and brought to fruition in this research.

A comprehensive breakdown of these contributions is included in Section 6.5.

1.7 Thesis overview

This thesis is divided into 6 chapters. This chapter provides an overview of the study presented in the thesis as well as the motivation behind it. Chapter 2 provides an overview of the video detection process, supplemented by a literature overview with special attention to the core modules of the video detection process. In Chapter 3, special attention is given to research pertaining to video segmentation as it has a direct influence on all the other modules. Once the proposed frames have been identified by the video segmentation algorithm, these frames are analysed to extract a unique fingerprint as discussed in Chapter 4. Chapter 5 describes the video detection process and integration of the various modules. Chapter 6 concludes the research conducted throughout this thesis.

1.8 List of publications

During the course of this research, multiple publication contributions have been made.

1.8.1 Peer-reviewed conference contributions

- M.G. De Klerk, W.C. Venter and A.J. Hoffman, "Digital video shot boundary detector investigation" in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*, 2014, pp. 132-137
- M.G. De Klerk, W.C. Venter and A.J. Hoffman, "Automatic shot boundary detection for streaming digital video" in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*, 2015, pp. 219-224
- M.G. De Klerk, W.C. Venter and A.J. Hoffman, "Resizing-effects analysis on video

fingerprinting for use in video recognition" in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*, 2016, pp. 242-247

1.8.2 Journal article contribution

- M.G. De Klerk, W.C. Venter and A.J. Hoffman, "Parameter analysis of the Jensen-Shannon divergence for shot boundary detection in streaming media applications" accepted for publication by *The South African Institute of Electrical Engineers (SAIEE)*, 2018.

Chapter 2

State of the art

"That is part of the beauty of all literature. You discover that your longings are universal longings, that you're not lonely and isolated from anyone. You belong." —F. Scott Fitzgerald

The research question addressed by this thesis pertains to the possibility to improve the speed of video advertisement detection while maintaining robustness to such an extent that it can be implemented in real-time. This chapter outlines the literature relevant to video detection.

2.1 Introduction to video detection

Video detection is an ambiguous term used to refer to the art of detecting something within a video, be it movement or something else. Within the context of this thesis, the term video detection will be used as the collective term to describe the process by which an unknown video is detected, analysed and recognised if it is a known video.

In its most primitive state, video detection can be accomplished by means of correlation as is common in signal processing. Correlation entails the comparison of one signal or video to another in order to evaluate the covariance between them. This comparison technique is extremely inefficient as it has to compare the unknown media to each known media file as illustrated in Figure 2.1 in a frame-by-frame manner. Furthermore the media has to be perfectly

synchronised, or in phase for a positive detection to be made.

Figure 2.1: Basic video comparison

A better technique to accomplish video detection is to extract distinguishing features from the video. This process is more commonly known as video fingerprinting as discussed in Section 2.3. Video fingerprinting allows numerical metrics to be calculated from each digital frame. Fingerprints of unknown media can then be compared to reference fingerprints. The advantages brought forth by the implementation of video fingerprinting does however come with a cost. The generation of fingerprints takes some time to process and requires storage.

The underlying structure of video, be it digital or analogue, holds the key to alleviating the costs incurred by fingerprinting. Rather than fingerprinting and comparing in a frame-by-frame manner, one can reduce the number of frames to be analysed without losing data integrity. By utilising frame selection methods along with the video fingerprinting thereof, the conceptual detection process can be illustrated as shown in Figure 2.2.

As seen in Figure 2.2, the last core element within the detection process is a database containing the relevant fingerprints and metadata used for detection. More on the frame selection process in Section 2.2, fingerprinting thereof in Section 2.3 and the database in Section 2.4.

2.1.1 Legacy video detection algorithm

As alluded to in Section 1.2, Moolman et al. proposed a video detection algorithm utilising some of the aforementioned characteristics in [3]. This technique proved to be fairly effective within its scope of execution, but unfortunately very limited with regards to its capabilities. Some of the limitations include the rigid resizing of videos as well as the use of parameters without justification thereof. The most critical factor that warrants investigation, is the applicability of the shot boundary detection algorithm for frame selection in the presence of

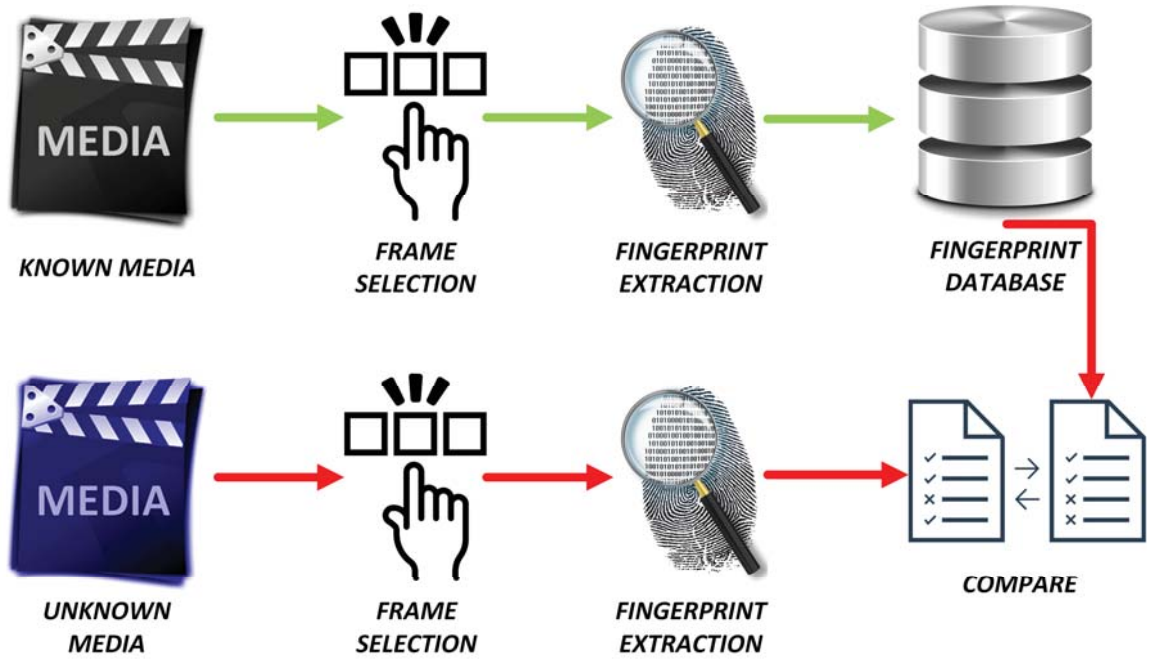


Figure 2.2: Advanced video comparison

gradual transitions.

The video segmentation process is implemented by means of a shot boundary detection algorithm. By accurately and reliably detecting the shot boundaries contained within the video, a subset of frames are identified that can be further processed by the rest of the advertisement detection process. Hence, the smaller subset of frames allows for faster subsequent processing in the system, but only if the video segmentation can be reliably and accurately reproduced. Since the video segmentation process is the initial step in the video detection process, it is imperative to be able to detect the shot boundaries, including the shot boundaries contained within gradual transitions.

In order to address these limitations, each aspect of the detection process is investigated, starting with the video segmentation process and its efficacy with regards to gradual transitions.

2.2 Video segmentation

One of the core techniques that is encountered when working with video analysis software; video storage and management systems; or the on-line video indexing systems, is called a shot boundary detector (SBD) [11]. Automatic shot boundary detection plays a pivotal role in

completing tasks such as video abstraction and key-frame selection [12]. Within the context of the research conducted in this thesis, the SBD technique is employed to identify the boundary frames of the various shots from which the video is composed. The premises on which these SBD techniques function are discussed below in Sections 2.2.2 and 2.2.3 with an in-depth analysis thereof in Chapter 3.

By identifying unique frames within the video, the subsequent video fingerprinting process is expedited. Consider a 1 minute video clip with a frame rate of 30 frames per second (FPS). If a verbose analysis would be done, then all 1800 frames would need to be fingerprinted. However, assuming the video clip consisted of two 30 second advertisements, each consisting of 5 scenes (shots), the number of frames to analyse could be greatly reduced. For this example use case, there would be 11 shot boundaries that would need to be fingerprinted. Hence the computationally intensive video fingerprinting need only be done on 11 frames compared to 1800 in the verbose case.

In order to better understand and appreciate the functionality of a SBD, one has to understand the basic underlying structure of video files. The ambiguous term *video* has its origin from the Latin word *videre* meaning 'to see' combined with the English word *audio* which refers to the process of hearing, ultimately forming a word that describes a coalesced union of audio and visual material known as video. While some research investigate the hybrid use of audio and visual components for the detection of scene changes as was presented by Chen et al. in [13], the audio component is not as reliable as the video component. This is due to possible audio-visual synchronisation anomalies as well as some *silent* scenes that are devoid of prominent audio. For the purpose of this investigation, the term video will semantically refer to the visual aspect thereof.

2.2.1 Digital video framework

Despite the fact that digital video is stored as a sequence of 1s and 0s and not on film, the structure of the digital videos remains the same as that of traditional film video. Hence a generic video V can be defined as a collection of various smaller sections called shots s:

$$V = \{s_1, s_2, \ldots, s_{n-1}, s_n\}, \;\; n \in \mathbb{Z}. \tag{2.1}$$

A key concept used to describe a video shot is that of perceived visual continuity. Visual

continuity in layman terms refers to the continuous or logical flow of objects or scenery being depicted. Hence visual continuity can be defined as the logical and continuous temporal perception of visual phenomena such as simultaneity, successiveness, temporal order, subjective present, anticipation temporal continuity and duration thereof [14].

The break in temporal continuity encountered at the ends of each shot is defined as a shot boundary. By adhering to the visual continuity concept, various metrics can be employed to detect these shot boundaries. This is done by calculating the inter-frame variances ϕ of the sequence, which should be small compared to the inter-shot variances Φ.

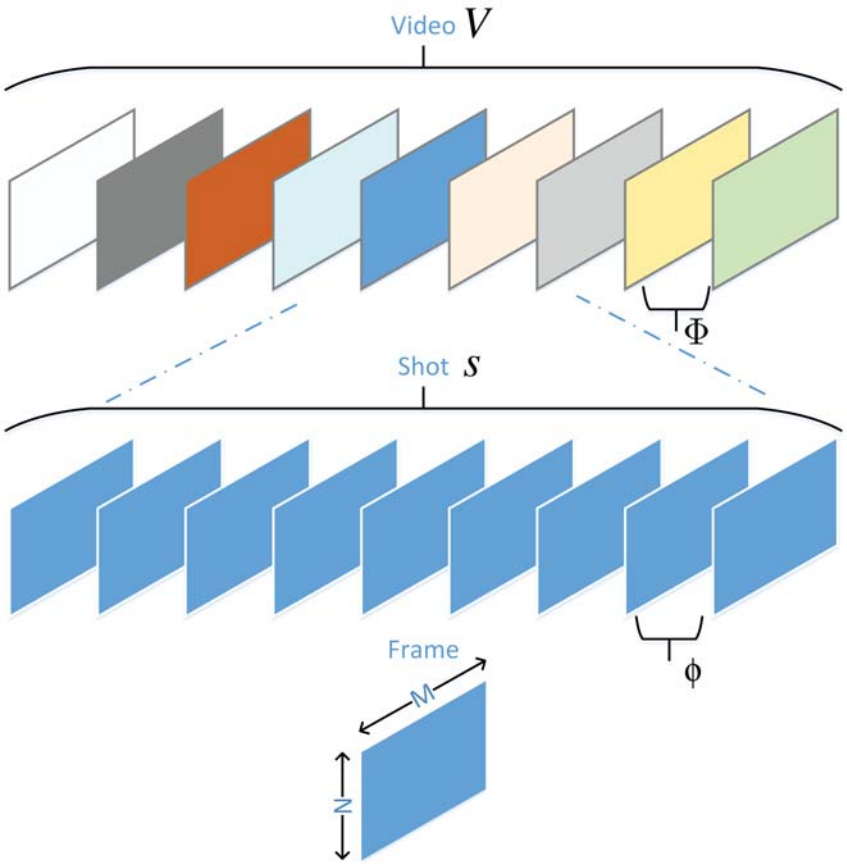


Figure 2.3: Video structure breakdown

At the lowest level, the structure of a shot consists of multiple sequential static frames f:

$$s = \{f_1, f_2, \ldots, f_{n-1}, f_n\}, \quad n \in \mathbb{Z}. \tag{2.2}$$

Each static frame can be represented by a $M \times N$ matrix where each picture element (pixel) $p_{i,j}$ can be addressed by its relative position in the frame by its co-ordinates i and j. In colour frames, each pixel has a multi-chromaticity value (RGB (Red-Green-Blue) or CMYK

(Cyan-Magenta-Yellow-Key (black)) color space) that defines the colour thereof. Similarly for grayscale frames, each pixel has a monochromatic value. This generic breakdown of the video structure is illustrated in Figure 2.3.

Now that technical format of a shot boundary has been addressed, one can appreciate a shot boundary for its perceptual contributions to video media. Although a shot boundary is just a term to describe the boundary between consecutive shots, the way in which it is implemented can be used to convey supplementary subliminal detail of the sequences.

Transitions

Human beings are able to detect the boundaries between various shots due to cognitive analysis. However, computers lack these cognitive analysis capabilities of humans and thus need to analyse the video in a different manner.

The transition between shots describes the way in which the shot boundary is implemented. There are two main categories of transitions used in videos: abrupt and gradual transitions. In the simplest form, a shot transition can be described as the visual manifestation of an abrupt change in visual content, hence called an abrupt transition.

In order to soften the visual discontinuity caused by a shot boundary, techniques can be employed to alter the frames surrounding the shot boundary. A basic gradual transition technique is called fading where the opacity or luminosity of sequential frames in the one shot is gradually decreased while the inverse is done on the following shot. A common example of such a fade is commonly referred to a as fade-to-black where the current shot is faded to a black frame.

The advancements in video editing techniques have brought forth multiple transition techniques to create visually appealing shot transitions but which tend to complicate the automatic detection thereof. These include transitions like: additive dissolve, cross dissolve, fade-to-black and fade-from-black, zoom in or out, frame slide, page peal and iris box transitions to name but a few [15].

These transitions can be broadly classified into two main categories:

- Abrupt transitions:

Abrupt transitions, sometimes referred to as hard-cuts, describe the original shot boundary between consecutive shots.

- Gradual transitions:

  The term gradual transitions do not refer to a specific transition technique, instead it is collectively used for transitional effects that soften the break in temporal continuity between shots. There are a multitude of gradual transition techniques available to accomplish this, however the main principle on which these techniques rely can be categorised in the following types:

  - *Fades:*

    Fades are generally encountered, but not limited to, the beginning and end of a video sequence. The fade commonly encountered at the start of a sequence is referred to as a fade-in. A fade-in is implemented when a black frame sequence is progressively altered by overlaying the a prescribed amount of starting frames form the actual video sequence. The opacity of these overlaid frames are incrementally increased resulting in a visually soft transition from black to the shot sequence. Similarly the reverse of this process is called a fade-out or fade-to-black and generally encountered at the end of a video sequence.
  - *Dissolves:*

    Dissolve transitions work on the same principle as fades with the exception that there is no constant black shot sequence used. During a cross-dissolve, the preceding sequence's opacity is gradually reduced while the succeeding sequence's opacity is simultaneously increased. The midpoint of the cross-dissolve will essentially contain a composite frame from both sequences, each with an opacity of 50%.

Now that the commonly occurring transitions have been identified, one can investigate various detection methodologies. The main detection methodologies encountered in literature are grouped according to how the interpret the image - pixel by pixel or as groups of pixels.

2.2.2 Pixel-based methods

The simplest method that can be employed with the goal of detecting shot boundaries is the comparison of successive frames. This can be accomplished by comparing the value of each corresponding pixel in both frames and calculating the difference thereof. In this case the difference $D_{n,n+1}$ will be the aforementioned inter-frame variance $\phi_{n,n+1}$.

$$D_{n,n+1} = \sum_{i=1}^{N}\sum_{j=1}^{M}|f_{n+1}(p_{i,j}) - f_n(p_{i,j})| \tag{2.3}$$

If the total difference between the two frames are above a certain threshold τ, it might be possible that a shot boundary has been detected:

$$D_{n,n+1} = \begin{cases} \text{Possible shot boundary}, & \text{if } D_{n,n+1} \geq \tau \\ \text{No boundary}, & \text{otherwise.} \end{cases} \tag{2.4}$$

It is easy to see how this pixel based method can become computationally expensive as well as very susceptible to noise since each pixel is evaluated as a singular entity [16]. Alternatively the pixels can be analysed as groups of entities, allowing for a reduced impact due to noise and camera motion [17].

2.2.3 Histogram-based methods

On the other end of the spectrum, there are multiple techniques that start of by grouping the pixels present in a frame according to prescribed criterion. In doing so, the amount of samples to analyse has been reduced. This has the advantage of faster processing speed as well as a reduction in the effect of noise present in the image.

While investigation key frame selection techniques, Xu et al. compared a few algorithms used to find the underlying shot boundaries [18]. One of the techniques, called the Jensen-Shannon Divergence (JSD) algorithm, proved to be an effective histogram-based boundary detection technique as investigated by De Klerk et al. in [19].

G. Ciocca utilises a further technique called Temporal Pattern Analysis to evaluate the frame difference measures to discriminate between boundaries and flashes or lens flares [20]. This is a handy technique, but unfortunately requires *future* frames which does not adhere to real-time requirements as discussed in Section 1.2.1.

More detail regarding the various video segmentation techniques are discussed in Chapter 3.

2.3 Video fingerprinting

The process of video fingerprinting entails the extraction of distinguishing features from video. Within the context of video frame fingerprinting, these distinguishing features take the form of prominent pixel groupings in the frame. These pixel groupings are commonly referred to as keypoints within the frame. These keypoints form the basis of the digital video fingerprint. However the way in which they are interpreted and combined influences the robustness thereof as explained in more detail in Chapter 4.

There are many different methods which can be employed to extract these keypoints. In literature there are two main techniques which are utilised, namely Scale-invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF). These techniques have been used as the basis for other techniques such as Binary Robust Invariant Scalable Keypoints (BRISK) [21] and Hexagonal Scale-invariant Feature Transform (H-SIFT) [22].

Even though some of these techniques exhibit novelty for their respective areas of application, the core techniques are still considered to be the best starting point with regards to keypoint extraction. As with the multitude of available techniques, there are multiple image-processing libraries available such as OpenCV, EmguCV and AForge.NET. Although OpenCV has much more support available, its ease of use lacks behind that of EmguCV [23]. EmguCV contains most of the functionality of OpenCV, but is easier to implement. Both SIFT and SURF algorithms are contained within the EmguCV image-processing library.

Both the SIFT and SURF algorithms have been implemented with great success in literature. As the name suggests, the SURF algorithm might be faster than SIFT, but an investigation by Panchal et al. concluded that the SIFT algorithm was better at detecting features [24]. The accuracy of the detected features have a higher priority than the execution speed of the extraction algorithm since the frames on which it will execute have already been reduced by the video segmentation technique. Although the SURF algorithm was contemplated, based on the aforementioned investigation by Panchal et al., the SIFT algorithm is utilised throughout the remainder of the research conducted.

Yet another contributing factor of the SIFT algorithm as implemented by EmguCV library, is the ability to specify the number of features to retain. This helps to simplify subsequent hash

generation by limiting the number of keypoints to process.

2.3.1 Scale-Invariant Feature Transform (SIFT)

In 1999 David G. Lowe published a paper *"Object Recognition from Local Scale-Invariant Features"* [25] in which he proposed a novel method to recognize objects within an image. The algorithm Lowe employed is able to extract distinguishing features from an object-image, and correlate those same features with those found on a composite image which contains the aforementioned object-image.

This unique algorithm, called SIFT, extracts distinguishing features from within an image. These features are partially invariant to illumination, 3D projective transforms and common object variations, while remaining distinct [25].

The SIFT algorithm employs the following methodology to locate distinguishing features from within an image [26]:

1. **Scale-space extrema detection** - implementation of a difference-of-Gaussian function to search over all scales and image locations for potential interest points.
2. **Keypoint localisation** - a detailed model is fit to each candidate location to determine the location and scale thereof since keypoints are selected based on their stability.
3. **Orientation assignment** - the local image gradient directions determine the orientations that is assigned to each keypoint location.
4. **Keypoint descriptor** - the measurements of local image gradients at the selected scale in the regions around each keypoint is transformed into a representation that allows for significant levels of local shape distortion.

It is important to note the origin of these distinguishing features in order to evaluate the effects that resizing has on them as discussed later in Section 4.4.

2.4 Data storage and retrieval

Throughout the various analyses, as well as in the final video detection application, large amounts of data will be encountered. The best way to manage all of this data is by

implementing a database. Although there are various different database systems available, the decision was made to utilise Microsoft SQL Server Express 2014. Not only does SQL Server Express allow for database sizes of up to 10GB, but has the advantage that it is free. By choosing this option, future implementations of the program can be deployed on the commercial version of SQL server, with only minor changes required since the core syntax is the same.

The implementation of a database during the analyses phases of this investigation has the added benefit of providing a platform that can be utilised to validate the simulation results. This is attributed to the uniform analyses of the datasets generated, ensuring that all data is evaluated using the same stored procedures - hence removing any personal bias one might have towards a certain technique.

The relational nature of the database allows one to keep track of all meta-data pertaining to the analyses being conducted, hence allowing a more comprehensive comparison.

The only limitations encountered while using SQL Server Express, is with regard to size constraints. The physical database file limit is easy to overcome, simply by employing multiple databases for the various analyses datasets. An additional size constraint is imposed when running large stored procedures on the data sets. The *page-file* memory is limited as well. Hence when analysing the relevant datasets already contained within the database, it might require a progressive approach. Where required, the relevant analyses-stored procedures were sequentially executed, and caching or committing the results to the database, rather than executing everything in memory.

2.4.1 Data retrieval

Since speed is of essence, the data utilised during the detection process must be structured in such a way as to allow for optimal retrieval thereof.

Ideally the data access speed of a system should be as fast as possible for both the data access and retrieval. The data access speed can however be altered for the use within the video detection system. This is attributed to the data volumes associated with the process. The number of videos to be fingerprinted for the detection process is greatly overshadowed by the number of videos to be analysed. Hence for this application, the data access speed can be reduced in order to increase the data retrieval speed.

By simplifying the data to be stored during the data access stage, the retrieval process can be sped up. This simplification is accomplished by reducing the keypoint data as extracted by the SIFT algorithm and creating hashes from it. This hash generation process is discussed in detail in Chapter 4.

Hash matrix

The hash matrix utilized in this system is structured as a hash table. A hash table is a dictionary data structure used to store key and value pairs [27]. It works on the premise that an index key refers to a single value entry. When an index key maps to more than one value, the phenomenon is called a collision. Although there are many collision resolution schemes, this video detection process actually utilises this collision phenomenon.

Since a digital fingerprint is composed of multiple hashes, the possibility exists that some of these hashes might be similar between videos. In order to account for this phenomenon, the hash matrix allows for M entries associated to N index keys, hence resulting in a $M \times N$ hash matrix. The index keys used for this table are the extracted hash values. The indexing and comparison of these hashes is discussed in more detail below in Section 2.5.1.

2.4.2 Scalability

The current implementation of the hash matrix is comprised of $N = 65536$ (2^{16}) index keys, each corresponding to $M = 64$ buckets to contain frame IDs. This allows the system to hold 4194304 frame IDs. Each of these frame IDs are represented by an integer value, requiring a 4 byte memory space. This corresponds to a full hash matrix requiring 16MB.

The size required is not nearly close to the 10GB limit per database imposed by Microsoft SQL Express. This implies that the database can be scaled up to contain many more video hashes.

It is however important to note that scaling the database can have an adverse influence on the detection speed of the system. The implications of scaling the hash matrix becomes apparent when the video identification is done.

2.5 Video detection

Many of the aforementioned components are self-explanatory as to how they function within the video detection system such as the shot boundary detection and the keypoint extraction. One crucial component that might not be as apparent is the hash matrix used to link the extracted fingerprints (hash values) to the corresponding Video Frame ID (vfID).

2.5.1 Database hash matching

The video fingerprint (hash) matching procedure is the last step within the video detection process. The journey up to this point involved taking a known video, finding the various shot boundaries therein, extract keypoints from those frames and finally use the keypoints to create hash values. These hash values can now be used to populate the hash matrix.

The hash matrix for this investigation is row indexed from $0 - 65535$ with columns $0 - 63$ to form a matrix as illustrated in Figure 2.4. The number of rows are determined by the number of bits used in a hash as described later in Section 4.3 - in this instance a 16 bit hash is used. The number of columns are however an arbitrary number and depends on the use case. Higher column counts might adversely affect the performance of the data retrieval since all the columns are retrieved for each hash (row) query.

Each hash value is entered into the hash matrix by using the 16 bit hash value as the index key to the matrix while the corresponding vfID is entered into the first empty bucket (column). If the fingerprinted frame has enough visual diversity, the keypoint extraction process will result in a multitude of keypoints. These keypoints in turn relate to multiple hashes per fingerprinted frame. The legacy implementation called for 50 unique hashes to be constructed per fingerprinted frame.

Hash retrieval process

Direct video identification from a singular hash is highly unlikely, as in this test setup, each hash can correspond to 64 possible videos. Fortunately each fingerprinted frame has multiple keypoints, causing multiple hashes that correspond to multiple database entries.

Since each fingerprinted frame contains multiple hashes, each of these hashes are used to extract its corresponding row from the hash matrix. These extracted rows are compared to

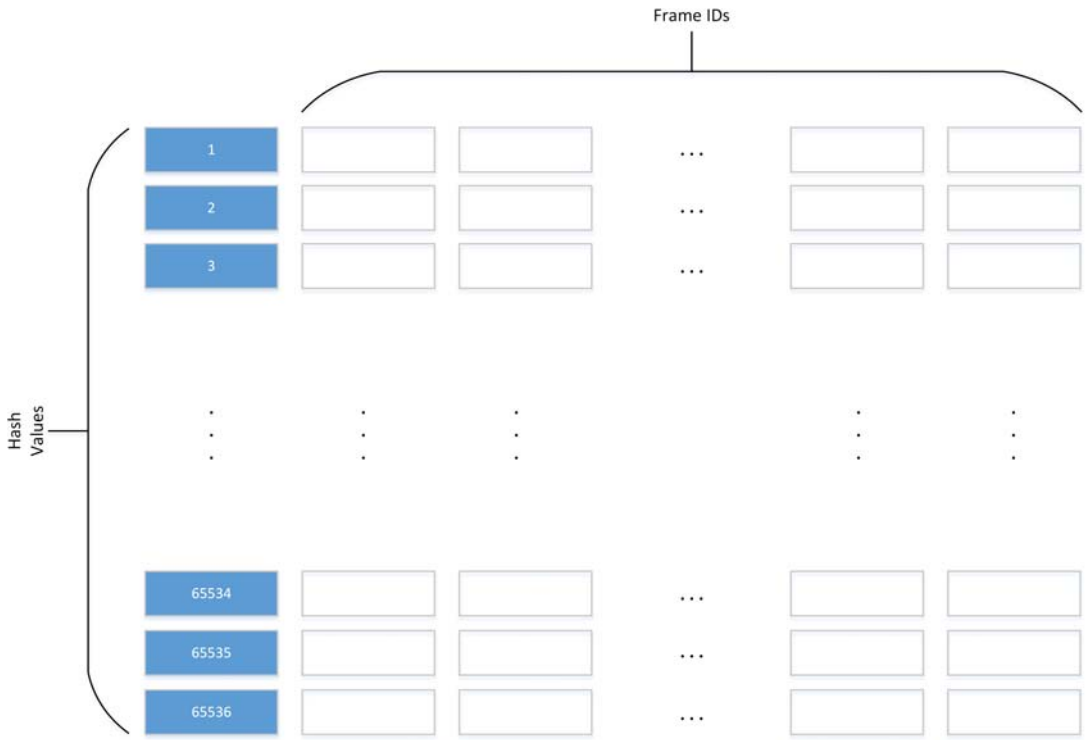


Figure 2.4: Hash matrix

determine which vfIDs are common between the extracted rows. The hash matrix example illustrated in Figure 2.5 illustrates how 4 hash values were used to extract their respective rows.

786	52	999	659	41	558
1520	4545	772	44453	999	0
50698	887	999	0	0	0
62001	1238	659	4552	999	0

Figure 2.5: Hash matrix extraction example

For this example, the vfID 999 is encountered 4 times while the vfID 659 only occurs twice. Due to this possibility, it is preferential to match not only multiple hashes on an extracted frame, but to collate a sequence of multiple frames as well.

As the video stream is being monitored, each identified frame is fingerprinted and compared to the database. All possible detections are noted alongside their respective vfIDs. The vfID that occurs the most from the hash lookup for the frame in question, is considered to be the detected vfID as illustrated in Figure 2.6. The particulars regarding the number of detections is

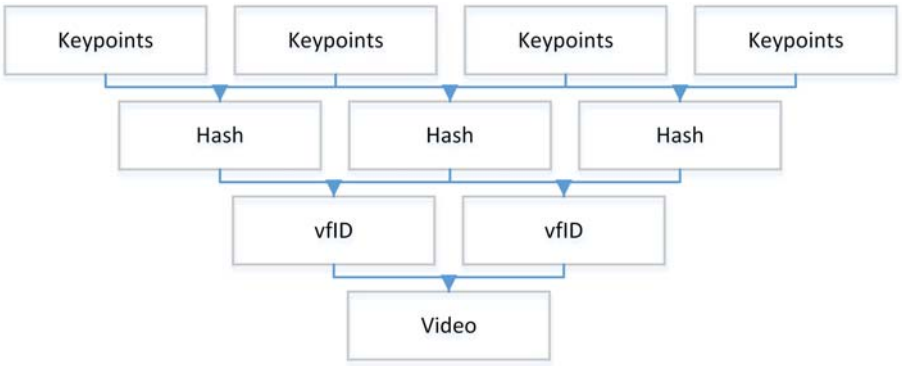


Figure 2.6: Detection hierarchy

discussed in more detail in Chapter 5.

2.6 Test data classification

2.6.1 Generated test media

The term *ground truth* is commonly used when referencing information which is provided by direct observation as opposed to information obtained by inference. When using this term within the context of video detection, it can have multiple meanings. When used within the video segmentation context, the ground truth of a video refers to the actual shot boundary locations within the video file. On the other hand, when used within the video detection context, it refers to the actual videos contained within the stream.

In order to verify and validate the accuracy of the Shot Boundary Detection (SBD) by the algorithms, various test videos were generated from which the exact shot boundary indexes are know as well as durations of any transitions. These generated test media is discussed further within their respective scopes of investigation.

2.6.2 Representative test media

As alluded to in Section 1.3, the unbounded nature of the input videos makes it extremely difficult to analyse the various algorithms with the hope of optimising it. In order to make it a bounded problem, multiple advertisements were sourced from local and global listings in order to obtain a representative sample of commonly encountered videos. For local advertisements, multiple videos were sourced from Velocity Films SA, a South-African based commercial

production company [28]. In order to expand the sample size, some global representative adverts were also obtained as listed on AdWeek [29] and Ads of the World [30]. A detailed list of the representative test media is tabulated in Tables A-3 and A-5 attached in Appendix A.

2.6.3 Streaming evaluation test media

While the goal is to optimise the video detection process to function in real-time on streaming media, it is impractical to use actual streaming media as a data source. When supplied with actual streaming data, the algorithm being evaluated will be in a process and wait state, waiting for new data to become available. This would preclude accurate timing analysis of the aforementioned algorithms. Furthermore the parameter sensitivity analyses and other investigations would take unnecessarily long to evaluate.

This problem was overcome by creating stream test media, emulating that which might be encountered in an actual stream. This was done by combining the representative test media into a singular test sequence. These test sequences were encoded using various formats as tabulated in Table 2.1.

Table 2.1: Artificial streams

FORMAT	HEIGHT	WIDTH	FPS
DVNTSC	720	480	29.97
DVPAL	720	576	25
HDV 1080p	1440	1800	25
HDV 720p	1280	720	25

In order to simulate the effects that bad weather or bad reception might have on the streams, each of the aforementioned artificial streams were subjected to various levels of noise. This was done by using the Noise-effect feature in Adobe Premier Pro CC 2014. The artificial streams were encoded with 0%, 5%, 10% and 15% noise levels.

Just as the artificial streams listed in Table 2.1 have various frame rates and dimensions, so it is with the various broadcasters. Different countries transmit using different formats and sizes, even within the countries it can vary with the availability of set-top boxes such as DSTV. With all these various dimensions, it is common to encounter a phenomenon called letterboxing, where the aspect ratio of the video is maintained by adding black bars to the top and bottom

to fit the desired aspect ratio. This phenomenon can also be seen in the test video streams. The original advert as seen in Figure 2.7 was resized to fit the stream by means of letterboxing, with the results shown in Figure 2.8.

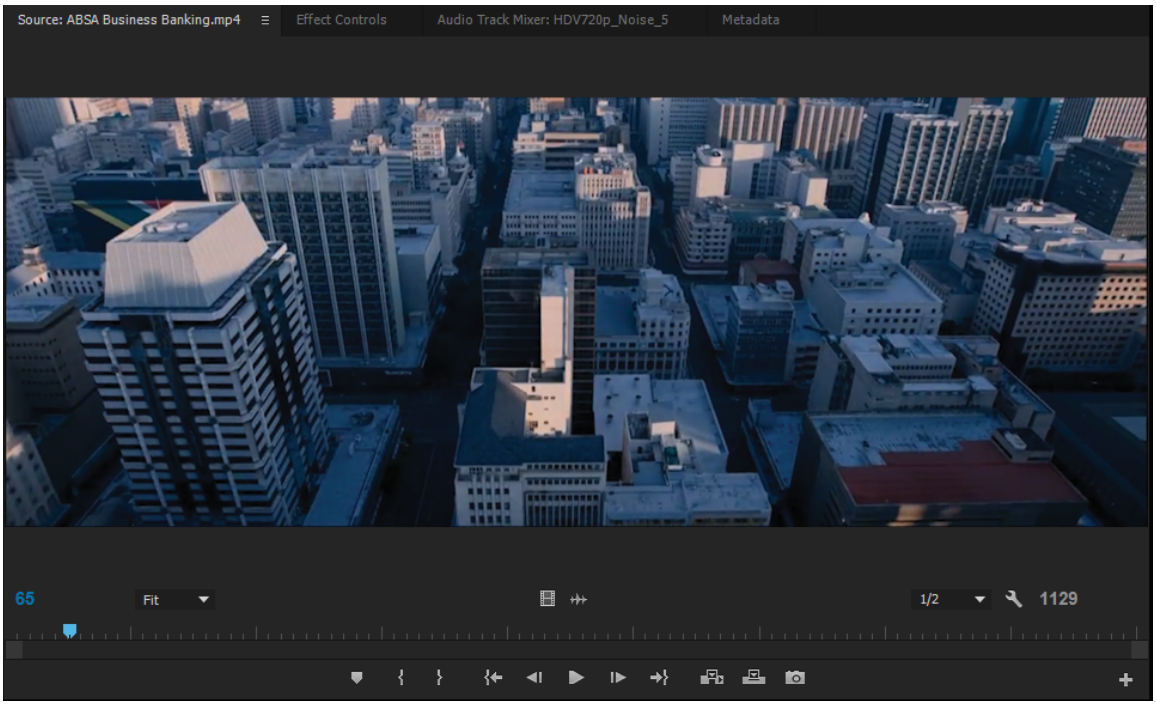


Figure 2.7: Test advert 1 - Original 1280 x 480

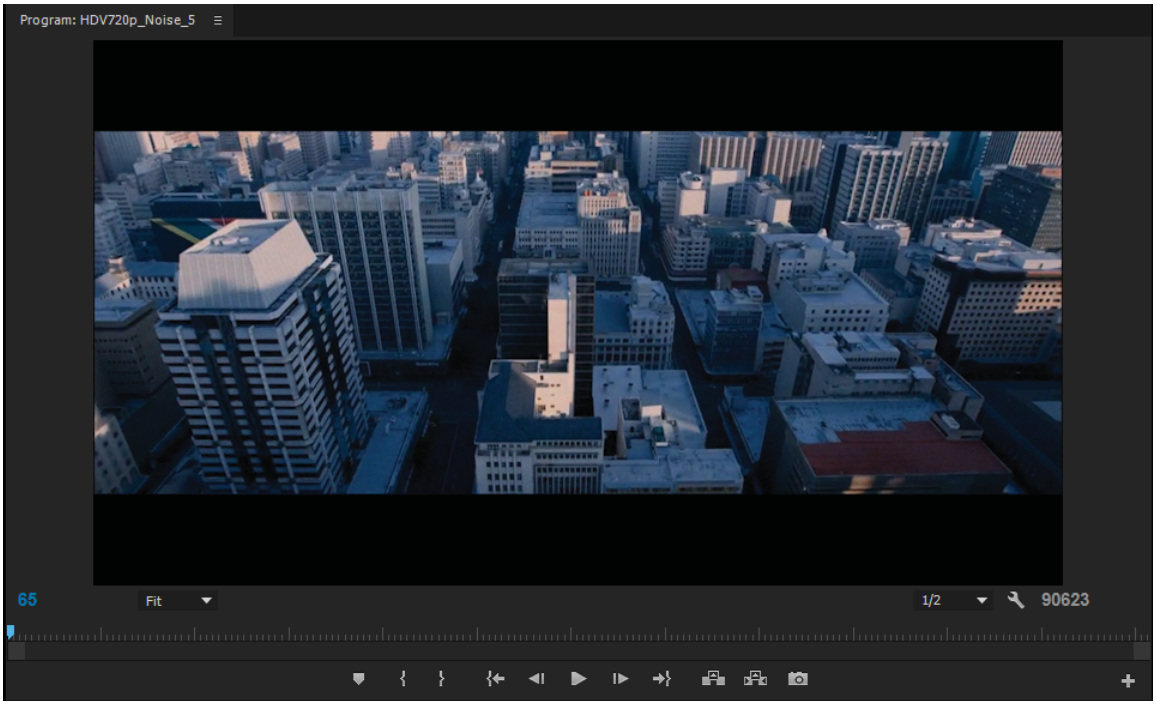


Figure 2.8: Test advert 1 - In HDV stream - 1280 x 720

2.6.4 Legacy evaluation test media

Originally the legacy implementation was evaluated using video files recorded from live video streams as they appeared on the users' television. These live video streams were recorded in a Flash Video format (.flv) with the dimensions 320×240. The unique adverts to be detected within these streams, were extracted in the same manner and in the same format. By extracting the adverts to be detected form the actual streams, it becomes easier to detect as there are no additional resizing or black bars added - the adverts added to the database is an exact match. If an advert happens to have black bars shown in the stream, those black bars are seen as part of the advert which will be added to the database.

2.7 Concluding remarks

The proposed video detection process is an involved process, incorporating multiple stand-alone modules. Each of these core modules have been utilised in literature to some extent, hence providing a starting point for the optimisation processes.

There are multiple challenges facing the optimisation process, in particular the extrapolation of the techniques to function on a wide variety of input media. This challenge is approached by exploiting the modular nature of the video detection process. Each of the core modules will be optimised with regards to the test media subsets, before being integrated and analysed once more.

The logical flow of this optimisation sequence follows the same basic flow of the video detection process as is illustrated in Figure 2.2, starting with the frame selection technique in the video segmentation chapter.

Chapter 3

Video segmentation

"Divide et impera!" - The famous motto used by the Roman general, Julius Caesar, as well as French emperor Napoléon Bonaparte meaning to divide and rule, or in the common usage, divide and conquer! This same philosophy is implemented when analysing the media by means of video segmentation - dividing video into smaller visually contiguous sections.

The question now beckons - why is it relevant to know what a shot boundary is, or even more so, where it is located? Multiple video indexing systems utilise shot boundary detectors to determine the boundaries of the various shots from which it is composed. These shots are then evaluated in order to determine the most representative frame that occurs within that shot. This frame can then be used to summarise the content of the shot and is called a key-frame. Effectively, a composite video can be analysed and a collection of key-frames returned, *surmising* the shots contained therein. For the analysis of the video stream, the location of the key-frame is not as important, only the position of the shot boundaries need to be detected.

3.1 Shot boundary detection algorithms

The premise of a shot boundary detection algorithm is to analyse a video sequence and determine the shot boundaries and their respective locations. This might seem like a trivial task to accomplish for a cognitive human, but is challenging for a computer algorithm. A

computer *perceives* a video one frame at a time, represented as a matrix containing the color or intensity values of each pixel.

Multiple techniques have been developed in order to address this need. The way in which they work are as varied as their applications. Throughout the years these techniques have been developed and tailored to work with specific types of videos, be it low resolution monochrome or high-definition color videos. Although these are extremely diverse, their core functionality can be categorised based on how they interpret the frames - pixel or histogram based. An introduction to these two categories of analysis techniques has been given in Section 2.2.2 and 2.2.3.

Lefévre et al. reviewed multiple video segmentation techniques in [31], concluding that inter-frame difference is indeed one of the fastest methods although it may be characterised by poor quality. On the other end of the spectrum are feature or motion based methodologies which are more robust, but are computationally expensive. Apart from being computationally expensive, some of the more robust techniques not only require the video as a whole to detect peaks in analysis outputs, but require training for the statistical learning methods. Since training can alter the criteria for boundary detection, it can in effect cause miss-matches as the algorithm adapts, hence changing the output of the results. The aforementioned arguments reinforce the notion to opt for a fast procedure based technique. Hence various techniques were evaluated to determine their suitability for video shot boundary detection within streaming media.

3.1.1 Standard deviation of pixel intensities

As the name suggests, the Standard Deviation of Pixel Intensities (SDPI) is a pixel-based boundary detection technique. Standard deviation σ is a commonly used statistical metric used to quantify the dispersion of a set of data values relative to the mean thereof. Instead of calculating the pixel-wise difference between the frames, each frame is analysed individually at first. The first step is to calculate the mean pixel intensity μ_F of the monochrome frame F:

$$\mu_F = \frac{1}{N} \sum_{i=1}^{N} p_i \tag{3.1}$$

where N is the number of pixels in the frame. It is important to note that since the frame constitutes the whole population and not merely a sample, the numerator N is used and not

$N-1$ as would be the case for a sample. Subsequently the standard deviation of the frame is calculated by determining the *distance* of each pixel p from the mean intensity:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - \mu_F)^2} \tag{3.2}$$

This metric in itself does not provide adequate information to make a definitive call indicating a shot boundary. The same challenge is encountered when employing the Jensen-Shannon divergence, but a solution to this problem is proposed later in Section 3.1.3.

3.1.2 Jensen-Shannon divergence

Following the video framework definition as set forth in Section 2.2.1, one can describe the aforementioned SDPI metric as an inter-frame variance ϕ metric. Similarly a metric can be calculated by employing the Jensen-Shannon Divergence (JSD) to determine the inter-frame variance between consecutive frames. This in turn can be used to detect if these frames constitute a shot boundary. By combining the inter-frame variance with an adaptive threshold τ, the metric can be used to determine if a shot boundary has been detected. Thus, a generalised form of Equation 2.4 can be given as:

$$\phi_{n,n+1} = \begin{cases} \text{Possible Shot Boundary,} & \text{if } \phi_{n,n+1} \geq \tau \\ \text{No Boundary,} & \text{otherwise.} \end{cases} \tag{3.3}$$

Jensen-Shannon divergence theoretical background

In order to optimise an algorithm, it is imperative to know how it functions. The same holds true for the Jensen-Shannon Divergence. As the name suggests, the JSD algorithm is a combination of the Shannon's entropy and the Jensen inequality.

In 1984, Claude Shannon defined information measures for instance, mutual information and entropy. The Shannon entropy [32] is a method used in information theory to express the information content, or the diversity of the uncertainty of a single random variable, i.e. a measure of information choice and uncertainty [33].

The Shannon entropy function H of the probability distribution $P = (p_1, p_2, p_3, \ldots, p_n)$

consisting of n possibilities in the distribution is calculated by:

$$H(P) = -K \sum_{i=1}^{N} p_i \log_b p_i \tag{3.4}$$

where K is a positive constant [34], b denoting the logarithmic base [35] and $n \in \mathbb{N}$. In essence, this entropy measure can be explained as the measure of uncertainty to predict which probability in occurrence will be encountered.

The other part of this technique relies on the Jensen's inequality measure as was proposed by Johan Jensen in 1905 in the paper *Sur les fonctions convexes et les inégalités entre les valeurs moyennes* [36]. Fundamentally, the Jensen-inequality states that if a given function g is convex on the range of a random variable Y, then the expected value $\mathbb{E}$ of the function will be greater than or equal to the function of the expected value:

$$\mathbb{E}\left[g(Y)\right] \geq g\left(\mathbb{E}\left[Y\right]\right) \tag{3.5}$$

where the variance will always be positive. This property can be illustrated by evaluating the Jensen-inequality for the convex function $g(y) = y^2$. This will always be positive since $\mathbb{E}\left[Y^2\right] - \left(\mathbb{E}\left[Y\right]\right)^2 \geq 0, \quad \forall\, y \in Y$.

In order to understand the relevance of the Jensen-inequality to the functionality of boundary detection, a specialised version of the Shannon entropy is however required, namely relative entropy. Relative entropy is a further application of Shannon's entropy which can be used to quantify the distance between two probability distributions. This relative entropy, also referred to as the Kullback-Leibler distance, is denoted by $D_{KL}(p,q)$ and calculates the distance between two probability distributions p and q that are defined over the alphabet χ:

$$D_{KL}(p,q) = \sum_{x \in X} p(x) log \frac{p(x)}{q(x)} \tag{3.6}$$

where $x \in \chi$. By following the conventions, as used by [33] and [37] that

$$0 log\left(\frac{0}{0}\right) = 0 \text{ and } x log\left(\frac{x}{0}\right) = \infty \tag{3.7}$$

if $x > 0$, it becomes apparent from equation 3.6 that the relative entropy satisfies the information inequality or divergence:

$$D_{KL}(p,q) \geq 0 \tag{3.8}$$

if and only if $p = q$. This implies that only when both distributions are identical, the Kullback-Leiber distance can be zero. For all other instances it will be greater than zero. This information inequality or divergence, is sometimes referred to as the information divergence [33] and is later used with the Jensen divergence.

The integration of the entropy measures from Shannon with the inequality measure of Jensen, in effect, enables the measuring of the expected entropy of the probability, which will be greater than or equal to the entropy of the expected probability.

Thus, by combining Shannon's entropy with Jensen's inequality, one is left with a commonly used index of the diversity of a multinomial distribution. Consider a multinomial distribution $p = (p_1, \ldots, p_n)$, where each $p_i \geq 0$ and the total sum of the distribution is $\sum\ p_i\ =\ 1$. Burbea et al. expressed the concavity of the Shannon entropy provides a decomposition of the total diversity in a mixed distribution $\frac{(p+q)}{2}$ as :

$$H_n\left(\frac{p+q}{2}\right) = \frac{1}{2}\left[H_n(p) + H_n(q)\right] + \mathcal{J}_n(p,q) \tag{3.9}$$

in [38]. The average diversity within the distributions is represented by the first component $\frac{1}{2}\left[H_n(p) + H_n(q)\right]$ in equation 3.9, while the latter component is called the Jensen difference.

The Jensen difference $\mathcal{J}_n(p,q)$ corresponds to the Shannon entropy $H_n(p)$ and can be expressed as:

$$\mathcal{J}_n(p,q) = H_n\left(\frac{p+q}{2}\right) - \frac{1}{2}\left[H_n(p) + H_n(q)\right]. \tag{3.10}$$

The Jensen difference is non-negative and vanishes if $p\ =\ q$ and thus provides a natural measure of divergence between the two distributions [38] as with the relative entropy in equation 3.8.

Thus, by combining the entropy calculation with the Jensen inequality measure between two consecutive frames' histograms as derived by Qing Xu [39], the JSD equation is produced:

$$JSD(f_{i-1}, f_i) = H\left(\frac{P_{f_{i-1}} + P_{f_i}}{2}\right) - \frac{H\left(P_{f_{i-1}}\right) + H\left(P_{f_i}\right)}{2} \tag{3.11}$$

where f_{i-1} is the previous frame and f_i the current frame. A measure is created as to how far the probabilities are from their likely joint source, equalling zero only if all the probabilities are equal. The probabilities of the frames are respectively $P_{f_{i-1}}$ and P_{f_i}. The aforementioned probabilities are calculated from the applicable frames' histograms as expressed in equation 3.12:

$$P(f_i) = \frac{Histogram\,(f_i)}{Height\,(f_i) \times Width\,(f_i)} \tag{3.12}$$

where the histogram of each color component is divided by the number of pixels in that frame f_i. The total number of pixels is calculated by multiplying the number of horizontal pixels by the number of vertical pixels in the frame.

Jensen-Shannon divergence implementation

Previous investigations by De Klerk et al. [19] confirmed that not only does the technique produce high recall and precision rates, but the time required for the analysis is less than the duration of the segment being analysed, hence satisfying the *real-time* requirement as stated in Section 1.2.1.

As was the case with the SDPI technique, the implementation of the JSD technique alone will only produce inter-frame variances. As a standalone metric, this is not very useful, but when employed in conjunction with a threshold-algorithm, it is able to detect shot boundaries.

Now that the theory behind the JSD algorithm has been evaluated, it becomes obvious that the inter-frame variance metric itself does not present parameters which can be varied with the hope of optimisation without affecting the core thereofs. This is however not the case with the threshold-algorithm as discussed in the following section.

3.1.3 Threshold

In the previous sections, the calculation of the SDPI and JSD inter-frame variance measures were discussed. While both of these metrics provide useful information about their respective frames, it does not contribute much towards the detection of a shot boundary as a standalone metric. One way of utilising these inter-frame variance metrics is to combine them with a threshold-algorithm. The remainder of this section discusses such a threshold-algorithm being employed with the JSD metric as expressed in Equation 3.11.

In order to ascertain if the inter-frame variance constitutes a shot boundary, we refer back to Equation 2.4. For a metric to be effective in detecting a shot boundary, it has to be evaluated against a representative threshold τ.

Cerenkova et al. investigated the use of entropy-based metrics for shot detection and also

utilised an adaptive threshold, but the threshold was manually modified for each video sequence they evaluated [40]. Some other existing techniques calculate the inter-frame variance between each frame in the video and then calculate the average thereof for the entire video:

$$\tau = \frac{1}{N}\left[\sum_{i=1}^{N}\phi_i\right] = \frac{1}{N}\left[\sum_{i=1}^{N} JSD(f_{i-1}, f_i)\right] \tag{3.13}$$

where $N \in \mathbb{N}$ represents the total number of frames within the video and the inter-frame variance ϕ is the $JSD(f_{i-1}, f_i)$ calculated by Equation 3.11. It is important to note that if the inter-frame variance $\phi_i = JSD(f_{i-1}, f_i)$ is evaluated where $i = 1$, it will result in a very large value as the zero-frame f_0 does not exist and is generally represented by a black frame. This is however not the case when encountering a fade-from-black transition at the start of the video due to the gradual change - low inter-frame variance.

Although this type of global thresholding allows for a well-represented threshold, it does however not conform to the real-time criteria as set forth in subsection 1.2.1. This short fall can be overcome by calculating a moving average from a selection of the last preceding frames against which the inter-frame variances can be evaluated. This *auto-adjusting threshold* $\bar{\tau}$ can be represented by:

$$\bar{\tau}_i = \frac{1}{\omega}\left[\sum_{j=1}^{\omega} JSD(f_{i-j-1}, f_{i-j})\right] \tag{3.14}$$

where ω is the number of preceding frames to be evaluated for the moving average. The value of ω will be evaluated later in Section 3.2.3. Similarly the moving-average threshold-algorithm can be employed with the SDPI metric.

3.1.4 JSD-SDPI hybrid

While evaluating the boundary detection capabilities of the JSD and SDPI techniques, it was noticed that each technique has certain strengths and weaknesses as corroborated by the results of the analysis in Section 3.3.3.

The SDPI metric is particularly well suited for gradual transitions as the metric is very sensitive to these changes. This sensitivity does however come at a price. Such a high sensitivity is prone to creating false positives during detection all the while being computationally expensive.

On the other hand, the JSD technique requires less execution time and easily detects abrupt

transitions, but is more likely to miss the gradual transitions. This hurdle can be overcome by employing a lower threshold value with the JSD technique, which will be able to detect smaller variations, but in doing so, also create more false positives.

Thus a union of these two techniques should be able to better detect gradual transitions while not incurring too much extra processing time.

The premise behind the JSD-SDPI Hybrid technique is to utilise the JSD algorithm with a low threshold to serve as an indicator of possible locations to run the SDPI algorithm. In doing so, only the likely areas in the stream is subjected to the more-computationally expensive technique. This process is visually represented in Figure 3.1.

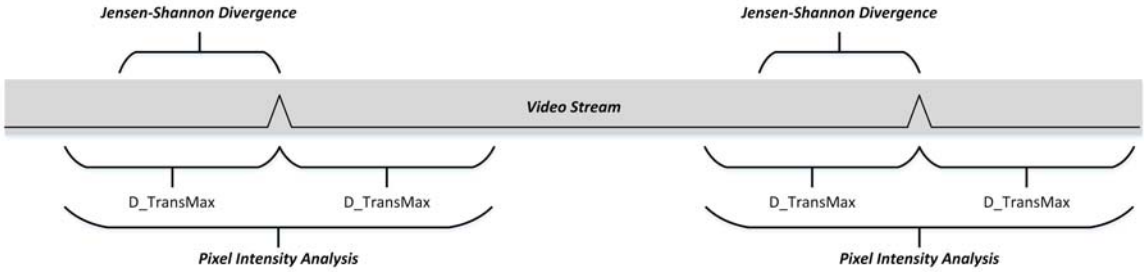


Figure 3.1: High-level illustration of the JSD-SDPI hybrid technique

It is important to note that the location of the JSD flag forms the center-point of the SDPI analysis. This is attributed to the JSD's low-threshold value as it can trigger at the very start of a gradual transition. Thus to account for this, a prescribed number of frames before and after the JSD-flag must be analysed as indicated by the maximum transition duration $D_{TransMax}$ in Figure 3.1.

Using *future* frames in effect goes against the real-time criterion of a detection system, however there is a workaround. This system can be implemented by utilising a frame buffer equal to twice the maximum transition duration. In effect this buffer would accumulate $D_{TransMax}$ worth of frames before analysing them, hence causing a slight delay in detection / broadcast.

In order for this technique to function properly, some premisses had to be established with regards to the possible boundary transitions. Two of the most important pertain to the allowable durations of a transition and is characterised by $D_{TransMax}$ and $D_{TransMin}$. These premisses are further discussed during the results analysis in Section 3.3.3. In essence, by choosing the maximum and minimum duration of a transition to account for, an adequate

number of frames can be buffered and evaluated.

3.2 Analysis designs

A simulation platform was created in Visual Basic that allows various video formats to be analysed. At the core of this simulation platform is EMGU CV (version 3.0.0.2158). EMGU CV is a cross platform .Net wrapper for OpenCV (Open source computer vision) which is a library consisting of a multitude programming functions aimed at computer-vision applications [41].

In order to simulate the channel used by streaming media applications, the simulation platform loads a video and supplies the algorithm currently being evaluated with a sequential stream of video frames. Once the frames have been evaluated, they are discarded to free up memory. Research has shown that the compression and encoding properties of certain video streams and files can be exploited to further increase the processing speed of the algorithm in certain instances [42]. Irrespective of this advantage, it was decided that the algorithm being evaluated in this platform, be evaluated at *face value* when supplied with the sequential frames excluding any accompanying compression and encoding information.

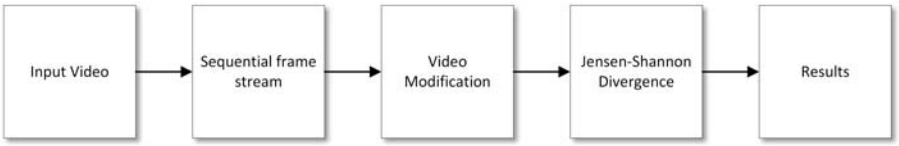


Figure 3.2: High-level flow chart of the simulation setup

In order to evaluate the processing duration of the algorithms, the logical flow as illustrated in Figure 3.2 was implemented to run linearly as a single thread. In doing so, the efficiency of the technique is evaluated and not merely the multi-threading capabilities of the hardware platform. Specifics of the hardware used in the analysis is summarised in Table 3.1.

The analyses that follows, were each created to test certain aspects of the video segmentation process and techniques with the results thereof discussed separately in Section 3.3.

3.2.1 Analysis 1: JSD - Colour space comparison

In these modern times, digital media is predominantly available in colour. In contrast to this fact, the majority of video segmentation techniques utilise a monochromatic video for the analysis. The aim of analysis 1 is to evaluate if there is any substantial gain in accuracy by

Table 3.1: Technical specifications of the simulation hardware

COMPONENT	SPECIFICATION
CPU	Intel Core i7-4470 3.4GHz
RAM	16GB DDR3 1600MHz
Motherboard	MSI Z87-GD65
Storage	Samsung 750 Evo SSD
GPU	Gigabyte Radeon R9 290x
Operating System	Windows 8.1 x64

using colour media all the while keeping track of the execution time thereof. This analysis formed part of the SATNAC 2014 article [19] and is included in Appendix C.1.

This analysis was implemented by utilising specially made test sequences in order to simplify the ground-truthing process and ensure accuracy. A linear depiction of these test sequences is illustrated in Figure 3.3.

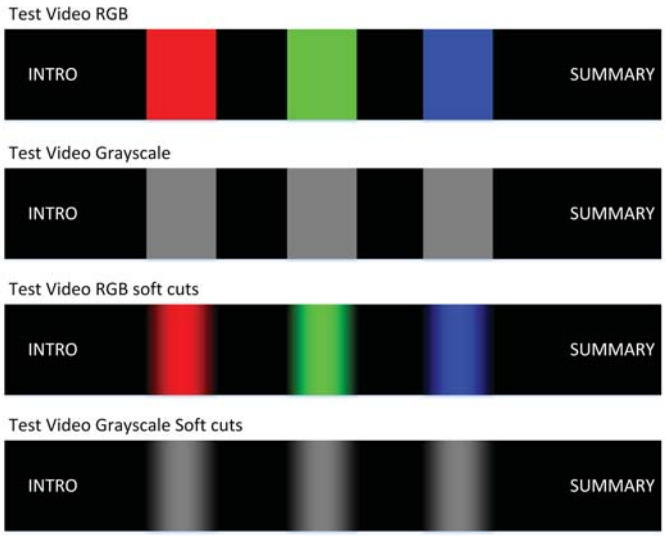


Figure 3.3: Test video sequences

The sequences were constructed to have the following properties:

- *Test Video RGB:*
  Black video sequence transitioning to individual R, G and B sequences featuring hard-cuts;
- *Test Video Grayscale:*

Black video sequence containing gray sequences featuring hard-cut transitions;

- *Test Video RGB soft cuts:*
  Soft transitions from black to individual R, G and B sequences featuring a cross-dissolve with a transition duration of 25 frames;
- *Test Video Grayscale soft cuts:*
  Black video sequence to gray sequences featuring cross-dissolves.

The hypothesis behind the full color versus monochromatic representation was to test how the accuracy of the technique holds up with the loss of data as well as the effect thereof on the execution speed. It is important to note that the gray sections in the aforementioned test videos were created by a RGB value of R=128; G=128; B=128 on the RGB 8bit color map. This RGB-grayscale representation allowed the JSD technique be compared on RGB-gray (3 separate histograms) and monochromatic-gray (1 histogram) videos. All video sequences were converted to monochromatic representations where applicable in the various algorithms by the principle explained in equation 3.15:

$$monochromatic = \frac{R + G + B}{3} \tag{3.15}$$

where R, G and B are the individual component values of each pixel.

3.2.2 Analysis 2: SDPI - Transition classification

In order to better detect a transition, it is imperative to know how the transition will be *perceived* by the detection algorithm. In order for the SDPI technique to be applied with the goal of detecting shot boundaries, the metrics of the algorithm first had to be evaluated. During this transition classification analysis, the SDPI algorithm was evaluated when supplied with various different shot boundaries which include the following:

- Hard-cuts;
- Additive-dissolve;
- Cross-dissolve;
- Film-dissolve
- Center zoom;

- Dip to black;
- Iris-box;
- Page-peel;
- Slide.

The goal of this analysis is to evaluate the SDPI technique and characterise the aforementioned shot boundaries and the transitions that are contained within. Since the JSD technique has been previously used as boundary detection technique, it was evaluated alongside the SDPI technique in order to compare the difference measures. This analysis formed the basis of the SATNAC article by De Klerk et al. in [43], which is attached in Appendix A Section C.2.

Analysis data

Adobe Premier was used to join two video segments created from the Windows 7 sample video named *Wildlife*. These video segments were subjected to the aforementioned transitions to create test videos. The videos were encoded using the Flash Video format (.flv), using a framerate of 29.97 Frames Per Second (FPS) with a resolution of 1280×720. The shot boundary between the two segments was created with the second segment starting on frame 60, hence a hardcut shot boundary. All the other gradual transitions started on frame 49 and ended on frame 73.

3.2.3 Analysis 3: JSD - Parameter sensitivity analysis of the JSD algorithm

As previously stated, the JSD algorithm in itself is a useful frame difference measure, but has to be implemented alongside a threshold technique for it to be effective. This union is investigated by performing a sensitivity analysis on the relevant parameters. The analysis described in this section forms part of an article which has been accepted for publication by the South African Institute of Electrical Engineers (SAIEE) - Africa Research Journal, with the submitted article included in Appedix C.4.

Lienhart et al. performed a comparison of some automatic shot boundary detection algorithms in [44]. The main focus of the algorithms in this comparison was the accuracy thereof with no mention of the processing speed or duration. Due to the real-time criteria, the execution duration of the technique has to be considered alongside the accuracy thereof.

It is clear from Equation 3.11 that Jensen-Shannon divergence has limited parameters which

can be analysed with the hope of optimising it. The only variable aspects are the probabilities calculated from the frames. Hence the best area for evaluation and optimisation of the technique can be identified as the manner in which this inter-frame variance is compared - i.e. the threshold. Thus the evaluation of the Jensen-Shannon divergence technique's efficiency as a shot boundary detection technique is accomplished by performing a basic sensitivity analysis of the following parameters pertaining to the threshold:

- Auto-adjusting threshold frame count (ω);
- Minimum threshold scalar (η);
- Average threshold scalar (α);
- Resize factor (RF).

The RF isn't a direct parameter utilised in any of the threshold equations, but rather a scalar describing the size of the video used for the analysis as a percentage of the original. During the threshold comparison, the auto-adjusting threshold expressed by Equation 3.14 is multiplied by the average threshold scalar α allowing the operator to dictate how much greater the inter-frame variance must be to be classified as a shot boundary. This evaluating threshold Y has a further constraint imposed where a minimum threshold is used to evaluate the inter-frame variance, if the evaluating threshold was too low:

$$\mathrm{Y} = \begin{cases} \alpha\bar{\tau}, & \text{if } \alpha\bar{\tau} > \eta \\ \eta, & \text{otherwise.} \end{cases} \tag{3.16}$$

Thus the final outcome of the boundary detection test is represented by:

$$DetectedBoundary = \begin{cases} \text{Yes}, & \text{if } JSD(f_{n-1}, f_n) \geq \mathrm{Y} \\ \text{No}, & \text{otherwise.} \end{cases} \tag{3.17}$$

Analysis data

Due to infinite video possibilities that can be encountered, a diverse collection of videos were chosen to evaluate the video segmentation technique. These videos encompasses numerous transitional effects as well as various sizes and frame rates. In order to establish a baseline, a few test videos were created in Adobe Premier where the same two shots were joined using

the aforementioned transitions methods into test sequences as previously described in Section 2.6.1.

3.3 Analysis results

Various analyses were conducted, each evaluating certain criteria pertaining to video segmentation. This section discusses the results obtained from these analyses as well as the collaborative implications thereof.

3.3.1 Evaluation metrics

During the analyses, various metrics were employed to evaluate the effectiveness of the technique or algorithm. The most prevalent of these metrics are the recall and precision rates while also keeping track of the execution duration. Recall and precision rates for boundary detection are defined in the sections below. These evaluation metrics also pertain to other detection analyses, such as the detection of videos within a stream, as described in Chapter 5.

While the techniques might be consistent with regards to the frames identified as the shot boundaries, it is however imperative to determine if these detections coincide with the actual shot boundaries. The video detection algorithm uses the detected shot boundaries to identify frames to fingerprint. Hence the goal is to ensure that the shot boundaries are recalled correctly and precisely in order to ensure the correct shot's frames are being fingerprinted.

Recall

Recall rates provide a ratio of the number of relevant shot boundaries (correct detections) to the total number of relevant shot boundaries within the video as expressed in Equation 3.18.

$$Recall = \frac{Correct}{Correct + Missed} \tag{3.18}$$

Precision

The precision rates of the algorithm provide the ratio of relevant shot boundaries to the total number of irrelevant shot boundaries (false detections) retrieved as expressed in Equa-

tion 3.19 [45].

$$Precision = \frac{Correct}{Correct + FalsePositive} \tag{3.19}$$

Execution time

The execution time of each analysis will be recorded alongside the recall and precision rates thereof. This will help to justify a trade-off analysis between the technique parameters in terms of accuracy versus the execution time required. This becomes particularly important for real-time applications.

The execution time includes the following:

- Opening the video and reading the frames as they are required;
- Converting from RGB to monochrome where required;
- Resizing of the incoming frames where required.

3.3.2 Analysis 1 results: JSD - Colour space comparison

The execution times listed were obtained by averaging the execution times of the JSD SBD technique over 10 iterations. The results of the RGB test video shot boundary detection analysis are listed in Table 3.2. This confirms the intuitive assumption that an RGB comparison with 3 histograms takes longer to execute compared to a monochromatic comparison with a single histogram. The *JSD Grayscale* technique listed includes the time required for the RGB-to-Grayscale conversion with the `rgb2gray` function. The weak recall rate produced by the RGB techniques can be attributed to the threshold-criteria.

The threshold-criteria for RGB videos is essentially the same as for the monochromatic videos. The only difference is that it is implemented on all three colour component histograms. The threshold concept for monochromatic videos to detect a shot boundary (SB) is given by Equation 3.20:

$$SB = \begin{cases} True & \text{if } JSD(f_{i-1}, f_i) > T_{aveGray} \\ False & \text{otherwise} \end{cases} \tag{3.20}$$

where $T_{aveGray}$ is the average moving threshold computed from the *moving average window*.

Similarly for RGB videos, each color channel represented by Equations 3.21, 3.22 and 3.23 must be evaluated:

$$SB_R = \begin{cases} True & \text{if } JSD_R(f_{i-1}, f_i) > T_{aveR} \\ False & \text{otherwise} \end{cases} \tag{3.21}$$

$$SB_G = \begin{cases} True & \text{if } JSD_G(f_{i-1}, f_i) > T_{aveG} \\ False & \text{otherwise} \end{cases} \tag{3.22}$$

$$SB_B = \begin{cases} True & \text{if } JSD_B(f_{i-1}, f_i) > T_{aveB} \\ False & \text{otherwise} \end{cases} \tag{3.23}$$

in order to calculate if a shot boundary has been found. A shot boundary has been identified if Equation 3.24 is satisfied:

$$SB_{RGB} = \begin{cases} True & \text{if } SB_R \ \& \ SB_G \ \& \ SB_B = True \\ False & \text{otherwise.} \end{cases} \tag{3.24}$$

The duration of the test sequence used is 37 seconds.

Table 3.2: Test Video - RGB

SBD Technique	**Execution Time (s)**	**Precision (%)**	**Recall (%)**
JSD Grayscale	3.83	100	87.5
JSD RGB	5.08	100	14.29
JSD Grayscale Resized	1.11	100	87.5
JSD RGB Resized	1.85	100	14.29

The native grayscale test video produced better recall rates as can be seen in Table 3.3. The only difference between the RGB and grayscale test videos are the colours. The RGB video has definitive red, green and blue scenes, while the grayscale video has only gray scenes amidst the black timeline. These gray scenes contain equal values for all the RGB channels (R=G=B=128). Hence the boundary detection conditions, as required by Equation 3.24 is easily satisfied for if one channel is satisfied, the remaining two will be satisfied as well. The duration of the test sequence used is 37 seconds.

Table 3.3: Test Video - Grayscale

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.85	100	87.5
JSD RGB	5.17	100	87.5
JSD Grayscale Resized	1.09	100	100
JSD RGB Resized	1.78	100	14.29

Soft cut test video results

The nature of a gradual transition makes it difficult to distinctively classify a certain frame as the shot boundary. The recall and precision rates listed in Tables 3.4 and 3.5 were calculated by determining if the detected boundary was within the frame-range constituting the gradual transition. The duration of the test sequence used is 37 seconds.

Table 3.4: Test Video - RGB Soft Cuts

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.80	70	87.5
JSD RGB	4.93	100	87.5
JSD Grayscale Resized	1.10	77.78	87.5
JSD RGB Resized	1.81	50	91.3

The lower precision values are attributed to the false detections during the gradual transition of both the monochromatic and RGB videos. Only one shot boundary was allowed within the span of the gradual transition as the definition of a shot boundary implies that each shot has a start and stop boundary. A gradual transition from black to white is illustrated in Figure 3.4 over a period of 10 frames. Adhering to the aforementioned guidelines regarding the gradual transition, only one frame should be identified as the shot boundary e.g. frame nr. 5, even though frame 6 might also provide a difference value higher than the current threshold. The duration of the test sequence used is 37 seconds.

Hence the low precision is caused by the identification of a second shot boundary alongside the current shot boundary within the same transition period.

All the test videos have an introduction title and information summary at the end of the sequence. It is debatable whether or not these text sequences can be classified as a shot or

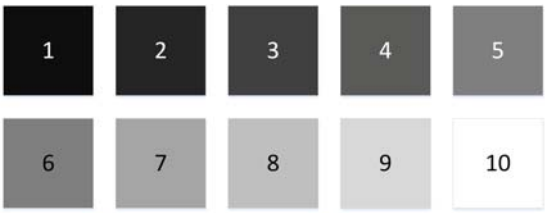


Figure 3.4: Gradual Transition

Table 3.5: Test Video - Grayscale Soft Cuts

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.77	53.85	87.5
JSD RGB	4.83	53.85	87.5
JSD Grayscale Resized	1.10	80	100
JSD RGB Resized	1.81	80	12.5

not. Although the majority of the techniques failed to identify the addition or removal of these text sequences, with the exception of both the resized implementations of the JSD algorithm on the grayscale test sequences.

Other video results

The same analysis was done on a video generally available on the internet in order to determine how they fared compared to the best case scenarios of the test videos. The action-sports themed RedBull commercial contains multiple lens-flares and flash photography that in turn gave rise to a few false detections. The results listed in Table 3.6 indicate worse recall and precision rates for the original video when compared to the smaller resized versions thereof.

Upon further investigation into this peculiarity, it came to light that the frame rate of the video has an effect on the accuracy of the techniques being evaluated. The original RedBull Commercial was a 1280×720 video file with a frame rate of 30 FPS. During the resize process, the video was down-sampled to 25 FPS, the same as all the test videos.

A higher video frame rate constitutes more frames during a gradual transition, in effect causing less change from frame-to-frame. Thus in order for the JSD technique to function correctly at a higher frame rate, the size of the moving average window used for thresholding should be increased while the threshold might be lowered to detect the *slower changes* in a frame-to-frame

context. The duration of the test sequence used is 60 seconds.

Table 3.6: The World of RedBull TV Commercial 2013

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	7.31	72.73	92.31
JSD RGB	10.49	74.19	88.46
JSD Grayscale Resized	1.87	76.67	100
JSD RGB Resized	2.94	87.5	100

The final video used in the comparison was a wildlife video, depicting various scenes with slow and fast moving animals, hard cuts between shots and no lens-flares. The results, as listed in Table 3.7, shows a perfect score for both recall and precision across all the SBD techniques. The duration of the test sequence used is 30 seconds.

Table 3.7: Wildlife

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.11	100	100
JSD RGB	4.29	100	100
JSD Grayscale Resized	0.88	100	100
JSD RGB Resized	1.40	100	100

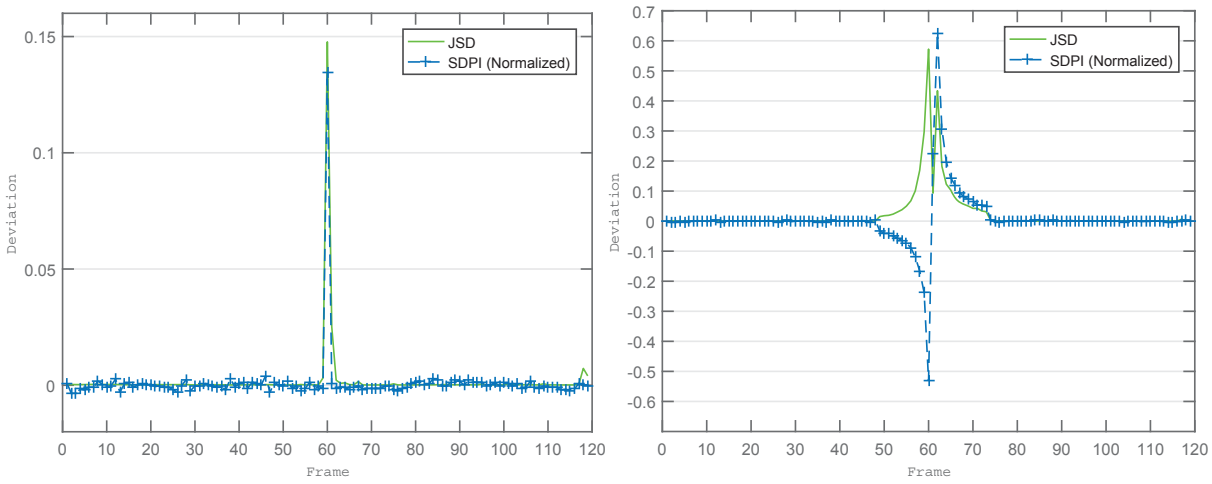
Analysis 1 conclusion

From this analysis it is clear that utilising all the available color histograms during a shot boundary detection analysis can provide slightly higher precision rates. This is however not a consistent observation as it is a lot more sensitive to the threshold values.

In summary the possibility of a slightly higher precision rate does not justify the additional execution time required.

3.3.3 Analysis 2 results: SDPI - Transition classification

As expected, the analysis of a hard-cut transition results in a *clean* discontinuity for both the JSD as well as the SDPI algorithms, as illustrated in Figure 3.5a. Furthermore the amplitude of the discontinuity greatly exceeds that of the surrounding values i.e. producing a great *signal-to-noise* ratio.



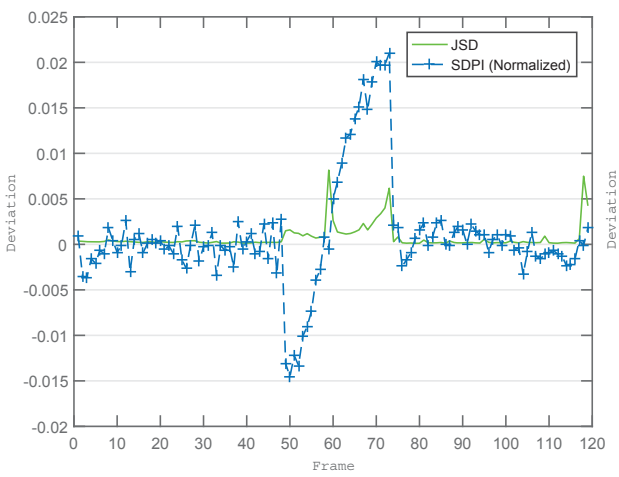
(a) Hard-cuts (b) Dip-to-black

Figure 3.5: Dissolve transition analyses results

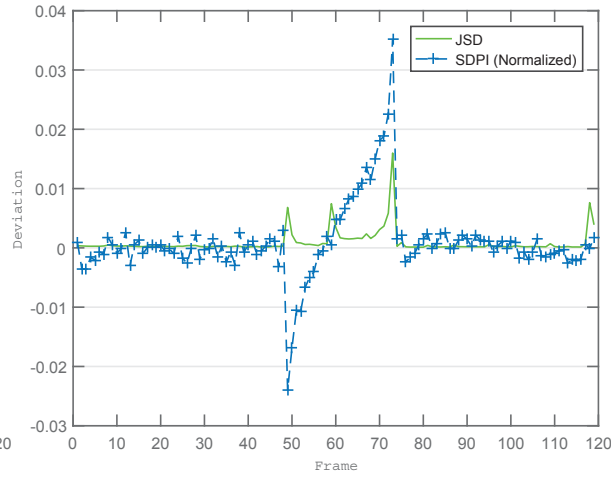
This is however not the case for gradual transitions. The same analysis of the cross-dissolve feature as seen in Figure 3.6a, resulted in multiple discontinuities. The amplitude of these discontinuities are still greater than the surrounding values, however this does pose a challenge in the sense of identifying the start and end of the transition amongst the noise. The film-dissolve transition shown in Figure 3.6b has fairly similar characteristics to cross-dissolve, but both differ greatly from the characteristics of the additive-dissolve transition in Figure 3.6c.

In certain instances even the JSD algorithm results in multiple peaks that constitute the transition as seen in Figure 3.5b. The dip-to-back transition is essentially a fade-to-black followed by a fade-from-black. Although this transition is visually observed as a single entity, it is detected as two separate JSD spikes due to the black frame(s) between the fades. The analysis results of the other transitions are shown in Figures 3.7a, 3.7b, 3.7c and 3.7d. From these various transitions it becomes apparent that the detection characteristics of the transitions are extremely discrepant. During these analyses, the transition duration was kept constant across the various transitions, with the exception of the hardcut transition. These characteristic discrepancies are further exacerbated when the duration of the transition is altered.

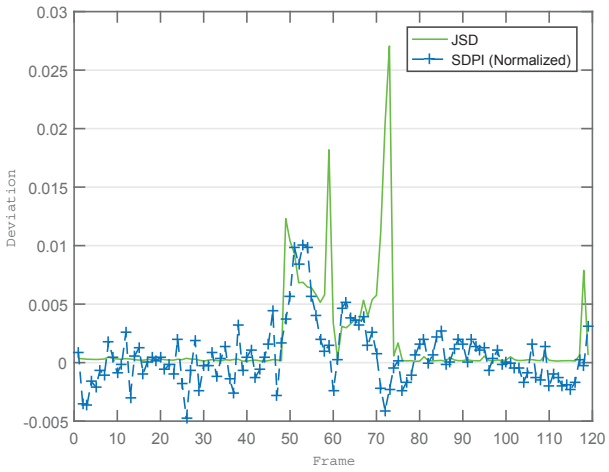
The variable nature of the transitions requires an interval analysis rather than an instantaneous analysis. This calls for some premisses to be introduced in order to account for these interval analyses. Transitions are implemented to smooth the visual discontinuity as observed by the user. A common frame-rate encountered in digital video is 25 FPS and higher. The human



(a) Cross-dissolve



(b) Film-dissolve



(c) Additive-dissolve

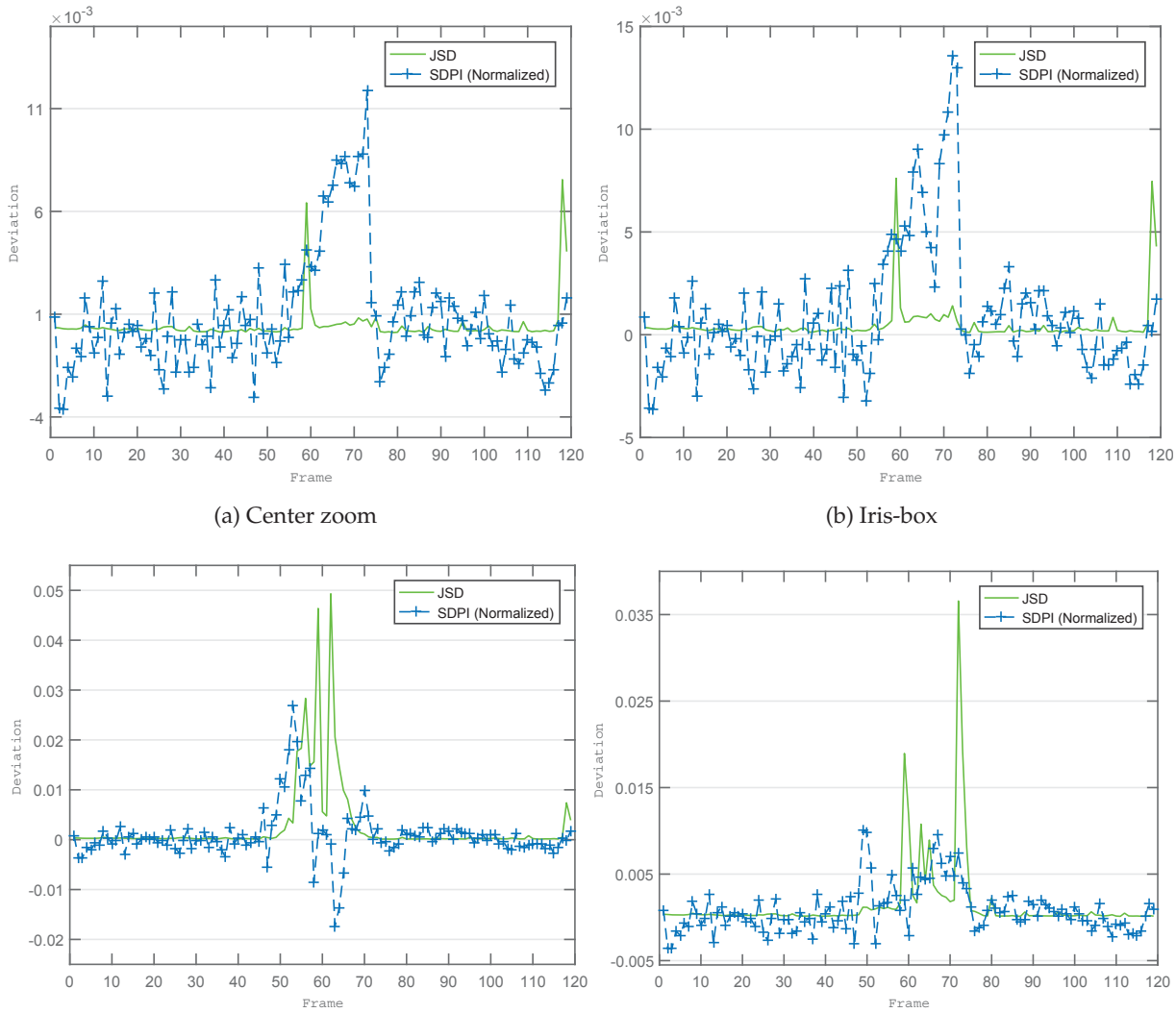
Figure 3.6: Various dissolve transition analyses results

eye perceives motion as being fluid at about 18 FPS for typical cinematic films because of its blurring [46].

By means of experimental observation it was found that in order to preserve the fluidity during a transition, it can be assumed that the duration of the transition should be at least $\frac{1}{3}$ of the video's frame rate. Shorter durations are perceived as a *flash* due to the abrupt change. This implies that for the test videos, this duration $D_{TransMin}$ is

$$D_{TransMin} = \frac{FPS}{3} = \frac{29}{3} \approx 10 \tag{3.25}$$

frames. Inversely the maximum duration of a transition has to be stipulated as well. Without



(a) Center zoom

(b) Iris-box

(c) Page peel

(d) Slide

Figure 3.7: Other transition analyses results

this parameter, the inherent ever-changing nature of a video will cause the boundary detection to be open ended - i.e. a time-lapse video of a sunset spanning multiple minutes can be misinterpreted as an extremely long fade-to-black. This maximum transition duration $D_{TransMax}$ is set at the frame rate of the video implying a maximum transition duration of 1 second which is

$$D_{TransMax} \approx 30 \tag{3.26}$$

frames.

Transition properties

A characteristic common to most of these transitions is an extreme local minimum and maximum within the duration $D_{TransMax}$. Theoretically the center of the shot boundary should be located between these two points. When analysing a composite video with a structure as detailed in Table 3.8, the threshold problem becomes apparent as seen in Figure 3.8a. This same analysis results are illustrated using a log scale in Figure 3.8b. The amplitude of the transition indicators for both the JSD as well as SDPI is dependent on the type of transition as well as the composition of the sequential frames being analysed.

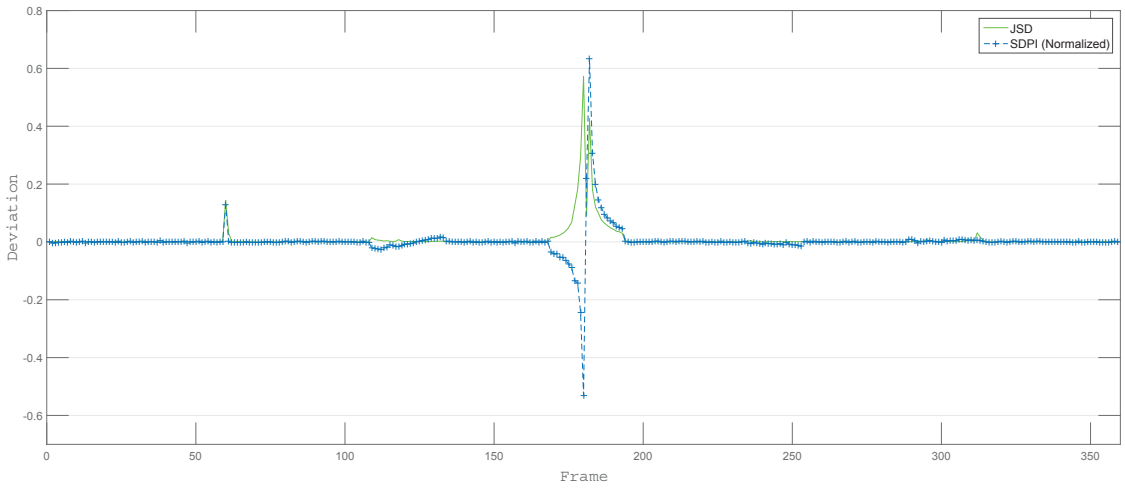
Table 3.8: Composite Video transition indices

TRANSITION	***TRANSITION INDICES***		
TYPE	***START***	***SHOT BOUNDARY***	***END***
Hard-cut	-	60	-
Cross-Dissolve	109	120	132
Dip to Black	169	180	193
Iris Box	229	240	253
Slide	289	300	313

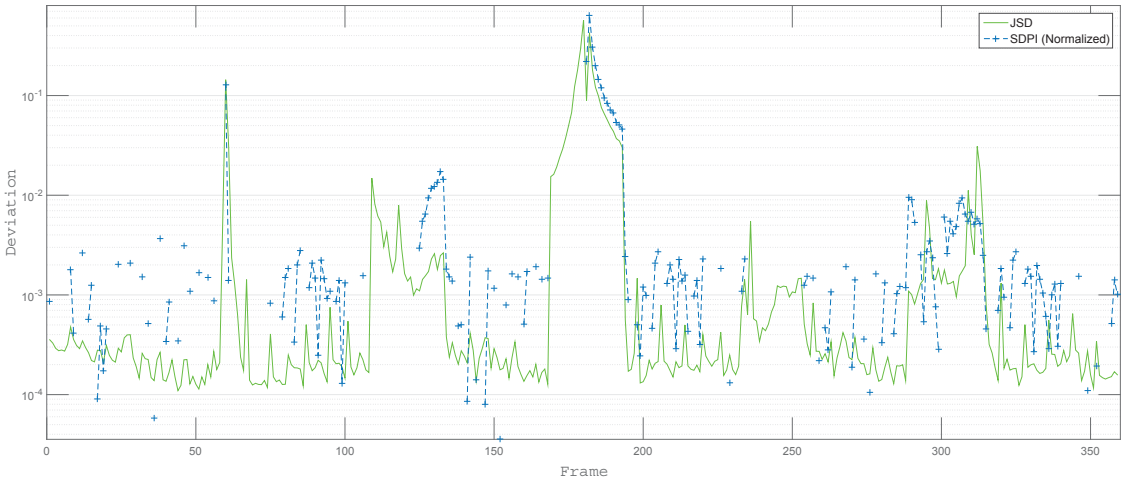
It was observed that the JSD technique as previously implemented in [19] with an average threshold of 3.5 and minimum threshold of 0.02, was able to detect the hard-cut at frame 60 and the center of the dip-to-black transition at frame 180 while producing no positives on the other transitions. When the minimum threshold was lowered to 0.005, the JSD algorithm was able to detect the deviation caused by the various transitions, but it also identified multiple false positives.

Analysis 2 conclusion

The greatest advantage of using the JSD technique is the fast processing speed. This is however contrasted with the inability to detect multiple soft transitions. When utilising a lower threshold, the deviations caused by the soft transitions are detected alongside the false positives. Even though the deviations are detected, the start and end of the transition that caused the deviation are still unknown.



(a) Composite test video analysis



(b) Additive-dissolve

Figure 3.8: Composite test video analyses results

The implementation of the standard deviation of pixel intensities should allow the detection of the soft transitions present in the video but at the cost of processing speed. Thus by combining the processing speed of the JSD technique with the detection capabilities of the SDPI technique, a hybrid technique is formed that should be able to detect the hard and soft transitions while keeping processing time to a minimum. The particulars of the hybrid technique alluded to are discussed in Section 3.1.4.

3.3.4 Analysis 3 results: JSD - Parameter sensitivity analysis of the JSD algorithm

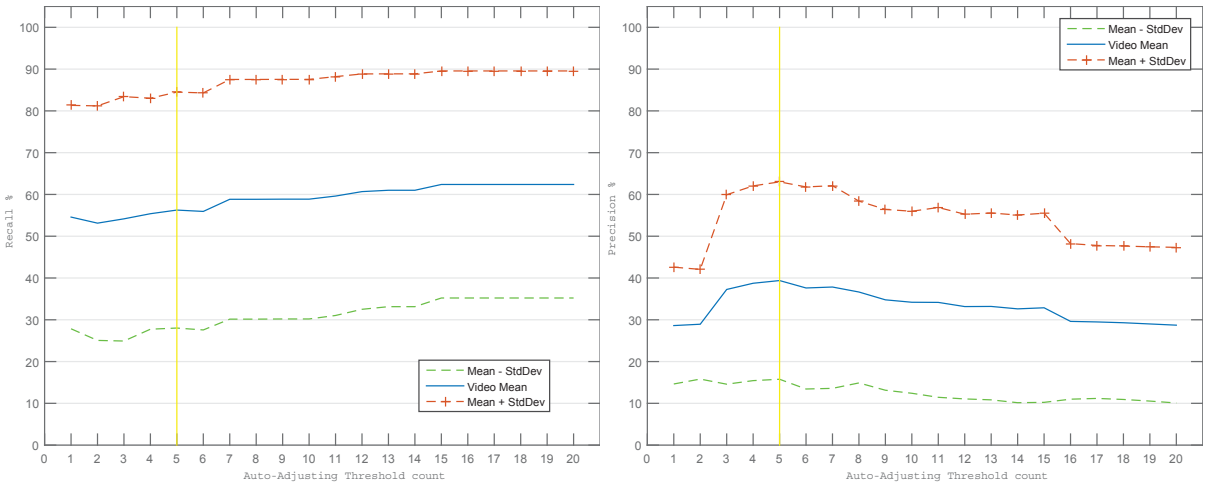
The analysis results in this section are grouped according to the parameter under investigation. The majority of all the graphs in this section are representative of the average values produced by each of the 31 test videos as referred to in Section 3.2.3. The influences of video resizing is incorporated in the other analysis.

Auto-adjusting threshold frame count (ω)

The first parameter that was investigated is the number of frames used by the auto-adjusting threshold as this outcome was used as an input parameter for the subsequent analyses.

While some of the test videos resulted in very good recall and precision rates, other, especially some videos with abnormal gradual transitions, performed poorly. In order to provide a representative sample, the average recall rates were calculated of all 31 videos with an accompanying standard deviation. The same was done with the precision values.

The average recall response of the system for a resize factor (RF) equal to 1 is illustrated in Figure 3.9a which seems to be increasing fairly linearly with regards to an increase in the number of frames.



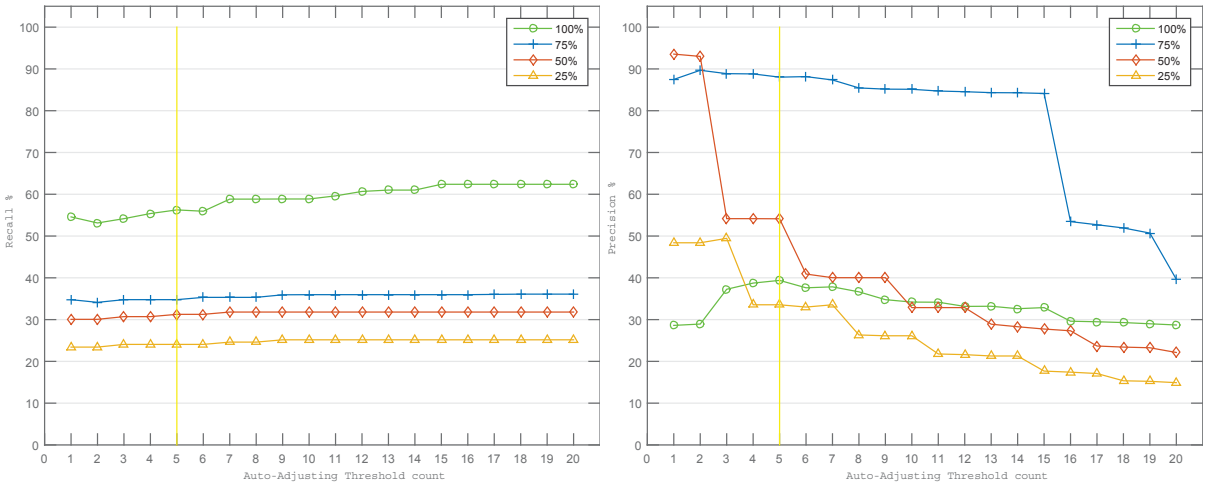
(a) Recall (b) Precision

Figure 3.9: Auto-adjusting threshold analysis values for RF = 1

This is however not the case for the accompanying precision response as seen in Figure 3.9b. An initial increase in the precision rate is observed but then followed by a steady decline. A

maximum precision response was observed at a $\omega = 5$.

Similarly the recall behaviour for the other resize factors follow the same trend seen in Figure 3.10a with increasing linear patterns but lower average values.



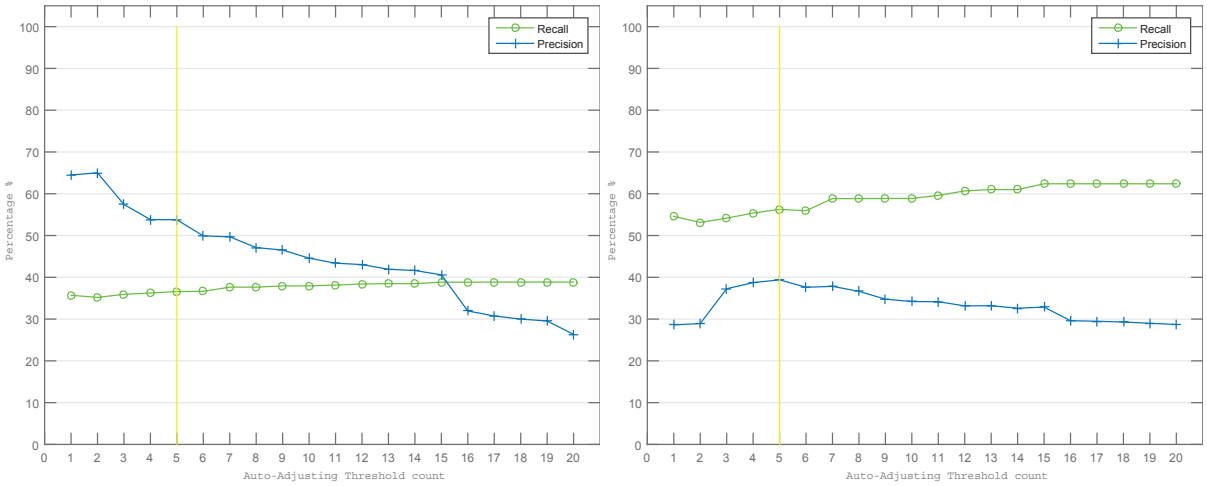
(a) Recall (b) Precision

Figure 3.10: Auto-adjusting threshold analysis values for all RF

Logic infers that the precision values will also follow suit. Generally this assumption is correct that all the resize factors follow a steady decline, however the concave form with a local maximum is not as clear. There is an anomaly with regard to RF = 75% as shown in Figure 3.10b. The precision rates for this RF is noticeably higher than the other RF values. The possible cause for this might be attributed to the bilinear resizing algorithm that is applied to the frame. By resizing the frame, the image is condensed, but at RF = 75% still retains a large amount of unique information for an inter-frame variance to be calculated.

The ideal frame count will result in maximum recall and precision values, hence the intersection of the two graphs as seen in Figure 3.11a, which represents the average recall and precision values across all RF values.

The intersection is visible at a frame count $\omega = 15$ with an average recall and precision rate of approximately 39%. Due to the aforementioned anomaly at the 75% resize factor, the intersection point for this RF was specifically evaluated. The point of intersection for RF = 75% was found to be at a frame count $\omega = 21$ with an average recall and precision rate of approximately 35%. These rates are however deemed too low. The recall and precision rates



(a) Average of all RF values

(b) RF = 1

Figure 3.11: Auto-adjusting threshold analysis recall and precision values

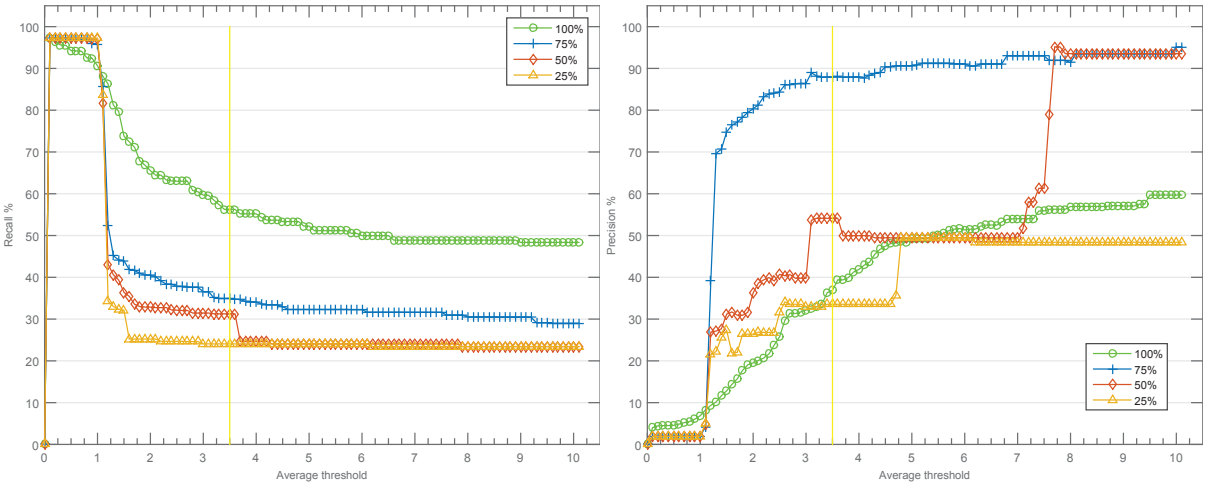
for RF = 100% is shown in figure 3.11b with a recall rate of 67% and a precision rate of 40% at a frame count $\omega = 5$. From this it is clear that $\omega = 5$ combined with RF = 100% provides the best trade-off between recall and precision values.

Average threshold sensitivity (α)

The second parameter to be evaluated was the average threshold sensitivity while using the previously calculated frame count $\omega = 5$. The recall rates obtained show a quick decline before appearing to stabilise as seen in Figure 3.12a.

The precision rates present with a large initial increase that appears to level off at higher threshold levels shown in Figure 3.12b.

The intersecting point for RF = 75% is situated at $\alpha \approx 1.2$ which results in recall and precision rates of approximately 49%. The best rates are obtained where RF = 100% where $\alpha = 5.7$ resulting in recall and precision rates of 51%. The intersecting point when evaluating the average for all RF values is located at $\alpha = 2$ with recall and precision rates of approximately 41% as illustrated in Figure 3.13.



(a) Recall (b) Precision

Figure 3.12: Average threshold analysis for all RF values

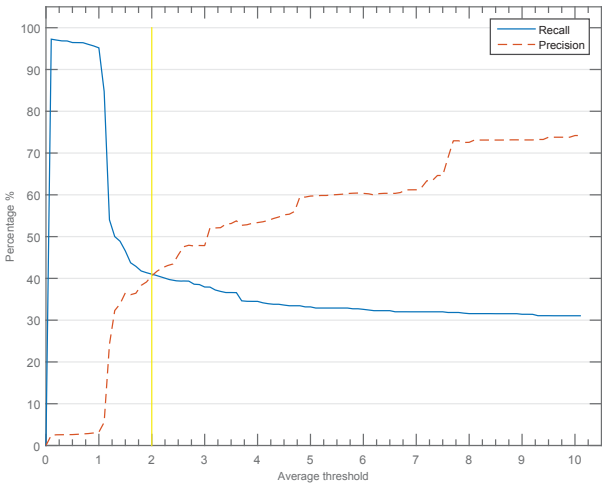
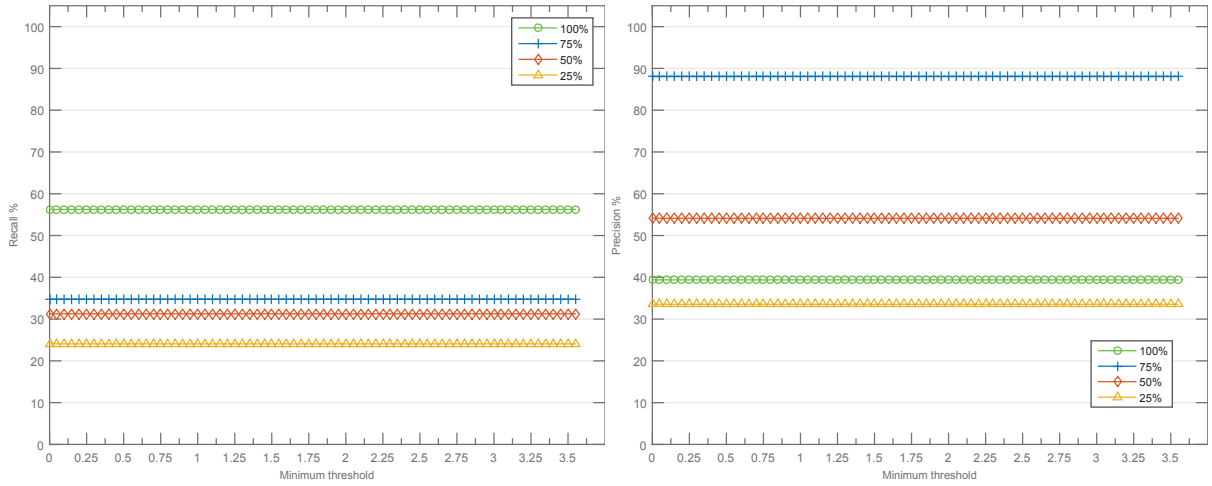


Figure 3.13: Average threshold analysis recall and precision rates for the average RF values

Minimum threshold sensitivity (η)

The sensitivity of the detection technique to variations in the minimum threshold was analysed while using an average threshold $\alpha = 3.5$ and a frame count $\omega = 5$. This analysis indicated that any variations in the minimum threshold $0 \leq \eta \leq \alpha$ does not relate to any change in the recall and precision rates for any RF as shown in Figures 3.14a and 3.14b.



(a) Recall (b) Precision

Figure 3.14: Minimum threshold analysis rates for all RF values

Analysis duration

The analysis durations for the auto-adjusting threshold frame count analysis is illustrated in Figure 3.15 from which it is clear that the analysis duration follows a fairly linear pattern with the exception of the initial threshold value of RF=1.

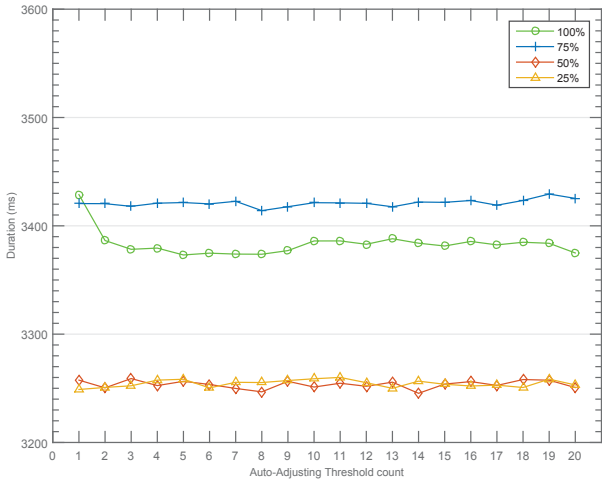


Figure 3.15: Auto-adjusting threshold analysis duration times for all RF values

The same linear behaviour is encountered during all the other investigations. The general trend for all parameter analyses as seen in Figure 3.16 indicates that the RF = 75% actually took the longest to complete.

With regards to the real-time component, the maximum average duration was encountered where RF = 75% at approximately 3435ms. When comparing this maximum to the average duration of the test video 9985ms, it is clear that the real-time criteria will be met as the analysis duration is at least 2.9 times faster than the playback duration. In order to ensure that unbiased timing was achieved, each value for each parameter under investigation was analysed 10 times with the average thereof taken as the analysis duration for that instance.

Analysis 3 conclusion

The Jensen-Shannon divergence proved to be a technique capable of detecting shot boundaries with fairly good recall and precision rates. The analysis duration of the technique more than satisfied the real-time requirements while using only the *historic* (received) data for calculations.

It is important to note that the average CPU usage was at fairly constant 35% as well as the average RAM usage at 33% throughout the analysis. This indicates that the timing for the analysis is representative of the actual execution and not perplexed by the system bottlenecking by e.g. caching to a page file if the RAM was filled.

The results indicate that a the RF = 75% has some outliers with regards to the α- and ω precision rates as well as longer execution times.

The optimal parameters based on the simulation results contained in this section are as follow:

- $\omega = 5$;
- $\alpha = 5.7$;
- η =No-effect $\therefore \eta = 0$.

3.4 Concluding remarks

When analysing a color video with a technique such as the JSD, the various color components can be integrated into a singular monochrome component to be analysed. This is done since the slight increase in accuracy can't be justified by the accompanying increase in processing time.

After determining that a singular monochrome channel will provide adequate information for the SBD algorithms, the SDPI and JSD algorithms could be evaluated. Both the SDPI and JSD

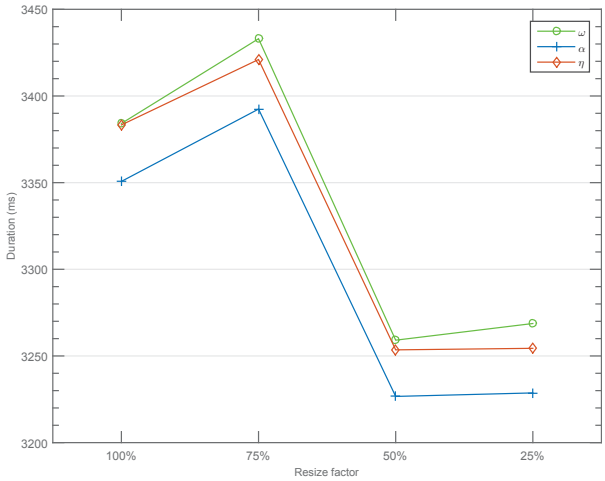


Figure 3.16: Average analysis duration times for all RF values

techniques were able to detect abrupt transitions without any problems. The SDPI technique however produced more prominent peaks that indicate possible boundary locations when analysing gradual transitions as compared to the JSD algorithm.

Based on the properties of the aforementioned techniques, the possibility of a hybrid Jensen-Shannon Divergence Standard Deviation of Pixel Intensities (JSDSDPI) technique was investigated. The increase in accuracy for both the SDPI and JSDSDPI algorithm does not justify the increased processing time. The JSD algorithm can be adjusted to better detect gradual transitions by sacrificing a little precision. In the greater scheme of the video detection process, this is a justifiable trade-off since the resulting precision when integrated with a hash algorithm, can be corrected as discussed in Chapter 5.

To summarise, within the video detection context, the best combination with regards to speed while maintaining the optimal ratio between recall and precision, is found when employing the JSD algorithm on monochrome frames with the following parameters:

- $\omega = 5$;
- $\alpha = 5.7$;
- $\eta = 0$.

Chapter 4

Video Fingerprinting

"Character isn't something you were born with and can't change, like your fingerprints. It's something you weren't born with and must take responsibility for forming." —Jim Rohn

Similarly each video is born with a set of 'fingerprints' called keypoints, but the way in which they come together, forms a unique character pertaining to that video called hashes.

The process of fingerprinting is a well established practise where an imprint of a person's fingers is taken. The patterns contained within these imprints are analysed in order to isolate distinguishing features thereof. The combination of these distinguishing features is referred to as a fingerprint.

The same concept of fingerprinting can be applied to digital video frames. The patterns present in a video frame are however more complex than the arches, loops and whorls created by the friction ridges on a finger, since the pattern generated by the pixels can be an infinite number of things. This calls for a different approach to identify distinguishing features from a digital image - i.e. keypoint extraction.

When referring to the optimisation of the video fingerprint, it is more centred around the robustness and reproducibility than the speed thereof. This also encompasses the resizing effect analysis in Section 4.4.

4.1 Keypoint extraction

Keypoints are spatial interest points within an image which are prominent amongst the rest of the image. As mentioned in Section 2.3, there are two core techniques used for keypoint extraction, called SIFT and SURF.

Both of these core techniques are contained within the EMGU CV Libraries. EMGU CV is a cross platform .Net wrapper for OpenCV (Open source computer vision) which is a library consisting of a multitude programming functions aimed at computer-vision applications [41]. Throughout the analyses, various versions were employed: initially version 2.3.0.1416 was used, with the aforementioned algorithms contained in the Emgu.CV.Features2D library. In the newer version 3.1.0.2282, SIFT has been moved to the restricted library Emgu.CV.XFeatures2D since they are considered to be non-free modules [47]. These non-free modules are however free to use for academic purposes.

When these EMGU CV techniques are employed, they take different input parameters but return the same essential parameters:

- X-Coordinate x_i
- Y-Coordinate y_i
- Key point size $\overline{kp_i}$
- Key point direction α_i

for each keypoint identified. All parameters and references to these techniques that follow, are done within the purview of EMGU CV version 3.1.0.2282.

All subsequent keypoint extraction operations are performed using the SIFT algorithm following the investigation by Panchal et al. that concluded that the SIFT algorithm was better at detecting features [24].

While the keypoints can be used as a standalone metric, it will be fairly sensitive to noise. Furthermore, as a point-metric, using the keypoint as a standalone metric drastically increases the likelihood of similar keypoints being detected in other videos. These hurdles can be overcome by creating hash descriptors from these keypoints.

4.2 Hash descriptors

Merely identifying the keypoints contained within a frame will not suffice for fast and reliable fingerprint descriptors. As a standalone identifier, a keypoint is fairly susceptible to the effects of noise and other distortions. Hence not only is the extraction of *KP*s important, but the representation thereof also has an impact on robustness and performance [48]. In some cases, the *KP*s are matched between images or to *KP*-database by minimizing the distance between the descriptors [49]. Unfortunately by using only the distance as a metric, the uniqueness of the *fingerprint* is fairly weak. This is overcome by combining the aforementioned *KP* properties in order to generate four unique descriptors for the collection of *KP*s:

$$KP = \{kp_1, kp_2, \ldots, kp_{n-1}, kp_n\}, \;\; n \in \mathbb{Z} \tag{4.1}$$

where each keypoint is indexed based on its size while ordered in a declining manner so that the following constraint holds true:

$$\overline{kp}_n \geq \overline{kp}_{n+1} \tag{4.2}$$

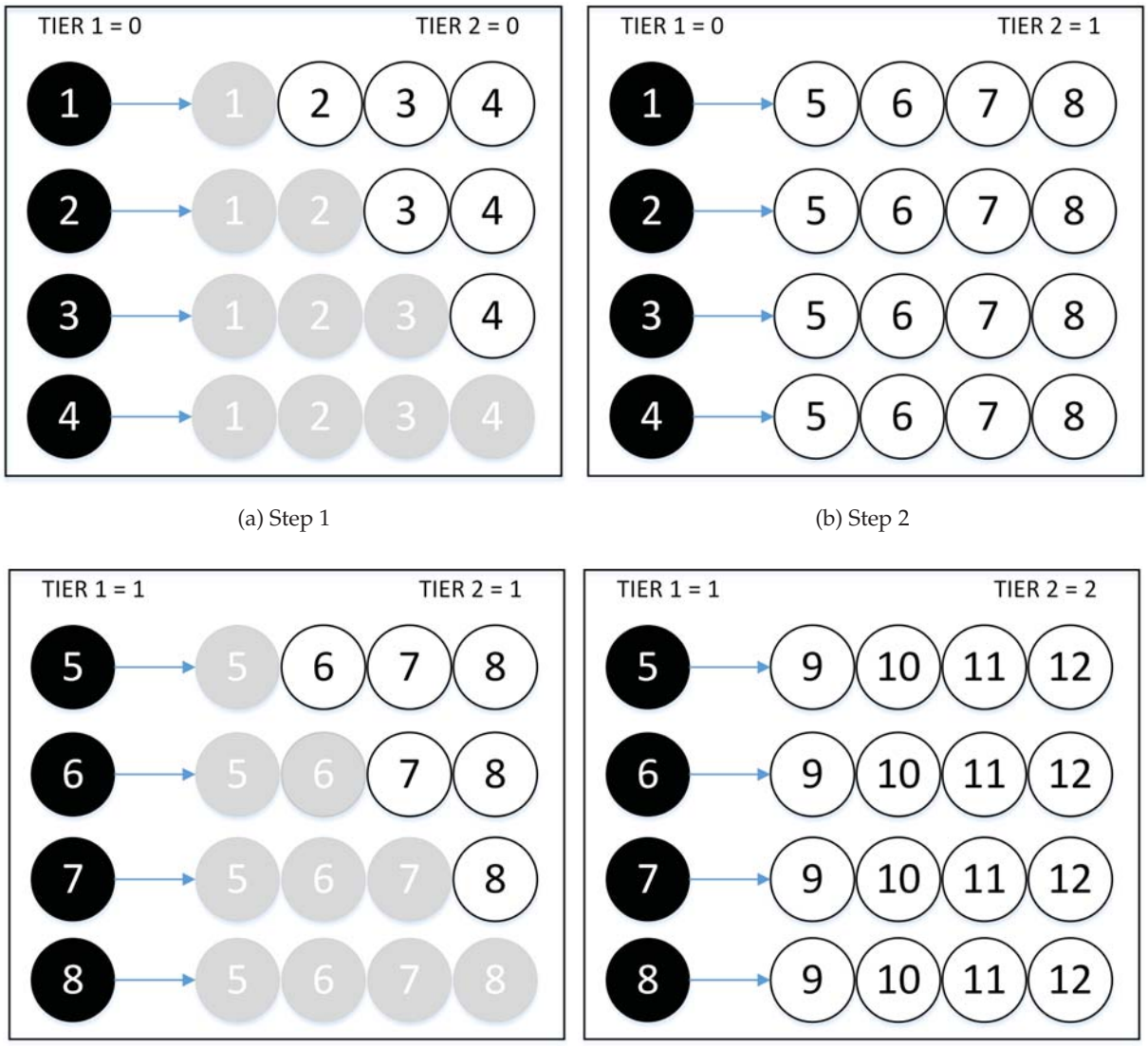
for all *KP*s in the frame.

An excerpt of this work has been presented in the SATNAC 2016 article [50] which is included in Appendix C.3.

4.2.1 Keypoint pairs

As stated by equation 4.2, the keypoints are ordered from largest to the smallest based on the size parameter. This is done in order to ensure that the most prominent keypoints are used first since the largest keypoints are the most prominent, hence most reproducible points of interest on the frame. As a further measure to ensure reproducibility of the hash descriptors from possible noise-contaminated sources, the keypoints pairs are not formed by a purely linear indexing relation, but rather the implementation of a two-tier system.

The two-tier approach maximizes the pair relation to large keypoints. This is done by dividing the available keypoints into levels. Initially both tiers are set to level 0 as illustrated in Figure 4.1a. Each keypoint in the primary tier (tier 1) is paired to each keypoint in the secondary tier (tier 2), with the exception of self-pairing when both tiers are on the same level. Once the current tiers have been paired, the secondary tier is increased to a lower level as illustrated in Figure 4.1b. In doing so, the smaller keypoints are now *anchored* to the larger ones. The primary



(a) Step 1

(b) Step 2

(c) Step 3

(d) Step 4

Figure 4.1: Two-Tier structure of keypoint pairs

tier is either on the same level, or one level behind the secondary tier's depth as is illustrated in Figure 4.1c and 4.1d.

This approach causes multiple permutations of keypoint pairs and is repeated until all the keypoints have been utilised or, the desired number of keypoint-pairs have been reached to calculate hashes from. For the use case in this example, the two-tier approach with a depth level of 4 will create 7 keypoint pairs with keypoint 1 as the anchor or parent keypoint as illustrated in Figure 4.2.

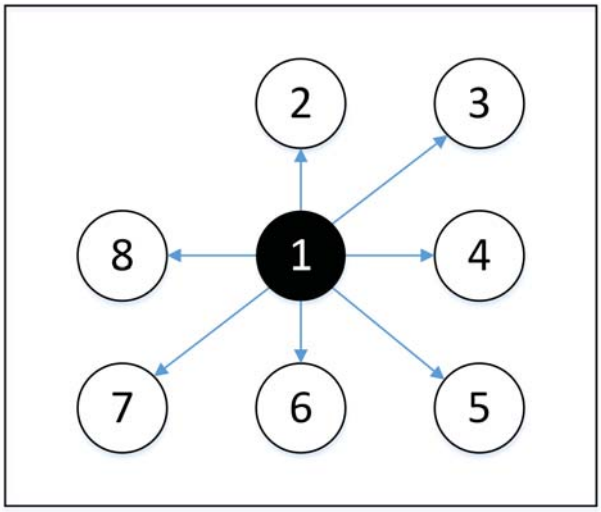


Figure 4.2: KP-1 pair relations

The argument exists - why not use a fixed relation for each keypoint - i.e. link one keypoint to the next 10 subsequent keypoints? Such an approach will work, and concentrate the keypoint pairs within the prominent areas which is positive. But this comes at a cost. If any distortion should compromise the prominent area of the frame, there will be very little usable features left. Thus by adjusting the depth of the two-tier approach, one can specify how many keypoint pairs to be created using the larger keypoints, while still providing some diversity through linking them to smaller keypoints.

4.3 Hash generation

The manner in which these KP pairs are related to one another has a prominent effect on the hash that is generated. The various components from which the hash is comprised is discussed in the following section. Some of these components originated in the legacy implementation, but have evolved to become more robust and generally applicable.

4.3.1 Relative magnitude (λ)

The relative magnitude λ represents the magnitude of one KP to another. Due to the constraint expressed in Equation 4.2, the maximum bits used to represent the relative magnitude will not overrun the bit ratio $\mathcal{B}_{\mathcal{R}}$. This relative magnitude is calculated by:

$$\lambda = \frac{\overline{kp}_{n+1}}{\overline{kp}_{n}} \times \mathcal{B}_{\mathcal{R}} \tag{4.3}$$

where the bit ratio is expressed as a function of the number of bits used per hash descriptor $\mathcal{B}_{\mathcal{H}}$:

$$\mathcal{B}_{\mathcal{R}} = \mathcal{B}_{\mathcal{H}} \times 4. \tag{4.4}$$

4.3.2 Relative direction (ω)

The relative direction is the resulting angular component of the difference vector calculated from the individual KP-directions. These KP-directions are indicated by green arrows as illustrated in Figure 4.3.

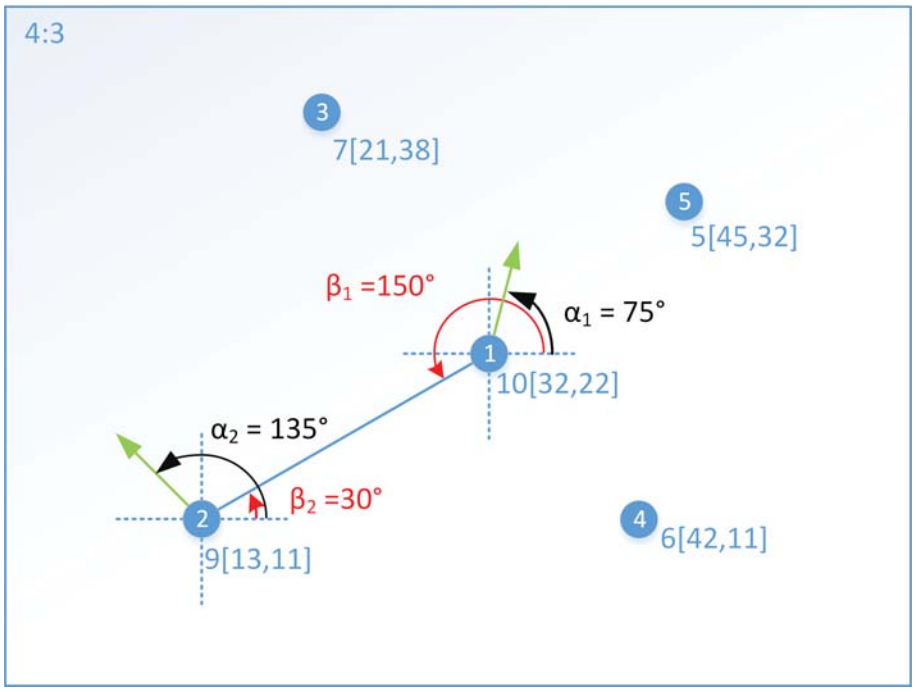


Figure 4.3: Frame key point example

The difference in angular direction α_{diff} is calculated by

$$\alpha_{diff} = \alpha_{n+1} - \alpha_n \tag{4.5}$$

which allows the relative direction ω to be expressed as a function of the $\mathcal{B}_{\mathcal{R}}$:

$$\omega = \alpha_{diff} \times \frac{\mathcal{B}_{\mathcal{R}}}{2\pi}. \tag{4.6}$$

4.3.3 Keypoint distance (Ω)

The KP distance relates the position of the following KP_{i+1} to the current KP_i. The distance Δ between the keypoints is calculated from the difference in the $[X;Y]$ coordinates of the KP:

$$x_{diff} = x_{n+1} - x_n \tag{4.7}$$

$$y_{diff} = y_{n+1} - y_n \tag{4.8}$$

with the

$$\Delta = \sqrt{x_{diff}^2 + y_{diff}^2}. \tag{4.9}$$

The native implementation by Moolman et al. expressed this distance as a function of two hash generation constraints R_{min} and R_{max}. The aforementioned constraints are used to describe the minimum R_{min} and maximum R_{max} distance that the succeeding KP may be from the parent, and still be related to it. If the KP falls outside of the range $R_{max} \geq \Delta \geq R_{min}$, none of the 4 hash parameters will be generated using this KP-pair, and the hash-generation moves on to the next KP-pair.

If these constraints are satisfied, the keypoint distance Ω is natively expressed as:

$$\Omega = \frac{\Delta - R_{min}}{R_{max} - R_{min}} \times \mathcal{B}_{\mathcal{R}}. \tag{4.10}$$

4.3.4 Keypoint direction (θ)

The last hash parameter relates the relative direction between the KP pair. The traditional inverse tangent function, Arctan $atan(\alpha)$ or commonly written as $tan^{-1}(\alpha)$, has been replaced with the more comprehensive inverse tangent function Arctan2 $atan2(\alpha)$. This allows the relative direction to be calculated and expressed in all 4 quadrants of the Cartesian coordinate system since $atan$ cannot distinguish polar opposite directions. This allows the relative KP direction θ to be calculated as:

$$\theta = \frac{-atan2\left(x_{diff}, y_{diff}\right) - \alpha_n}{2\pi} \times \mathcal{B}_{\mathcal{R}}. \tag{4.11}$$

Now that the 4 unique descriptors have been generated, they can be combined to form a singular representative descriptor called a hash. This resulting hash value # is created by *typecasting* the value of each parameter:

$$\bar{\bar{\lambda}} = CByte(\lambda, \mathcal{B}_{\mathcal{H}}) \tag{4.12}$$

to the specified number of bits per hash $\mathcal{B}_{\mathcal{H}}$ where the double over-line annotation $\bar{\bar{}}$ is used for the bit representations. These *typecast* values are then coalesced & into a single hash:

$$\bar{\bar{\#}} = [\ \bar{\bar{\lambda}}\ \&\ \bar{\bar{\omega}}\ \&\ \bar{\bar{\Omega}}\ \&\ \bar{\bar{\theta}}\ ]. \tag{4.13}$$

Finally this hash value is *typecast* back and the numerical value utilised as the hash-matrix index in the storage system.

4.4 Resizing effects on hash generation

4.4.1 Problem definition

As indicated by Moolman et.al. in [3], the native implementation of this technique proved to be effective under certain test conditions. In these test cases, input video files for both fingerprinting and detection instances, were resized to a fixed resolution and analysed using the parameters as tabulated in Table 4.1.

Further analysis indicated that the scale invariant nature of the SIFT algorithm is not fully inherited by the native implementation's hash as the KP distance proved to be negatively impacted by frame resizing.

The size of an arbitrary video frame can affect both the feature-richness and processing time

Table 4.1: Native implementation parameters

Fingerprinting Parameters	
Frame width	320
Frame height	240
R_{max}	100
R_{min}	10
$\mathcal{B}_{\mathcal{H}}$	4
$\mathcal{B}_{\mathcal{R}}$	16

thereof. The higher pixel count in larger frames can result in more unique keypoints, however the finer granularity tends to increase the processing time thereof. The rigid resizing of the legacy implementation worked with fixed minimum and maximum radii that dictate the keypoint pairs. These rigid radii values might not result in the best keypoint pairs when analysing videos with larger resolutions. The logic behind the effects of resizing investigation pertains to the multitude of different resolutions at which television broadcasts are made. In an ideal world, there would be a resolution from which scale invariant fingerprints could be generated. This would allow larger, but especially smaller resolution streams to be detected. In effect - generate a high resolution master fingerprint against which lower resolution fingerprints could be evaluated. In doing so, one would be able to retain fine features while expediting the analysis process by using smaller resolutions for the stream analysis.

Thus the problem to be addressed during this analysis pertains to the effect that resizing has on the reproducibility of hash values when using varying input resolutions. This analysis formed the basis of the SATNAC 2016 article by De Klerk et al. in [50], which is attached in Appendix A Section C.3.

In order to address the scalability problem exhibited by the native approach, the origin of the error had to be determined. Logic dictates that the *KP*-distance Ω should be calculated as a factor of the actual frame dimensions hence, becoming a scalable descriptor. This is a fairly simple solution when the aspect ratio of the frames remain constant.

4.4.2 Aspect ratio

Unfortunately the aspect ratio does not always remain constant, especially when a fixed input video resolution is defined, effectively forcing input videos to be distorted. There are 3 types of aspect ratios used in literature:

- *Pixel aspect ratio (PAR)*: Ratio of the actual pixels' dimensions. Generally square pixels are used with a $PAR = 1:1$. Any distortion in PAR only affects the perceived image by the user, but the SAR remains the same.
- *Storage aspect ratio (SAR)*: This ratio is calculated from the amount of pixels from which the image is comprised, i.e. the horizontal and vertical pixel counts.
- *Display aspect ratio (DAR)*: This aspect ratio is a combination of $DAR = SAR \times PAR$ and

describes how an image will be perceived.

The ambiguous term *aspect ratio* generally refer to the SAR as is the case in this thesis. Another related ambiguous term is *resize*. In general usage it refers to the scaling of an image while retaining the SAR. When the scaling results in a change of SAR, the process is more aptly referred to as size distortion, but to prevent confusion with the optical aberration techniques (e.g. barrel distortion), the term *remap* is used. There are various techniques that may be used to perform this remapping.

4.4.3 Remapping algorithms

In order to ascertain the effect of resizing on hash generation, it is imperative to know how the resizing is implemented. This is due to the *invasive* nature of the interpolation based algorithms. During upscaling, pixels are fabricated based on surrounding values, while during the downscaling operations, pixels are averaged and discarded - hence a lossy operation. These *invasive* operations modify the underlying frame's data, and ultimately affects the keypoints extracted thereof.

There are a multitude of remapping algorithms that can be employed to upscale or downscale an image. Some of these algorithms are contained within EMGU CV as optimised modules as part of the inter enumeration class [51]:

- *Nearest:*
  Nearest-neighbour interpolation is the simplest and fastest algorithm remapping algorithm. It assigns the value of the nearest pixel to the pixel in the remapped frame. The simplicity and fast processing comes at a price - the resulting frame is prone to contain jagged edges [52].
- *Linear:*
  Contrary to what the name suggests, the *linear* algorithm employed by the EMGU CV library is not the standard linear method that only analyses the 2 closest pixels, but rather bilinear interpolation. Bilinear interpolation surveys the 4 closest pixels from which a weighted average is created based on the nearness and brightness of the surrounding pixels.
- *Cubic:*

The cubic resampling technique surveys 4 neighbouring pixels to estimate the resulting pixel values. When compared to linear interpolation methods, it results in a more accurate result but requires more processing time [52] [53].

- *Area:*
  Bicubic interpolation works in a manner similar to that of the cubic algorithm, but it surveys 16 neighbouring pixels to generate a moire-free result. One drawback with bicubic interpolation is the halation effect present at certain color borders.
- *Lanczos4:*
  The Lanczos4 interpolation algorithm results in the best detail preservation and minimal generation of aliasing artifacts for geometric transformations that do not involve strong downsampling [54].

Each of the aforementioned techniques have their strengths and weaknesses. The most obvious weakness is the processing time required. An analysis by Tanbakuchi [55] illustrates the various processing times incurred for the algorithms. Techniques such as the area interpolation might be the better choice for downsampling as it retains a lot of information, but presents a sharpening effect when images are upscaled. The bilinear interpolation technique on the other hand, has a softening effect and is computationally less expensive. This softening effect can help to reduce the effect of noise present in the image during the keypoint extraction phase.

Bilinear interpolation is an image re-sampling technique that is generally used to enlarge images by creating extra pixels with values derived from the surrounding pixels. Contrary to the term *linear* in the name, the technique itself is not linear, but rather quadratic [56]. While the values are linearly sampled in both axis, the interpolation as a whole can be considered quadratic at the sample location. This quadratic nature creates a *smoothing* effect which might soften local valleys and peaks a bit during enlargement extrapolation. This technique can however also be applied to re-sample an image to a smaller version. The aforementioned *smoothing* effect is not present during the down-sampling process, but may actually result in some aliasing effects [57]. Spatial aliasing could impact the keypoint detection of the SIFT algorithm, but this can be negated by applying a Gaussian smoothing filter [21].

The remapping of a frame F with M horizontal and N vertical pixels results in a remapped frame $F'(M' \times N')$. This is done by obtaining the weighted sum of the four nearest neighbours' pixel values as illustrated in Figure 4.4. The values of the pixel X at coordinates $[m_f; n_f]$ is

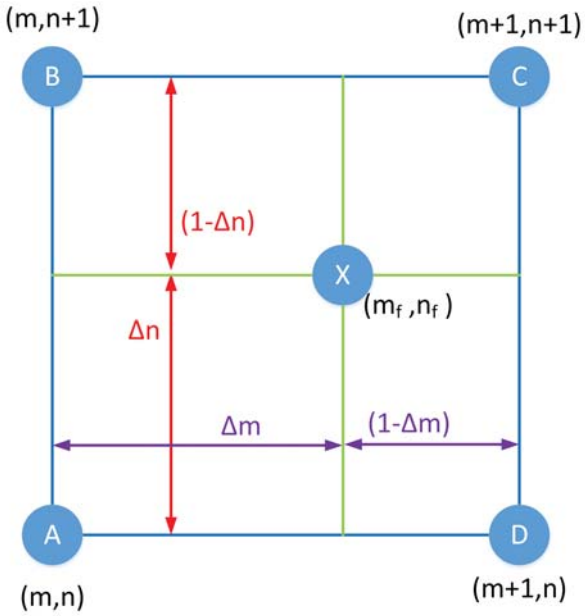


Figure 4.4: Bilinear interpolation addressing schematic

calculated by the parametric equations of the straight line segments running from $A \Rightarrow B$ and $D \Rightarrow C$ [58]:

$$X_{(m_f,n_f)} = aA + bB + cC + dD \tag{4.14}$$

where the weighted values for each neighbour as indicated on Figure 4.4 is [59]:

$$\begin{aligned} a &= (1 - \Delta m)(1 - \Delta n) \\ b &= (1 - \Delta m)\Delta n \\ c &= (\Delta m \Delta n) \\ d &= (1 - \Delta n)\Delta m. \end{aligned}$$

Thus the remapping of an image, can alter the SAR. Although the nearest-neighbour weighted value might have a smoothing effect when remapping to a larger image, this effect is global. This is similar to applying a Gaussian filter to the image. In extreme cases this might reduce the magnitude of the KP. If the remapping results in a change in SAR, not only might the KP size be affected, but the other KP-metrics as well.

By understanding how the frame remapping works, the resulting change in SAR can be quantified and the effects thereof on parameters such as the KP distance Ω analysed.

4.4.4 Modified keypoint distance

Even when the SAR is maintained during the remapping procedure, hence resizing, the KP-distance Ω hash descriptor as expressed in Equation 4.10 is found to be scale-variant. When a frame is resized, the Ω will be different for every resize factor RF where $RF \in \mathbb{R}$ and $RF > 0$. This RF-scalar is used to represent the amount by which the horizontal and vertical pixel count is changed.

The best approach to deal with this limitation is to express the modified KP-distance $\mathring{\Omega}$ as a function of the SAR with the help of a scalar κ producing

$$\mathring{\Omega} = \kappa\Omega \times \mathcal{B}_{\mathcal{R}}. \tag{4.15}$$

A relation of Ω to R_{max} and R_{min} can be retained if these radii is related to the SAR.

4.4.5 Radii parameter relation

Now that the remapping of the frame has been addressed, one has to re-asses the distance metrics employed within the hash generation. One particularly important set of distances are the minimum and maximum radii.

The native analysis called for the user to hardcode the boundary radii R_{min} and R_{max} used for KP-pairing. Since the chosen values proved to be successful for the resolution of 320×240, this property was chosen as a starting reference to determine possible relations.

The logic behind the boundary radii is to improve the hash diversity by forcing KPs to form KP-pairs close to the origin KP. Analysis of the experimental value for $R_{max} = 100$ in the native implementation, shows that R_{max} can be expressed as a factor of the maximum frame distance:

$$\Delta_{max} = \sqrt{M^2 + N^2} \tag{4.16}$$

with

$$R_{max} = \frac{\Delta_{max}}{4} = \frac{\sqrt{320^2 + 240^2}}{4}. \tag{4.17}$$

This relation from Equation 4.17 is graphically represented in Figure 4.5 as an overlay of the radii on the example frame from Figure 4.3.

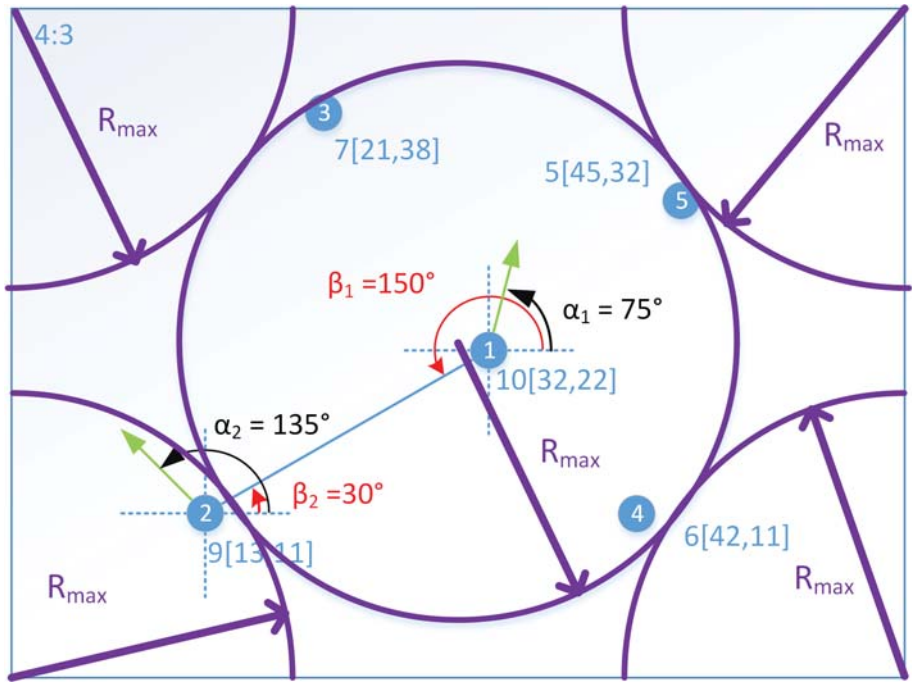


Figure 4.5: Relative R_{max} representation 4 : 3 aspect ratio

When applying the same relation of Equation 4.17 on a frame with a commonly used aspect ratio of 16 : 9, the graphical representation indicates that the radii provides better coverage (91.9%) of the frame as seen in Figure 4.6, without any overlapping. This coverage percentage is greater than the 81.8% from Figure 4.5, but this coverage has no bearing on the detection result as a keypoint can be located at any location and not necessarily the 4 corners and center of the frame as is depicted.

It is interesting to note that for the 16 : 9 aspect ratio, R_{max} can be expressed as a function of the maximum frame distance as well as a function of the frame height. This relation is expressed as:

$$R_{max} = \frac{\Delta_{max}}{4} = \frac{N}{2} \tag{4.18}$$

where N is the frame height in pixels. By applying this height-relation to the 4 : 3 aspect ratio frame, a better radii coverage is observed with minimum overlap as seen in Figure 4.7.

Similar to the native implementation, the relation of R_{min} to R_{max} is set at 10%:

$$R_{min} = \frac{R_{max}}{10}. \tag{4.19}$$

It is important to note that the radii displayed in the various figures merely represent the

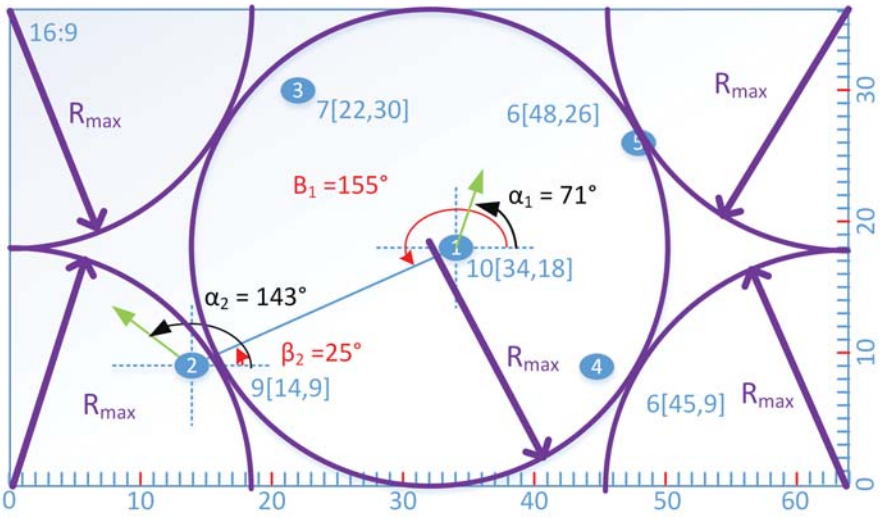


Figure 4.6: Relative R_{max} representation on a 16 : 9 aspect ratio

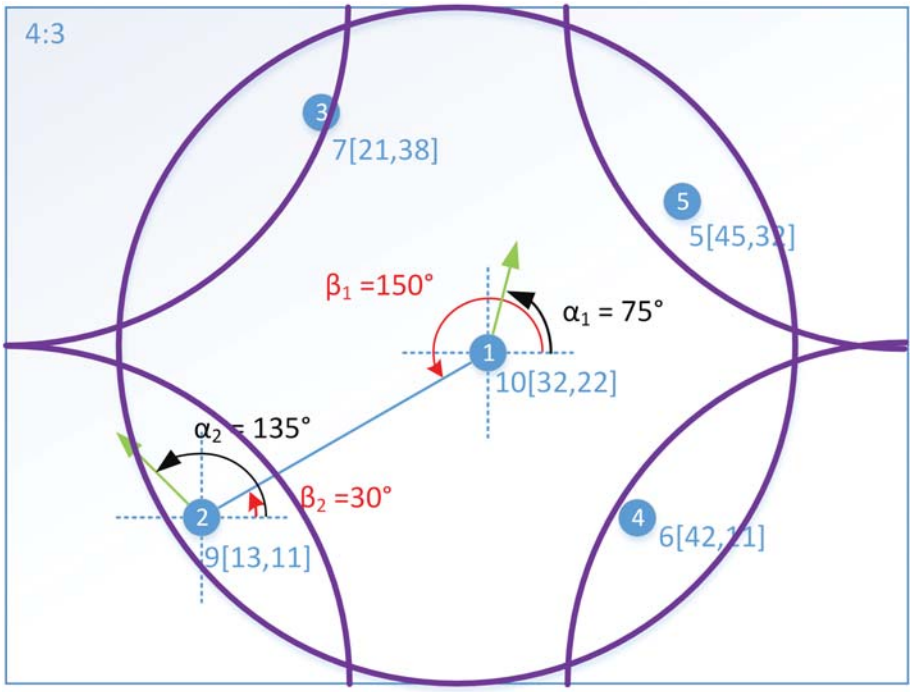


Figure 4.7: Relative R_{max} representation as a function of the frame height on a 4 : 3 aspect ratio

maximum KP-radii if the KP were to be situated on the four corners as well as the center of the frame.

4.4.6 Evaluation instances

In order to isolate the effects that resizing has on keypoint extraction, the native video recognition algorithm's keypoint extraction methodology was applied to a series of images as described later in Section 4.4.7. As with the native implementation, the frames were remapped to a fixed frame size 320×240 in order to serve as a benchmark:

$$\Omega_{Benchmark} = \frac{\Delta - R_{min}}{R_{max} - R_{min}} \times \mathcal{B}_{\mathcal{R}}. \tag{4.20}$$

Subsequent analyses will test the implementation of the two radii parameter relations by calculating the modified KP- distances by using the maximum frame distance:

$$\Omega_{MaxDist} = \frac{\Delta - R_{min}}{\left(\frac{\Delta_{max}}{4}\right) - R_{min}} \times \mathcal{B}_{\mathcal{R}} \tag{4.21}$$

and the frame height as a parameter:

$$\Omega_{Height} = \frac{\Delta - R_{min}}{\left(\frac{N}{2}\right) - R_{min}} \times \mathcal{B}_{\mathcal{R}}. \tag{4.22}$$

4.4.7 Test data

Since the keypoint extraction techniques only require a single frame to function, it allows for analysis using static images. A list of the images that were used during this analysis is presented in Table A-1 in Appendix A [60], [61].

4.5 Results

4.5.1 Resizing analysis 1 results - Resizing effects on hash generation

The resizing effects on hash generation analysis has two components - a theoretical and a practical implementation component. The theoretical component analysed custom test images that were created with fixed points of interest. These images were made with 5 different resolution sets as tabulated in Table 4.2. In order to evaluate the resizing effects on the parameters, the evaluation points were set as one at the origin $P_1[0;0]$ and the other at the

diagonally opposite corner $P_2[Width; Height]$. All the tabulated results that follow is expressed in terms of these resolutions.

The first result set was obtained by analysing the legacy implementation. The results thereof are tabulated in Table 4.3 from which it is clear to see that the only descriptor affected by the resizing is the *KP*-distance as was anticipated.

The *KP*-distance as expressed in Equation 4.21 was evaluated with the results tabulated in Table 4.4. It is evident that Ω is now scale invariant by using the related parameters. The height relation as expressed in Equation 4.22 performed in a similar fashion, with results in Table 4.5.

It is important to note that even though these descriptors are now calculated as a relation to the frame size, the actual value exceeds the allowed size dictated by the $\mathcal{B}_{\mathcal{R}}$.

This was addressed by relating the distance to only the maximum distance

$$\Omega_{Relational} = \frac{\Delta}{\Delta_{max}} \times \mathcal{B}_{\mathcal{R}}. \tag{4.23}$$

This was done since the radii R_{min} and R_{max} are implemented in the algorithm as *KP*-eligibility criterion. The results of this relational *KP*-distance in Equation 4.23 is tabulated in Table 4.6.

From this it is clear that the relational *KP*-distance in Equation 4.23 is best suited as it is scale invariant while the SAR is maintained. To further substantiate this claim, it was implemented on two other points with the results tabulated in Table 4.7. From this it is evident that the *KP*-distance metric stays the same even when resizing has occurred.

Conclusions and recommendations

The origin of the scalability abnormality from the legacy technique was in part attributed to the original calculation of the *KP*-distance as a function of the fixed radii parameters. Although

Table 4.2: Frame Set Resolutions

Frame Set	1	2	3	4	5
Width	320	640	1280	2560	5120
Height	240	480	960	1920	3840

Table 4.3: Legacy implementation results

R_{max}	100	100	100	100	100
R_{min}	10	10	10	10	10
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	69	140	283	567	1136
θ	6	6	6	6	6

Table 4.4: R_{max} as a function of Δ_{max} implementation results

R_{max}	100	200	400	800	1600
R_{min}	10	20	40	80	160
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	69	69	69	69	69
θ	6	6	6	6	6

Table 4.5: R_{max} as a function of the frame height implementation results

R_{max}	120	240	480	960	1920
R_{min}	12	24	48	96	192
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	57	57	57	57	57
θ	6	6	6	6	6

Table 4.6: R_{max} as a function of only Δ_{max} implementation results

R_{max}	80	160	320	640	1280
R_{min}	8	16	32	64	128
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	16	16	16	16	16
θ	6	6	6	6	6

Table 4.7: R_{max} as a function of only Δ_{max} implementation results for points P_1 and P_2

P_1	[50;60]	[100;120]	[200;240]	[400;480]	[800;960]
P_2	[200;11]	[400;22]	[800;44]	[1600;88]	[3200;176]
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	6	6	6	6	6
θ	1	1	1	1	1

these techniques worked well for the fixed frame size of 320×240, it did not fare as well with other frame sizes.

A relation was derived where the radii can be expressed as a function of the frame dimensions. This relation allows the radii to become scale invariant, but the incorporation thereof in the native implementation calculation as expressed in Equation 4.10, does not allow for scalability.

Since the core function of the radii is to serve as KP eligibility criterion, they were omitted and a different relation was used as expressed in Equation 4.23.

Although the hash descriptors are now scale invariant, they are however not distortion invariant. Remapping of the frames to a different SAR will incur errors. A further investigation is warranted to better relate the parameters with the aim to become distortion invariant.

Once the descriptors can be expressed as distortion invariant relations, the aforementioned radii expressions can be evaluated to determine if the slight overlap caused by the height relation performs better or worse than the maximum distance relation.

4.5.2 Resizing analysis 2 results - Resizing effects on hash generation in practical application

Although the theoretical approach produced positive results, the same was not true when the techniques were applied to normal video frames. The test images that were used created an optimal environment.

The theoretical approach validated that the hash composition could be effective when provided with consistent keyframes. Further analysis indicated that the scale invariant feature transform (SIFT) keypoint extraction method might be scale invariant when the keypoints are evaluated

as singular entities, but tend to be affected by scaling when used in hash pairs.

The components of the extracted keypoints are affected differently. The relative locations of the keypoints are constant with only minor variations in the X and Y-coordinates as they are expressed as a decimal value rather than an integer value. The keypoint sizes can also be affected, but it was found that most prominent keypoints tend to remain the most prominent. The component that is affected the most is the keypoint direction. The variations brought forth by the addition of interpolated pixels, or the removal of some pixels has a prominent effect on the extracted value.

This sensitivity of the extracted keypoints components to the remapping algorithm prompted the re-evaluation of the aforementioned hash components. The main goal was to allow the hash components to be scale invariant, hence not only must the hash components be scale invariant, but also only contain the keypoint components that are primarily unaffected by the remapping.

By expressing the hash components as ratios of one another or by normalizing the components relative to the frame it self, the components should be more scale invariant.

Relative Magnitude (λ)

The remapping procedure during e.g. an upscale operation will add interpolated pixels to the frame. Since this operation is uniform over the frame, the effects also tend to be uniform. The addition of pixels might soften a keypoint and hence reduce the size component thereof. But this effect is observed over all the keypoints. Hence by expressing the keypoints as a ratio as was done in Equation 4.3, the same relative magnitude ratio should be observed as before remapping.

Relative Direction (ω)

The relative direction ω as was expressed in Equation 4.6 is no longer a viable hash component as the individual keypoint directions are too sensitive to remapping. Merely removing this metric from the hash generation would result in a substantial loss of 25% hash diversity. Hence a new metric is required to fill this void.

One method to obtain an extra hash component is to use the angle of the vector formed when connecting the parent keypoint and the origin at [0,0]. Using the origin as the reference

point is the logical choice as it simplifies calculations. This premise does however have a few drawbacks. It will be extremely sensitive to noise as a slight variance in the keypoint position will cause a change in the angle since the angle range is only 90°.

This inspired the notion to use the center point of the frame as an anchor point against which all parent keypoints will be evaluated. This allows for an angle range of 360° to be utilised. Thus the relative direction to the anchor $\mathring{\omega}$ can be expressed as:

$$\mathring{\omega} = \frac{atan2\left(x_n - x_A, y_n - y_A\right) + \pi}{2\pi} \times \mathcal{B}_{\mathcal{R}} \tag{4.24}$$

where (x_A, y_A) are the coordinates of the anchor point. Equation 4.24 can be normalised in the horizontal and vertical directions by using the associated values of the frame dimensions to produce:

$$\mathring{\omega}_{norm} = \frac{atan2\left(x_{n\_norm} - 0.5, y_{n\_norm} - 0.5\right) + \pi}{2\pi} \times \mathcal{B}_{\mathcal{R}}. \tag{4.25}$$

The addition of π is to shift the range of the $atan2$ function from $[-\pi, \pi]$ to $[0, 2\pi]$ which is in turn normalised by the denominator.

Keypoint distance (Ω)

The distance between two keypoints as expressed in Equations 4.7, 4.8 and 4.9 is still valid. It is tempting to apply the same logic as previously implemented - to normalise the coordinate component with the hope of making the metric scale invariant. This approach will however cause problems with the bit resolution calculations as the distance can be quite small or large as a standalone metric.

This can be overcome by using the hash generation constraints R_{min} and R_{max} to normalize the distance. This holds true if the aforementioned hash constraints are expressed as functions of the image.

The keypoint distance Ω can then still be used as natively expressed in Equation 4.10 with the hash constraints expressed as either maximum frame distance or frame height in Equations 4.21 and 4.22 respectively.

Keypoint direction (θ)

Since the direction component of the extracted keypoint is no longer being used, the keypoint distance equation had to be modified to remove the angular difference α component. This resulted in Equation 4.26:

$$\theta = \frac{atan2\left(x_{diff}, y_{diff}\right) + \pi}{2\pi} \times \mathcal{B}_{\mathcal{R}}. \tag{4.26}$$

which only evaluates the direction component of the vector linking the child keypoint to the parent.

Results

In order to evaluate the efficiency of the modified hash generation method, it was evaluated against the legacy implementation. This entailed evaluating the remapping algorithms mentioned in Section 4.4.3. A collection of high-resolution images as tabulated in Table A-2 were remapped using various resize factors RF before being subjected to the keypoint extraction technique.

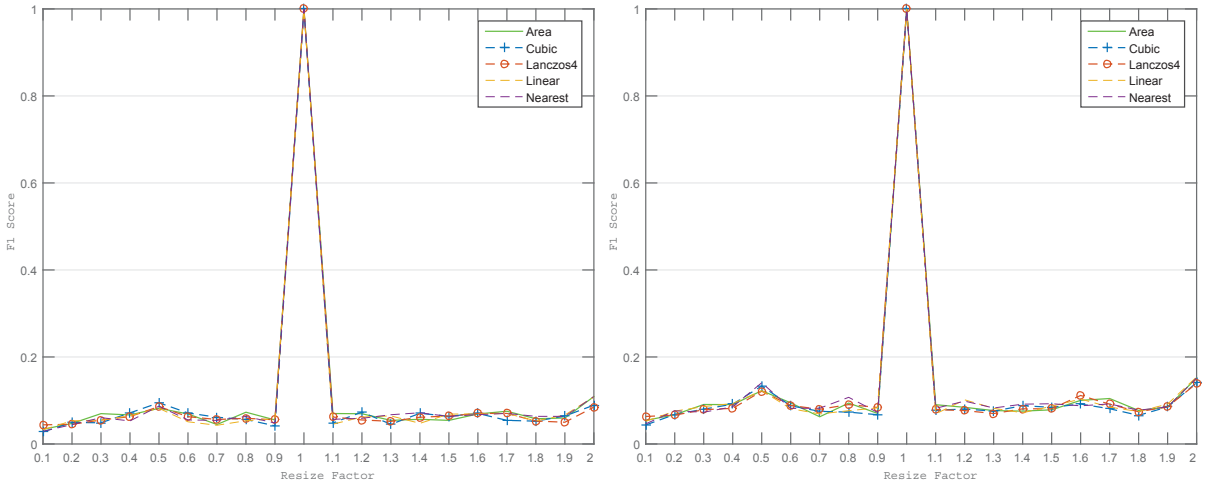
The keypoints as extracted from these remapped images were supplied to the legacy and modified hash generation algorithms. These generated hash values are then compared while using the $RF = 1$ hashes as the groundtruth values.

While the same recall and precision metrics were calculated as described in Section 3.3.1, they are represented as a compounded metric called the $F1$ score in order to reduce clutter on the graphs. The $F1$ score represents the harmonic mean of the recall and precision rates as a single metric [62] and is expressed as:

$$F1 = \frac{2 \times Recall \times Precision}{Recall + Precision}. \tag{4.27}$$

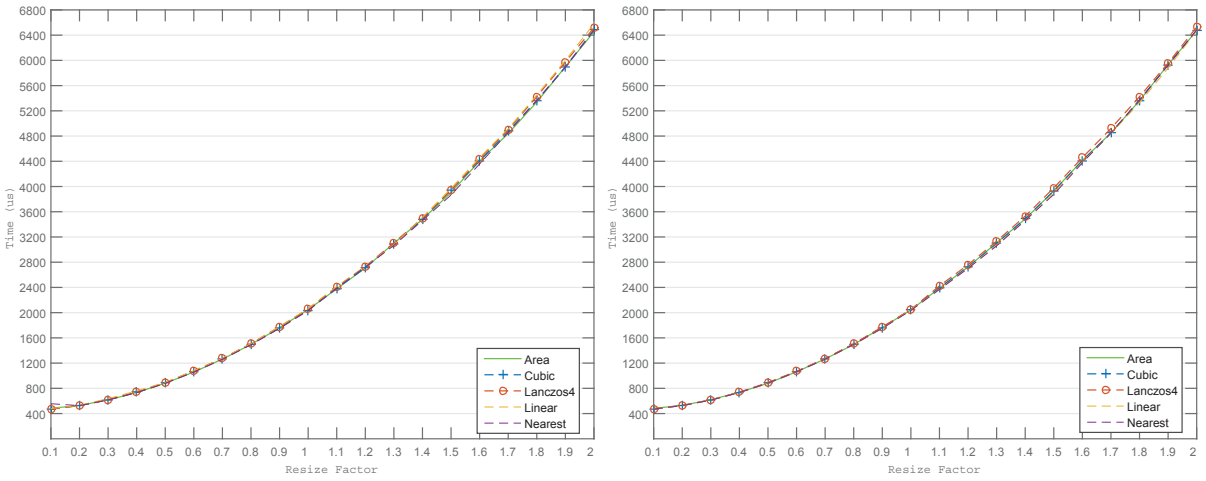
The results for the legacy implementation is illustrated in Figures 4.8a, 4.8b, 4.9a and 4.9b. From these images it is apparent that there is almost no noticeable difference between the various remapping techniques. The same behaviour was observed for the modified hashing algorithm. In order to simplify the the comparison between the legacy and the modified hashing algorithms, an average was calculated for each RF value utilising the various remapping algorithm results.

Figure 4.10b shows that the execution time is the same for both hash generation techniques, it



(a) Frame height (b) Maximum frame distance

Figure 4.8: F1 score comparison for legacy hash generation using multiple remapping techniques



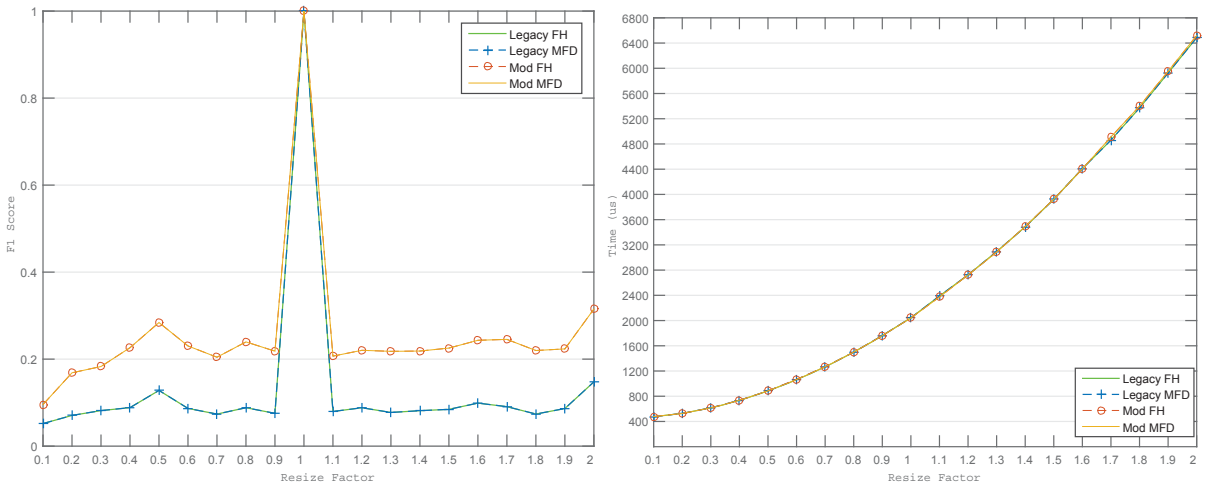
(a) Frame height (b) Maximum frame distance

Figure 4.9: Time comparison for legacy hash generation using multiple remapping techniques

is clear from Figure 4.10a that the modified hashing algorithm responds better to resizing than the legacy hashing method.

Conclusions and recommendations

Figure 4.10a clearly illustrates that the modified hash generation approach produced better F1 scores than the legacy approach. At first glance it is also apparent that the F1 scores are



(a) F1 score comparison

(b) Time comparison

Figure 4.10: F1 score and time comparison for legacy hash and modified hash generation using multiple remapping techniques

drastically lower than the score for $RF = 1$. This can be attributed to the fact that the hashes for $RF = 1$ was used to groundtruth the analysis. Since the F1 score is comprised of both the recall and precision metrics, it does not only evaluate the number of correct detections, but also penalises due to missed or wrong detections.

Hence an accurate interpretation of this F1 score indicates that while not all 50 hashes from the groundtruth set have been detected, on average 20% of them have been matched when using the modified hash generation approach.

4.6 Concluding remarks

The various remapping algorithms made available by the EMGU CV library were found to perform equally well under the test conditions by producing fairly similar results while there was no noticeable time difference observed. The initial assessment to use the bilinear interpolation method as the remapping algorithm has been validated.

While the SIFT keypoint extraction algorithm can be used to extract scale invariant keypoint data for certain applications, certain components thereof seem to be adversely affected by the remapping algorithms used to alter the frame sizes.

The theoretical hash analysis indicated that the initial modified approach could work assuming

the consistent keypoint data. Since the theoretical and practical results did not correlate, the modified hash generation approach was altered to make it more invariant to these keypoint components.

This is seen in Figure 4.10a which clearly illustrates that the modified hash generation approach produced approximately 10% better F1 scores than the legacy approach. This was achieved by implementing the following hash descriptors:

- **Relative Magnitude (λ)** - Equation 4.3;
- **Relative Direction (ω)** - Equation 4.25;
- **Keypoint distance (Ω)** - Equation 4.10 with MFD distance technique in Equation 4.21.
- **Keypoint direction (θ)** - Equation 4.26.

The analysis of the modified hash generation algorithm also indicated that the maximum frame distance MFD technique for radii relation produced slightly better results as can be seen when comparing Figure 4.8a and 4.8b for values other than $RF = 1$.

When referring to the legacy implementation and the original rigid resizing of 320×240, it is worth knowing that the original videos and streams utilised by Moolman et al. was by default sized as 320×240. Hence the legacy implementation operated at a resize factor $RF = 1$.

To conclude:

- The analyses proved that the idea postulated in Section 4.4.1, where a high resolution video could be used for the database fingerprint while streams are analysed at smaller resolutions, is not a viable avenue.
- This implies that the reference video should be the same size as the video stream being analysed. The changes to the hash descriptors proved to result in approximately 10% better $F1$ scores. This should help with minute size differences, possibly brought forth by rounding of pixel bins during the remapping algorithms.
- The radii parameter relation enables the hash algorithms to function on video resolutions larger than 320×240 while maintaining hash diversity.

Chapter 5

Video detection

"I often say that when you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meagre and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science, whatever the matter may be" — William Thomson

The hash matching procedure outlined in Section 2.5.1 describes the manner in which video fingerprint hashes are stored and retrieved. As was the case with the Jensen-Shannon divergence, the individual metrics, or in this instance, the possible video identifications, can be better utilised if it is contextualized.

Since the database already contains all the meta data pertaining to each video such as the vfIDs and their respective positions within the video, this data can be utilised to collate the extracted vfIDs from the hash matrix.

The interval between the extracted vfIDs can be normalised using the stream's frame rate and compared to the normalised interval between fingerprinted frames as recorded in the database. This dual-phase authentication procedure helps to eliminate false detections and the flow thereof is illustrated in Figure 5.1. In order to account for some variance with regards to the interval normalisation procedures, the comparison is done with a variance of 5 frames.

This variance might seem like a large value, but only serves as a secondary authentication since a video could be detected by a single frame with enough recognisable hashes.

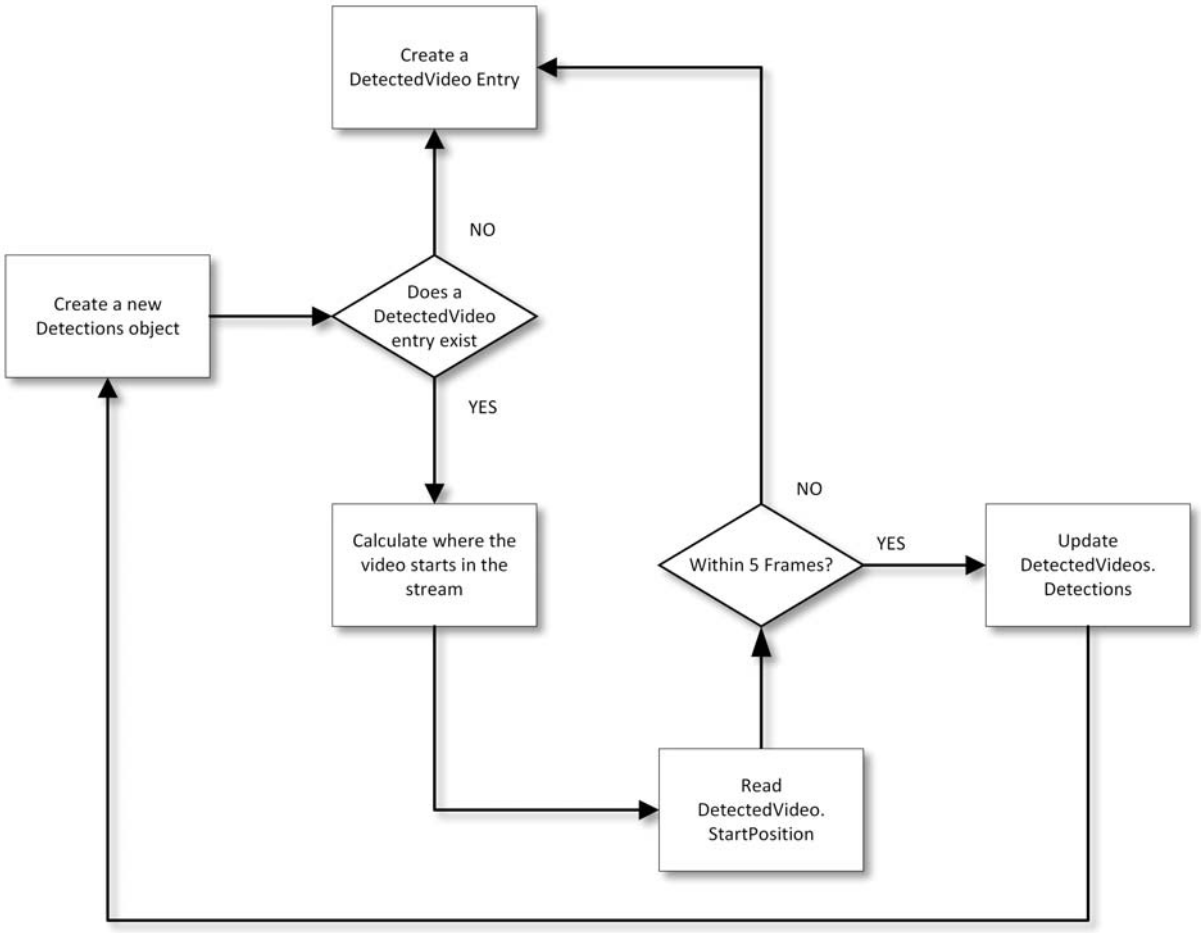


Figure 5.1: Detection logic flow

5.1 Legacy - implementation

In order to establish a baseline for the analysis comparison, the legacy implementation of such a video detection application, as presented by Moolman et.al. [3], was used.

5.1.1 Legacy - SBD

The legacy implementation employed the Jensen-Shannon Divergence as a shot boundary detection algorithm with the parameters as tabulated in Table 5.1.

Table 5.1: Legacy JSD Parameters

Auto adjusting threshold frame count (ω)	3
Average threshold sensitivity (α)	1
Minimum threshold sensitivity (η)	0.012

5.1.2 Legacy - Hash generation

Although the legacy implementation utilises the same hash table concept, the detection classification criterion is slightly different than the illustration in Figure 5.1. The legacy implementation calls only for two consecutive hash-matched frames as illustrated in Figure 5.2.

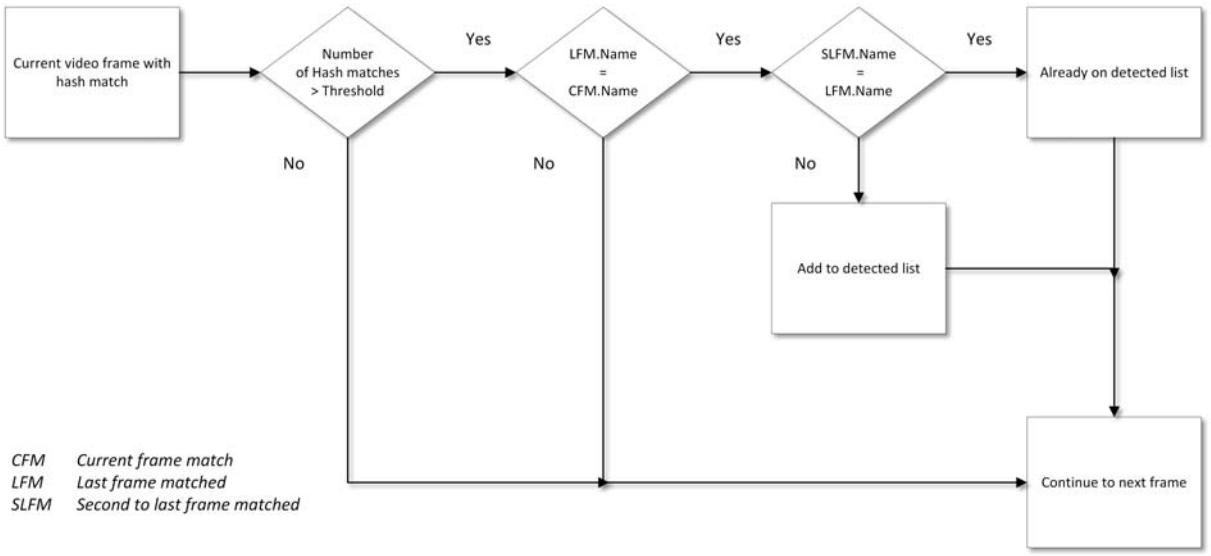


Figure 5.2: Legacy detection logic flow

One critical drawback of this approach lies in the implementation of the *detected list*. If a video has been detected and is noted on the list, any secondary detection of the same video will not be noted as a new detection until the detected video list has been cleared.

5.2 Optimised - implementation

The culmination of the various optimisation approaches are referred to as the optimised implementation.

5.2.1 Optimised - SBD

The Jensen-Shannon divergence was selected as the SBD technique to process the video streams. The JSD technique that has been implemented, was guided by the results from Chapter 3 with the values tabulated in Table 5.2.

Table 5.2: Optimised JSD Parameters

Auto adjusting threshold frame count (ω)	5
Average threshold sensitivity (α)	5.7
Minimum threshold sensitivity (η)	0.0

5.2.2 Optimised - Hash generation

This implementation employs the modified hash generation implementation as set forth in Section 4.5.2.

5.2.3 Detection criterion

As opposed to the hash confirmation procedure for the legacy implementation outlined in Figure 5.2, the optimised approach has further constraints on the retrieved hashes.

The optimised implementation evaluates if the corresponding start locations are within 5 frames from one another as illustrated in Figure 5.1 before it is added to the *hit-list*. In order for a positive identification of a video, an additional constraint was imposed where at least 2 *hit-list* entries should correlate to confirm a detection. This *hit-list* was represented by a circular buffer with a length of 5. This was done to ensure that sporadic false detections will not influence the detections.

5.3 Video detection comparison

In order to ascertain the effectiveness of the optimisation of the various components of the video detection process, this optimised technique was evaluated alongside the legacy implementation using the same dataset.

A third configuration has been included in the comparison. A derivation of the optimised approach was included that contains the modified hash generation, but reduced parameters

for the JSD technique.

5.3.1 Stream analysis data

The dataset utilised for the stream analysis was comprised of all the 1280×720 resolution advertisements from the South Africa specific advertisement set, as well as global advertisements as tabulated in Table A-3 and Table A-5 respectively. As was the case with the other data sets used for analysis, these videos were integrated into a continuous stream, and 4 test streams were generated with various noise levels using the noise feature from Adobe Premiere Pro [63]. The database was however populated with all the videos from the aforementioned tables to introduce some variance into the system.

5.4 Video detection comparison results

5.4.1 Recall

The recall rates of the stream analysis process provides the ratio of number correct video detections to the number of relevant videos contained within the analysis stream similar as to what was expressed in Equation 3.18. The recall rates as illustrated in Figure 5.3 clearly indicate that both the modified implementation techniques resulted in better recall rates up to the 10% noise mark.

5.4.2 Precision

The precision rates describe the ratio of relevant video detections to the number of irrelevant video detections similarly as to what was expressed in Equation 3.19.

The precision rates for the legacy implementation is slightly higher than the optimised implementations for 0% and 15% noise. However for 5% and 10% noise, the optimised implementations produced better results as illustrated in Figure 5.4.

5.4.3 Execution time

One of the most critical aspects of this investigation pertains to the *real-time* criterion as set forth in Section 1.2.1. The execution time of the various algorithms were recorded and expressed as a factor of the actual duration of the stream being analysed. With the exception of the optimised

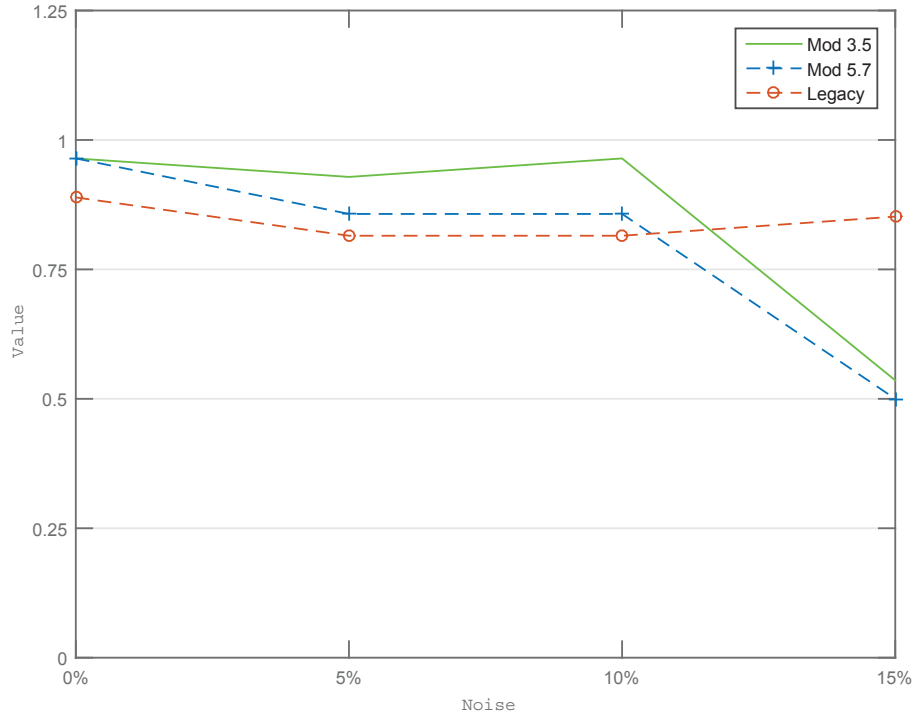


Figure 5.3: Recall comparison of the legacy and modified implementation

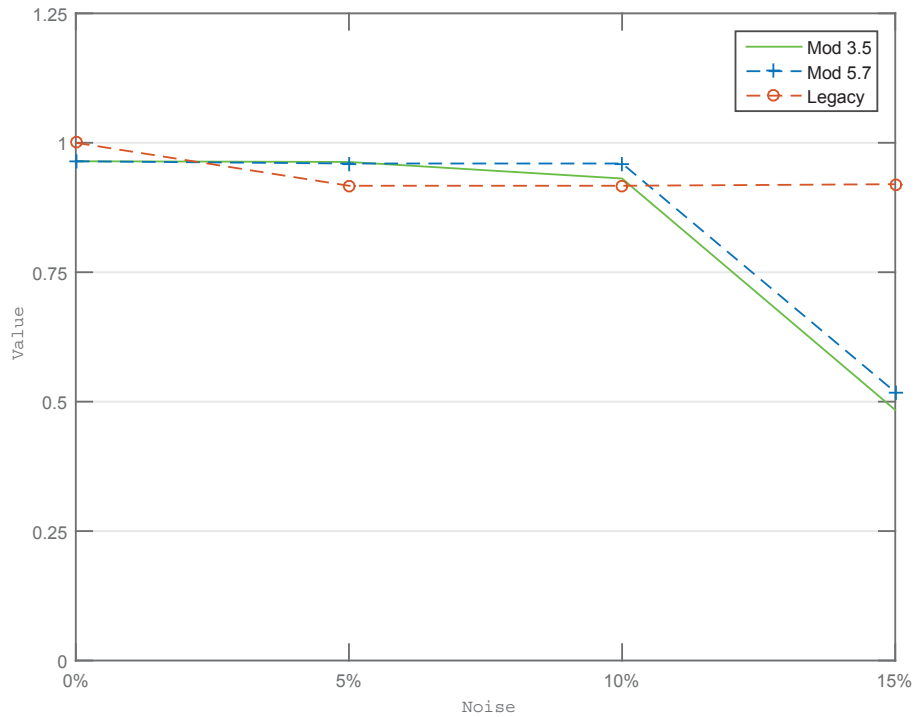


Figure 5.4: Precision comparison of the legacy and modified implementation

implementation *mod 3.5*, all the analyses adhered to the *real-time* criterion.

It is clear from Figure 5.5 that the optimised implementation *mod* 5.7 had the fastest execution time of all the analyses. The noticeable difference between the the two optimised implementation analyses will be discussed later in Section 5.6.

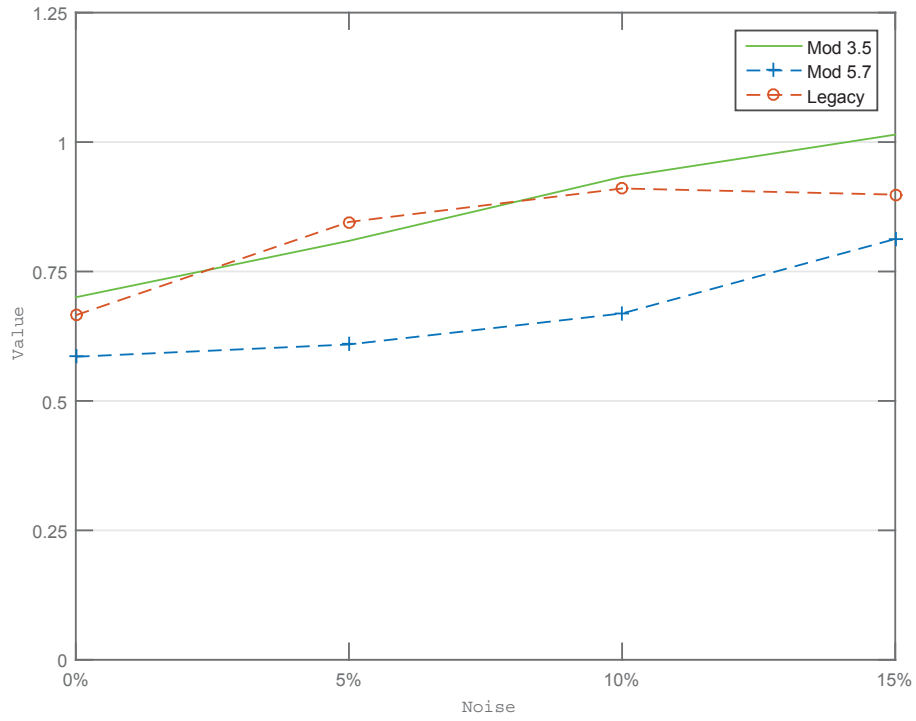


Figure 5.5: Execution time comparison of the legacy and modified implementation

5.5 Slow-changing video use case

During the analysis there were a few particular videos identified that proved to be extremely difficult to detect. A manual investigation revealed that the videos contained very long gradual transitions. This was exacerbated by the monotonous nature of the video. The worst of them all was a particular advertisement featuring a solid color background with slowly changing animations that cover only a small portion of the frame.

The long gradual transitions could however be detected using the JSD algorithm. While the *mod*5.7 implementation faired slightly worse than the legacy implementation, the *mod*3.5 did however surpass them both with the results tabulated in Table 5.3.

Table 5.3: Detected boundary count for the slow-changing video use case

VIDEO NR.	LEGACY	MOD 5.7	MOD 3.5	ACTUAL BOUNDARIES
1	27	18	34	14
2	14	8	18	5
3	3	4	10	6

From Table 5.3 it is clear that the *mod*3.5 implementation was able to detect more frames which can be fingerprinted for better detection.

5.6 Concluding remarks

The results of the various stream analyses confirm that the optimised video detection approach has various advantages and improvements over the legacy implementation. Although the precision metric indicated that the legacy implementation produced slightly better precision rates for the 0% noise analysis, it is totally acceptable. The precision rate is slightly better, since the recall rate is lower as illustrated in Figure 5.4 - hence it is slightly more precise, but it has less frames to be precise on. This will be a problem with videos containing slowly changing scenes or graphics as there will be less frames recalled.

In the light of the analyses performed, the recall metric is actually more important than the precision metric as the precision can be improved with constraints within the detection criterion. Whereas good precision rates coupled with low recall rates would be indicative of missed detections.

The attempt to increase the recall rates have been included in the aforementioned analyses as *mod 3.5*. Although this analysis utilised the same optimised hash generation technique and the same JSD algorithm, the average threshold value of the JSD algorithm has been lowered from 5.7 to 3.5. This clearly resulted in increased recall rates as indicated in Figure 5.3. Unfortunately one drawback associated to this increased recall rates, is the increased detection time since more possible shot boundaries are identified and fingerprinted. The increased recall rates proved to be useful in addressing the slow-changing video problems as discussed in Section 5.5.

As for the precision rates, both the optimised implementations were analyses using the same detection criterion. This is why there is a slight difference beteeen *mod*3.5 and *mod*5.7.

Essentially when lowering the average threshold for the JSD algorithm, it is a good idea to refine the detection classification criterion to weed out the extra recalled possibilities.

Chapter 6

Conclusions and Recommendations

"The end is never the end. It's always the beginning of something" — Kate Lord Brown

6.1 Summary of research

The primary aim of the research presented in this thesis was to optimise the video detection process for use with streaming digital media. This was done by not only optimising the individual components of video detection, but by providing an understanding of the optimised integration of the various components to meet the *real-time* processing time constraints.

A summary of the research presented in this thesis is illustrated in Figure 6.1. From the literature it becomes apparent that the process of advertisement detection is not an isolated component, but rather a system of components. Figure 6.1 illustrates the logical flow through the various components of this system. It became apparent from an early stage that the many of these components have been tailored for their specific applications, making their application to a universal algorithm cumbersome.

The main drawback pertaining to the shot boundary detection as well as the integrated video detection process lies within the temporal domain. Apart from the *real-time* criterion that must

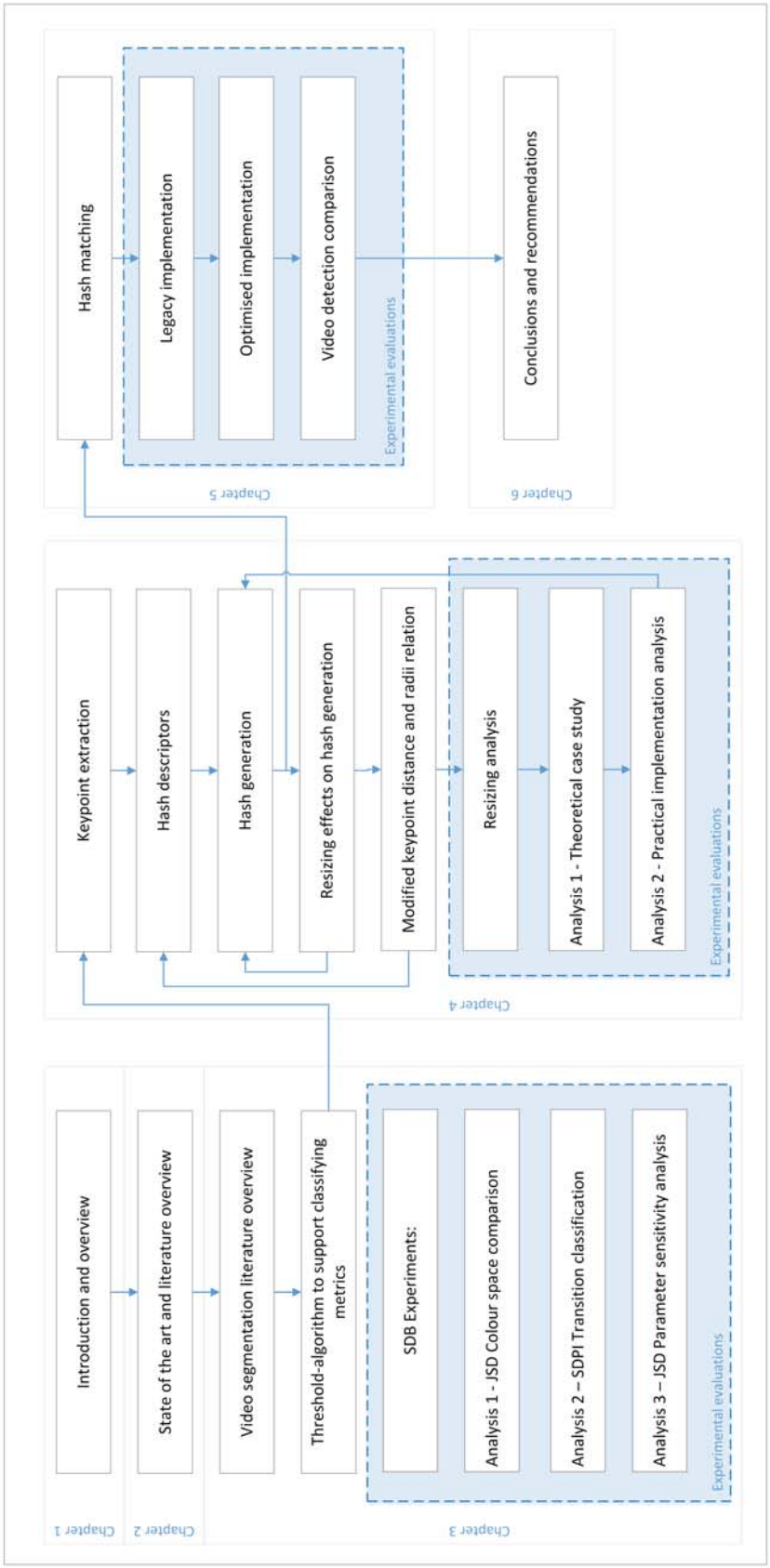


Figure 6.1: Flowchart summarising the research presented in this thesis

be satisfied by all the components, and the integration thereof, the biggest impact was caused by the instantaneous nature of stream analysis which is devoid of *future* frames. The *real-time* aspects of the main SBD techniques were evaluated and a trade-off made to sacrifice some accuracy for the sake of the processing simplicity of the JSD SBD algorithm.

6.2 Video segmentation

The contradictory notion of sacrificing processing cycles with the aim of saving processing cycles proved to be a fruitful venture. The implementation of a video segmentation algorithm required some processing power, but in effect reduced the total amount of data to process with expensive algorithms, hence saving valuable processing time as well as reducing data access and storage.

The video segmentation stage has a substantial impact on the subsequent stages of the video detection process. Due to this sequential impact factor, various aspects of the segmentation process was evaluated. The first component that was investigated correlates with the first visually obvious component of a video - the colour composition thereof. Although the combination of the available colour channels can be used for a better classification of the shot boundaries, it does not justify the processing overhead increasing threefold.

Since the choice has been made to utilise a single monochromatic video channel for all subsequent analyses, the two most promising algorithms were evaluated to determine their efficiency in terms of accurate shot boundary detection all the while taking processing time into account. As expected from the literature, the SDPI SBD technique proved to be more sensitive to gradual transitions, hence able to detect shot boundaries more accurately than the JSD technique.

The JSD technique had no problems with abrupt transitions, but performed slightly worse for gradual transitions. It was however observed that these transitions could still be detected by lowering the average threshold against which the JSD metric is evaluated. In doing so, the recall rates increased, but the precision rates declined. This decline in precision for the sake of increased recall is discussed later in Section 6.4.2. Hence the choice was made to employ the JSD algorithm rather than the SDPI or the JSDSDPI hybrid algorithm since the JSD algorithm with a low threshold could also detect gradual transitions.

With the choice made to employ the JSD algorithm for SBD, a better understanding was required regarding the all parameters pertaining to the JSD algorithm and the accompanying threshold mechanism. This was accomplished in the form of a parameter sensitivity analysis which isolated and investigated the effect of all relevant parameters in Section 3.3.4.

The optimal configuration of parameters were generated for the supplied datasets. Unfortunately this optimal point tends to drift as the datasets change, hence making the calculated values, the *optimal entry values* for further optimisations.

6.3 Video fingerprinting

Now that the stream segmentation process has been optimised, the focus shifted to what must be done with the isolated frames. A literature survey indicated that there are two main schools of thought when it comes to extracting keypoints from an image - SIFT and SURF. It was however decided to employ the SIFT algorithm for keypoint extraction as discussed in Section 2.3.

Once the frame has been processed and keypoints extracted, these keypoints were grouped using a unique two-tier structure to improve keypoint-pair diversity. These keypoint pairs could hence forth be used to derive various hash descriptors which would later become the hash values which form the frame's fingerprint.

The hash generation of the legacy implementation served as a base from which the hash generation could be evaluated and improved. Many analyses were done and a theoretical derivation was done that would provide better results when the size of the frame would be altered. It was however noted that although the hash characteristics would support resizing, the underlying data supplied from the keypoints did not accommodate this change as investigated in Section 4.4.

The nature of the SIFT algorithm is to be scale invariant, however the manner in which the keypoint components were grouped and incorporated into the hash descriptors proved to be sensitive to the pixel variations induced by the remapping algorithms. This prompted the re-evaluation of the hash components to reduce the effect of resizing on the resulting hash. Although the hash is still affected by resizing, it is clear from Figure 4.10a that the resulting algorithm showed a clear improvement above the theoretical derivation.

6.4 Video detection

6.4.1 Legacy implementation

A previous investigation by Moolman et al. indicated their proposed video detection algorithm was capable of satisfying some of the criteria shared by the research addressed by this thesis. This allowed the legacy implementation to be analysed and to serve as a baseline for the optimised video detection algorithm. The optimised implementation contains multiple modifications and upgrades to address the shortcomings of the legacy implementation.

6.4.2 Optimised implementation

The optimised video detection algorithm entails the integration of the various sub-modules which have been optimised in the preceding chapters of this document. There is however reference to two versions of the optimised video detection algorithm: *mod 5.7* and *mod 3.5*.

This subdivision of the optimised implementation is due to the variable nature of the videos being analysed. The optimised values for the JSD SBD algorithm was derived based on the data set utilised. The JSD optimisation resulted in an optimal trade-off between recall and precision. This optimal state was achieved using an average threshold value of 5.7.

Unfortunately when integrated with the other video detection components, the optimal JSD parameters can actually have a negative impact on the detection process. The aforementioned average threshold parameter is in effect used to determine if a frame constitutes a shot boundary. This threshold value is indirectly related to the likelihood of a frame being classified as shot boundary.

Based on Figures 3.12a and 3.12b, the decision was made to evaluate a lower average threshold value of 3.5. Although 3.5 results in slightly less precision, it does present with better recall rates. This decision would cause an increase in processing duration since there would be more frames selected for fingerprinting. From a video segmentation aspect, this would reduce the precision of the detected boundaries, but increase the recall thereof.

This is advantageous for the video detection process as it is more important to analyse possible frames for fingerprints along side a few false detections, rather than reducing

the false detections and risk missing critical frames. The effects of these false detections are easily mitigated by employing stricter detection classification criterion as discussed in Section 5.2.3 and 5.6. The advantage of employing the lower threshold *mod*3.5 implementation became apparent in the particular slow-changing video use case in Section 5.5. The *mod*3.5 implementation was able to identify multiple additional shot boundaries which was missed by the legacy implementation. This in turn greatly improved the capability of detecting these videos.

6.4.3 Video detection comparison

Finally both implementations of the optimised video detection algorithm were analysed alongside the legacy implementation. These analyses were evaluated in terms of recall and precision rates of the detected videos while keeping track of the various execution times.

The recall rates for both the optimised detection algorithm surpassed that of the legacy implementation up to noise levels levels of 10%. As for the precision values, both the optimised implementations had practically the same precision values, reaffirming the notion that the precision is affected by the detection classification criterion if there are adequate recalled frames. The precision values for the legacy implementation as well as the optimised implementations correlated fairly closely, only diverging for noise levels greater than 10%. The noise levels affect the keypoint extraction process, which in turn has a noticeable effect on the hashes. With the advent of satellite television, such as DSTV, the chances of encountering noise is fairly small. It was however included in the analyses for completeness should a different broadcasting medium be used.

The most important metric of this comparison is the execution time of the algorithms. The optimised implementation *mod*5.7 showed an overall improvement in execution time across all the noise levels, and is noticeably faster than the legacy implementation as illustrated in Figure 5.5. From this figure it is clear that the *real-time* criterion is satisfied for the *mod*5.7 implementation.

The effect of lowering the average threshold of the JSD SBD algorithm for the *mod*3.5 implementation is clearly visible on the aforementioned figure as an almost linear increase in execution time. With the exception of noise levels greater than approximately 14%, the *mod*3.5 implementation also satisfied the *real-time* criterion.

6.5 Research contributions

Two main contributions were made during the course of this thesis, focussing on the synergy of the video detection components. These contributions were made at separate stages during the research presented in this thesis, but in essence are the result of each other. The culmination of these research contributions paved the way for a video detection system capable of detecting videos in *real-time* with great accuracy. Furthermore, the optimisation of the JSD algorithm enabled the detection of previously difficult-to-detect videos as discussed in Section 5.5 which can also be seen as a minor contribution.

6.5.1 Provide a new insight into the Jensen-Shannon Divergence for shot boundary detection

The name Claude Shannon is commonly encountered in literature pertaining information theory while Johan Jensen is famous for his inequality measure. An integration of these brilliant mathematicians' work provides a unique metric that can be employed to quantify the difference measure between subsequent frames.

There are numerous entropy-based difference metrics available that are in some part based on the principle of the JSD. The research contribution however pertains more to the deeper understanding that was obtained about how this difference metric behaves in the realm of video analysis.

6.5.2 Integration of multiple techniques to create a robust video detection technique

Building on the renewed insight obtained form the JSD technique, it could be integrated into an *eco-system* that strives towards an optimal configuration for video detection. The term *eco-system* was deliberately used as it captures the interaction-based variable nature of the video detection system dependent on the nature of the streams being analysed.

The techniques that are integrated to create this *eco-system* include:

- Jensen-Shannon divergence as shot boundary detection metric:
  - Combining the JSD algorithm with an adaptive-threshold algorithm in order to

adapt to the changing video streams.

- Video fingerprinting:
  - Extract keypoints from within the segmented video frames using the Scale Invariant Feature Transform;
  - Enhance the robustness of these fingerprints by calculating further hash descriptors from these keypoints.
- Implementation of a hash table database structure:
  - The implementation of hash table within a database did not only provide a means of fast data access, but was employed in such a way as to provide additional metadata to aid with detection classification.

The variable nature of the *eco-system* was reaffirmed by the implementation of the optimised video detection algorithm. Although the optimisation of the individual, seemingly independent, components showed promising results, the integration of thereof into a singular system had additional effects which were addressed in this research.

6.6 Future work

The main goal of the research presented in this thesis was to investigate and optimise the video detection algorithm for use with streaming digital media. In order to accomplish this goal, the focus was placed on the various algorithmic components and their interaction.

An area warranting further investigation is the hash generation process. While this research managed to reduce the effect that image scaling has on the generated hashes, it would be beneficial if this process could be made more distortion invariant.

The programming methodology employed throughout the research was done deliberately as to evaluate the various algorithmic components. Since that has been accomplished, a paradigm shift can be made with regards to the programming methodology. Implementation of the various algorithms using a lower level language such as C++ should increase the overall performance. This performance increase might be drastically compounded if the multi-threaded nature of modern CPUs, as well as the parallel processing of GPUs such as the CUDA architecture is employed.

6.7 Closure

Due to the variable nature of video as an input parameter, it was no trivial undertaking to evaluate the possibility of *real-time* video detection for all possible videos. In order to address this unbounded nature of the input variable, representative datasets were used to evaluate the video detection algorithm.

This systematic approach was able to affirm that *real-time* video detection on streaming digital media is possible. Although the optimisations referenced in this thesis might be subject to change dependent on the nature of the input videos, the research provides an understanding of the *eco-system* and a stepping stone towards further optimisation of video detection algorithms.

Bibliography

[1] American marketing association. [Online]. Available: https://www.ama.org/AboutAMA/Pages/Definition-of-Marketing.aspx

[2] SABC October 2016 Rate Card. [Online]. Available: http://web.sabc.co.za/digital/stage/advertising/October_16_TV_RATES_BOOKLET_V2.pdf

[3] R. Moolman and W. Venter, "Video fingerprinting using robust hashing of scale invariant features of frames," in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*. www.satnac.org.za: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2012.

[4] A. Gambier, "Real-time control systems: a tutorial," in *Control Conference, 2004. 5th Asian*, vol. 2, 2004, pp. 1024–1031.

[5] G. E. Moore, "Cramming more components onto integrated circuits, reprinted from electronics, volume 38, number 8, april 19, 1965, pp. 114 ff." *IEEE Solid-State Circuits Newsletter*, vol. 3, no. 20, pp. 33–35, 2006.

[6] N. Myhrvold, "The next fifty years of software," in *ACM97 Conference*, 1997.

[7] D. Reisinger, "Keeping up with moore's law proves difficult for intel," http://www.cnet.com/news/keeping-up-with-moores-law-proves-difficult-for-intel/, Tech. Rep., August 2016.

[8] M. Murgia, "End of moore's law? what's next could be more exciting," http://www.telegraph.co.uk/technology/2016/02/25/end-of-moores-law-whats-next-could-be-more-exciting/, Tech. Rep., March 2016.

[9] B. Chachuat, B. Srinivasan, and D. Bonvin, "Adaptation strategies for real-time optimization," *Computers & Chemical Engineering*, vol. 33, no. 10, pp. 1557–1567, 2009.

[10] A. Bryman, "Integrating quantitative and qualitative research: how is it done?" *Qualitative research*, vol. 6, no. 1, pp. 97–113, 2006.

[11] A. M. Amel, B. A. Abdessalem, and M. Abdellatif, "Video shot boundary detection using motion activity descriptor," *arXiv preprint arXiv:1004.4605*, 2010.

[12] C. Chan and A. Wong, "Shot boundary detection using genetic algorithm optimization," in *Multimedia (ISM), 2011 IEEE International Symposium on*. IEEE, 2011, pp. 327–332.

[13] S.-C. Chen, M.-L. Shyu, W. Liao, and C. Zhang, "Scene change detection by audio and video clues," in *Multimedia and Expo, 2002. ICME'02. Proceedings. 2002 IEEE International Conference on*, vol. 2. IEEE, 2002, pp. 365–368.

[14] E. Pöppel, "A hierarchical model of temporal perception," *Trends in cognitive sciences*, vol. 1, no. 2, pp. 56–61, 1997.

[15] Top 5 video transitions. [Online]. Available: http://content.videoblocks.com/video-basics/top-5-video-transitions-and-the-most-effective-ways-to-use-them/

[16] A. K. Tripathi, U. Ghanekar, and S. Mukhopadhyay, "Switching median filter: advanced boundary discriminative noise detection algorithm," *Image Processing, IET*, vol. 5, no. 7, pp. 598–610, 2011.

[17] U. Gargi, R. Kasturi, and S. H. Strayer, "Performance characterization of video-shot-change detection methods," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 10, no. 1, pp. 1–13, 2000.

[18] Q. Xu, P. Wang, B. Long, M. Sbert, M. Feixas, and R. Scopigno, "Selection and 3d visualization of video key frames," in *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*. IEEE, 2010, pp. 52–59.

[19] M.G. de Klerk and W.C. Venter and A.J. Hoffman, "Digital video shot boundary detector investigation," in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2014*. http://www.satnac.org.za/: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2014, pp. 105–110.

[20] G. Ciocca, "A robust multi-feature cut detection algorithm for video segmentation," *ELCVIA: Electronic Letters on Computer Vision and Image Analysis*, vol. 9, no. 1, pp. 32–46, 2010.

[21] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Proceedings of the 2011 International Conference on Computer Vision*, ser. ICCV '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 2548–2555.

[22] A. Azeem, M. Sharif, J. Shah, and M. Raza, "Hexagonal scale invariant feature transform (h-sift) for facial feature extraction," *Journal of applied research and technology*, vol. 13, no. 3, pp. 402–408, 2015.

[23] S. Shi, *Emgu CV Essentials*. Packt Publishing Ltd, 2013.

[24] P. Panchal, S. Panchal, and S. Shah, "A comparison of sift and surf," *International Journal of Innovative Research in Computer and Communication Engineering*, vol. 1, no. 2, pp. 323–327, 2013.

[25] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Ieee, 1999, pp. 1150–1157 vol.2.

[26] D.G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, nov 2004.

[27] P. E. Black. Hash table - dictionary of algorithms and data structures. [Online]. Available: https://xlinux.nist.gov/dads/HTML/hashtab.html

[28] Velocity films sa. [Online]. Available: http://www.velocityfilms.com/

[29] Adweek. [Online]. Available: http://www.adweek.com/

[30] Ads of the world. [Online]. Available: http://adsoftheworld.com/media/tv

[31] S. Lefèvre, J. Holler, and N. Vincent, "A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval," *Real-Time Imaging*, vol. 9, no. 1, pp. 73–98, 2003.

[32] C. E. Shannon, "A mathematical theory of communication," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, jan 2001. [Online]. Available: http://doi.acm.org/10.1145/584091.584093

[33] M. Feixas, A. Bardera, J. Rigau, Q. Xu, and M. Sbert, *Information Theory Tools for Image Processing*. Morgan and Claypool Publishers, 2014, vol. 6, no. 1.

[34] M. Menéndez, J. Pardo, L. Pardo, and M. Pardo, "The jensen-shannon divergence," *Journal of the Franklin Institute*, vol. 334, no. 2, pp. 307–318, 1997.

[35] T. D. Schneider, "Information theory primer," 1995.

[36] J. L. W. V. Jensen, "Sur les fonctions convexes et les inégalités entre les valeurs moyennes," *Acta Mathematica*, vol. 30, no. 1, pp. 175–193, 1906.

[37] I. Spanou, A. Lazaris, and P. Koutsakis, "Scene change detection-based discrete autoregressive modeling for mpeg-4 video traffic," in *Communications (ICC), 2013 IEEE International Conference on*. IEEE, 2013, pp. 2386–2390.

[38] J. Burbea and C. R. Rao, "On the convexity of some divergence measures based on entropy functions." DTIC Document, Tech. Rep., 1980.

[39] M. Vila, A. Bardera, Q. Xu, M. Feixas, and M. Sbert, "Tsallis entropy-based information measures for shot boundary detection and keyframe selection," *Signal, Image and Video Processing*, vol. 7, no. 3, pp. 507–520, 2013.

[40] Z. Cernekova, C. Nikou, and I. Pitas, "Shot detection in video sequences using entropy based metrics," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 3. IEEE, 2002, pp. III–421.

[41] Open source computer vision library. [Online]. Available: http://opencv.org/about.html

[42] M. Chunmei, D. Changyan, and H. Baogui, "A new method for shot boundary detection," in *Industrial Control and Electronics Engineering (ICICEE), 2012 International Conference on*. IEEE, 2012, pp. 156–160.

[43] M.G. de Klerk and W.C. Venter and A.J. Hoffman, "Automatic shot boundary detection for streaming digital video." http://www.satnac.org.za/proceedingsn.html: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2015, pp. 219–224.

[44] R. W. Lienhart, "Comparison of automatic shot boundary detection algorithms," in *Electronic Imaging'99*. International Society for Optics and Photonics, 1998, pp. 290–301.

[45] V. Raghavan, P. Bollmann, and G. S. Jung, "A critical investigation of recall and precision as measures of retrieval system performance," *ACM Transactions on Information Systems (TOIS)*, vol. 7, no. 3, pp. 205–229, 1989.

[46] 100fps. [Online]. Available: http://www.100fps.com/how_many_frames_can_humans_see.htm

[47] Where did sift and surf go in opencv 3. [Online]. Available: http://www.pyimagesearch.com/2015/07/16/where-did-sift-and-surf-go-in-opencv-3/

[48] R. Chellappa, A. Veeraraghavan, and G. Aggarwal, "Pattern recognition in video," in *Pattern Recognition and Machine Intelligence*. Springer, 2005, pp. 11–20.

[49] A. Kelman, M. Sofka, and C. V. Stewart, "Keypoint descriptors for matching across multiple image modalities and non-linear intensity variations," in *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 2007, pp. 1–7.

[50] M.G. De Klerk, W.C. Venter and A.J. Hoffman, "Resizing-effects analysis on video fingerprinting for use in video recognition," in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2016*. Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2016, pp. 242–247.

[51] Emgu cv library documentation - inter enumeration. [Online]. Available: http://www.emgu.com/wiki/files/3.1.0/document/html/eab4d5b4-81bc-52f9-1545-26230079ca30.htm

[52] Interpolation methods. [Online]. Available: http://northstar-www.dartmouth.edu/doc/idl/html_6.2/Interpolation_Methods.html

[53] J.-t. Shangguan, Y.-l. Li, Y.-g. Wang, and H.-l. Li, "Fast algorithm of modified cubic convolution interpolation," in *Image and Signal Processing (CISP), 2011 4th International Congress on*, vol. 2. IEEE, 2011, pp. 1072–1075.

[54] Interpolation algorithms in pixinsight. [Online]. Available: http://pixinsight.com/doc/docs/InterpolationAlgorithms/InterpolationAlgorithms.html

[55] A. Tanbakuchi. Comparison of opencv interpolation algorithms. [Online]. Available: http://tanbakuchi.com/posts/comparison-of-openv-interpolation-algorithms/

[56] R. Olivier and C. Hanqiang, "Nearest neighbor value interpolation," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 4, 2012.

[57] R. C. Gonzalez and R. E. Woods, "Digital image processing," 2002.

[58] D. Salomon and G. Motta, *Handbook of data compression*. Springer Science & Business Media, 2010.

[59] K. Gribbon, C. Johnston, and D. G. Bailey, "A real-time fpga implementation of a barrel distortion correction algorithm with bilinear interpolation," in *Image and Vision Computing New Zealand*, 2003, pp. 408–413.

[60] (2016) Public-domain test images for homeworks and projects. [Online]. Available: https://homepages.cae.wisc.edu/~ece533/images/

[61] Mcguire computer graphics archive. [Online]. Available: http://graphics.cs.williams.edu/data/images.xml#13

[62] Z. C. Lipton, C. Elkan, and B. Naryanaswamy, "Optimal thresholding of classifiers to maximize f1 measure," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 2014, pp. 225–239.

[63] Video effects and transitions. [Online]. Available: https://helpx.adobe.com/africa/premiere-pro/using/video-effects-transitions.html#noise_grain_effects

[64] N. A. T. LTD. Testimages project. [Online]. Available: http://testimages.tecnick.com

Appendix A

Appendix

A Test images

Table A-1: Test images

NR.	IMAGE NAME	FORMAT	SIZE (KB)	DIMENSIONS (height x width)	SOURCE
1	1440x1080	png	60	1440x1080	[61]
2	1920x1080	png	63	1920x1080	[61]
3	airplane	png	440	512x512	[60]
4	arctichare	png	259	594x400	[60]
5	baboon	png	623	512x512	[60]
6	barbara	bmp	258	512x512	[60], [61]
7	barbara	png	182	512x512	[60], [61]
8	boat	png	174	512x512	[60]
9	cameraman	png	37	256x256	[61]
10	cat	png	648	490x733	[60]
11	fruits	png	462	512x512	[60]
12	frymire	png	247	1118x1105	[60]
13	gamma	png	41	990x768	[61]
14	girl	png	609	768x512	[60]
15	goldhill	bpm	258	512x512	[60]
16	goldhill	png	169	512x512	[60]
17	height(Mermaid Dune Stereo Pair)	png	2050	4097x2049	[61]
18	left (Grand Canyon Heightfield)	png	141	538x343	[61]
19	lena	bmp	258	512x512	[60]
20	lena	jpg	90	512x512	[60], [61]
21	lena	png	501	512x512	[60], [61]
22	mandrill	png	612	512x512	[61]
23	mountain	bmp	302	640x480	[60]
24	mountain	png	199	640x480	[60]
25	normalCompressionMap0	png	686	1024x1024	[61]
26	normalCompressionMap1	png	200	512x512	[61]
27	normalCompressionMap2	png	55	256x256	[61]
28	normalCompressionMap3	png	15	128x128	[61]
29	normalCompressionMap4	png	5	64x64	[61]
30	peppers	png	527	512x512	[60], [61]
31	pool	png	183	510x383	[60]
32	right (Grand Canyon Heightfield)	png	141	539x343	[61]
33	sails	bmp	386	768x512	[60]
34	sails	png	788	768x512	[60]
35	serrano	png	105	629x794	[60]
36	SMPTE-color-bars	png	34	120x1080	[61]
37	star-chart	png	1493	5075x4725	[61]
38	tulips	png	664	768x512	[60]
39	watch	png	681	1024x786	[60]
40	zelda	png	136	512x512	[60]

Table A-2: High resolution test images

NR.	IMAGE NAME	FORMAT	SIZE (MB)	DIMENSIONS (height x width)	SOURCE
1	almonds	png	27	2448x2448	[64]
2	apples	png	25	2448x2448	[64]
3	baloons	png	25	2448x2448	[64]
4	bananas	png	27	2448x2448	[64]
5	billiard_balls_a	png	24	2448x2448	[64]
6	billiard_balls_b	png	25	2448x2448	[64]
7	building	png	25	2448x2448	[64]
8	cards_a	png	28	2448x2448	[64]
9	cards_b	png	26	2448x2448	[64]
10	carrots	png	26	2448x2448	[64]
11	chairs	png	24	2448x2448	[64]
12	clips	png	28	2448x2448	[64]
13	coins	png	28	2448x2448	[64]
14	cushions	png	25	2448x2448	[64]
15	ducks	png	25	2448x2448	[64]
16	fence	png	25	2448x2448	[64]
17	flowers	png	27	2448x2448	[64]
18	garden_table	png	28	2448x2448	[64]
19	guitar_bridge	png	27	2448x2448	[64]
20	guitar_fret	png	25	2448x2448	[64]
21	guitar_head	png	27	2448x2448	[64]
22	keyboard_a	png	27	2448x2448	[64]
23	keyboard_b	png	27	2448x2448	[64]
24	lion	png	25	2448x2448	[64]
25	multimeter	png	27	2448x2448	[64]
26	pencils_a	png	25	2448x2448	[64]
27	pencils_b	png	24	2448x2448	[64]
28	pillar	png	26	2448x2448	[64]
29	plastic	png	27	2448x2448	[64]
30	roof	png	25	2448x2448	[64]
31	scarf	png	29	2448x2448	[64]
32	screws	png	29	2448x2448	[64]
33	snails	png	25	2448x2448	[64]
34	socks	png	29	2448x2448	[64]
35	sweets	png	26	2448x2448	[64]
36	tomatoes_a	png	25	2448x2448	[64]
37	tomatoes_b	png	25	2448x2448	[64]
38	tools_a	png	27	2448x2448	[64]
39	tools_b	png	27	2448x2448	[64]
40	wood_game	png	24	2448x2448	[64]

B Test videos

B.1 Adverts

South Africa specific

Table A-3: Test Adverts - South Africa Specific

NR.	NAME	DURATION	WIDTH	HEIGHT	FRAMERATE
1	ABSA Business Banking	00:00:45	1280	480	25
2	ABSA MegaU Free	00:01:02	1920	1080	25
3	ABSA Provider	00:00:30	1280	720	25
4	ABSA Tooth fairy	00:00:37	1280	720	25
5	Allan Gray The Chase	00:01:00	1100	718	25
6	Audi Q3	00:00:45	1280	546	25
7	Cadbury Bubly Aquarium	00:00:45	1920	1080	25
8	CSIR AFIS	00:01:00	1280	720	25
9	Defy Believe in Better	00:00:47	1280	720	25
10	DSTV Power	00:01:00	1280	720	25
11	DSTV You	00:01:00	1280	720	25
12	Five Roses Kitty	00:00:30	1280	720	25
13	investec fire pool	00:00:45	1920	1080	25
14	Nedbank Reins	00:03:00	1280	478	25
15	RealCostCampaign Skinny Jeans	00:00:30	1280	720	25
16	Shoprite Promise	00:01:00	1280	720	25
17	Transact Perseverance	00:00:50	1920	1080	25
18	Vodacom #JustSwitch TV Ad	00:01:00	1280	720	25

Table A-4: Test Adverts - South Africa Specific Sources

NR.	SOURCE	Listing
1	https://vimeo.com/117470677	https://vimeo.com/velocityfilmssa/videos
2	https://vimeo.com/163246132	https://vimeo.com/velocityfilmssa/videos
3	https://vimeo.com/133988992	https://vimeo.com/velocityfilmssa/videos
4	https://vimeo.com/106293583	https://vimeo.com/velocityfilmssa/videos
5	https://vimeo.com/113006582	https://vimeo.com/velocityfilmssa/videos
6	https://vimeo.com/135564577	https://vimeo.com/velocityfilmssa/videos
7	https://vimeo.com/173036920	https://vimeo.com/velocityfilmssa/videos
8	https://vimeo.com/117790953	https://vimeo.com/velocityfilmssa/videos
9	https://vimeo.com/104912404	https://vimeo.com/velocityfilmssa/videos
10	https://vimeo.com/151889984	https://vimeo.com/velocityfilmssa/videos
11	https://vimeo.com/144734627	https://vimeo.com/velocityfilmssa/videos
12	https://vimeo.com/136812199	https://vimeo.com/velocityfilmssa/videos
13	https://vimeo.com/163824982	https://vimeo.com/velocityfilmssa/videos
14	https://vimeo.com/130409779	https://vimeo.com/velocityfilmssa/videos
15	https://vimeo.com/131073592	https://vimeo.com/velocityfilmssa/videos
16	https://vimeo.com/132806720	https://vimeo.com/velocityfilmssa/videos
17	https://vimeo.com/174787130	https://vimeo.com/velocityfilmssa/videos
18	https://www.youtube.com/watch?v=gIexUhmuhXk	https://www.youtube.com/watch?v=gIexUhmuhXk

Global

Table A-5: Test Adverts -Global

NR.	NAME	LENGTH	WIDTH	HEIGHT	FRAMERATE
1	Ad of the Day This Up-and-Coming Beer Brand Got Its First Ad	00:01:02	1920	1080	23
2	Cisco - Pep Talk	00:01:00	1280	720	23
3	Dark Arts Coffee Coffee is in our blood Ads of the World	00:03:25	1920	1012	25
4	Gillette- Perfect Isn't Pretty - Rio 2016 Olympic Games - Sia Unstoppable	00:03:09	1280	720	25
5	Gold Feelings	00:01:00	640	360	24
6	Handpicked for Greatness #GreatnessIsBrewing	00:00:14	1280	720	23
7	I Love Movies IMAX Movies - IMAX Brand Film	00:01:00	1280	676	24
8	Inspected for Greatness #GreatnessIsBrewing	00:00:14	1280	720	23
9	iPhone 6s - Onions	00:01:00	1280	720	23
10	IT COMES FROM BELOW - BRYCE HARPER	00:00:47	1280	720	23
11	Liftoff - Café de Colombia #GreatnessIsBrewing	00:01:12	1280	720	23
12	Nike Basketball - Worth the wait	00:01:00	1280	720	23
13	Nike Football Presents- The Switch ft. Cristiano Ronaldo, Harry Kane, Anthony Martial & More	00:05:57	1280	720	23
14	Nike Presents Da Da Ding	00:02:52	1280	720	25
15	Olympics 2016 on the BBC - The Greatest Show on Earth - BBC Sport	00:01:33	1280	720	25
16	OneToughMother	00:02:01	1280	720	23
17	Red Bull - science behind break dance	00:01:56	1920	1080	25
18	Red Bull - Sounds of football	00:01:34	1920	1080	25
19	Responsibilities	00:00:31	1280	720	23
20	Taste-Tested for Greatness #GreatnessIsBrewing	00:00:14	1280	720	23
21	Telkom Oculus	00:00:30	1920	1080	25
22	The Athlete Machine - Red Bull Kluge	00:05:57	1280	720	29
23	The First Flight - Heathrow Airport Summer 2016	00:01:30	1280	720	25
24	UNISA Define Tomorrow (Extended Video)	00:02:09	1280	720	25
25	Virgin Media - Usain Bolt - #BeTheFastest	00:01:40	1280	720	25

Table A-6: Test Adverts -Global Sources

NR.	SOURCE	Listing
1	https://vimeo.com/170941597	http://www.adweek.com/news/advertising-branding/ad-day-heres-very-no-frills-ad-very-no-frills-beer-172350
2	https://youtu.be/1qemmj6ue9s	http://www.adweek.com/news/advertising-branding/ad-day-ewan-mcgregor-waxes-poetic-human-achievement-cisco-172080
3	https://vimeo.com/169239735	https://adsoftheworld.com/media/tv/dark_arts_coffee_coffee_is_in_our_blood
4	https://youtu.be/xRXfevLDZBc	http://www.adweek.com/news/advertising-branding/ad-day-gillettes-grand-olympic-spot-takes-dark-look-athletes-sacrifices-172533
5	https://youtu.be/SW7Cuopu8No	https://adsoftheworld.com/media/tv/cocacola_gold_feelings
6	https://youtu.be/CajcuCyDsec	http://www.adweek.com/news/advertising-branding/ad-day-how-coffee-maybe-probably-got-us-moon-47-years-ago-172594
7	https://youtu.be/MQUQx4eGAqQ	http://adsoftheworld.com/media/tv/imax_first_commercial_shot_on_an_imax_camera
8	https://youtu.be/UioBtL3BW5o	http://www.adweek.com/news/advertising-branding/ad-day-how-coffee-maybe-probably-got-us-moon-47-years-ago-172594
9	https://youtu.be/2gHeBVyqlRo	http://www.adweek.com/news/advertising-branding/ad-day-world-weeping-over-beauty-video-shot-iphone-6s-171080
10	https://youtu.be/nw65LZucfpQ	https://adsoftheworld.com/media/tv/under_armour_it_comes_from_below
11	https://youtu.be/HyzAnlNQ85A	http://www.adweek.com/news/advertising-branding/ad-day-how-coffee-maybe-probably-got-us-moon-47-years-ago-172594
12	https://youtu.be/ZyGL6B7OH5A	http://www.adweek.com/news/advertising-branding/ad-day-cleveland-fans-are-stunned-be-champs-nikes-salute-cavaliers-172099
13	https://youtu.be/scWpXEYZEGk	http://www.adweek.com/news/advertising-branding/ad-day-cristiano-ronaldo-swaps-lives-kid-nikes-epic-euro-2016-film-171901
14	https://youtu.be/1UvPZ8fD4B8	https://adsoftheworld.com/media/tv/nike_da_da_ding
15	https://youtu.be/EkRVqn5gZJY	https://adsoftheworld.com/media/tv/bbc_the_greatest_show_on_earth
16	https://youtu.be/ESNw1VC67yc	http://www.adweek.com/news/advertising-branding/ad-day-teleflora-finds-fresh-use-vince-lombardi-powerful-ad-mothers-day-171034
17	https://vimeo.com/145732000	https://adsoftheworld.com/media/online/red_bull_science_behind_break_dance
18	https://vimeo.com/167930556	https://adsoftheworld.com/media/online/red_bull_sounds_of_football
19	https://youtu.be/zYfuKWV5fns	http://www.adweek.com/news/advertising-branding/ad-day-droga5-has-johnsonville-employees-dream-their-own-ridiculous-ads-171678
20	https://youtu.be/9WDbCq2WVt0	http://www.adweek.com/news/advertising-branding/ad-day-how-coffee-maybe-probably-got-us-moon-47-years-ago-172594
21	https://vimeo.com/165747539	http://adsoftheworld.com/media/tv/telkom_oculus
22	https://youtu.be/MOjmSsQSptw	https://adsoftheworld.com/media/tv/red_bull_kluge_the_athlete_machine
23	https://youtu.be/lMaVlVBmDr0	http://www.adweek.com/news/advertising-branding/ad-day-heathrow-airport-crafts-sweet-first-tv-spot-set-david-bowie-song-172567
24	https://youtu.be/DZqUlgUHYfQ	https://adsoftheworld.com/media/tv/unisa_define_tomorrow
25	https://youtu.be/AdGQ43smLoc	http://www.adweek.com/news/advertising-branding/ad-day-958-seconds-lasts-lifetime-usain-bolts-new-ad-virgin-media-172425

C Research contributions

C.1 SATNAC 2014

Digital video shot boundary detector investigation

M.G. de Klerk, W.C. Venter and A.J. Hoffman
School of Electrical, Electronic and Computer Engineering
North-West University, Potchefstroom Campus, South Africa
Tel: +27 18 299 1978, Fax: +27 86 521 2569
Email: {20555466, 10063218, Alwyn.Hoffman}@nwu.ac.za

***Abstract*—Many algorithms have been proposed and evaluated for the detection of shot boundaries in digital video. Although these techniques have been verified, there remains a lack of standardised data to classify which techniques are best suited for certain applications. The Jensen-Shannon divergence (JSD) is one such technique used for shot boundary detection. In this article the JSD technique was adapted to handle monochromatic and RGB videos. This adaptation made it possible for the JSD technique to be evaluated in the RGB and monochromatic (grayscale) color spaces as well as the effect of video resizing in terms of recall, precision and execution time.**

***Index Terms*—shot boundary detector (SBD); Jensen-Shannon Divergence**

I. Introduction

Digital video has become part of our everyday lives. It is not just found in television broadcasts any more but it is commonplace, everywhere from our homes to our workplace, and even on the go with the modern mobile devices. This advent of digital communication by means of video has brought forth a new set of challenges.

Managing video content is a laborious task that has grown beyond what is manually practical. Various techniques have been developed that can perform this task, but tend do be computationally intensive and slow to execute [1]. These techniques are discussed in literature, but a lack of standardised testing makes a quantitative comparison from literature impossible.

One of the core techniques utilized to manage video content is a shot boundary detector (SBD). The aim of this investigation is to quantitatively compare one such a SBDs by subjecting it to a multitude of controlled tests.

This article provides an overview of a SBD and previous work done in this regard in section II-B. This is followed by a breakdown of the methodology followed while evaluating the JSD SBD in section III. Lastly the experimental results of the investigation and a conclusion thereof is provided in sections IV and V.

II. Background

Video segmentation refers to the process of breaking down a video sequence into the shots from which it is comprised [2]. This can be achieved by manually identifying shot boundaries or by implementation of a SBD. A SBD is an algorithm used to identify the shot boundary between consecutive shots in a video stream. This boundary is generally a considerable discontinuity in the visual-content flow of a video sequence [3].

In order to grasp the application of a SBD, it is important to understand the basic structure of digital video. The basic structure of digital video is inherently similar to that of traditional film video. The smallest component of any video is called a frame, which is a static image. A collection of multiple consecutive frames, captured by a camera in a single continuous take, is called a shot [4]. When multiple shots are combined into a single video, it is referred to as a sequence, or commonly referred to as a video as seen in figure 1.

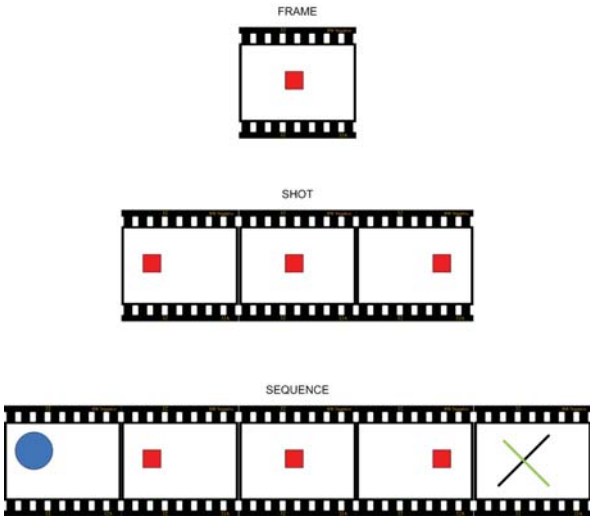


Fig. 1. Hierarchical structure of a video sequence

A. SBD overview

Shot boundaries, as illustrated in figure 2, can be classified into two distinct classes: hard and soft cuts. A hard-cut refers to an abrupt change from one shot to another [5]. Soft transitions refers to a collection of transition methods used to soften the transition boundary between frames. This is commonly done by dissolving one frame into (fade-in) or from (fade-out) a black image [6]. Another technique involves fading one video section into another, called cross-fading. Soft-cuts are generally more difficult to detect than hard-cuts and can be easily falsely detected due to a drastic illumination difference e.g. a lens flare.

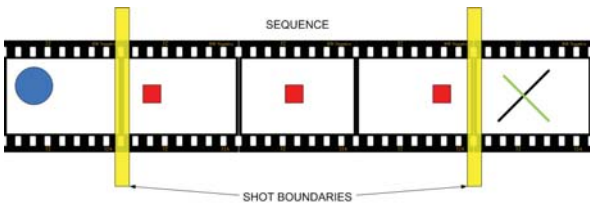


Fig. 2. Video structure hierarchy

A SBD algorithm identifies the start and end of a new shot. Once the start and end positions of a shot are known, it can be further analysed in manners similar to the SBD to determine the most representative frame of that shot, called the key-frame, for video indexing purposes.

B. Related work

A lot of work has been done in the field of SBD algorithms. The majority of SBD algorithms are designed to function on color videos. Each pixel in these color frames are represented by a red, green and blue (RGB) value. This provides techniques like the Jensen-Shannon Divergence (JSD) algorithm with 3 separate *datasets* i.e. histograms of the red (R), green (G) and blue (B) components of each pixel. This abundance of information makes it easier to detect soft transitions between shots, but it comes at a price. An RGB video has 3 times more data to process than a grayscale video, thus requiring more time to process all the data.

The aim of this investigation is to determine how these various techniques can be adapted to function on grayscale data to increase processing speed but retaining the accuracy of the algorithm.

C. SBD techniques

There are various techniques available to detect the shot boundaries in videos. The majority of these techniques can be grouped under pixel based or histogram based techniques.

Pixel-based methods

The simplest way of comparing two consecutive frames is by comparing the corresponding pixels in both frames and measuring the difference. A shot boundary may have been detected if the number of changed pixels exceeds a prescribed threshold. This technique is sensitive to camera motion [5] and noise since each pixel is evaluated as a single entity.

Statistical methods like the mean and standard deviation of the pixels' gray levels can be used to provide a measure of variance between frames. This method proved to be more tolerant of noise, but takes longer to perform calculations due to the complexity of the statistical formulas.

Histogram-based methods

Histograms SBD techniques applies the same basic principles as pixel-based techniques, but now the pixels are grouped into bins.

The most commonly used metric for shot boundary detection is the histogram difference between frames. Histograms are drawn of each frame by counting pixels with certain values based upon the number of bins chosen - typically 256 bins to quantize the image. Grayscale images deliver a single histogram per frame where color images require 3 histograms.

By drawing a histogram of each frame and subtracting them, one is left with a difference histogram indicating the frame-to-frame content change. Frame pairs with a high content change are usually indicative of a shot boundary. A linear combination or straight forward summations of these differences are then used as a measure of the total content change across the frames [7]. A predefined value for these metrics, called a threshold, is generally used to discern if this difference constitutes a shot boundary. These thresholds can be set manually or calculated automatically.

These histograms can then be compared using various techniques like the JSD, Chi-squared (χ^2) comparison [8] and various other techniques.

This article investigates the JSD technique as it was adapted for both RGB (3 histograms) and monochromatic (1 histogram) videos. Since the JSD technique is a histogram based method, it is less susceptible to noise and camera movement, with the advantage of faster processing speeds which makes it ideal for applications such as advertisement tracking.

JSD algorithm

The JSD algorithm provides a means to determine the frame difference measure (FDM) between consecutive frames, which in turn can be used to detect if these frames constitute a shot boundary. As the name suggests, the JSD algorithm is a combination of the Shannon's entropy and the Jensen inequality.

The Shannon entropy [9] is a method used in information theory as a measure of information choice and uncertainty. The Shannon entropy function H of the probability distribution $P = (p_1, p_2, p_3, \ldots, p_n)$ consisting of n possibilities in the distribution is calculated by:

$$H(P) = -K \sum_{i=1}^{N} p_i \log_b p_i \tag{1}$$

where K is a positive constant [10].

This entropy calculation is combined with the Jensen inequality measure between two consecutive frames' histograms as derived by Qing Xu [4] producing the JSD equation:

$$JSD(f_{i-1}, f_i) = H\left(\frac{P_{f_{i-1}} + P_{f_i}}{2}\right) - \frac{H\left(P_{f_{i-1}}\right) + H\left(P_{f_i}\right)}{2} \tag{2}$$

where f_{i-1} is the previous frame and f_i the current frame. The probabilities of the frames are respectively $P_{f_{i-1}}$ and P_{f_i}. The aforementioned probabilities are calculated from the applicable frames' histograms as expressed in equation 3:

$$P(f_i) = \frac{Histogram\,(f_i)}{Height\,(f_i) \times Width\,(f_i)} \tag{3}$$

where the histogram of each color component is divided by the number of pixels in that frame f_i. The number of pixels is calculated by multiplying the number of horizontal pixels by the number of vertical pixels in the frame.

The results from equation 2 are then used to create a moving average of the previous frames that serves as a threshold level to test the current frame-to-frame difference against. Trial and error has indicated that a *moving average window* of 5 frames produced satisfactory results for an *auto-adjusting threshold*.

III. Methodology

A high level abstraction of the basic methodology followed during the SBD analysis can be seen in figure 3.

MATLAB was used as the primary programming environment due to the availability of mathematical and statistical tools as well as video functionality.

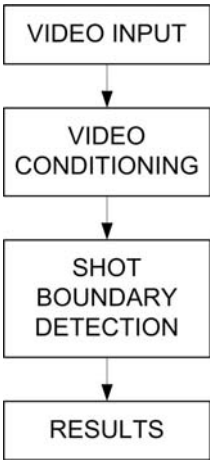


Fig. 3. SBD high level program flow

- *Video input*
  Each video is read into MATLAB's workspace as a video object with the `VideoReader` function, containing all the frames and relevant video information. Due to constraints of the aforementioned function, only the following video formats are supported:
  - All Platforms:
    - Motion Joint Photographic Experts Group (JPEG) 2000 (.mj2);
    - Audio Video Interleave (.avi);
  - All Windows® platforms:
    - Moving Picture Experts Group Phase 1 (MPEG1) (.mpg);
    - Windows Media® Video formats:
      - Advanced Systems Format (.asf);
      - Windows Media Video (.wmv);
      - Advanced Stream Redirector (.asx);
  - Windows 7 or later:
    - Apple Quicktime Movie (.mov);
    - Moving Picture Experts Group Phase 4 (MPEG4 including H.264 encoded video) (.mp4);
  - Moving Picture Experts Group Phase 1 (.mpg);
- *Video conditioning*
  Due to the input format constraints imposed by using the `VideoReader` MATLAB function, all the videos used in the comparison had to be converted to a suitable format as listed above. Many digital videos on the internet are packaged in a Adobe Flash video format (.flv) [11] which is not natively supported by MATLAB. These videos were converted to the AVI container format, using their original codecs in *Any Video Converter Ultimate 4.4.2* which utilizes the FFmpeg. FFmepeg provides a complete, cross-platform solution to record, convert and stream audio and video [12].
  The digital videos were subjected to some conversions during the analysis to determine the effect thereof on the accuracy and execution speed of the SBD. The RGB-to-Grayscale conversions were done in MATLAB on the video loaded into memory using `rgb2gray`, leaving the original video in its original state for further calculations. Resizing of the video to a smaller resolution of 320x240 was done beforehand by using *Any Video Converter Ultimate 4.4.2*.
  An arbitrary resolution of 320x240 was chosen for the resize conversion, since it fit all the source video's aspect ratio of 4:3.
- *Shot boundary detection*
  The aforementioned JSD algorithm was implemented on the video data that has been loaded into MATLAB's workspace.
- *Results*
  The frame index of each boundary detected by the JSD algorithm was stored as well as the processing time of the SBD algorithm. The frame indexes were used to visually confirm the frame identified as a shot boundary by displaying it alongside the previous and following frame as seen in figure 4. All the videos were also analysed by hand to identify all the shot boundaries. This manual process is called ground-truthing [1]. Based upon the results obtained from the comparison, it was possible to calculate the precision and recall factors as defined in section IV.

Fig. 4. SBD visual comparison example

The aforementioned execution time of the SBD algorithm does not include the *overhead* of the simulation that includes the loading and conditioning (wrapper conversion) of the video. This is attributed to the fact the current video analysis approach implemented in MATLAB was focused on comparing the speed, recall and precision of the SBD technique on the various videos.

IV. Results

The Performance of the JSD algorithm, based upon recall and precision rates as expressed in equations 4 and 5 [13], was determined when the algorithm was applied to different test videos.

Recall rates provides a ratio of the number of relevant shot boundaries (correct detection) to the total number of relevant shot boundaries in the video, while precision rates provides the ratio of relevant shot boundaries to the total number of irrelevant shot boundaries (false detections) retrieved.

$$Recall = \frac{Correct}{Correct + Missed} \tag{4}$$

$$Precision = \frac{Correct}{Correct + FalsePositive} \tag{5}$$

In order to simplify the ground-truthing process and ensure accuracy, multiple test sequences were created to test the functionality of the technique. A linear depiction of these test sequences is illustrated in figure 5.

The sequences were constructed to have the following properties:

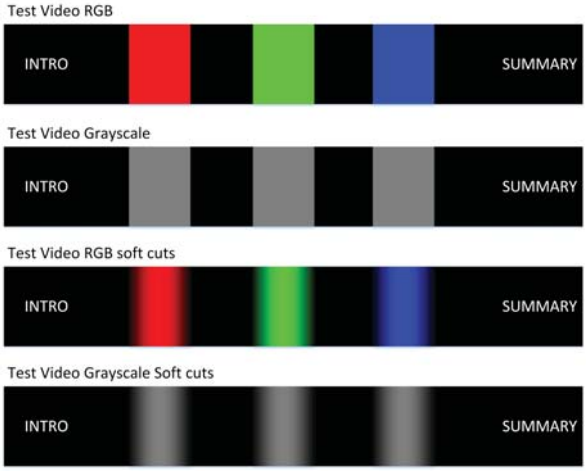


Fig. 5. Test video sequences

- *Test Video RGB:*
  Black video sequence to R, G and B sequence transitions featuring hard cuts;
- *Test Video Grayscale:*
  Black video sequence to gray sequence hard-cut transitions;
- *Test Video RGB soft cuts:*
  Soft transitions from black to R, G and B and back to black again. Cross-dissolve transition duration is 25 frames;
- *Test Video Grayscale soft cuts:*
  Black video sequence to gray sequence and back to black cross-dissolves.

The hypothesis behind the full color versus monochromatic representation was to test how the accuracy of the technique holds up with the loss of data as well as the effect thereof on the execution speed. It is important to note that the gray sections in the aforementioned test videos were created by a RGB value of R=128; G=128; B=128 on the RGB 8bit color map. This RGB-grayscale representation allowed the JSD technique be compared on RGB-gray (3 separate histograms) and monochromatic-gray (1 histogram) videos. All video sequences were converted to monochromatic representations where applicable in the various algorithms by the principle explained in equation 6:

$$monochromatic = \frac{R+G+B}{3} \tag{6}$$

where R,G and B are the individual component values of each pixel.

A. Test video results

The execution times listed was obtained by averaging the execution times of the JSD SBD technique for 10 iterations. The results of the RGB test video shot boundary detection analysis are listed in table I, confirming the previous statement that an RGB comparison of the 3 histograms takes longer to execute compared to a monochromatic comparison with 1 histogram. The *JSD Grayscale* technique listed includes the time required for the RGB-to-Grayscale conversion with the `rgb2gray` function. The weak recall rate produced by the RGB techniques can be attributed to the thresholding criteria.

The thresholding for RGB videos is essentially the same as for the monochromatic videos. The only difference is that it is implemented on all three components. The threshold concept for monochromatic videos to detect a shot boundary (SB) is given by equation 7:

$$SB = \begin{cases} True & \text{if } JSD(f_{i-1}, f_i) > T_{aveGray} \\ False & \text{otherwise} \end{cases} \tag{7}$$

where $T_{aveGray}$ is the average moving threshold computed from the *moving average window*. Similarly for RGB videos, each color channel represented by equations 8, 9 and 10 must be evaluated:

$$SB_R = \begin{cases} True & \text{if } JSD_R(f_{i-1}, f_i) > T_{aveR} \\ False & \text{otherwise} \end{cases} \tag{8}$$

$$SB_G = \begin{cases} True & \text{if } JSD_G(f_{i-1}, f_i) > T_{aveG} \\ False & \text{otherwise} \end{cases} \tag{9}$$

$$SB_B = \begin{cases} True & \text{if } JSD_B(f_{i-1}, f_i) > T_{aveB} \\ False & \text{otherwise} \end{cases} \tag{10}$$

in order to calculate if a shot boundary has been found. A shot boundary has been identified if equation 11 is satisfied:

$$SB_{RGB} = \begin{cases} True & \text{if } SB_R \;\&\; SB_G \;\&\; SB_B = True \\ False & \text{otherwise.} \end{cases} \tag{11}$$

TABLE I
TEST VIDEO - RGB

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.83	100	87.5
JSD RGB	5.08	100	14.29
JSD Grayscale Resized	1.11	100	87.5
JSD RGB Resized	1.85	100	14.29

The native grayscale test video produced better recall rates as can be seen in table II. The only difference between the RGB and grayscale test videos are the colors. The RGB video has definitive red, green and blue scenes, while the grayscale video has only gray scenes amidst the black timeline. These gray scenes contain equal values for all the RGB channels (R=G=B=128). Hence the boundary detection conditions as required by equation 11 is easily satisfied for if one channel is satisfied, the remaining two will be satisfied as well.

TABLE II
TEST VIDEO - GRAYSCALE

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.85	100	87.5
JSD RGB	5.17	100	87.5
JSD Grayscale Resized	1.09	100	100
JSD RGB Resized	1.78	100	14.29

Soft cut test video results

The nature of a gradual transition makes it difficult to distinctively classify a certain frame as the shot boundary. The recall and precision rates listed in tables III and IV were calculated by determining if the detected boundary was within the frame-range constituting the gradual transition.

TABLE III
TEST VIDEO - RGB SOFT CUTS

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.80	70	87.5
JSD RGB	4.93	100	87.5
JSD Grayscale Resized	1.10	77.78	87.5
JSD RGB Resized	1.81	50	91.3

The lower precision values are attributed to the false detections during the gradual transition of both the monochromatic and RGB videos. Only one shot boundary was allowed within the span of the gradual transition as the definition of a shot boundary implies that each shot has a start and stop boundary. A gradual transition from black to white is illustrated in figure 6 over a period of 10 frames. Adhering to the aforementioned guidelines regarding the gradual transition, only one frame should be identified as the shot boundary e.g. frame nr. 5, even though frame 6 might also provide a difference value higher than the current threshold.

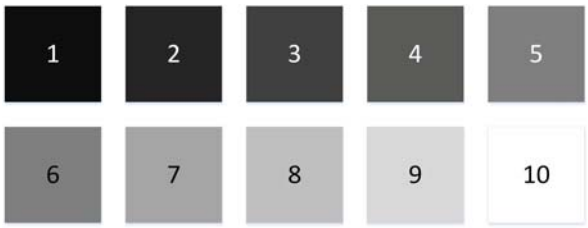


Fig. 6. Gradual Transition

TABLE IV
TEST VIDEO - GRAYSCALE SOFT CUTS

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.77	53.85	87.5
JSD RGB	4.83	53.85	87.5
JSD Grayscale Resized	1.10	80	100
JSD RGB Resized	1.81	80	12.5

Hence the low precision is caused by the identification of a second shot boundary alongside the current shot boundary within the same transition period.

All the test videos have an introduction title and information summary at the end of the sequence. It is debatable whether or not these text sequences can be classified as a shot or not. Although the majority of the techniques failed to identify the addition or removal of these text sequences, with the exception of both the resized implementations of the JSD algorithm on the grayscale test sequences.

B. Other video results

The same analysis was done on a video generally available on the internet in order to determine how they fared compared to the best case scenarios of the test videos. The action-sports themed RedBull commercial contains multiple lens-flares and flash photography that in turn gave rise to a few false detections. The results listed in table V indicate worse recall and precision rates for the original video than compared to the smaller resized versions thereof.

Upon further investigation into this peculiarity, it came to light that the frame rate of the video has an effect on the accuracy of the techniques being evaluated. The original RedBull Commercial was a 1280x720 video file with a frame rate of 30 frames per second (fps). During the resize process, the video was down-sampled to 25 fps, the same as all the test videos.

A higher video frame rate constitutes more frames during a gradual transition, in effect causing less change from frame-to-frame. Thus in order for the JSD technique to function correctly at a higher frame rate, the size of the moving average window used for thresholding should be increased while the threshold might be lowered to detect the *slower changes* in a frame-to-frame context.

TABLE V
THE WORLD OF REDBULL TV COMMERCIAL 2013

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	7.31	72.73	92.31
JSD RGB	10.49	74.19	88.46
JSD Grayscale Resized	1.87	76.67	100
JSD RGB Resized	2.94	87.5	100

The final video used in the comparison was a wildlife video, depicting various scenes with slow and fast moving animals, hard cuts between shots and no lens-flares. The results, as listed in table VI, shows a perfect score for both recall and precision across all the SBD techniques.

TABLE VI
WILDLIFE

SBD Technique	Execution Time (s)	Precision (%)	Recall (%)
JSD Grayscale	3.11	100	100
JSD RGB	4.29	100	100
JSD Grayscale Resized	0.88	100	100
JSD RGB Resized	1.40	100	100

V. CONCLUSION

The SBD technique comparison confirmed the logical assumption that RGB comparisons take longer to execute than a monochromatic comparison. Overall the RGB techniques had higher precision values than the grayscale techniques, but their recall rates plummeted in certain instances.

Amongst other factors, it is clear that lens-flares and flash photography can adversely affect the outcome of these detection algorithms by providing false positives. Heng et al. [14] proposed various techniques to remove false alarms caused by abrupt illumination changes which can be implemented.

It can be concluded that for most instances of this comparison, the JSD grayscale technique on resized videos at 25fps produced the best results in the shortest amount of time.

There are a few impact factors on SBD that require further investigation:

- Effect of the frame rate;
- Effect of various file wrappers and codecs;
- Text over video.

The general convention for video composition focusses the subject in the middle of the frame, hence the most detail and movement tends to be focussed here. This convention calls for further investigation into a special attention areas as described by Gu et al. in [15].

VI. Acknowledgement

I would like to thank my supervisor, Prof. W.C. Venter for his help and guidance throughout the duration of the project.

References

[1] U. Gargi, R. Kasturi, and S. H. Strayer, "Performance characterization of video-shot-change detection methods," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 10, no. 1, pp. 1–13, 2000.

[2] I. Koprinska and S. Carrato, "Temporal video segmentation: A survey," *Signal processing: Image communication*, vol. 16, no. 5, pp. 477–500, 2001.

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

[4] M. Vila, A. Bardera, Q. Xu, M. Feixas, and M. Sbert, "Tsallis entropy-based information measures for shot boundary detection and keyframe selection," *Signal, Image and Video Processing*, vol. 7, no. 3, pp. 507–520, 2013.

[5] J. S. Boreczky and L. A. Rowe, "Comparison of video shot boundary detection techniques," *Journal of Electronic Imaging*, vol. 5, no. 2, pp. 122–128, 1996.

[6] W. J. H. N. Ngan, "Shot boundary refinement for long transition in digital video sequence," *IEEE TRANSACTIONS ON MULTIMEDIA*, vol. 4, no. 4, pp. 434–445, 2002.

[7] M. R. Naphade, R. Mehrotra, A. M. Ferman, J. Warnick, T. S. Huang, and A. M. Tekalp, "A high-performance shot boundary detection algorithm using multiple cues," in *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, vol. 1. IEEE, 1998, pp. 884–887.

[8] I. Williams, N. Bowring, and D. Svoboda, "A performance evaluation of statistical tests for edge detection in textured images," *Computer Vision and Image Understanding*, vol. 122, pp. 115–130, 2014.

[9] C. E. Shannon, "A mathematical theory of communication," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 5, no. 1, pp. 3–55, jan 2001. [Online]. Available: http://doi.acm.org/10.1145/584091.584093

[10] M. Menéndez, J. Pardo, L. Pardo, and M. Pardo, "The jensen-shannon divergence," *Journal of the Franklin Institute*, vol. 334, no. 2, pp. 307–318, 1997.

[11] J. Emigh, "New flash player rises in the web-video market," *Computer*, vol. 39, no. 2, pp. 14–16, 2006.

[12] Ffmpeg. [Online]. Available: http://www.ffmpeg.org/

[13] V. Raghavan, P. Bollmann, and G. S. Jung, "A critical investigation of recall and precision as measures of retrieval system performance," *ACM Transactions on Information Systems (TOIS)*, vol. 7, no. 3, pp. 205–229, 1989.

[14] W. J. Heng and K. N. Ngan, "High accuracy flashlight scene determination for shot boundary detection," *Signal Processing: Image Communication*, vol. 18, no. 3, pp. 203–219, 2003.

[15] X. Gu, Z. Chen, and Q. Chen, "Refinement of extracted visual attention areas in video sequences," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 966–969.

M.G. de Klerk is currently studying towards an PhD degree in Electrical and Electronic Engineering at the North-West University after receiving his M.Eng. degree on renewable energy simulations in 2013.

C.2 SATNAC 2015

Automatic shot boundary detection for streaming digital video

[1]M.G. De Klerk*, [2]W.C. Venter*, [3]A.J. Hoffman*
*School of Electrical, Electronic and Computer Engineering, North-West University
[1]20555466@nwu.ac.za
[2]Willie.Venter@nwu.ac.za
[3]Alwyn.Hoffman@nwu.ac.za

***Abstract*—The art of shot boundary detection (SBD) is a fundamental principle commonly encountered when working with digital videos. Various different SBD techniques exist and have been implemented in specific instances. A common problem encountered with traditional SBD techniques lies in the various types of shot boundaries, especially gradual transitions. Implementation of the standard deviation of pixel intensities (SDPI) was used to characterise the transitional effects in order to classify them when encountered in a digital video stream. This classification of the shot boundaries helps to understand and overcome the uncertainty of boundary location during gradual transitions.**

***Index Terms*—shot boundary detection); Jensen-Shannon Divergence; Standard deviation of pixel intensities**

I. Introduction

Throughout the years multiple techniques have been developed and implemented with the goal of detecting digital video shot boundaries. The majority of these techniques were developed to function on digital videos where the whole video file is available. This allows for an analysis of the complete video file with the advantage of calculating global thresholds against which the techniques can evaluate their respective metrics.

An additional advantage of those techniques is that the frame(s) that are currently being evaluated, can be compared to previous frames as well as *future* frames. This seemingly irrelevant property becomes apparent when considering video segmentation detection techniques for application on streaming media.

Streaming media is refers to audio or video media being sent over the internet and played immediately on the client side without saving the complete media file to local storage media first [1]. These advances in technology has prompted the need to re-evaluate existing techniques and their applicability to the modern standards. The techniques utilised throughout this paper were chosen on the premise of being supplied with streaming video data. Apart from the logical restrictions of *future* frames, streaming video poses yet another challenge. In order to process the streaming media, the techniques applied thereto has to be executed in *real-time*. Throughout this paper the concept of *real-time* refers to the requirement that the time $t_{calculations}$ required for any and all calculations done on the media, be done in less time than the standard playback duration of the media $t_{duration}$

$$t_{calculations} \leq t_{duration}. \tag{1}$$

II. Background

Shot boundary detection is the process by which a digital video is analysed to determine where the current video shot segment ends and the following one begins. In this context a shot is defined as the collection frames that constitute a contiguous recording depicting a visually continuous action in space and time [2].

The majority of digital videos that are encountered are subjected to some form of video editing. The most common types of video editing pertains to the way in which various shots are coalesced to create a video. When two or more unique shot sequences are coalesced, a break in the visual continuity is encountered. This break in visual continuity is called a shot boundary.

The manner in which these shot boundaries are traversed is called the transition. When no transitional effects are performed on the shot boundaries, they are referred to as hard-cuts. It is however commonly encountered that transitional effects are performed to soften the change in visual continuity between the consecutive shots. The collective term used to refer to these types of transitions is aptly called soft-transitions.

The three most commonly encountered transitions are:

- Hard-cuts;
- Fades:
  A fade from black is encountered when black frame is progressively changed by incrementally overlaying consecutive frames with increasing intensities until the initial black shot is no longer visible. This type of transition is generally encountered at the beginning, and the reverse thereof at the end of a video sequence.
- Dissolves:
  During a dissolve the intensity of the preceding sequence gradually decreases while at the same time the succeeding sequences' intensity increases.

Video segmentation analysis

In this article the term video segmentation analysis is used to describe the process of analysing digital video with the

goal of locating the aforementioned shot boundary transitions with their respective boundaries. The process sounds fairly trivial when human cognition can be employed to identify and classify these boundaries. However the process becomes fairly more complex when an unsupervised algorithm is to be employed.

Video segmentation analysis techniques can be broadly classified into two main categories based on how they quantify and process the relevant video frames: pixel-based and histogram-based methods.

The logical procedure employed to analyse a digital video with the intention of identifying shot boundaries is to compare two consecutive frames. If the difference between the consecutive frames falls below a certain threshold it can be assumed that the two frames are from the same shot sequence. The threshold used for the difference comparison is due to the inherent nature of a video where each frame is expected to vary slightly from the previous frame.

Pixel-based techniques: Each shot sequence is comprised of multiple frames, which in turn are comprised of pixels. The number of pixels present in a frame stays constant throughout the video. The only variation encountered is the changing of the pixels' intensity [3].

The simplest method to compare two consecutive frames is by comparing the corresponding pixels in both frames and measuring the difference thereof. This technique is however sensitive to camera motion [4] and noise since each pixel is evaluated as a single entity.

Pixel-based techniques can however be made more robust by employing statistical methods like the mean and standard deviation of the pixels' intensity in order to ascertain a measure of variance between frames. This robustness proved to be more tolerant of noise, but at the cost of processing time due to the complexity of the statistical formulas.

Histogram-based methods: Histogram based SBD techniques apply the same basic principles as pixel-based techniques, but now the pixels are grouped into bins.

The most commonly used metric for shot boundary detection is the histogram difference between frames. Histograms are drawn of each frame by counting pixels with certain values based upon the number of bins chosen - typically 256 bins to quantize the image. Monochrome images deliver a single histogram per frame where color images require 3 such histograms.

The difference between these consecutive frames' histograms gives an indication of the frame-to-frame content change. Shot boundaries are generally associated with high rates of content change. The linear combination of these inter-frame differences are then used as a measure of the total content change across the frames [5].

As with the pixel-based techniques, the aforementioned histogram differences has to be compared against a threshold to indicate a shot boundary while ruling out expected inter-frame differences brought on by object or camera movement. Irrespective of the method employed, the accuracy thereof will be directly related to the accuracy of the threshold used. Statistical techniques can once more be employed to analyse these histograms, but with the added benefit of greatly reduced amounts of data due to the frame being quantised by the 256 bin histogram. Some of these techniques include the Jensen-Shannon Divergence (JSD), Chi-squared (χ^2) comparison [6] and various other techniques.

A previous investigation into the digital video shot boundary detection revealed the usefulness of the histogram based Jensen-Shannon divergence as a technique for detecting shot boundaries [7]. Some of the key points that were revealed by the aforementioned investigation have direct implications for application to streaming media.

III. Methodology

A. Analysis Platform

An analysis platform was created in Visual Basic that allows various video formats to be analysed. At the core of this analysis platform is EMGU CV (version 2.3.0.1416). EMGU CV is a cross platform .Net wrapper for OpenCV (Open source computer vision) which is a library consisting of a multitude programming functions aimed at computer-vision applications [8].

The developed analysis platform loads a video and applies the algorithm currently being evaluated to a sequential stream of video frames. Once the frames have been evaluated, they are discarded to free up memory.

B. Analysis setup

A high-level representation of the analysis setup is presented in figure 1.

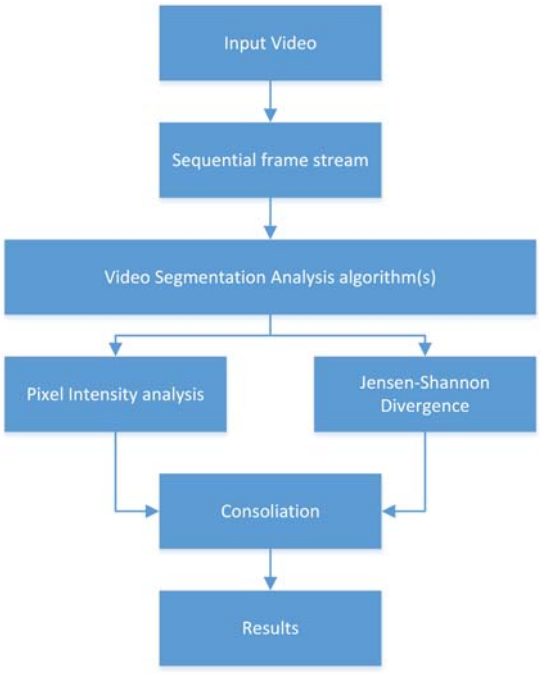


Fig. 1. High-level flow chart of the analysis setup

Lienhart et al. performed a comparison of some automatic shot boundary detection algorithms in [9]. The main focus of the algorithms in this comparison was the accuracy thereof with no mention of the processing speed or duration. There are some similarities between those techniques and the techniques employed in this analysis. One of these alternative techniques were implemented alongside the JSD technique.

1) Pixel Intensity Analysis: Standard deviation σ is a commonly used statistical metric to quantify the dispersion of a set of data values relative to the mean thereof. Instead of calculating the pixel-wise difference between the frames, each frame is analysed individually at first. The first step is to calculate the mean pixel intensity μ_F of the monochrome frame F:

$$\mu_F = \frac{1}{N}\sum_{i=1}^{N} p_i \tag{2}$$

where N is the number of pixels in the frame. It is important to note that since the frame constitutes the whole population and not merely a sample, the numerator N is used and not $N-1$ as would be the case for a sample. Subsequently the standard deviation of the frame is calculated by determining the *distance* of each pixel p from the mean intensity:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - \mu_F)^2} \tag{3}$$

2) Transition classification: In order to detect and possibly classify various types of shot boundaries encountered in a video stream, the general transition types had to be characterised. Test sequences were supplied to the analysis platform in order to generate algorithm results. The output of the various techniques were compared with each other in order to determine the transition specific behaviour.

As expected the analysis of the hard-cut transition results in a *clean* discontinuity for both the JSD as well as the SDPI as illustrated in figure 2. Furthermore the amplitude of the discontinuity greatly exceeds that of the surrounding values i.e. producing a great *signal-to-noise* ratio.

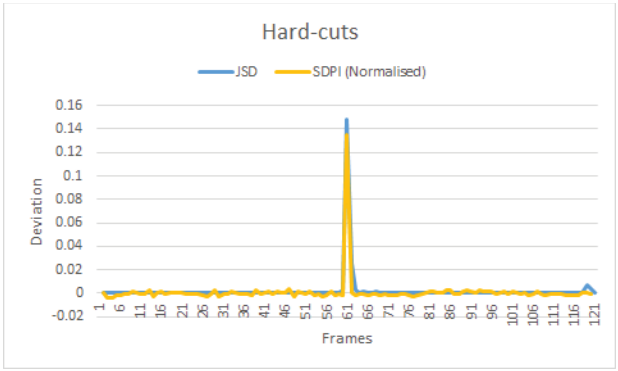


Fig. 2. Frame deviation results for a hard-cut transition

This is however not the case for gradual transitions. The same analysis of the cross-dissolve feature as seen in figure 3, resulted in multiple discontinuities. The amplitude of these discontinuities are still greater than the surrounding values, however this does pose a challenge in the sense of identifying the start and end of the transition amongst the noise.

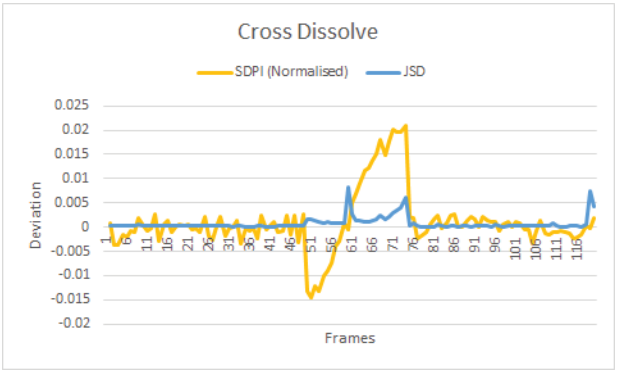


Fig. 3. Frame deviation results for a cross-dissolve transition

In certain instances even the JSD algorithm results in multiple peaks that constitute the transition as seen in figure 4. The dip-to-back transition is essentially a fade to black followed by a fade from black. Although this transition is visually observed as a single entity, it is detected as two separate JSD spikes due to the black frame(s) between the fades.

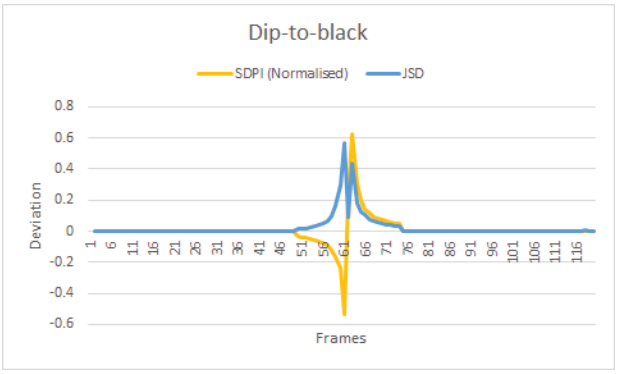


Fig. 4. Frame deviation results for a dip-to-black transition

The variable nature of the transitions requires an interval analysis rather than an instantaneous analysis. This calls for some assumptions to be made in the algorithm. Transitions are implemented to smooth the visual discontinuity as observed by the user. A common frame-rate encountered in digital video is 25 frames per second (FPS) and more. The human eye perceives motion as being fluid at about 18FPS for typical cinematic films because of its blurring.

By means of experimental observation it was found that in order to preserve the fluidity during a transition, it can be assumed that the duration of the transition will be at least $\frac{1}{3}$ of the video's frame rate. Shorter durations are perceived as a *flash* due to the abrupt change. This implies that for the test videos, this duration $D_{TransMin}$ is

$$D_{TransMin} = \frac{FPS}{3} = \frac{29}{3} \approx 10 \tag{4}$$

TABLE I
COMPOSITE VIDEO TRANSITION INDICES

TRANSITION TYPE	*TRANSITION INDICES*		
	START	*SHOT BOUNDARY*	*END*
Hard-cut	-	60	-
Cross-Dissolve	109	120	132
Dip to Black	169	180	193
Iris Box	229	240	253
Slide	289	300	313

frames. Inversely the maximum duration of a transition has to be stipulated as well. Without this parameter, the inherent ever-changing nature of a video will cause the boundary detection to be open ended - i.e. a time-lapse video of a sunset spanning multiple minutes can be misinterpreted as an extremely long fade to black. This maximum transition duration $D_{TransMax}$ is set at the frame rate of the video implying a maximum transition duration of 1 second which is

$$D_{TransMax} \approx 30 \tag{5}$$

frames. The length of this duration will be perceived as a lag of the same length if the analysis can be performed in *real-time*.

3) Transition properties: A common characteristic of these transitions is an extreme local minimum and maximum within the duration $D_{TransMax}$. Theoretically the center of the shot boundary should be located between these two points. When analysing a composite video with a structure as detailed in table I, the threshold problem becomes apparent as seen in figure 5. The amplitude of the transition indicators for both the JSD as well as SDPI is dependent on the type of transition as well as the composition of the sequential frames being analysed.

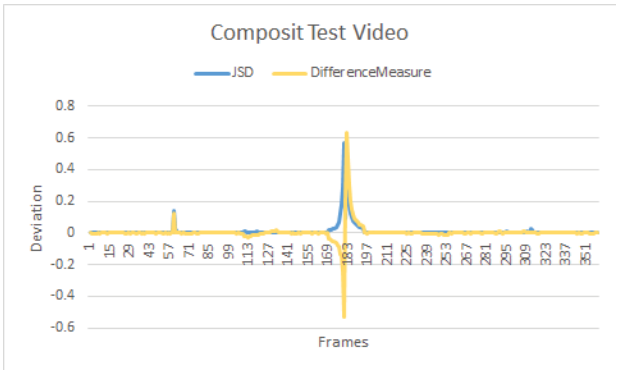


Fig. 5. Composite test video analysis

It was observed that the JSD technique as previously implemented in [7] with a average threshold of 3.5 and minimum threshold of 0.02, was able to detect the hard-cut at frame 60 and the centre of the dip-to-black transition at frame 180 while producing no positives on the other transitions. When the minimum threshold was lowered to 0.005, the JSD algorithm was able to detect the deviation caused by the various transitions, but it also identified multiple false positives.

The greatest advantage of using the JSD technique is the fast processing speed. This is however contrasted with the inability to detect multiple soft transitions. When utilising a lower threshold, the deviations caused by the soft transitions are detected alongside the false positives. Even though the deviations are detected, the start and end of the transition that caused the deviation is still unknown.

The implementation of the standard deviation of pixel intensities should allow for the detection of the soft transitions present in the video but at the cost of processing speed. Thus by combining the processing speed of the JSD technique with the detection capabilities of the SDPI technique, a hybrid technique is formed that should be able to detect the hard and soft transitions while keeping processing time to a minimum.

This is accomplished by lowering the threshold of the JSD technique, to detect the deviations caused by possible boundaries. Even though it will be prone to more false positives, these possible boundary indicators are utilised as landmarks where the SDPI technique can be implemented.

The landmarks generated by the JSD technique can be located anywhere from the start of the transition the end thereof. This poses a problem if only preceding or subsequent frames are evaluated. To overcome this problem an overlapping window of frames can be analysed by using a buffer of frames with duration D_{SDPI}

$$D_{SDPI} = D_{TransMax} \times 2. \tag{6}$$

The rational behind the D_{SDPI} is to serve as a safety measure to ensure the whole possible transition is identified in the event that the JSD algorithm only triggered on the trailing edge of the transition. Various flags will be implemented to ensure that the process will not be repeated if a JSD trigger is encountered within the next $D_{TransMin}$.

C. Data

A diverse collection of videos were chosen to evaluate the video segmentation analysis program. These videos encompasses various transitional effects to test the functionality. The test videos used in the transition classification section were created in Adobe Premier where the same two shots were joined using various transition techniques as seen in figure 6.

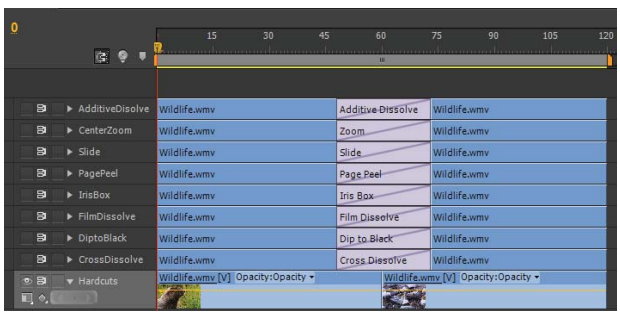


Fig. 6. Test video structure in Adobe Premier

These individual videos were used to characterise the shot boundary types. A second composition video containing multiple transitions as listed in table I was used to evaluate the behaviour of the algorithm.

D. Evaluation Metrics

Various metrics were employed to evaluate the effectiveness of the various algorithms that were evaluated in the analysis platform.

1) Recall: Recall rates provides a ratio of the number of relevant shot boundaries (correct detection) to the total number of relevant shot boundaries in the video as expressed in equation 7.

$$Recall = \frac{Correct}{Correct + Missed} \tag{7}$$

2) Precision: The precision rates of the algorithm provides the ratio of relevant shot boundaries to the total number of irrelevant shot boundaries (false detections) retrieved as expressed in equation 8 [10].

$$Precision = \frac{Correct}{Correct + FalsePositive} \tag{8}$$

3) Execution time: The execution time of each technique will be recorded alongside the recall and precision rates thereof. This will help to justify a trade-off analysis between the techniques in terms of accuracy versus the execution time required. This becomes particularly important for *"real-time"* applications.

The execution speed includes the following:

- Opening the video and reading the frames as it is required;
- Converting from RGB to monochrome where required.

IV. Results

A. Transition classification results

During the initial type classification of the different transitions, the characteristics of the hard-cut, cross-dissolve and dip-to-black transitions were derived.

The hard-cut transition was found to be the only consistent transition as it produced a singular peak with a high signal to noise ratio. The cross-dissolve transition resembles a reversed-N for the test video. This form is however drastically affected by the duration and *intensity* of the fade. During the longer fades, this cross-dissolve transition waveform becomes elongated with lower extrema that is more centred and thus starting to resemble a cosine wave form.

The dip-to-black transition waveform is characterised by a maximum and minimum with a high signal to noise ratio. Furthermore the number of frames between these extrema are very few. Unfortunately this duration is once again dependent on the duration that the scene stays black. If the frames stay black for longer than $D_{TransMax}$, it will be evaluated as two separate transitions - i.e. fade to- and fade from black.

Some additional transition types were evaluated to see how the algorithm will cope with them. These include the slide transition and the iris-box transitions with the results illustrated in figures 7 and 8.

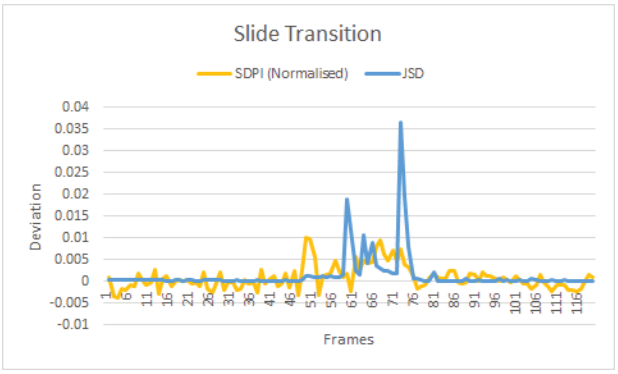


Fig. 7. Frame deviation results for a slide transition

These two transitions are achieved by moving one frame over the other for the slide transition while the iris-box transition incrementally expands the next sequence from the centre of the current frame. It is apparent from these figures that there is no discernible characterising features of these techniques other than two JSD extremes for the slide transition while the iris box produces a noisy gradient with a high signal to noise ratio and finally culminating in an abrupt return.

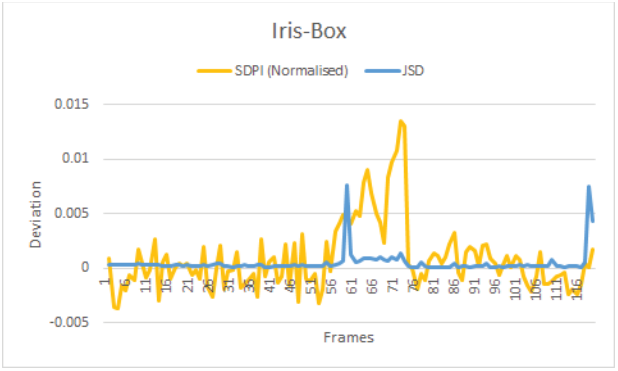


Fig. 8. Frame deviation results for an iris-box transition

B. Video segmentation analysis results

The application of the stand-alone JSD algorithm on the test video detailed in table I, confirmed the initial suspicions regarding the detection of soft transitions. The results of the JSD with a threshold of 0.02 as well as 0.005 is tabulated in table II alongside the actual boundaries. The lower minimum threshold allowed all the boundary transitions to produce deviation landmarks.

The hybrid JSD-SDPI algorithm with the lower minimum threshold proved to be successful when implemented on the composite test sequence. The results of this analysis is tabulated in table III. The JSD algorithm flagged all the possible boundaries, allowing the SDPI algorithm to evaluate the section for the transition boundaries.

TABLE II
JENSEN-SHANNON DIVERGENCE ANALYSIS RESULTS

TRANSITION TYPE	*TRANSITION INDICES*			*JSD*	
	START	*SB*	*END*	*0.02*	*0.005*
Hard-cut	-	60	-	61	60,61
Cross-Dissolve	109	120	132	N/D	110
Dip to Black	169	180	193	181	170,171,181
Iris Box	229	240	253	N/D	237
Slide	289	300	313	N/D	296,308

TABLE III
COMPOSITE VIDEO TRANSITION ANALYSIS RESULTS

TRANSITION TYPE	*TRANSITION INDICES*			*FRAME OFFSET*		
	START	*SB*	*END*	*START*	*SB*	*END*
Hard-cut	-	60	-	0	0	0
Cross-Dissolve	112	122	132	3	2	0
Dip to Black	180	181	182	11	1	-11
Iris Box	234	244	253	5	4	0
Slide	289	290	292	0	-10	313

The detected boundaries of the transition do not coincide perfectly with the actual boundaries. The frame offset differences listed in table III show that for the majority of the transitions, the actual shot boundary was detected within a few frames while the start and end boundaries were identified closer to the centre than the actual positions.

With the exception of the slide transition, recall and precision rates of 100% were obtained. If the detected boundaries are coupled with a safety margin of e.g. the maximum transition duration, with the centre located at the detected SB location, all the transitions will be encapsulated.

The execution times of the algorithms are shown in table IV. It is clear that the *real-time* criterion has been satisfied.

TABLE IV
EXECUTION TIMES

ALGORITHM	*PROCESSING TIME*	*VIDEO DURATION*
JSD	4.101 s	12 s
SDPI	4.609 s	

V. CONCLUSION AND RECOMMENDATION

The shortcomings of the JSD as a standalone video segmentation analysis technique was confirmed during the analysis. The JSD produced multiple false positives during the gradual transitions.

The analysis duration of a pure SDPI analysis proved to be greater than the duration of the video being analysed, hence failing the *real-time* criterion.

The hybrid algorithm that utilises the JSD algorithm as a point of interest marker for a sectional analysis by the SDPI algorithm proved to be very successful. Not only was the processing time less than the duration of the video being analysed, it produced nearly perfect recall and precision rates on the test video.

It is recommended that a safety margin be added to the detected shot boundary to account for the slight offsets in the detected boundaries. Based on the previous results, a safety margin of length equal to $D_{TransMin}$ will ensure that the boundaries of the transitions are adequately quantised.

Future work entails incorporation of the boundary form classification into the detection algorithm in order to help classify the detected boundary transition type and in doing so provide more accurate transition indices.

ACKNOWLEDGEMENT

I would like to thank my supervisor, Prof. W.C. Venter for his help and guidance throughout the duration of this investigation as well as the financial support from Prof. A. Hoffman.

REFERENCES

[1] G. J. Conklin, G. S. Greenbaum, K. O. Lillevold, A. F. Lippman, and Y. A. Reznik, "Video coding for streaming media delivery on the internet," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 11, no. 3, pp. 269–281, 2001.

[2] N. Patel and I. Sethi, "Video shot detection and characterization for video databases," *Pattern Recognition*, vol. 30, no. 4, pp. 583–592, 1997.

[3] I. Koprinska and S. Carrato, "Temporal video segmentation: A survey," *Signal processing: Image communication*, vol. 16, no. 5, pp. 477–500, 2001.

[4] J. S. Boreczky and L. A. Rowe, "Comparison of video shot boundary detection techniques," *Journal of Electronic Imaging*, vol. 5, no. 2, pp. 122–128, 1996.

[5] M. R. Naphade, R. Mehrotra, A. M. Ferman, J. Warnick, T. S. Huang, and A. M. Tekalp, "A high-performance shot boundary detection algorithm using multiple cues," in *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, vol. 1. IEEE, 1998, pp. 884–887.

[6] I. Williams, N. Bowring, and D. Svoboda, "A performance evaluation of statistical tests for edge detection in textured images," *Computer Vision and Image Understanding*, vol. 122, pp. 115–130, 2014.

[7] M. de Klerk, "Digital video shot boundary detector investigation," in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2014*. http://www.satnac.org.za/: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2014, pp. 105–110.

[8] Open source computer vision library. [Online]. Available: http://opencv.org/about.html

[9] R. W. Lienhart, "Comparison of automatic shot boundary detection algorithms," in *Electronic Imaging'99*. International Society for Optics and Photonics, 1998, pp. 290–301.

[10] V. Raghavan, P. Bollmann, and G. S. Jung, "A critical investigation of recall and precision as measures of retrieval system performance," *ACM Transactions on Information Systems (TOIS)*, vol. 7, no. 3, pp. 205–229, 1989.

M.G. de Klerk received his M.Eng. degree on renewable energy simulations in 2013 and is currently studying towards a PhD degree in Computer and Electronic Engineering at the North-West University with a focus on the optimisation of video detection techniques.

C.3 SATNAC 2016

Resizing-effects analysis on video fingerprinting for use in video recognition

[1]M.G. De Klerk*, [2]W.C. Venter*, [3]A.J. Hoffman*
*School of Electrical, Electronic and Computer Engineering, North-West University
[1]20555466@nwu.ac.za
[2]Willie.Venter@nwu.ac.za
[3]Alwyn.Hoffman@nwu.ac.za

***Abstract*—The advent of the digital era brought forth a multitude of advancements in terms of quality, availability and variety of videos. There consequently exits a need for tools to effectively manage all these files. One aspect of such a management tool is video recognition. These files may have different sizes and aspect ratios which can negatively effect the detection thereof. The video recognition algorithm presented in this paper was developed for advertisement tracking in streaming digital media. This paper investigates the effects of frame resizing on the hash generation for video fingerprinting and proposes relational parameters to support video scalability.**

***Index Terms*—SIFT, video recognition, video resizing, hash-generation**

I. Introduction

The term *video detection* is commonly misused in discussions when referring to the process by which an unknown video is identified. This process is more aptly called *video recognition* whereas *video detection* is more relevant to the process where the content of the video is analysed to e.g. track object motion.

Video content analysis is the capability by which a video is analysed to detect the temporal and spatial events present therein. The process of video recognition is inherently more associated with the spatial aspect as this is the source of the *fingerprint*.

The conceptual idea behind video recognition is to compare an unknown video against known videos and return the name of the video if a match is found. In signal theory this idea is accomplished by means of cross-correlation where the unknown series is compared against a known series as a function of the lag of one relative to the other. It is apparent that this method can not be effectively used to compare videos in this manner as it will not be practical, merely due to the execution time, not to mention the introduction of video distortions into the mix.

The ideal solution would be to effectively compare an unknown video against as many known videos as possible, within the least amount of time while producing accurate results. One such a concept has been presented by Moolman et al. in [1]. The work presented in this paper is a continuation of the is concept with the aim of identifying some of its limitations in terms of the input videos. These limitations are discussed alongside the basic overview of the techniques employed in section II.

II. Background

The streaming media classifier is used to describe the method of data delivery. The conventional method of video delivery has progressed from physical media such as film and DVD, to the digital delivery methods where video material can be downloaded. This digital download initially required the whole file to be downloaded before it could be accessed. but have since evolved to include a digital equivalent of traditional television broadcasts where the data is sequentially received, displayed and discarded. There are various methods employed to reduce the transmission cost such as the H.264 motion compensating encoding standard [2]. In order to focus on the evaluation of the techniques itself and not the encoding scheme, streaming media will be defined for the purpose of this paper as follows:

Streaming digital media. *Streaming digital media is a media delivery method where the video frames are sequentially supplied to the client where they can be viewed or analysed and then discarded.*

Following this definition, the digital stream can be viewed as a *digital pipe* that delivers content in the order it was sent. Since this *digital pipe* merely serves as the delivery method, the data contained therein can vary. One such a variation that one might encounter in terms of digital video, is video with an aspect ratio different that what was expected. More information regarding the aspect ratio is discussed in subsection III-A.

In order to analyse streaming digital media, any and all analyses must be done in real-time, implying that all calculations on the data must require less time than the actual playback duration thereof [3]. This implies that the video recognition algorithm must fingerprint the data, compare it to the known video database and return an answer in less time than it would take to watch the video. The most time consuming of the aforementioned tasks is the fingerprinting of the video frames.

A. Video Fingerprinting

The process of fingerprinting is a well established practise where an imprint of a persons' fingers is taken. The patterns found on each finger is then analysed to isolate distinguishing features thereof. The combination of these distinguishing features is referred to as a fingerprint.

The same concept of fingerprinting can be applied to digital images. The patterns present in a digital frame is however more complex than the arches, loops and whorls created by the friction ridges on a finger, since the pattern generated by the pixels can be an infinite number of things.

This calls for a different approach to identify distinguishing features from a digital image. Multiple such techniques have been developed, such as the Scale Invariant Feature Transform (SIFT) [4], [5] and Speeded-Up Robust Features (SURF).

During this investigation the SIFT algorithm is employed for distinguishing feature extraction as initially implemented by Moolman et al. [1]. These extracted features are called key points (KP). The focus will however be oriented more towards the utilisation of the KP-data generated by the SIFT algorithm than the generation thereof.

Hash Generation: The SIFT algorithm analyses a digital image and detects KP within the image that meet prescribed classifying criterion. Each of these detected KPs contain the following properties:

- X-Coordinate x_i
- Y-Coordinate y_i
- Key point size $\overline{kp_i}$
- Key point direction α_i

Not only is the extraction of KPs important, but the representation thereof also has an impact on robustness and performance [6]. In some cases, the KPs are matched between images or to KP-database by minimizing the distance between the descriptors [7]. Unfortunately by using only the distance as a metric, the uniqueness of the *fingerprint* is fairly weak. This is overcome by combining the aforementioned KP properties in order to generate four unique descriptors for the collection of KPs:

$$KP = \{kp_1, kp_2, \ldots, kp_{n-1}, kp_n\}, \quad n \in \mathbb{Z} \tag{1}$$

where each key point is indexed based on its size while ordered in a declining manner so that the following constraint holds true:

$$\overline{kp}_n \geq \overline{kp}_{n+1} \tag{2}$$

for all KPs in the frame. The accuracy of the hash used is influenced by the bits used to represent it. Since this is a user-variable parameter, the affected equations are expressed as a function of the bit resolution $\mathcal{B}_\mathcal{R}$.

1) Relative Magnitude (λ): The relative magnitude λ represents the magnitude of one KP to another. Due to the constraint expressed in equation 2, the maximum bits used to represent the relative magnitude will not overrun the $\mathcal{B}_\mathcal{R}$. This relative magnitude is calculated by:

$$\lambda = \frac{\overline{kp}_{n+1}}{\overline{kp}_n} \times \mathcal{B}_\mathcal{R} \tag{3}$$

where the bit ratio is expressed as a function of the number of bits used per hash descriptor $\mathcal{B}_\mathcal{H}$:

$$\mathcal{B}_\mathcal{R} = \mathcal{B}_\mathcal{H} \times 4. \tag{4}$$

2) Relative Direction (ω): The relative direction is the resulting angular component of the difference vector calculated from the individual KP-directions. These KP-directions are indicated by green arrows as illustrated in figure 1.

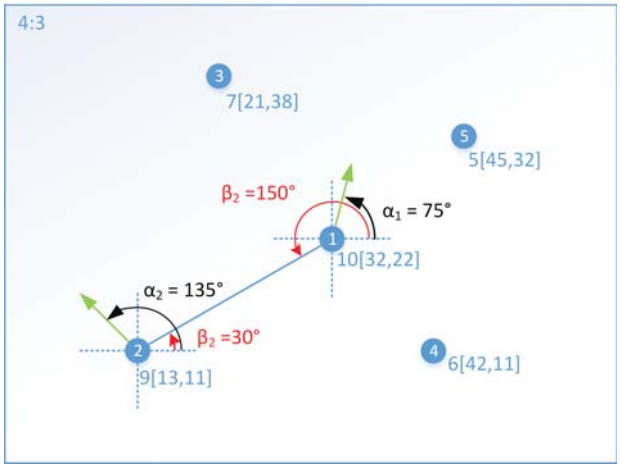


Fig. 1. Frame key point example

The difference in angular direction α_{diff} is calculated by

$$\alpha_{diff} = \alpha_{n+1} - \alpha_n \tag{5}$$

which allows the relative direction ω to be expressed as a function of the $\mathcal{B}_\mathcal{R}$:

$$\omega = \alpha_{diff} \times \frac{\mathcal{B}_\mathcal{R}}{2\pi}. \tag{6}$$

3) Key point Distance (Ω): The KP distance relates the position of the following KP_{i+1} to the current KP_i. The distance Δ between the key points is calculated from the difference in the $[X;Y]$ coordinates of the KP:

$$x_{diff} = x_{n+1} - x_n \tag{7}$$

$$y_{diff} = y_{n+1} - y_n \tag{8}$$

with the

$$\Delta = \sqrt{x_{diff}^2 + y_{diff}^2}. \tag{9}$$

The native implementation by Moolman et al. expressed this distance as a function of two hash generation constraints R_{min} and R_{max}. The aforementioned constraints are used to describe the minimum R_{min} and maximum R_{max} distance the succeeding KP may be from the parent, and still be related to it. If the KP falls outside of the range $R_{max} \geq \Delta \geq R_{min}$, none of the 4 hash parameters will be generated using this KP-pair, and the hash-generation moves on to the next KP-pair.

If these constraints are satisfied, the key point distance Ω was natively expressed as:

$$\Omega = \frac{\Delta - R_{min}}{R_{max} - R_{min}} \times \mathcal{B}_\mathcal{R}. \tag{10}$$

TABLE I
NATIVE IMPLEMENTATION PARAMETERS

Fingerprinting Parameters	
Frame width	320
Frame height	240
R_{max}	100
R_{min}	10
$\mathcal{B}_{\mathcal{H}}$	4
$\mathcal{B}_{\mathcal{R}}$	16

4) Key point Direction (θ): The last hash parameter relates the relative direction between the KP pair. The traditional inverse tangent function, Arctan $atan(\alpha)$ or commonly written as $tan^{-1}(\alpha)$, has been replaced with the more comprehensive inverse tangent function Arctan2 $atan2(\alpha)$. This allows the relative direction to be calculated and expressed in all 4 quadrants of the Cartesian coordinate system since $atan$ cannot distinguish polar opposite directions. This allows the relative KP direction θ to be calculated as:

$$\theta = \frac{-atan2\left(x_{diff}, y_{diff}\right) - \alpha_n}{2\pi} \times \mathcal{B}_{\mathcal{R}}. \tag{11}$$

Now that the 4 unique descriptors have been generated, they can be combined to form a singular representative descriptor called a hash. This resulting hash value $\#$ is created by *typecasting* the value of each parameter:

$$\bar{\bar{\lambda}} = CByte(\lambda, \mathcal{B}_{\mathcal{H}}) \tag{12}$$

to the specified number of bits per hash $\mathcal{B}_{\mathcal{H}}$ where the double over-line annotation $\bar{\bar{}}$ is used for the bit representations. These *typecast* values are then coalesced $\&$ into a single hash:

$$\bar{\bar{\#}} = [\ \bar{\bar{\lambda}}\ \&\ \bar{\bar{\omega}}\ \&\ \bar{\bar{\Omega}}\ \&\ \bar{\bar{\theta}}\ ]. \tag{13}$$

Finally this hash value is *typecast* back and the numerical value utilised as the hash-matrix index in the storage system.

B. Problem Definition

As indicated by Moolman et.al., the native implementation of this technique proved to be effective under certain test conditions. In these test cases, input video files for both fingerprinting and detection instances, were resized to a fixed resolution and analysed using the parameters as tabulated in table I.

Further analysis indicated that the scale invariant nature of the SIFT algorithm is not fully inherited by the native implementation's hash as the KP distance proved to be negatively impacted by frame resizing.

The problem to be addressed in this article pertains to the effect that resizing has on the reproducibility of hash values when using varying input resolutions.

III. METHODOLOGY

In order to address the scalability problem exhibited by the native approach, the origin of the error had to be determined. Logic dictates that the KP-distance Ω should be calculated as a factor of the actual frame dimensions hence, becoming a scalable descriptor. This is a fairly simple solution when the aspect ratio of the frames remain constant.

A. Aspect Ratio

Unfortunately the aspect ratio does not always remain constant, especially when a fixed input video resolution is defined, effectively forcing input videos to be distorted. There are 3 types of aspect ratios used in literature:

- *Pixel aspect ratio (PAR)*: Ratio of the actual pixels' dimensions. Generally square pixels are used with a $PAR = 1:1$. Any distortion in PAR only affects the perceived image by the user, but the SAR remains the same.
- *Storage aspect ratio (SAR)*: This ratio is calculated from the amount of pixels from which the image is comprised, i.e. the horizontal and vertical pixel counts.
- *Display aspect ratio (DAR)*: This aspect ratio is a combination of $DAR = SAR \times PAR$ and describes how an image will be perceived.

The ambiguous term *aspect ratio* generally refer to the SAR as is the case in this paper. Another related ambiguous term is *resize*. In general usage it refers to the scaling of an image while retaining the SAR. When the scaling results in a change of SAR, the process is more aptly referred to as size distortion, but to prevent confusion with the optical aberration techniques (e.g. barrel distortion), the term *remap* is used. This resizing and remapping is implemented using a technique called bilinear interpolation.

B. Bilinear Interpolation

Bilinear interpolation is an image re-sampling technique that is generally used to enlarge images by creating extra pixels with values derived from the surrounding pixels. Contrary to the term *linear* in the name, the technique itself is not linear, but rather quadratic. This quadratic nature creates a *smoothing* effect which might soften local valleys and peaks a bit during enlargement extrapolation. This technique can however also be applied to re-sample an image to a smaller version. The aforementioned *smoothing* effect is not present during the down-sampling process, but may actually result in some aliasing effects [8]. Spatial aliasing could impact the the key point detection of the SIFT algorith, but this can be negated by applying a Gaussian smoothing filter [9].

The remapping of a frame F with M horizontal and N vertical pixels results in a remapped frame $F'(M' \times N')$. This is done by obtaining the weighted sum of the four nearest neighbours' pixel values as illustrated in figure 2. The values of the pixel X at coordinates $[m_f; n_f]$ is calculated by the parametric equations of the straight line segments running from $A \Rightarrow B$ and $D \Rightarrow C$ [10]:

$$X_{(m_f, n_f)} = aA + bB + cC + dD \tag{14}$$

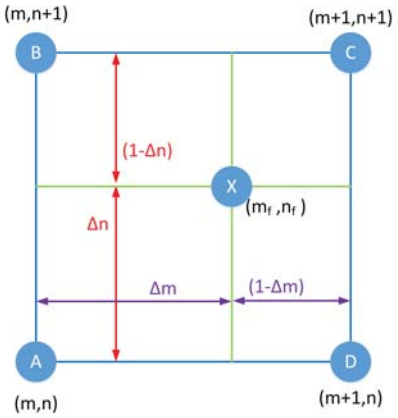


Fig. 2. Bilinear interpolation addressing schematic

where the weighted values for each neighbour is [11]:

$$a = (1 - \Delta m)(1 - \Delta n)$$
$$b = (1 - \Delta m)\Delta n$$
$$c = (\Delta m \Delta n)$$
$$d = (1 - \Delta n)\Delta m.$$

Thus the remapping of an image, can alter the SAR. Although the nearest-neighbour weighted value might have a smoothing effect when remapping to a larger image, this effect is global. This is similar to applying a Gaussian filter to the image. In extreme cases this might reduce the magnitude of the KP. If the remapping results in a change in SAR, not only might the KP size be affected, but the other KP-metrics as well.

By understanding how the frame remapping works, the resulting change in SAR can be quantized and the effects thereof on parameters such as the KP distance Ω analysed.

C. Modified key point distance

Even when the SAR is maintained during resizing, the KP-distance Ω hash descriptor, as expressed in equation 10, is found to be scale-variant. When a frame is resized, the Ω will be different for every resize factor RF where $RF \in \mathbb{R}$. This RF-scalar is used to represent the amount by which the horizontal and vertical pixel count is changed.

The best approach to deal with this limitation is to express the modified KP-distance $\mathring{\Omega}$ as a function of the SAR with the help of a scalar κ producing

$$\mathring{\Omega} = \kappa\Omega \times \mathcal{B}_{\mathcal{R}}. \tag{15}$$

A relation of Ω to R_{max} and R_{min} can be retained if these radii is related to the SAR.

D. Radii parameter relation

The native analysis called for the user to hardcode the boundary radii R_{min} and R_{max} used for KP-pairing. Since the chosen values proved to be successful for the resolution of 320×240, this property was chosen as a starting reference to determine possible relations.

The logic behind the boundary radii is to improve the hash diversity by forcing KPs to form KP-pairs close to the origin KP. Analysis of the experimental value for $R_{max} = 100$ in the native implementation, shows that R_{max} can be expressed as a factor of the maximum frame distance:

$$\Delta_{max} = \sqrt{M^2 + N^2} \tag{16}$$

with

$$R_{max} = \frac{\Delta_{max}}{4} = \frac{\sqrt{320^2 + 240^2}}{4}. \tag{17}$$

This relation from equation 17 is graphically represented in figure 3 as an overlay of the radii on the example frame from figure 1, covering 81.8% of the frame area.

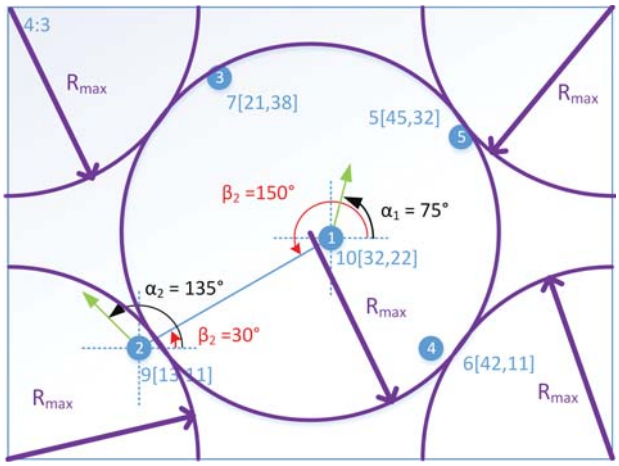


Fig. 3. Relative R_{max} representation $4:3$ aspect ratio

When applying the same relation of equation 17 on a frame with a commonly used aspect ratio of $16:9$, the graphical representation indicates that the radii provides better coverage (91.9%) of the frame as seen in figure 4, without any overlapping.

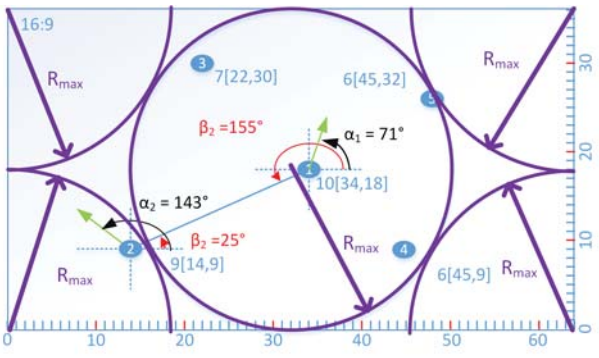


Fig. 4. Relative R_{max} representation on a $16:9$ aspect ratio

It is interesting to note that for the $16:9$ aspect ratio, R_{max} can be expressed as a function of the maximum frame distance as well as a function of the frame height. This relation is expressed as:

$$R_{max} = \frac{\Delta_{max}}{4} = \frac{N}{2} \tag{18}$$

where N is the frame height in pixels. By applying this height-relation to the $4:3$ aspect ratio frame, a better radii coverage is observed with minimum overlap as seen in figure 5.

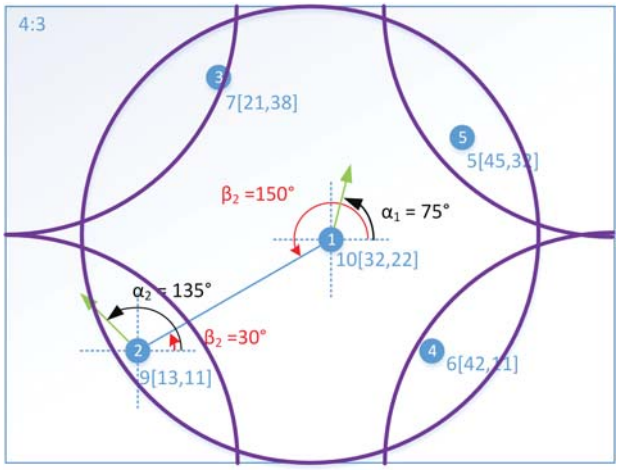


Fig. 5. Relative R_{max} representation as a function of the frame height on a $4:3$ aspect ratio

Similar to the native implementation, the relation of R_{min} to R_{max} is set at 10%:

$$R_{min} = \frac{R_{max}}{10}. \tag{19}$$

It is important to note that the radii displayed in the various figures merely represent the maximum KP-radii if the KP were to be situated on the four corners as well as the center of the frame.

E. Evaluation instances

The native video recognition algorithm was implemented on a series of test videos, once again remapped to a fixed frame size 320×240, in order to serve as a benchmark:

$$\Omega_{Benchmark} = \frac{\Delta - R_{min}}{R_{max} - R_{min}} \times \mathcal{B}_{\mathcal{R}}. \tag{20}$$

Subsequent analyses will test the implementation of the two radii parameter relations by calculating the modified KP-distances by using:

$$\Omega_{MaxDist} = \frac{\Delta - R_{min}}{\left(\frac{\Delta_{max}}{4}\right) - R_{min}} \times \mathcal{B}_{\mathcal{R}} \tag{21}$$

and

$$\Omega_{Height} = \frac{\Delta - R_{min}}{\left(\frac{N}{2}\right) - R_{min}} \times \mathcal{B}_{\mathcal{R}}. \tag{22}$$

IV. Results

All the analyses were performed on 5 sets of images with the resolutions tabulated in table II. In order to evaluate the resizing effects on the parameters, the evaluation points were set as one at the origin $P_1[0;0]$ and the other at the diagonally opposite corner $P_2[Width; Height]$. All the tabulated results that follow is expressed in terms of these resolutions.

TABLE II
Frame Set Resolutions

Frame Set	1	2	3	4	5
Width	320	640	1280	2560	5120
Height	240	480	960	1920	3840

TABLE III
Native implementation results

R_{max}	100	100	100	100	100
R_{min}	10	10	10	10	10
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	69	140	283	567	1136
θ	6	6	6	6	6

The first result set was obtained by analysing the native implementation. The results thereof is tabulated in table III from which it is clear to see that the only descriptor affected by the resizing is the KP-distance as was anticipated.

The KP-distance as expressed in equation 21 was evaluated with the results tabulated in table IV. It is evident that Ω is now scale invariant by using the related parameters. The height relation as expressed in equation 22 performed in a similar fashion, with results in table V.

It is important to note that even though these descriptors are now calculated as a relation to the frame size, the actual value exceeds the allowed size dictated by the $\mathcal{B}_{\mathcal{R}}$.

This was addressed by relating the distance to only the maximum distance

$$\Omega_{Relational} = \frac{\Delta}{\Delta_{max}} \times \mathcal{B}_{\mathcal{R}}. \tag{23}$$

This was done since the radii R_{min} and R_{max} are implemented in the algorithm as KP-eligibility criterion. The results of this relational KP-distance in equation 23 is tabulated in VI.

From this it is clear that the relational KP-distance in equation 23 is best suited as it is scale invariant while the SAR is maintained. To further substantiate this claim, it was implemented on two other points with the results tabulated in VII. From this it is evident that the KP-distance metric stays the same even when resizing has occurred.

TABLE IV
R_{max} as a function of Δ_{max} implementation results

R_{max}	100	200	400	800	1600
R_{min}	10	20	40	80	160
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	69	69	69	69	69
θ	6	6	6	6	6

TABLE V
R_{max} AS A FUNCTION OF THE FRAME HEIGHT IMPLEMENTATION RESULTS

R_{max}	120	240	480	960	1920
R_{min}	12	24	48	96	192
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	57	57	57	57	57
θ	6	6	6	6	6

TABLE VI
R_{max} AS A FUNCTION OF ONLY Δ_{max} IMPLEMENTATION RESULTS

R_{max}	80	160	320	640	1280
R_{min}	8	16	32	64	128
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	16	16	16	16	16
θ	6	6	6	6	6

TABLE VII
R_{max} AS A FUNCTION OF ONLY Δ_{max} IMPLEMENTATION RESULTS FOR POINTS P_1 AND P_2

P_1	[50;60]	[100;120]	[200;240]	[400;480]	[800;960]
P_2	[200;11]	[400;22]	[800;44]	[1600;88]	[3200;176]
λ	14	14	14	14	14
ω	8	8	8	8	8
Ω	6	6	6	6	6
θ	1	1	1	1	1

V. CONCLUSIONS AND RECOMMENDATIONS

The origin of the scalability abnormality from the native technique was in part attributed to the original calculation of the KP-distance as a function of the fixed radii parameters. Although these techniques worked well for the fixed frame size of 320×240, it did not fare as well with other frame sizes.

A relation was derived where the radii can be expressed as a function of the frame dimensions. This relation allows the radii to become scale invariant, but the incorporation thereof in the native implementation calculation as expressed in equation 10, does not allow for scalability.

Since the core function of the radii is to serve as KP eligibility criterion, they were omitted and a different relation was used as expressed in equation 23.

Although the hash descriptors are now scale invariant, they are however not distortion invariant. Remapping of the frames to a different SAR will incur errors. A further investigation is warranted to better relate the parameters with the aim to become distortion invariant.

Once the descriptors can be expressed as distortion invariant relations, the aforementioned radii expressions can be evaluated to determine if the slight overlap of caused by the height relation performs better or worse than the maximum distance relation.

ACKNOWLEDGEMENT

I would like to thank my supervisor, Prof. W.C. Venter for his help and guidance throughout the duration of this investigation as well as the financial support from Prof. A. Hoffman.

REFERENCES

[1] R. Moolman and W. Venter, "Video fingerprinting using robust hashing of scale invariant features of frames," in *Southern Africa Telecommunication Networks and Applications Conference (SATNAC)*. www.satnac.org.za: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2012.

[2] B. Juurlink, M. Alvarez-Mesa, C. C. Chi, A. Azevedo, C. Meenderinck, and A. Ramirez, "Understanding the application: An overview of the h. 264 standard," in *Scalable Parallel Programming Applied to H. 264/AVC Decoding*. Springer, 2012, pp. 5–15.

[3] A. H. MG de Klerk, W.C. Venter, "Automatic shot boundary detection for streaming digital video." http://www.satnac.org.za/proceedingsn.html: Southern Africa Telecommunication Networks and Applications Conference (SATNAC), 2015, pp. 219–224.

[4] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Ieee, 1999, pp. 1150–1157 vol.2.

[5] ——, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, nov 2004.

[6] R. Chellappa, A. Veeraraghavan, and G. Aggarwal, "Pattern recognition in video," in *Pattern Recognition and Machine Intelligence*. Springer, 2005, pp. 11–20.

[7] A. Kelman, M. Sofka, and C. V. Stewart, "Keypoint descriptors for matching across multiple image modalities and non-linear intensity variations," in *2007 IEEE conference on computer vision and pattern recognition*. IEEE, 2007, pp. 1–7.

[8] R. C. Gonzalez and R. E. Woods, "Digital image processing," 2002.

[9] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *Proceedings of the 2011 International Conference on Computer Vision*, ser. ICCV '11. Washington, DC, USA: IEEE Computer Society, 2011, pp. 2548–2555.

[10] D. Salomon and G. Motta, *Handbook of data compression*. Springer Science & Business Media, 2010.

[11] K. Gribbon, C. Johnston, and D. G. Bailey, "A real-time fpga implementation of a barrel distortion correction algorithm with bilinear interpolation," in *Image and Vision Computing New Zealand*, 2003, pp. 408–413.

M.G. de Klerk received his M.Eng. degree on renewable energy simulations in 2013 and is currently studying towards a PhD degree in Computer and Electronic Engineering at the North-West University with a focus on the optimisation of video detection techniques.

C.4 SAIEE 2017

South African Institute of Electrical Engineers

AFRICA RESEARCH JOURNAL

Date: 15 October 2017

Editor-in-Chief's decision from the SAIEE Africa Research Journal (ARJ)

Details of submitted paper:

Paper number/ID (our reference)	2017-703
Authors	M.G. De Klerk, W.C. Venter and A.J. Hoffman
Title	Parameter Analysis of the Jensen-Shannon Divergence for Shot Boundary Detection in Streaming Media Applications

Thank you for your submission to the SAIEE ARJ.

Your submission was coordinated with a specialist editor, who reached the decision. The decision is a consensus decision, derived as a result of feedback from at least two independent reviewers.

In summary, the recommendations from the reviewers were:

- Accept - minor changes as indicated
- Accept - no changes
- Accept - minor changes as indicated

Recommendation: The manuscript be accepted with minor changes, on condition that the authors implement the comments of all three reviewers as proposed in section.

☐ Professional proofreading is also required.

We would appreciate your resubmission within 30 days of receipt of this letter.

We would like to thank you for considering our journal, and would like to invite you for future considerations similarly.

Sincerely yours,

Prof Bea Lacquet

Editor-in-Chief, *SAIEE Africa Research Journal*
http://www.saiee.org.za/arj
University of the Witwatersrand
Private Bag 3 Wits 2050 South Africa

PARAMETER ANALYSIS OF THE JENSEN-SHANNON DIVERGENCE FOR SHOT BOUNDARY DETECTION IN STREAMING MEDIA APPLICATIONS

M.G. De Klerk* , **Prof. W.C. Venter†** and **Prof. A.J. Hoffman‡**

* *School of Electric, Electronic and Computer Engineering, North West University, Potchefstoom, South Africa E-mail: 20555466@nwu.ac.za*

† *School of Electric, Electronic and Computer Engineering, North West University, Potchefstoom, South Africa E-mail: Willie.Venter@nwu.ac.za*

‡ *School of Electric, Electronic and Computer Engineering, North West University, Potchefstoom, South Africa E-mail: Alwyn.Hoffman@nwu.ac.za*

Abstract: Shot boundary detection is an integral part of multimedia, be it video management or video processing. Multiple boundary detection techniques have been developed throughout the years, but are only applicable to very specific instances. The Jensen-Shannon divergence (JSD) is one such a technique that can be implemented to detect the shot boundaries in digital videos. This paper investigates the use of the JSD algorithm to detect shot boundaries in streaming media applications. Furthermore, the effects of the various parameters used by the JSD technique, on the accuracy of the detected boundaries, are quantified by the recall and precision metrics all the while keeping track of how they affect the execution time.

Key words: Jensen-Shannon Divergence; shot boundary detection, threshold parameters

1. INTRODUCTION

Throughout the years, humanity has made advances in every field of research and brought forth new technologies. One example of this technological progression is the advancement from film-based videos to the digital age where videos are no longer confined to film. Along with the physical progression between the media, the techniques associated with videos had to adapt as well.

One of the core techniques that is encountered when working with video analysis software; video storage and management systems; or the on-line video indexing systems, is called a shot boundary detector [4].

As the technology has progressed and processing capabilities became readily available, a multitude of techniques could be implemented without worrying too much about the processing requirements.

There are however still applications that require the same detection capabilities, but where processing time is of the utmost importance. A further constraint arises for these techniques if they are to be implemented on streaming digital media, as discussed in Section 2.2.

Due to the varying nature of videos, each available technique will perform differently based on the input data. In this investigation the sensitivity of one such technique, the Jensen-Shannon divergence, is evaluated for a generic set of test videos. The knowledge gained from this analysis provides a set of seed parameter values, as well as an understanding of the various parameters' effects on the algorithm when used in conjunction with streaming media.

2. BACKGROUND

In order to understand and appreciate the functionality of a shot boundary detector (SBD), one has to understand the basic underlying structure of video files. Automatic shot boundary detection plays a pivotal role in completing tasks such as video abstraction and keyframe selection [6]. The ambiguous term *video* has its origin from the Latin word *videre* meaning 'to see' combined with the English word *audio* which refers to the process of hearing, ultimately forming a word that describes a coalesced union of audio and visual material known as video. For the purpose of this investigation, the term video will semantically refer to the visual aspect thereof.

2.1 Video structure

A generic video file V can be defined as a collection of various smaller sections called shots s:

$$V = \{s_1, s_2, \ldots, s_{n-1}, s_n\}, \;\; n \in \mathbb{Z}. \tag{1}$$

These shots are defined as video segments that are visually contiguous and generally captured during a single take. This implies that although the visual content might be varying as is the nature of videos, the inter-frame variances ϕ should be small compared to the inter-shot variances Φ. The underlying structure of a shot consists of multiple sequential static frames f:

$$s = \{f_1, f_2, \ldots, f_{n-1}, f_n\}, \;\; n \in \mathbb{Z}. \tag{2}$$

Each of these static frames can be represented by a $M \times N$ matrix where each picture element (pixel) $p_{i,j}$

can be addressed by its relative position in the frame by its co-ordinates i and j. In colour frames, each pixels has a multi-chromaticity value (RGB (Red-Green-Blue) or CMYK (Cyan-Magenta-Yellow-Key (black)) color space) that defines the colour thereof. Similarly for grayscale frames, each pixel has a monochromatic value. This generic breakdown of the video structure is illustrated in Figure 1.

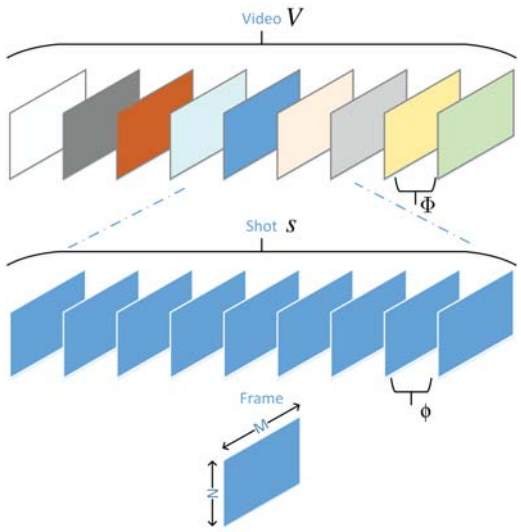


Figure 1: Video structure breakdown

2.2 *Shot Boundary Detection*

Human beings are able to detect the boundaries between various shots due to cognitive analysis. However, computers lack the cognitive analysis capabilities of humans and thus need to analyse the video in a different manner.

A shot boundary can be defined as the break in visual continuity between two sequential shot sequences. In the simplest form, this boundary can be depicted as two sequential frames that contain drastically different visual content. This break in visual continuity can thus be detected by measuring the inter-frame variance between sequential frames. When the inter-frame variance between two frames is sufficiently larger than the preceding inter-frame variances, it can be an indication of a shot boundary. This variance will then be called an inter-shot variance.

Following this simple premise, a computer can then analyse videos and based on the results thereof, infer possible shot boundary locations. There are however numerous contributing factors that hamper the detection of boundaries.

Transitions: The transition between shots describe the way in which the shot boundary is implemented. There are two main categories of transitions used in videos: abrupt and gradual transitions. In the simplest form, a shot transition can be described as the visual manifestation of an abrupt change in visual content, hence called an abrupt transition.

In order to soften the visual discontinuity caused by a shot boundary, techniques can be employed to alter the frames surrounding the shot boundary. A basic gradual transition technique is called fading where the opacity or luminosity of sequential frames in the one shot is gradually decreased while the inverse is done on the following shot. A common example of such a fade is commonly referred to a as fade-to-black where the current shot is faded to a black frame.

The progress of video editing techniques have brought forth multiple transition techniques to create visually appealing shot transitions, but which tend to complicate the automatic detection thereof. These include transitions like:

- additive dissolve;
- cross dissolve;
- fade-to-black and fade-from-black;
- zoom in or out;
- slide;
- page peal;
- iris box.

and many other interesting transitions.

With the exception of dissolves and fade-to-black or fade-from-black, the other transitions are not generally encountered in general video sources such as television programs and movies. It is more commonplace in advertisements or presentation videos.

Detection Methodologies: The simplest method that can be employed with the goal of detecting shot boundaries is the comparison of successive frames. This can be accomplished by comparing the value of each corresponding pixel in both frames and calculating the difference thereof. In this case the difference $D_{n,n+1}$ will be the aforementioned inter-frame variance $\phi_{n,n+1}$.

$$D_{n,n+1} = \sum_{i=1}^{N} \sum_{j=1}^{M} |f_{n+1}(p_{i,j}) - f_n(p_{i,j})| \tag{3}$$

If the total difference between the two frames are above a certain threshold τ, it might be possible that a shot boundary has been detected:

$$D_{n,n+1} = \begin{cases} \text{Possible Shot Boundary}, & \text{if } D_{n,n+1} \geq \tau \\ \text{No Boundary}, & \text{otherwise.} \end{cases} \tag{4}$$

It is easy to see how this pixel based method can become computationally expensive as well as very susceptible to noise since each pixel is evaluated as a singular entity

[19]. Alternatively the pixels can be analysed as groups of entities, allowing for a reduced impact due to noise and camera motion [9].

Lefévre et al. reviewed multiple video segmentation techniques in [11], concluding that inter-frame difference is indeed one of the fastest methods although it may be characterised by poor quality. On the other end of the spectrum are feature or motion based methodologies which are more robust, but are computationally expensive. Furthermore, some of the more robust techniques not only require the video as a whole to detect peaks in analysis outputs, but require training for the statistical learning methods. Since training can alter the output of the results depending on the training set used. Hence different instantiations could be subject to different results while using the same algorithm if trained using a different set. Thus the aforementioned arguments reinforce the notion to opt for a fast procedure based technique. One such a technique is the histogram based Jensen-Shannon divergence. A previous investigation by De Klerk et al. [14] revealed the viability and usefulness of this technique as a boundary detection algorithm.

Real-time: The concept of real-time is a fairly relative one. In this context the term does not refer to the validity of time, but rather to a relational expression. For the purpose of this investigation, the term real-time will be defined as the analysis time domain where the time required for all computations t, is equal to or preferably less than the actual playback duration $t_{duration}$ of the video being analysed

$$t_{calculations} \leq t_{duration}. \tag{5}$$

Another constraint imposed on the analysis techniques pertaining to the streaming media aspect, is that the analysis technique will only have historic data available to perform the analysis. This becomes a notable factor when calculating the aforementioned threshold, as some traditional techniques rely on a global threshold calculated from the whole composite video which is now unavailable.

2.3 Jensen-Shannon Divergence

The Jensen-Shannon Divergence algorithm provides a means to determine the inter-frame variance ϕ between consecutive frames. This in turn can be used to detect if these frames constitute a shot boundary. Thus, a generalised form of equation 4 can be given as:

$$\phi_{n,n+1} = \begin{cases} \text{Possible Shot Boundary,} & \text{if } \phi_{n,n+1} \geq \tau \\ \text{No Boundary,} & \text{otherwise.} \end{cases} \tag{6}$$

As the name suggests, the JSD algorithm is a combination of the Shannon's entropy and the Jensen inequality.

In 1984, Claude Shannon defined information measures for instance, mutual information and entropy. The Shannon entropy [17] is a method used in information theory to express the information content, or the diversity of the uncertainty of a single random variable, i.e. a measure of information choice and uncertainty [8].

The Shannon entropy function H of the probability distribution $P = (p_1, p_2, p_3, \ldots, p_n)$ consisting of n possibilities in the distribution is calculated by:

$$H(P) = -K \sum_{i=1}^{N} p_i \log_b p_i \tag{7}$$

where K is a positive constant [13] and b denoting the logarithmic base [16]. In essence, this entropy measure can be explained as the measure of uncertainty to predict which probability in occurrence will be encountered.

The other part of this technique relies on the Jensen's inequality measure as was proposed by Johan Jensen in 1905 in the paper *Sur les fonctions convexes et les inégalités entre les valeurs moyennes* [10]. Fundamentally, the Jensen-inequality states that if a given function g is convex on the range of a random variable Y, then the expected value $\mathbb{E}$ of the function will be greater than or equal to the function of the expected value:

$$\mathbb{E}[g(Y)] \geq g(\mathbb{E}[Y]) \tag{8}$$

where the variance will always be positive. This property can be illustrated by evaluating the Jensen-inequality for the convex function $g(y) = y^2$. This will always be positive since $\mathbb{E}\left[Y^2\right] - (\mathbb{E}[Y])^2 \geq 0, \quad \forall y \in Y$.

In order to understand the relevance of the Jensen-inequality to the functionality of boundary detection, a specialised version of the Shannon entropy is however required, namely relative entropy. Relative entropy is a further application of Shannon's entropy which can be used to quantify the distance between two probability distributions. This relative entropy, also referred to as the Kullback-Leibler distance, is denoted by $D_{KL}(p,q)$ and calculates the distance between two probability distributions p and q that are defined over the alphabet χ:

$$D_{KL}(p,q) = \sum_{x \in X} p(x) log \frac{p(x)}{q(x)} \tag{9}$$

where $x \in \chi$. By following the conventions, as used by [8] and [18] that

$$0 log\left(\frac{0}{0}\right) = 0 \text{ and } x log\left(\frac{x}{0}\right) = \infty \tag{10}$$

if $x > 0$, it becomes apparent from equation 9 that the relative entropy satisfies the information inequality or divergence:

$$D_{KL}(p,q) \geq 0 \tag{11}$$

if and only if $p = q$. This implies that only when both distributions are identical, the Kullback-Leibler distance can be zero. For all other instances it will be greater than zero. This information inequality or divergence, is sometimes referred to as the information divergence [8] and is later used with the Jensen inequality, hence becoming the Jensen divergence.

The integration of the entropy measures from Shannon with the inequality measure of Jensen, in effect, enables the measuring of the expected entropy of the probability, which will be greater than or equal to the entropy of the expected probability.

Thus, by combining Shannon's entropy with Jensen's inequality, one is left with a commonly used index of the diversity of a multinomial distribution. Consider a multinomial distribution $p = (p_1, \ldots, p_n)$, where each $p_i \geq 0$ and the total sum of the distribution is $\sum p_i = 1$. Burbea et al. expressed the concavity of the Shannon entropy provides a decomposition of the total diversity in a mixed distribution $\frac{(p+q)}{2}$ as :

$$H_n\left(\frac{p+q}{2}\right) = \frac{1}{2}\left[H_n(p) + H_n(q)\right] + \mathcal{J}_n(p,q) \qquad (12)$$

in [5]. The average diversity within the distributions is represented by the first component $\frac{1}{2}\left[H_n(p) + H_n(q)\right]$ in equation 12, while the latter component is called the Jensen difference.

The Jensen difference $\mathcal{J}_n(p,q)$ corresponds to the Shannon entropy $H_n(p)$ and can be expressed as:

$$\mathcal{J}_n(p,q) = H_n\left(\frac{p+q}{2}\right) - \frac{1}{2}\left[H_n(p) + H_n(q)\right]. \qquad (13)$$

The Jensen difference is non-negative and vanishes if $p = q$ and thus provides a natural measure of divergence between the two distributions [5] as with the relative entropy in equation 11.

Thus, by combining the entropy calculation with the Jensen inequality measure between two consecutive frames' histograms as derived by Qing Xu [20], the JSD equation is produced:

$$JSD(f_{i-1}, f_i) = H\left(\frac{P_{f_{i-1}} + P_{f_i}}{2}\right) - \frac{H\left(P_{f_{i-1}}\right) + H\left(P_{f_i}\right)}{2} \qquad (14)$$

where f_{i-1} is the previous frame and f_i the current frame. A measure is created as to how far the probabilities are from their likely joint source, equalling zero only if all the probabilities are equal. The probabilities of the frames are respectively $P_{f_{i-1}}$ and P_{f_i}. The aforementioned probabilities are calculated from the applicable frames' histograms as expressed in equation 15:

$$P(f_i) = \frac{Histogram(f_i)}{Height(f_i) \times Width(f_i)} \qquad (15)$$

where the histogram of each color component is divided by the number of pixels in that frame f_i. The total number of pixels is calculated by multiplying the number of horizontal pixels by the number of vertical pixels in the frame.

2.4 Threshold

The Jensen-Shannon divergence, as expressed in equation 14, provides a unique inter-frame variance measure. In order to ascertain if the inter-frame variance constitutes a shot boundary, we refer back to equation 4. In order for a metric to be effective in detecting a shot boundary, it has to be evaluated against a representative threshold τ.

Some existing techniques calculate the inter-frame variance between each frame in the video and then calculate the average thereof:

$$\tau = \frac{1}{N}\left[\sum_{i=1}^{N} \phi_i\right] = \frac{1}{N}\left[\sum_{i=1}^{N} JSD(f_{i-1}, f_i)\right] \qquad (16)$$

where $N \in \mathbb{Z}$ represents the total number of frames within the video and the inter-frame variance ϕ is the $JSD(f_{i-1}, f_i)$ calculated by equation 14. It is important to note that if the inter-frame variance $\phi_i = JSD(f_{i-1}, f_i)$ where $i = 1$ will result in a very large value as the zero-frame f_0 does not exist and is generally represented by a black frame. This is however not the case when encountering a fade-from-black transition at the start of the video due to the gradual change - low inter-frame variance.

Although this type of global thresholding allows for a well-represented threshold, it does however not conform to the real-time criteria as set forth in subsection 2.2. This short fall can be overcome by calculating a moving average from a selection of the last preceding frames against which the inter-frame variances can be evaluated. This *auto-adjusting threshold* $\bar{\tau}$ can be represented by:

$$\bar{\tau}_i = \frac{1}{\omega}\left[\sum_{j=1}^{\omega} JSD(f_{i-j-1}, f_{i-j})\right] \qquad (17)$$

where ω is the number of preceding frames to be evaluated for the moving average. The value of ω will be evaluated in section 4.1.

3. METHODOLOGY

3.1 Simulation Platform

A simulation platform was created in Visual Basic that allows various video formats to be analysed. At the core of this simulation platform is EMGU CV (version 3.0.0.2158). EMGU CV is a cross platform .Net wrapper for OpenCV (Open source computer vision). This is a library consisting of a multitude of programming functions aimed at computer-vision applications [1].

Table 1: Technical specifications of the simulation hardware

COMPONENT	SPECIFICATION
CPU	Intel Core i7-4470 3.4GHz
RAM	16GB DDR3 1600MHz
Motherboard	MSI Z87-GD65
Storage	SSD 850 EVO SATA III 250GB
Operating System	Windows 8.1 x64

In order to simulate the channel used by streaming media applications, the simulation platform loads a video and supplies the algorithm currently being evaluated with a sequential stream of video frames. Once the frames have been evaluated, they are discarded to free up memory. Research has shown that the compression and encoding properties of certain video streams and files can be exploited to further increase the processing speed of the algorithm in certain instances [7]. Despite this advantage, it was decided that the algorithm being evaluated in this platform, be evaluated at *face value* when supplied with the sequential frames excluding any accompanying compression and encoding information.

In order to evaluate the processing duration of the algorithms, the logical flow, as illustrated in figure 2, was implemented to run linearly as a single thread. In doing so, the efficiency of the technique is evaluated and not merely the multi-threading capabilities of the hardware platform. Specifics of the hardware used in the analysis is summarised in table 1.

Figure 2: High-level flow chart of the simulation setup

3.2 Simulation setup

Lienhart et al. performed a comparison of a few automatic shot boundary detection algorithms in [12]. The main focus of the algorithms in this comparison was the accuracy thereof with no mention of the processing speed or duration. Due to the real-time criteria, the execution duration of the technique has to be considered alongside the accuracy thereof.

It is clear from equation 14 that Jensen-Shannon divergence has limited parameters for optimisation. The only variable aspects are the probabilities calculated from the frames. Hence the best area for evaluation and optimisation of the technique can be identified as the manner in which this inter-frame variance is compared - i.e. the threshold. Thus the evaluation of the Jensen-Shannon divergence technique's efficiency, as a shot boundary detection technique, is accomplished by performing a basic sensitivity analysis of the following parameters pertaining to the threshold:

- Auto-adjusting threshold frame count (ω);
- Minimum threshold scalar (η);
- Average threshold scalar (α);
- Resize factor.

During the threshold comparison, the auto-adjusting threshold expressed by equation 17 is multiplied by the average threshold scalar α allowing the operator to dictate how much greater the inter-frame variance must be to be classified as a shot boundary. This evaluating threshold Υ has a further constraint imposed where a minimum threshold is used to evaluate the inter-frame variance if the evaluating threshold was too low:

$$\Upsilon = \begin{cases} \alpha\bar{\tau}, & \text{if } \alpha\bar{\tau} > \eta \\ \eta, & \text{otherwise.} \end{cases} \tag{18}$$

Thus the final outcome of the boundary detection test is given by:

$$DetectedBoundary = \begin{cases} \text{Yes}, & \text{if } JSD(f_{n-1}, f_n) \geq \Upsilon \\ \text{No}, & \text{otherwise.} \end{cases} \tag{19}$$

3.3 Data

Due to the infinite video possibilities that can be encountered, a diverse collection of videos were chosen to evaluate the video segmentation technique. These videos encompasses various transitional effects as well as various sizes and frame rates. In order to establish a baseline, a few test videos were created in Adobe Premier where the same two shots were joined using various transitions. Two video segments created from the Windows 7 sample video named *Wildlife* was subjected to various transitions as mentioned in section 2.2. The videos were encoded using the Flash Video format (.flv), using a framerate of 29.97 frames per second (FPS) with a resolution of 1280×720. The shot boundary between the two segments was created with the second segment starting on frame 60, hence a hardcut shot boundary. All the other transitions started on frame 49 and ended on frame 73. Some actual advertisements containing rapid moving scenes, multiple transitions and lens flares were also evaluated such as *The World of RedBull TV Commercial* [3].

3.4 *Video Modifications*

The video modifications mentioned in figure 2 pertains to the colorspace as well as the size of the video. All the videos used in the analysis are natively colour videos in the RGB domain. This specific implementation of the Jensen-Shannon divergence utilises only a single representative information source to calculate the inter-frame variance. This implies that all input videos frames are converted or *flattened* to the monochrome colorspace. This allows the grayscale luminescence to be used as the basis for the probability distribution.

The resize factor (RF) represents the scalar according to which the original frame is resized using bilinear interpolation with the help of the `Inter.CV_INTER_LINEAR` function [2].

3.5 *Evaluation Metrics*

Various metrics were employed to evaluate the effectiveness of the shot boundary detection algorithm that was evaluated.

Recall:

Recall rates provides a ratio of the number of relevant shot boundaries (correct detection) to the total number of relevant shot boundaries in the video, as expressed in equation 20.

$$Recall = \frac{Correct}{Correct + Missed} \tag{20}$$

Precision: The precision rates of the algorithm provides the ratio of relevant shot boundaries to the total number of irrelevant shot boundaries (false detections) retrieved, as expressed in equation 21 [15].

$$Precision = \frac{Correct}{Correct + FalsePositive} \tag{21}$$

Execution time: The execution time of each analysis will be recorded alongside the recall and precision rates thereof. This will help to justify a trade-off analysis between the technique parameters in terms of accuracy versus the execution time required. This becomes particularly important for *"real-time"* applications.

The execution time includes the following:

- Opening the video and reading the frames as they are required;
- Converting from RGB to monochrome where required;
- Resizing of the incoming frames where required.

4. RESULTS

The analysis results in this section is grouped according to the parameter under investigation. The majority of all the graphs in this section is representative of the average values produced by each of the 31 test videos. The influences of video resizing is incorporated in the other analysis.

4.1 *Auto-adjusting threshold frame count*

The first parameter that was investigated is the number of frames used by the auto-adjusting threshold as this outcome was used as an input parameter for the subsequent analyses.

While most of the test videos resulted in very good recall and precision rates, other, especially some videos with abnormal gradual transitions, performed poorly. In order to provide a representative sample, the average recall rates were calculated of all 31 videos with an accompanying standard deviations. The same was done with the precision values. Further investigation indicated that there were 5 specific test videos which exhibited poor recall values,. This was due to the slow changing nature of the videos e.g. an advert with a static background where the biggest changes were contributed by some animated text (small percentage of the frame). These results were still incorporated into the analysis.

The average recall response of the system for a resize factor (RF) equal to 1 is illustrated in figure 3 which seems to be increasing fairly linearly with regards to an increase in the number of frames.

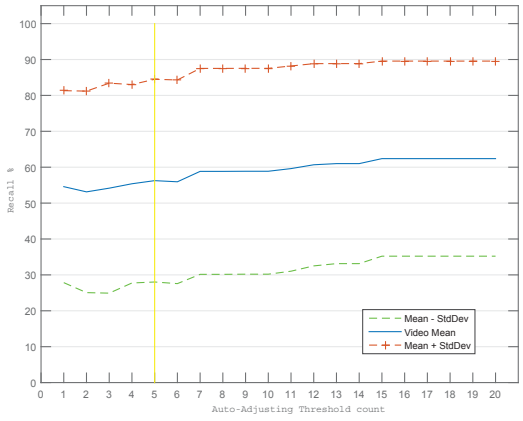


Figure 3: Auto-adjusting threshold analysis recall values for RF=1

This is however not the case for the accompanying precision response as seen in figure 4. An initial increase in the precision rate is observed but then followed by a steady decline. A maximum precision response was observed at a frame count of 5.

Similarly the recall behaviour for the other resize factors follow the same trend seen in figure 5 with increasing linear patterns but lower average values.

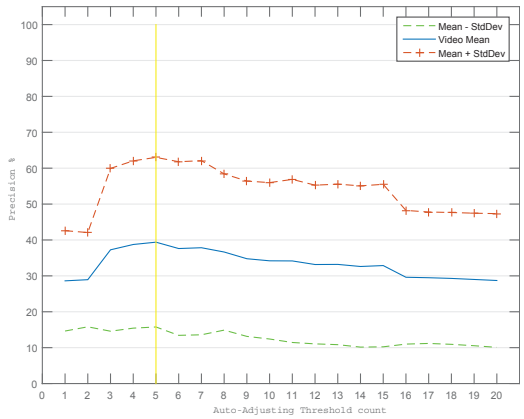


Figure 4: Auto-adjusting threshold analysis precision values for RF=1

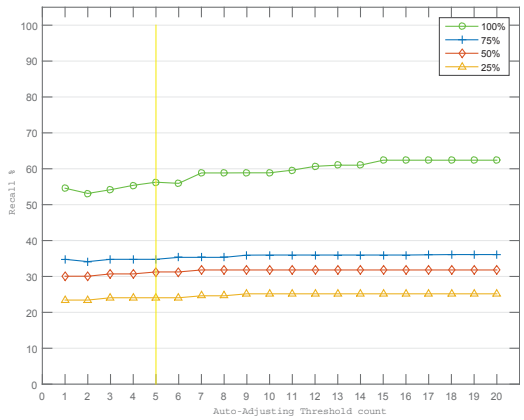


Figure 5: Auto-adjusting threshold analysis recall values for all RF

Logic infers that the precision values will also follow suit. Generally this assumption is correct - all the resize factors follow a steady decline, however the concave form with a local maximum is not as clear. There is an anomaly with regard to RF = 75% as shown in figure 6. The precision rates for this RF is noticeably higher than the other RF values. The possible cause for this might be attributed to the bilinear resizing algorithm that is applied to the frame. By resizing the frame, the image is condensed, but at RF=75% still retains a large amount of unique information for an inter-frame variance to be calculated.

The ideal frame count will result in maximum recall and precision values, hence the intersection of the two graphs as seen in figure 7, which represents the average recall an precision values across all RF values. The intersection is visible at a frame count $\omega = 15$ with an average recall and precision rate of approximately 39%. Due to the aforementioned anomaly at the 75% resize factor, the intersection point for this RF was specifically evaluated. Although not indicated on figure 7, the raw data indicated that the point of intersection for RF=75% was found to be at a frame count $\omega = 21$ with an average recall and precision rate of approximately 35%. These rates are however deemed too low. The recall and precision rates for RF=100% is shown in figure 8 with a recall rate of 67%

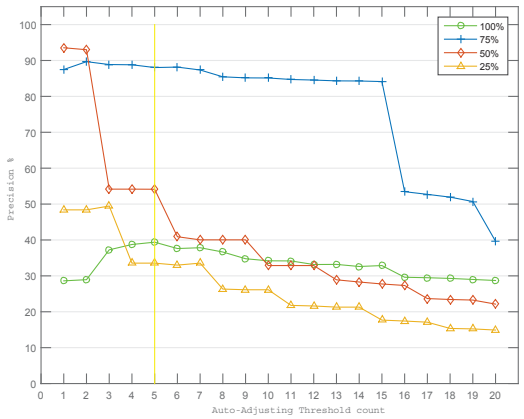


Figure 6: Auto-adjusting threshold analysis precision values for all RF

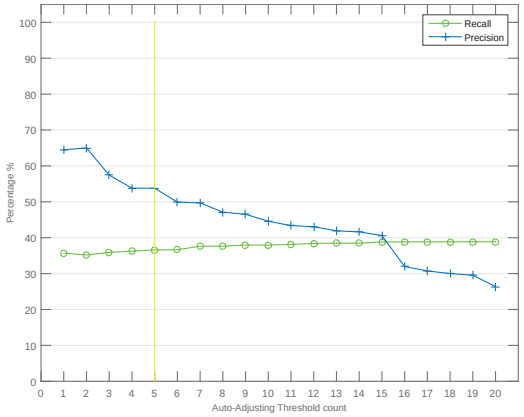


Figure 7: Auto-adjusting threshold analysis recall and precision values for the average RF

and a precision rate of 40% at a frame count $\omega = 5$.

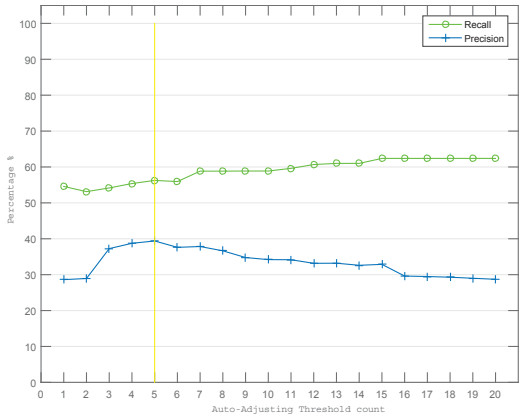


Figure 8: Auto-adjusting threshold analysis recall and precision values for RF=1

4.2 Average threshold sensitivity

The second parameter to be evaluated was the average threshold sensitivity while using the previously calculated frame count $\omega = 5$. The recall rates obtained show a quick decline before appearing to stabilise as seen in figure 9.

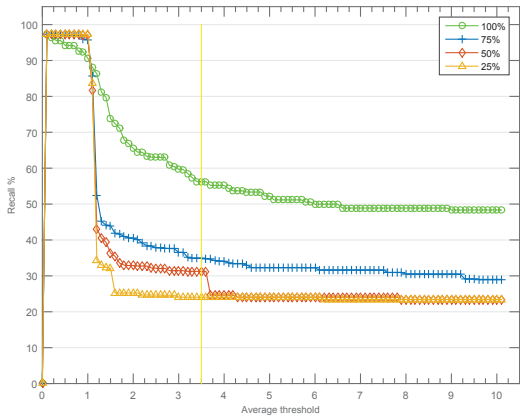


Figure 9: Average threshold analysis recall rates for all RF values

The precision rates present with a large initial increase that appears to level off at higher threshold levels shown in figure 10. The intersecting point for RF=75% is situated

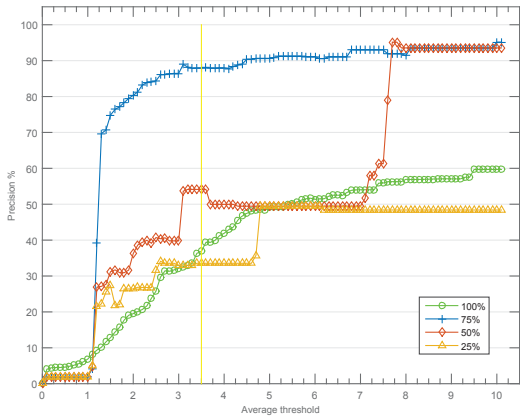


Figure 10: Average threshold analysis precision rates for all RF values

at $\alpha \approx 1.2$ which results in recall and precision rates of approximately 49%. The best rates are obtained where RF=100% where $\alpha = 5.7$ resulting in recall and precision rates of 51%. The intersecting point when evaluating the average for all RF values is located at $\alpha = 2$ with recall and precision rates of approximately 41% as illustrated in figure 11.

4.3 Minimum threshold sensitivity

The sensitivity of the detection technique to variations in the minimum threshold was analysed while using an average threshold $\alpha = 3.5$ and a frame count $\omega = 5$. This analysis indicated that any variations in the minimum threshold $0 \leq \eta \geq \alpha$ does not relate to any change in the recall and precision rates for any RF as shown in figures 12 and 13.

4.4 Analysis duration

The analysis durations for the auto-adjusting threshold frame count analysis is illustrated in figure 14 from which

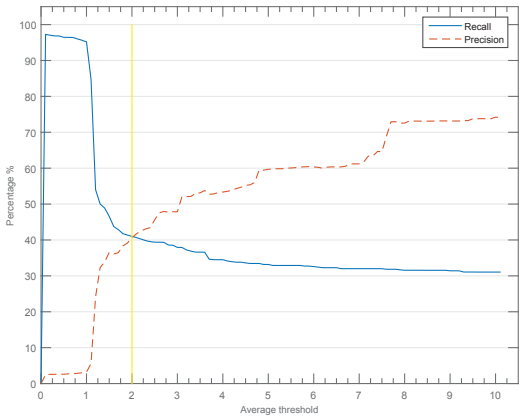


Figure 11: Average threshold analysis recall and precision rates for the average RF values

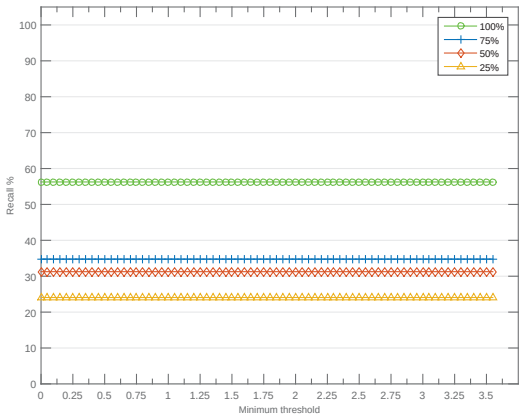


Figure 12: Minimum threshold analysis recall rates for all RF values

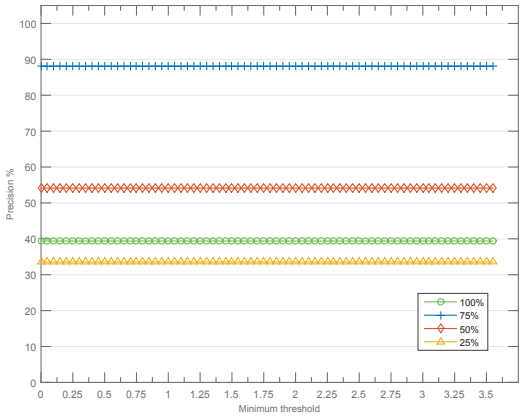


Figure 13: Minimum threshold analysis precision rates for all RF values

it is evident that the analysis duration follows a fairly linear pattern, where $\omega > 2$, with the exception of the initial threshold value of RF=1.

The same linear behaviour is encountered during all the other investigations. The general trend for all parameter analyses, as seen in figure 15, indicates that the RF = 75% actually took the longest to complete.

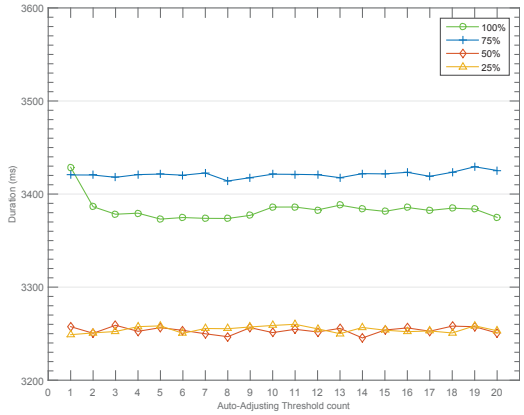


Figure 14: Auto-adjusting threshold analysis duration times for all RF values

With regard to the real-time component, the maximum average duration encountered at RF = 75% at approximately 3435ms. When comparing this maximum to the average duration of the test video 9985ms, it is clear that the real-time criteria will be met as the analysis duration is at least 2.9 times faster than the playback duration. In order to ensure that unbiased timing was achieved, each value for each parameter under investigation was analysed 10 times with the average thereof taken as the analysis duration for that instance.

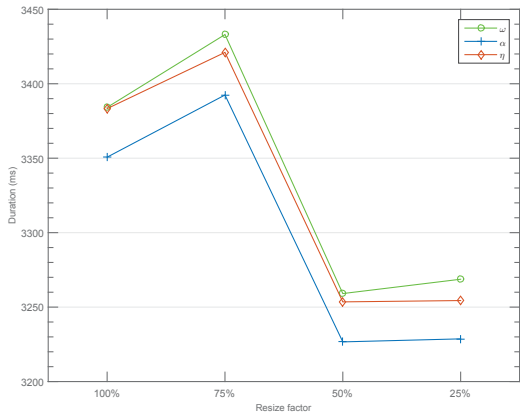


Figure 15: Average analysis duration times for all RF values

5. CONCLUSION AND RECOMMENDATION

The Jensen-Shannon divergence proved to be a technique capable of detecting shot boundaries with fairly good recall and precision rates. The analysis duration of the technique more than satisfied the real-time requirements while using only the *historic* (received) data for calculations.

It is important to note that the average CPU usage was, on average 35%, as well as the average RAM usage at 33% throughout the analysis. This indicates that the timing for the analysis is representative of the actual execution and not perplexed by the system bottlenecking by e.g. caching to a page file if the RAM was filled.

The results indicate that the RF = 75% has some outliers with regards to the α- and ω precision rates as well as longer execution times.

A finer resolution analysis of the resize factor would be beneficial to pinpoint the optimal RF in terms of optimal recall and precision rates while keeping the analysis duration as low as possible.

The optimal parameters based on the simulation results contained in this article is as follow:

- $\omega = 5$;
- $\alpha = 5.7$;
- η =No-effect $\therefore \eta = 0$.

Throughout the analysis a noticeable *abrupt* decline in precision rates was observed at $\omega = 5$, which warrants some further investigation.

6. ACKNOWLEDGMENTS

I would like to thank Prof. W.C. Venter for the assistance during this investigation as well as Prof. A.J. Hoffman for the continued financial contribution towards the project.

REFERENCES

[1] Open source computer vision library [online]. URL: `http://opencv.org/about.html`.

[2] Opencv geometric image transformations [online]. URL: `http://docs.opencv.org/3.1.0`.

[3] The world of redbull tv commercial 2013 [online]. URL: `https://www.youtube.com/watch?v=XSFIELPSGnU`.

[4] A. M. Amel, B. A. Abdessalem, and M. Abdellatif. Video shot boundary detection using motion activity descriptor. *arXiv preprint arXiv:1004.4605*, 2010.

[5] J. Burbea and C. R. Rao. On the convexity of some divergence measures based on entropy functions. Technical report, DTIC Document, 1980.

[6] C. Chan and A. Wong. Shot boundary detection using genetic algorithm optimization. In *Multimedia (ISM), 2011 IEEE International Symposium on*, pages 327–332. IEEE, 2011.

[7] M. Chunmei, D. Changyan, and H. Baogui. A new method for shot boundary detection. In *Industrial Control and Electronics Engineering (ICICEE), 2012 International Conference on*, pages 156–160. IEEE, 2012.

[8] M. Feixas, A. Bardera, J. Rigau, Q. Xu, and M. Sbert. *Information Theory Tools for Image Processing*, volume 6. Morgan and Claypool Publishers, 2014.

[9] U. Gargi, R. Kasturi, and S. H. Strayer. Performance characterization of video-shot-change detection methods. *Circuits and Systems for Video Technology, IEEE Transactions on*, 10(1):1–13, 2000.

[10] J. L. W. V. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, 30(1):175–193, 1906.

[11] S. Lefèvre, J. Holler, and N. Vincent. A review of real-time segmentation of uncompressed video sequences for content-based search and retrieval. *Real-Time Imaging*, 9(1):73–98, 2003.

[12] R. W. Lienhart. Comparison of automatic shot boundary detection algorithms. In *Electronic Imaging'99*, pages 290–301. International Society for Optics and Photonics, 1998.

[13] M. Menéndez, J. Pardo, L. Pardo, and M. Pardo. The Jensen-Shannon divergence. *Journal of the Franklin Institute*, 334(2):307–318, 1997.

[14] M.G. de Klerk and W.C. Venter and A.J. Hoffman. Digital video shot boundary detector investigation. In *Southern Africa Telecommunication Networks and Applications Conference (SATNAC) 2014*, pages 105–110, http://www.satnac.org.za/, 2014. Southern Africa Telecommunication Networks and Applications Conference (SATNAC).

[15] V. Raghavan, P. Bollmann, and G. S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Information Systems (TOIS)*, 7(3):205–229, 1989.

[16] T. D. Schneider. Information theory primer, 1995.

[17] C. E. Shannon. A mathematical theory of communication. *SIGMOBILE Mob. Comput. Commun. Rev.*, 5(1):3–55, jan 2001. URL: `http://doi.acm.org/10.1145/584091.584093`, `doi:10.1145/584091.584093`.

[18] I. Spanou, A. Lazaris, and P. Koutsakis. Scene change detection-based discrete autoregressive modeling for mpeg-4 video traffic. In *Communications (ICC), 2013 IEEE International Conference on*, pages 2386–2390. IEEE, 2013.

[19] A. K. Tripathi, U. Ghanekar, and S. Mukhopadhyay. Switching median filter: advanced boundary discriminative noise detection algorithm. *Image Processing, IET*, 5(7):598–610, 2011.

[20] M. Vila, A. Bardera, Q. Xu, M. Feixas, and M. Sbert. Tsallis entropy-based information measures for shot boundary detection and keyframe selection. *Signal, Image and Video Processing*, 7(3):507–520, 2013.