A Hash Based Technique for the Identification of Objectionable Imagery

D A N I E L F R A S E R, BCIS, Dip. IT

A thesis submitted to the Graduate Faculty of Design and Creative Technologies
Auckland University of Technology
in partial fulfilment of the requirements for the
Degree of Master of Forensic Information Technology (MFIT)

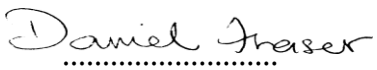
School of Computer and Mathematical Sciences

Auckland, New Zealand

2014

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person, nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning, except where due acknowledgement is made in the acknowledgements.



Signature

Acknowledgements

This thesis was conducted at the Faculty of Design and Creative Technologies in the school of Computer and Mathematical Sciences at Auckland University of Technology, New Zealand. Guidance and support was received from many people throughout the duration of the thesis, who without, would not make this thesis possible. I would like to thank my family, including my mother Jennifer, father Stuart and two brothers, Christopher and Jonathan and most importantly my partner Elena Williams, who all provided support and encouragement during the entirety of the Masters in Forensic Information course as well as throughout my thesis.

I would also like to thank Prof. Brian Cusack, my thesis supervisor, for the guidance and support given to the thesis project. Brian continually provided research direction and motivation which without this research would not have been completed to the standard which it stands.

In addition, I would like to thank my friend and former MFIT student, Arkar Kyaw who was always striving to create a positive environment and energizing discussions in the research topic area. Your positive attitude is second to none and I am forever in your debt, thank you. I would like to thank my fellow MFIT students who provided exceptional feedback throughout my research.

This research would not be possible without the support of Campbell McKenzie (Director of Forensic Technology Services) and PricewaterhouseCoopers for use of the PwC Digital Forensic Laboratory and the use of the Forensic Tool Kit software.

Also the assistance from several proof readers is acknowledged.

Abstract

The Internet has been one of the greatest man made achievements of the 20th century. It has revolutionized the computer and communications world like nothing before. The invention of the telegraph, telephone, radio, and computer set the stage for this unprecedented integration of capabilities (Leiner, et al., 2009, p.22). The Internet has given us the ability to easily communicate globally, access the world's libraries simultaneously, enabled surgery to be performed remotely, created virtual markets for the sale of goods and services, and allowed media on demand. In contrast, criminals have taken this opportunity to research into new ways of profiting by using the Internet as a platform. Accordingly to the Symantec Cyber Crime Report 2013, the global cost of cybercrime was US$113 Billion (NZ$1.35 Billion) in 2013 (Horbury, 2013). One of the fields that criminals are profiting in is child exploitation. This research aims to provide a better solution at identifying objectionable images on a global scale to reduce the amount of material being so easily distributed.

The current problem with identifying objectionable images is that there are an enormous number of images on the internet that require analysing. Forensic Investigators are currently using two techniques for identification. They are Content Based Image Retrieval (CBIR) and Concept Based Image Indexing (CBII). CBIR uses the visual aspects of an image for identification and CBII uses the metadata of the image for identification. The current problem is that CBIR is very accurate, but very slow and CBII is very inaccurate, but very fast in image retrieval. This research attempts to solve the issue by proposing a new hash based technique for the identification of objectionable imagery. A Hash Based Technique for the Identification of Objectionable Imagery. The Hashed Based Image Retrieval (HBIR) technique proposes to be faster than CBIR and more accurate than CBII.

A constructionist approach using the design science methodology (DSM) will be used to conduct the research. The DSM is a flexible methodology allowing the researcher to slightly redirect the research while it is being undertaken to create a robust and efficient artefact. Four phases of research were undertaken, the first phase was the software and pilot testing phase. The second phase was HBIR

testing using the proposed artefact. The third phase was CBIR testing using Forensic Tool Kit software. The fourth phase was CBII testing using Encase software.

The findings illustrated that the CBII technique was the fastest at processing the 10 batches of 10,000 images with processing times ranging between 21 seconds for 50KB batch of images to 407 seconds for the 950KB batch of images. The HBIR technique was slightly slower with processing times ranging me between 446 seconds for 50KB batch of images to 922 seconds for the 950KB batch of images. The CBIR was the slowest technique with processing times ranging me between 2024 seconds for 50KB batch of images to 3188 seconds for the 950KB batch of images. Overall the results proved that the HBIR technique was a more balanced and efficient image retrieval technique.

The research project has provided additional knowledge relating to identification and retrieval of objectionable images. The research has identified through experimentation a new technique that is feasible for implementation to assist forensic investigators and law enforcement in hope of reducing the distribution of objectionable images. Furthermore, it is recommended that further research be conducted in this area to strengthen the technique.

Table of Contents

Chapter Three: Research Methodology

Chapter Four: Research Findings

Chapter Five: Research Discussion

Chapter 6: Conclusion

List of Tables

List of Figures

Abbreviations

ACT	Average Comparison Time
AIT	Average Insertion Time
BMP	Bitmap
CA	Certificate Authority
CBII	Concept Based Image Indexing
CBIR	Content Based Image Retrieval
CPU	Central Processing Unit
CRC	Cyclic Redundancy
DOC	Word Document format
DS	Digital Signature
DSA	Digital Signature Algorithm
DSM	Design Science Methodology
E01	Evidence File Format
EID	Explicit Image Detection
Encase	Forensic Software
FNV1a	Fowler-Noll-Vo Hash
FTK	Forensic Tool Kit
GB	Gigabyte
HBIR	Hash Based Image Retrieval
HDD	Hard Disk Drive
IBM	International Business Machines
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronic Engineers
IFT	Inverted File Techniques
IMG	Image File Format
IP	Internet Protocol
ISO	International Standards Organisation File Format
JPG	Joint Photographic Experts Group
KB	Kilobyte
Kb	Kilobit
LFID	Level Fact Identifiers

m-block	Variable Message Block Size
MD4	Message Digest 4
MD5	Message Digest 5
MS	Microsoft
n-bit	Variable Bit Size
NIST	National Institute of Standards and Technology
NSA	National Security Agency
NZ	New Zealand
PNG	Portable Network Graphics format
QBIC	Query By Image Content
RAM	Random Access Memory
SHA	Secure Hashing Algorithm
SHS	Secure Hash Standard
SSD	Solid State Drive
TFM	Ternary Fact Model
TXT	Text File Format
USB	Universal Serial Bus (Drive)
xDFT	Extensible Dynamic Fault Tolerance Model
X-FST	Xerox Finite-State Tool
XOR	Exclusive OR

Chapter One

INTRODUCTION

1.0 INTRODUCTION

> "The Internet has revolutionized the computer and communications world like nothing before. The invention of the telegraph, telephone, radio, and computer set the stage for this unprecedented integration of capabilities" (Leiner, et al., 2009, p.22).

The Internet has also set a new stage for an unprecedented level of global crime which is often referred to as cybercrime. The Australian Institute of Criminology (2011) defines cybercrime as an umbrella term that refers to an array of criminal activity including offences against computer data and systems, computer-related offences, content offences and copyright offences. Unlike many other crimes to society such as murder, rape and theft, cybercrime does not require the person to be in the country at the time to commit the crime. Computer crimes can be achieved by residing in one country and committing a crime in one or more countries simultaneously. The Internet has created a virtual world which has made objectionable imagery boom as well as fraud, theft, money laundering and other profitable crimes. Accordingly to the Symantec Cyber Crime Report 2013, the global cost of cybercrime was US$113 Billion (NZ$1.35 Billion) in 2013 (Horbury, 2013). However, one of the most damaging crimes to society is child exploitation. On the 2nd September 1990, the United Nations *Convention on the Rights of the Child* treaty was introduced to protect children around the world from harm. In total, 193 countries have accepted the treaty including every member of the United Nations (apart from United States, Somalia, and South Sudan) (Convention on the Rights of the Child, 1990).

Section 1.1 will discuss my motivation to undertake the research in this area as well as why I believe more research is needed in this area in the future. This will be followed by gaps in current research (Section 1.2) and a summary of

the findings (Section 1.3). This chapter will then be concluded by reviewing the structure of the thesis (Section 1.4).

1.1 MOTIVATIONS OF THE RESEARCH

The research was motivated by three factors. The first factor is previous research that was completed by Terre des Hommes (a Dutch Non-Government Organisation), the second is using technology to help society, thirdly is taking responsibility for how technology has damaged society.

The research that Terre des Hommes helped to design, nicknamed “Sweetie”, was used to lure and trap criminals who exploited children. The research circulated around Facebook where they were asking people to sign a petition to persuade Interpol for further funding. The Terre des Hommes foundation used the software to identify 1000 criminals who were dealing in exploitation (Crawford, 2013).

Managing director, Albert Jaap van Santbrink, said:

> "*Our worst-case scenario is that the same will happen with this phenomenon as with child pornography, which is now a multi-billion industry in the hands of criminal gangs.*" (Crawford, 2013).

This was the source of inspiration to research in to areas of how technical tools could be developed to support police enforcement.

The second motivation is to want to create a solution to a problem that would help society as a whole. A large number of people are completely oblivious to the illegal activities that happen on the internet on a daily basis. Funding of research and the amendments to laws are needed in order to reduce the amount of crime, or otherwise as users become more interconnected these issues will intensify.

The third motivation is taking responsibility on how the internet has damaged society. Large companies such as Google, Yahoo and Microsoft have profited in building technology however they have done very little in taking responsibility for protecting society from their technology. For example until recently objectionable images could be searched for by using the Google or Yahoo search engines (Crawford, 2013). Furthermore, Google and Microsoft have

recently agreed to work with the National Crime Agency and the Internet Watch Foundation in the UK to try and identify networks which host objectionable images (Ward, 2013). To conclude there is still a great need for tools and technology to be researched and developed to protect the society from all illegal online behaviour.

1.2 GAPS IN CURRENT RESEARCH

In chapter 2, a general discussion is made that reviews the literature in relation to identification and retrieval of objectionable images. The general literature is then summarised in Issues and Gaps in Current Research (Section 3.1) where five individual categories of gaps are identified. The gaps identified are image processing, legal and political issues, accuracy, anti-forensics and creating a long term solution. It then continues by prioritising the issues and identifies if the issues are feasible areas to research as well as justifying the decision. The gaps that this research will investigate are processing, accuracy, and creating a long term solution. The overall goal in this area is to identify a technique that is a more productive and balanced system than current image retrieval techniques in hope of reducing the amount of stored and distributed objectionable images. The areas that will not be investigated are legal and political issues and anti-forensics. These two issues will not be investigated because this is a technical thesis and they are not directly related to digital forensics. Secondly, anti-forensics will not be investigated because Forensic Analysts are already aware that when certain anti forensic methods have been deployed by a criminal it is unfeasible to extract evidence. To publish the techniques that are already widely known would not be new research and would be anti-productive in solving crime.

To summarise, in the current environment, forensic investigators are using two techniques to identify objectionable images. The first is Content Based Image Retrieval (CBIR) which identifies the image by the visual aspects of the image (colours, objects, and textures). CBIR is very slow at analysing images however it is very accurate. The second technique is Concept Based Image Indexing (CBII). CBII identifies images by the metadata (file name, file size, date taken, dimensions, bit depth, name, item type (file format), date created, date modified, size, file owner and custom key words). In contrast to CBIR, CBII is very fast,

however is very inaccurate in image retrieval. The overall goal is to identify and more balanced technique that has greater accuracy than CBII and can process images quicker than CBIR to help law enforcement to reduce the distribution of objectionable images.

1.3 FINDINGS OF THE RESEARCH

The research findings discovered that a hash based image retrieval technique can process images faster than content based image retrieval, and with a greater accuracy than concept based image indexing. Therefore the goal of finding a more balanced image identification technique that can be used to identify objectionable imagery on a global scale was successful.

In order to investigate the potential of the hash based technique a custom tool was developed in Java. Testing was then performed on multiple hashing algorithms to identify the most suitable algorithm. Suitability was measured by the effectiveness of the hashing algorithm if it was implemented into a global objectionable image identification system. Therefore, the key aspects that were considered were speed, bit size, collision rate and algorithmic design. The hashing algorithm Secure Hashing Algorithm with 256 bit (SHA-256) was identified as the most effective algorithm. SHA-256 was able to hash 10 batches of 10,000 images of various file sizes ranging from 22.89 seconds to 498.56 seconds.

The next function that required experimentation was the identification of the most effective database. Three databases, MS SQL, PostgreSQL, MySQL, were tested thoroughly to identify which database was the fastest at inserting hash values and comparing hash values. PostgreSQL was identified as the quickest in both areas. PostgreSQL could insert 250,000 hash values into the database in 6233 seconds (2.4932×10^{-6} seconds per hash value) and could compare 10,000 hash values to 250,000 stored hash values in 446 seconds (1.784×10^{-7} seconds per hash value). The testing was concluded by the proposed HBIR Tool using the combination of the SHA-256 hashing algorithm with the PostgreSQL database. Forensic Tool Kit (FTK) and Encase software was then used to benchmark the HBIR Tool by comparing it to the CBIR and CBII techniques, respectfully. Testing showed that FTK could process the same 10 batches of 10,000 images in times ranging from 2023.5 seconds to 3187.8 seconds. In addition, Encase could

process the images in 21 to 407 seconds. In conclusion on average the HBIR could process the images 71.14 % faster than CBIR and due to the weaknesses of CBII the HBIR had significantly more accuracy.

1.4 STRUCTURE OF THE THESIS

In conclusion, Chapter 1 introduces gaps in current research in relation to identifying objectionable images on a global scale. It also reviewed the motivation of the research and reinforced how research in this area is critical to protecting society.

Chapter 2 introduces the literature review and is followed by the introduction and calculations of non-cryptographic and cryptographic calculations. Section 2.6 focuses on the underlying weakness of any hash based system by discussing hashing collisions and how they can be reduced. The next section discusses where hashing algorithms have already been effective in other areas of technology. It then evaluates the content based image retrieval and concept based image indexing techniques.

Chapter 3 introduces the methodology section by first summarising gaps in current research where a robust main question was developed. The main question was then analysed and five sub questions with their associated hypotheses were formed. The next section, research design (Section 3.3) is discussed where the design science methodology was chosen. A systematic procedure followed where the research phases was documented (Sections 3.5 to Section 3.7). Chapter 3 is then concluded with the data requirements, expected outcomes and limitations of the research.

Chapter 4 reports on the findings of the experiment. It begins by discussing the variations in data requirements from what was first proposed in chapter 3. It then reports on the research phase 1 software and pilot testing (Section 4.2 and Section 4.3), followed by phases 2,3 and 4 for the experiment testing of the HBIR, CBIR and CBII techniques (Section 4.4). Chapter 4 is then concluded with a brief technique comparison.

Chapter 5 discusses the findings of the four research phases in relation to the research question, sub questions and hypotheses. Therefore, the research questions and hypotheses developed in chapter 3 are answered (Section 5.1) and

discussed (Section 5.2). A critical reflection on the testing processes follows by reviewing the strengths and weaknesses of the experimental procedure. Section 5.4 evaluates the strengths and weaknesses of the overall research and Section 5.5 reviews Forensic Tool Kit and Encase. Chapter 5 is then concluded by a theory building section.

Finally, Chapter 6 summarises the research findings. Recommendations and future research directions are then discussed. The conclusion of the thesis is followed by a list of references and appendices.

Chapter Two

LITERATURE REVIEW

2.0 INTRODUCTION

The main objective of chapter 2 is to introduce, review and critique the current literature relating to hashing and image retrieval techniques.

Section 2.1 will begin by introducing the terminology and the early adaption of hashing and hashing algorithms. The cyclic redundancy check (CRC) hashing algorithm will be discussed in Section 2.2 which was one of the first hashing algorithms that was widely adopted by developers. The next section will investigate how the CRC is manually calculated. Section 2.4 will continue by discussing complex hashing algorithms and how Secure Hashing Algorithms (SHA) are calculated (Section 2.5). Section 2.6 will analyse hashing and collisions. A review of different uses of hashing algorithms in today's technology will follow in Section 2.7. Section 2.8 will discuss the strengths and weaknesses of hashing algorithms. Section 2.9 will evaluate Content Based Image Retrieval and Concept Based Image Indexing (Section 2.10). A legal section will follow where legal problems with objectionable images will be reviewed. Chapter 2 will concluded in Section 2.12.

2.1 INTRODUCTION TO HASHING AND HASHING ALGORITHMS

It is important to distinguish the difference between the terms hash and hashing algorithm. A hashing algorithm is a well-defined mathematical function that converts a string of data of arbitrary length and outputs a smaller alphanumeric representation, typically a fixed length data string. The hash (also known as a hash value, hash code, hash sum, or checksum) is the output from the hash function. The basic structure of a hash algorithm is *string > hash function > hash value*.

The first documented evidence of using the term hash function was in the article *A Business Intelligence System* authored by Hans Peter Luhn, written in 1958 (Ridge & Curry, 2012). Luhn used the term in conjunction with other

technologies to create what is called today an Information System. In the 1960s methodologies, processes, multiple architectures and technologies began to develop to identify uses for hashing algorithms to strengthen data organisation for business intelligence (Ridge & Curry, 2012).

In 1961 one of the first hashing algorithms was invented by W. Wesley Peterson in the form of cyclic redundancy checking (CRC) for error detection in stored and transmitted data (Peterson & Weldon, 1972). CRC's main objective was to create a simple and computational inexpensive algorithm that could be used for time dependant applications such as telecommunication devices (Monteiro, et al., 2001). Mathematically, CRC can be described as treating a binary string as a polynomial overGF(2) and performing polynomial division of the binary string (Koopman & Chakravarty, 2004).

2.2 CYCLIC REDUNDANCY CHECKING IN TELECOMMUNICATIONS

The implementation of error correction in telecommunications contained five main stages. The first stage was to create the CRC *n*-bit size hash code. The second stage was appending the hash code to the data block. The third stage is physical transmission of the package (data block + hash value) to the receiver over a communication medium. Once the receiver's device has received the package, the device would then run the same CRC algorithm over the data block to create a new hash value. The fifth stage is where the new hash value would be compared to the hash value that was received by the sender. If the senders and receivers hash values match, the receiver has confirmed the data received was *most likely* correct, depending on the length of the hash value. If the hash value from the sender and the receiver did not match, the receiver would drop the data package and make a new request to the sender for retransmission where the previous steps would then be repeated (Dublop & Smith, 1994, p.197). When implementing any hashing algorithm, it is critical to identify the most efficient hash length. The length of the hash is correlated to the accuracy. A longer hash value, the higher the accuracy of the transmitted data is correct. The length of the hash value is measured in bits and CRC typically uses 8, 16, 32, or 64 bit lengths.

Unfortunately, it is not as simple as always using a very large bit length. The length of the hash will determine the speed that the hash can be generated.

Therefore, if the telecommunications designer decides to use a very large hash value length, it will require more processing by both end devices which will increase the delay. If the delay is too long in voice telecommunications it can cause audio issues such as echoing.

Therefore, a decision has to be made to use a short 8-bit hash that is computed quickly and have low accuracy, or a larger hash which will provide higher accuracy but requires more time to process.

2.3 HOW CRC IS CALCULATED

There are three main stages involved in creating a CRC hash. The first stage is to convert the message into binary. The second stage is to calculate the divisor, also known as a polynomial. The polynomial will be used to divide the message. The third stage is to Exclusive OR (XOR) over the binary using the polynomial. XOR is the simple process of moving from left to right across the message adding the "message bit" to the "polynomial bit" in base 2. Unless the next bit in the message is equal to zero. If the next bit is equal to zero then that 0 bit is skipped and the processes continues moving along the message until a value of 1 is detected and once a 1 is detected continue to XOR.

Once the message has been converted into binary, the polynomial will need to be calculated. The polynomial will be the size of the hash length plus 1 (hash n-bit +1, CRC-8 will have a 9bit polynomial; CRC-16 will have a 17bit polynomial, and CRC-*n*-bit will have *n*-bit +1 polynomial). An example of CRC-4 is calculated below showing the hash value of 15, using a 5bit polynomial.

{ Message } {4bit hash}

1 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 1 – 0 0 0 0 0

Divisor = x^n +2 + 1 where x is 2 and *n* is 4

Therefore 2^4 +2 +1 = 19 or 10011

Once the polynomial has been calculated the XOR process beings. This process is shown in the 12 steps below.

Step 1:

1 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 0 – 0 0 0 0 - message

1 0 0 1 1 - divisor / polynomial

0 0 1 1 0 0 1 1 0 1 0 0 1 0 1 0 0 - result

Step 2:

0 0 1 1 0 0 1 1 0 1 0 0 1 0 1 0 0 - message

1 0 0 1 1 - divisor / polynomial

0 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 - result

Step 3:

0 0 0 1 0 1 0 1 0 1 0 0 1 0 1 0 0 - message

1 0 0 1 1 1 - divisor / polynomial

0 0 0 0 0 1 1 0 0 1 0 0 1 0 1 0 0 - result

Step 4:

0 0 0 0 0 1 1 0 0 1 0 0 1 0 1 0 0 - message

1 0 0 1 11 - divisor / polynomial

0 0 0 0 0 0 1 0 1 0 0 0 1 0 1 0 0 - result

Step 5:

0 0 0 0 0 0 1 0 1 0 0 0 1 0 1 0 0 - message

1 0 0 1 11 - divisor / polynomial

0 0 0 0 0 0 0 0 1 1 1 0 1 0 1 0 0 - result

Step 6:

0 0 0 0 0 0 0 0 1 1 1 0 1 0 1 0 0 - message

1 0 0 1 11 - divisor / polynomial

0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 - result

Step 7:

0 0 0 0 0 0 0 0 0 1 1 1 0 0 1 0 0 - message

1 0 0 1 11 - divisor / polynomial

0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 - result

Step 8:

0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 0 0 - message

1 0 0 1 1 1 - divisor / polynomial

0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 - result

Step 9:

0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 - message

1 0 0 1 1 1 - divisor / polynomial

0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 - result

Step 10:

0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 1 0 - message

1 0 0 1 11 - divisor / polynomial

0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 - result

Step 11:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 – 0 0 0 - message

1 0 0 1 1 1 - divisor / polynomial

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 - result

Step 12:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 – 1 1 0 0 - message

1 0 0 1 11 - divisor / polynomial

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 - result

Alternatively it can be converted to base 10 and calculated as:

1 0 1 0 1 0 1 1 0 1 0 0 1 0 1 0 0 = 87700

1 0 0 1 1 = 19

➔ m mod p = hash value

➔ 87700 mod19 = 15 or 1 1 1 1

(Banzel, 2007)

2.4 COMPLEX HASHING AND CRYPTOGRAPHIC HASHING ALGORITHMS

In the 1970s researchers investigated where hashing algorithms could be used to assist in other technologies. It was soon realised that hashing algorithms could assist in cryptography. The first definitions, analysis and construction for cryptographic hash functions were demonstrated in *Digitalized signatures* (Rabin, 1978), *How to Swindle Rabin* (Yuval, 1979) and *Secrecy, Authentication and Public Key Systems* (Merkle, 1979; Preneel, 2010). Similar to a standard hashing

algorithms such as CRC, cryptographic hashing algorithms are still required to be easily computed to find the hash value. However, for a hashing algorithm to be classed as cryptographic it requires 3 additional properties.

- It is infeasible to generate a message from a given hash.
- It is infeasible to modify a message without changing the hash,
- It is infeasible to find two different messages with the same hash (Gersting, 2007).

The three (above) additional properties require significantly higher mathematical complexity in order to be calculated. Therefore, cryptographic hashing algorithms require more computation or time to calculate the value when being compared to non-cryptographic algorithms. The main cryptographic algorithms presently in use are the Secure Hashing Algorithm (SHA) and Message Digest (MD) algorithm.

2.5 HOW SHA-1 IS CALCULATED

SHA-1 is a National Institute of Standards and Technology (NIST) approved, widely used, cryptographic hashing algorithm that outputs a 160 bit hash value that was designed by National Security Agency (NSA). There is a two stage process to calculate a SHA algorithm. The first stage entails padding the message, parsing the padded message into m-bit blocks of a fixed size and then setting the initialisation values to be used in the hash computation. The second stage is generating the message schedule from the padded message and using the schedule along with functions, constants and word operations to generate a message digest through iteration. A word is a group of 32 bits or 64 bits (NIST, 2012). For the full specifications and understanding on the SHA hashing algorithms please refer to Appendix A.

2.6 COLLISIONS

A Collision is where two different strings or messages are inserted in to the same algorithm and the algorithm produces the identical hash values for both messages. For example there are two images, one objectionable and the other is non-objectionable. The objectionable image is inserted into the SHA-1 hash algorithm and the outputted hash value is *09a433ebf81d3b7de8ca43f142aa71b00bb3ea*. The

second non-objectionable image is then inserted into the SHA-1 hashing algorithm and the outputted hash value is identical. Therefore, from a hashing computation point of view, both images are identical. In contrast, from the human perspective, they are easily viewed as completely different.

The chance of a hashing algorithm producing a collision can be severely reduced by either increasing the size of the hash value (*n*-bit) and/or mathematically confirming the hashing algorithm is unbiased in its selection of hash values. To easily demonstrate, a hash value with the size of 1-bit will be either a 1 or 0. The hash size of 2-bits will result in 11, 10, 01 or 00 (4 potential hashes). This can be mathematically described by the number of possible hashes are (2^n), where *n* is the number of bits. A hash size of 8bits will have (2^8) 256 possible hash values. However, if the hash length increases to 16 bits the number of possible hash values increase to (2^{16}) 65536. Therefore the likelihood of two messages creating the same hash value has decreased significantly by increasing the number of bits. In addition, the hashing algorithm requires treating all potential hash values as equal and not having a bias. If hashes have a different chance of being outputted, it increases the rate of collisions as well as subjecting the algorithm to cryptanalysis.

Ideally every hashing algorithm would be 100% collision resistant. However, this is an impossible task due to the Pigeonhole principle and the Birthday Paradox. The Pigeonhole principle states that if $n > m$ where *n* is the number of images and *m* is the number of unique hashes available, there will be occurrences where multiple images with produce the same hash value. For example the CRC-8 algorithm has 256 available unique hashes and there is requirement for 300 hash values. There will be at least 44 ($m - n$) images with matching hashes. Therefore to avoid the Pigeonhole principle it is critical to have at least $n \leq m$. Furthermore, to have $n \leq m$ will still create the possibility of collisions. This can be demonstrated by rolling a dice 6 times and expecting 6 different results. On average there is a 98.45% chance the roller will roll the same number at least twice. This is known as the Birthday Paradox. For example a dice is rolled 7 times, as one would expect with each roll of the dice the chance of a collision is increased. What is not expected is the chance that a collision will occur is not linear, i.e. 1/6, 2/6, 3/6, 4/6, 5/6, 6/6, or 7/6. Refer to Figure 2.1 and Table 2.1 below.

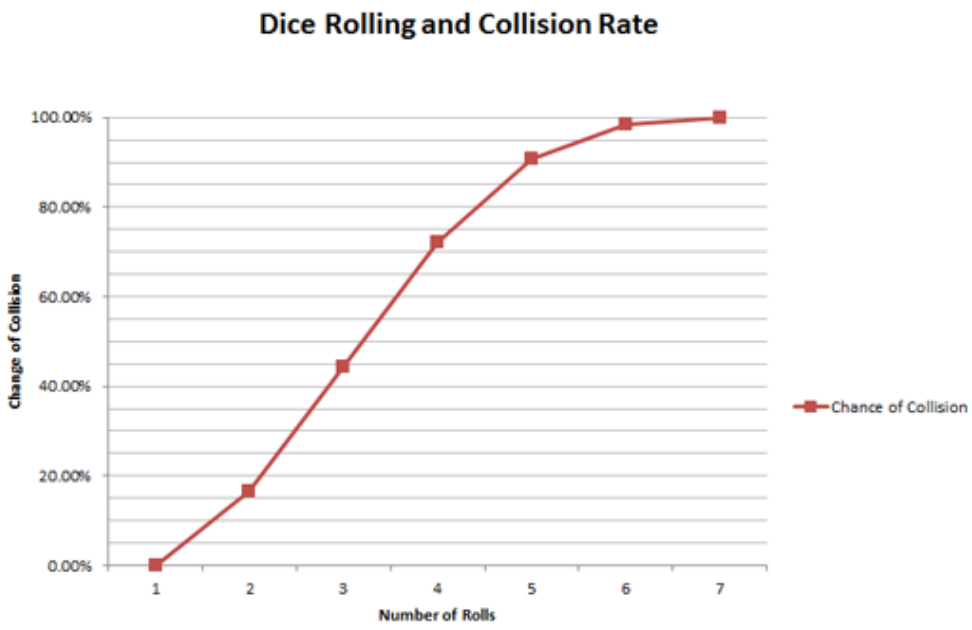


Figure 2.1 showing chance of getting 2 of the same numbers (a collision) when a dice is rolled 7 times

Table 2.1 showing chance of getting 2 of the same numbers when a dice is rolled 7 times

Dice Rolls	Chance of Collision
1	0.00%
2	16.66%
3	44.40%
4	72.20%
5	90.74%
6	98.46%
7	100.00%

The above graph and table illustrates that even if the number of possible hash values (dice throws) is equal to the number of hashes required, the likelihood of a collision is 98.46%. Therefore, the number of available hashes needs to be significantly more than the number of hash values required. As Figure 2.1 demonstrates, it is impossible to have a collision free hashing algorithm when creating at least two hash values. The hashing algorithm SHA-1 with 160 bit hash

value has 1.4615 x 10^{48} possible values and MD5-128bit has 3.4028 x 10^{38}. Refer to Table 2.2 for other examples of hashing algorithms.

Table 2.2 showing examples of hash values and their properties

Hash Name	Hash Length	Possible Values	Example Hash Value	Secure Hash
MD2	128 bit	3.4028237e+38	860aa94d2a8528faa8d25b8bef1548c2	No
MD4	128 bit	3.4028237e+38	4ecf9bbd7e8782af7aa230db7b80c29b	No
MD5	128 bit	3.4028237e+38	ba936adb281b251b71c937565e88e88c	No
MD6*	Variable 0< d ≤512 bits	Variable – Dependant on message		Unknown
SHA-1	160	1.4615016e+48	ddffeac8aff7b81ad217a5a99343e2a5c9a99bb4	Yes
SHA-224	224	2.6959947e+67	398739a13ecefe0f31af2641041b939c86a2702da8d1b4d8cf104242	Yes
SHA-256	256	1.1579209e+77	581b450c092c6c0a6b599b152ad49e8d19c12ff991b3857204d5af6bd1b70457	Yes
SHA-384	284	3.940201e+115	0b9e6470a0355edab753a6cfab14e08128aae503fcb46fa7c079d4bcd0ddcd7e1a0cad84200c54ed171be5ddd1e150e0	Yes
SHA-512	512	1.340781e+154	5c06ab1e3d4adacd86e42d6bfa4576a51a1596db24b00e8a369277dd8ed313f6541dcc2908453db84ad9502a4328ad62d245109322f96e431d0d7eb912c94dda	Yes
SHA-3	Arbitrary	Arbitrary		Yes

**MD-6 was submitted to NIST on 15/04/2009 and as of 07/07/2013 it has not* yet *been approved*. (http://groups.csail.mit.edu/cis/md6/)

2.7 USES OF HASHING VALUES

Hashing algorithms have been used to assist in multiple areas of information systems from digital signatures (DS), intrusion detection systems (IDS),

networking software, copyright infringement, large file downloads and random password generators.

2.7.1 Digital Signature

A digital signature is a mechanism where a message can be authenticated to prove the integrity of the sender (similar to a letter with a handwritten signature). Digital Signatures are used to assist in cryptography to secure 1 or 2 way communications. They can either be self-signed or signed by a trusted third party known as Certificate Authority (CA). The difference between self-signing and using a trusted third party is the location of the public key.

In two way encryption a digital signature contains a hash value which is encrypted with Alice's private key and then with Bobs public key. When Bob receives the message and digital certificate he will decrypt the message with his private key, then again with Alice's public key. Once Bob has decrypted the message he will then view the digital certificate contents and find the hashing algorithm that Alice used as well as Alice's Public key. Bob will insert the message into the same hashing algorithm to create a new hash. Once Bob has the new hash he will then compare his hash value with the hash value Alice sent. If the two hashes match the message has not been altered. If they do not match the message is invalid and another request will be made to Alice to resend the message.

Therefore a Digital Signature Algorithm (DSA) must be able to perform two main tasks, create a digital signature for Alice and verify the signature for Bob.

These two tasks are demonstrated below in figure 2.2.

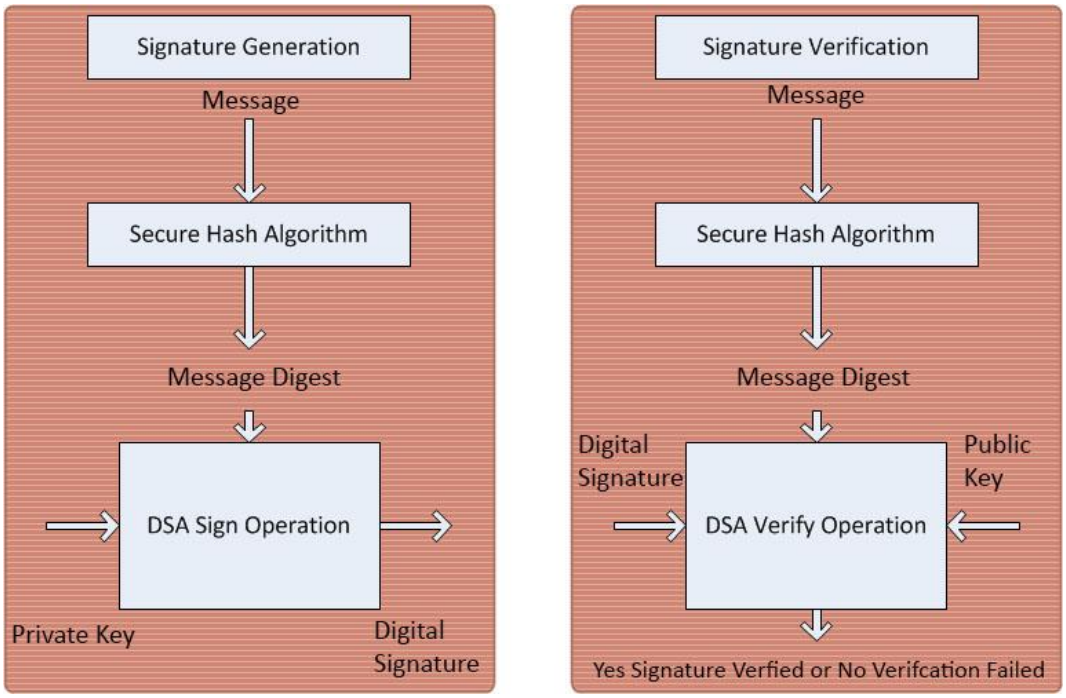


Figure 2.2 demonstrating SHA with the DSA (NIST, 1994, p.2)

For the digital signature to be classed as secure it must abide by the requirements in the Digital Signature Standard (DSS) published in 1994 (NIST, 1994). The DSS stipulates only hashing algorithms that are listed in the Secure Hash Standard may be used for secure communications if the system is required to be NIST compliant (NIST, 1995).

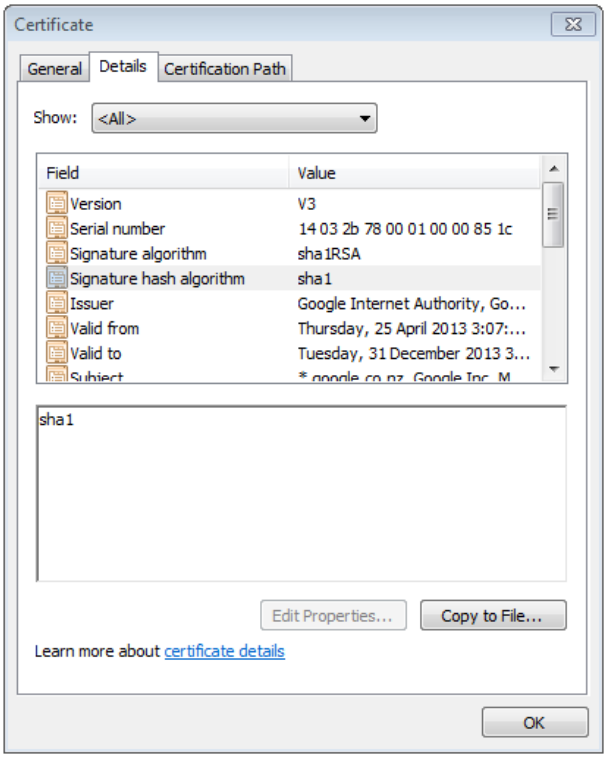


Figure 2.3 showing example of digital certificate

2.7.2 Intrusion Detection Systems and Hashing

Intrusion Detecting Systems (IDSs) are devices or software that monitor network infrastructure and system activities to detect unauthorised access. Software vendors such as Tripwire have utilised hashing algorithms by building software that logs hash values of system files. The Tripwire software periodically scans the system files matching the hash from the old log file to the current file. Any mismatch in values reflects a change in a system file. The advantage of this system is that the software will detect even a small change in the system files that may be overlooked by the human eye. Another advantage of using a hash based IDS system is that hashing algorithms use minimal system resources which will have little to no effect on the end user experience. When the IDS detects a mismatch of values it will notify the network or system administrator. The network administrator would then be able to investigate and identify if the change was legitimate or malicious. Website Administrators have also implemented similar systems by hashing the website code to stop cross –site scripting attacks. The same technique can be applied to databases, and firewall, router, switch configurations.

2.7.3 Networking

Another area where hashing algorithms have been implemented is to make routing more efficient to enable higher throughputs. In *Fast Hash Table Lookup using Extended Bloom Filter: An Aid to Network Processing*, the authors investigates different approaches to storing hash values. The authors explain that implementing hashing algorithms and novel hash tables can provide faster throughput for forwarding data packets than using naïve hash tables (Song, et al., 2005). It achieves this by "reducing the number of memory accesses needed for the most time consuming look ups" (Song et. al., 2005). Due to the reduction in the number of look ups, it has significantly reduced the required time to forward a data packet to the next node.

In *High Speed IP address Lookup Architecture Using Hashing* the authors demonstrate how a large router table with 37,000 entries can be compacted to a forwarding table of just 189kbytes. The authors continue to explain how

traditionally the Internet Protocol (IP) address itself was used as a memory pointer. However, due to the number of IP addresses in tables this technique has become impracticable because of the amount of memory space required. The hashing function takes multiple IP address routes and produces a shorter field that can be used as a subset in an index. By implementing a hash function it can perform one hash look up for every two memory accesses on average (Hyesook, et al., 2003).

Hashing is not just limited to routing forwarding tables. In *Performance of Hashing-based Schemes for Internet Load Balancing* the authors demonstrate how using CRC-16 can assist in distributing traffic loads according to unequal weights. The authors first look at direct hashing which is the simplest form of traffic splitting. Direct Hashing is hashing the five-tuple (source address, destination address, source port, destination port and protocol id). The authors experiment by comparing the Direct Hashing technique to 5 sub forms of the technique.

- Hashing the destination address.
- Hashing XOR folding of the destination address.
- Hashing XOR folding of the Source and Destination addresses.
- Using the Internet check sum.
- Using the CRC 16-bit algorithm.

The authors found that using the CRC-16 algorithm performed the strongest and fastest (Zhiruo, et al., 2000).

2.7.4 Copyright Infringement

In *Computer Based Copyright Control System in Social network and An Ordinal Measure Based on Grey Scale Value,* the authors looked at different approaches towards reducing piracy in a social media environment. The increase of bandwidth and removal of internet data caps, among other factors, have seen the numbers of users viewing video online increase dramatically over the last 10 years (Longfei, et al., 2009). YouTube, the largest social video provider receives 72 hours of video every minute. The YouTube Content ID system which monitors for copyright infringement processes 100 years of video every day (“Statistics”, 2013). The large amount of uploading of video content has created problems in

identifying objectionable or pirated material before it is published online. The authors look at different methods to identify pirated content to protect copy right infringement and the hosting provider from legal action. The authors continue by explaining how it is important for the copyright infringement software to identify inappropriate material rapidly and accurately and hashing algorithms can be used proficiently to solve the problem (Longfei, et al., 2009).

2.7.5 Large File Downloads

Hashing algorithms are commonly implemented on software vendor's websites for checking downloaded files. Software vendors have recently started to publish the MD5 and SHA1 hash values next to download links to enable the user to manually verify the integrity of the downloaded file (refer to Figure 2.4 below). The reason for potentially altering files may to include the injection of spyware or other malicious code. Large software companies (Google, Microsoft and Adobe) have taken the protection one step further by creating a secured encrypted connection between the software server and the end user. By encrypting the connection removes the threat of an eavesdropper from reading the data in addition to injecting spyware through man in the middle attacks.

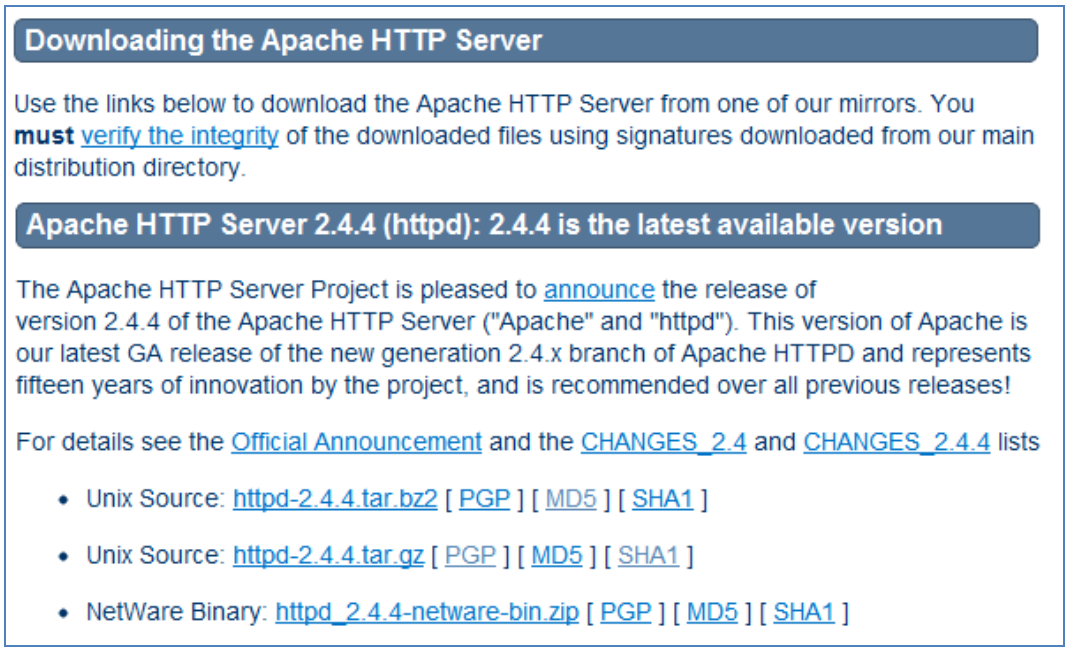


Figure 2.4 showing links to hash values next link to download file

(Apache Server Foundation, 2012)

2.7.6 Random Password Generators

Advanced computer users and system administrators have started to use hashing algorithms to create random passwords. Research has shown that even as users become more aware of the dangers of creating simple passwords, they are still choosing passwords that are very weak (Bonneau, 2012). Therefore, they are making themselves susceptible to brute force and dictionary attacks. Some researchers believe this will not change while passwords remain a common and "cost-effective solution" for access control to data (Bonneau, 2012). To combat the risk of brute forcing and dictionary attacks, some advanced computer users are now using hashing algorithms to randomly generate passwords. For example an advanced user would first create a strong password containing uppercase, lower case, numbers and symbols which do not contain any dictionary words. The administrator would then insert the custom password into a hashing algorithm. Once the hash is created, the user can select to either use the complete or part of the hash as a password. Therefore, the user has removed the threat of a direct dictionary attack and created a second layer of password security. To gain access to the password an attacker would need to know the original password, the hashing algorithm and how much of the hash was used or somehow capture the password with a key logger.

2.7.7 Accessing Physical Memory

There are numerous computation tasks that require the processing of very large amounts of data that are too large to be stored in random access memory (Barnat, et al., 2009). Therefore the only option is to store the data on off board hard drives. Storing the data off the motherboard creates an additional delay for processing. There are three levels of memory storage, Compact Processing Unit (CPU) Cache – Level 1 and 2, Random Access Memory (RAM) and Hard Disk Drives / Solid Disk Drives (HDD/SSD).

Accessing data from CPU cache takes approximately a nanosecond (10^{-9} seconds), accessing RAM can take 10^{-8} of a second, while accessing data from an off board hard disk is several milliseconds (10^{-3}) which is one million times slower (Vitter, 2001 pp. 209-271). Research has revealed that hashing can be used to achieve an increase in access speed.

In *External memory Algorithms and Data Structures: Dealing with Massive Data,* the author describes the current challenge with accessing data from external media is the slow input / output speeds (Vitter, 2001, pp. 209-271). The slow input/output speeds create a bottleneck where the CPU has to wait for the hard drive to access its stored data before it can be processed by the CPU. Vitter's proposed hash based technique functions by creating multiple hashing tables with hash reference pointers to the data block stored on the hard drive. Thus, instead of having one reference pointer per item, a hash reference pointer can be used to show the location of multiple items, resulting in only two accesses to the memory. The first time is to access the hash table and the second time is to access the data block. This technique could have significantly lower access time when compared to current techniques that require multiple accesses to the table and data block.

2.7.8 Digital Forensics and Anti Forensics

Hashing is a crucial tool available to experts in the field of digital forensics. It is best practice for a forensic investigator to perform all testing and analysis only on forensic copies of the evidence. A forensic copy of the evidence is an exact replication of the original evidence that is stored on a separate hard drive. Unlike other forensic industries, digital forensics allows the expert to perfectly replicate the original evidence with preservation. Any changes in the original evidence can stop the evidence from being admissible in a court of law. Furthermore, retesting maybe required if two forensic experts disagree on a certain aspect of the evidence. If the evidence has been changed in any form by the forensic expert during the original testing, additional testing may produce a different result. Furthermore, it can remove the credibility of the forensic expert resulting in all previous evidence submitted or handled by that investigator being redacted from the case. To manage this risk, a hashing algorithm is used to verify that the forensic copy of the evidence is an exact copy of the original evidence. Typically, in digital forensics an investigator will use the SHA-1 or MD5 hashing algorithms as these are less susceptible to collision and have become "court friendly".

Anti-forensics is a group of techniques that can be applied by criminals to computer crimes to complicate the forensic analysis stage of an investigation. There are numerous different techniques that are available from file encryption, transport encryption, editing file signatures and editing or cropping images. File

encryption is similar to any other form of encryption. It is the technique for converting plain text in to encrypted text. There are numerous different types of encryption technologies that have been proven to be unfeasible to extract evidence when they have been implemented correctly. Transport encryption is a type of encryption that typically automated where the message is encrypted before transmission and is decrypted after transmission. The advantage of this technique is that the data cannot be read by eavesdroppers. Another technique is to edit the file signature. A file signature refers to the format of a file. For example, text files will have the extension .txt and an image file might have an extension of .jpg. or .png. To manually edit the file signature, a user can easy download and open a file in a hex editor. Once the hex editor is open they are able to manually change the file magic number (the first 2-4 bytes of message) to represent another file format. If the file was using a different file signature, it may be overlooked by forensic software. Another simple technique is to edit or crop the image. Criminals may invert the colours of the image before transmission so systems such as Content Based Image Retrieval (CBIR) that rely on skin tone and textures have difficulty in detection.

2.8 STRENGTHS AND WEAKNESSES OF HASHING ALGORITHMS

The aforementioned examples of implementing hashing algorithms have shown they can be developed and adapted to assist a range of different technologies. The next section will look at the shortcomings and deficiencies of using hashing algorithms with a focus on image retrieval.

The first aspect that will be discussed is the accuracy of a hash based image retrieval system. The accuracy will be segmented into two sections. The first section will analyse the accuracy of hashing algorithms and the second section will review how the accuracy relates to detecting small changes in an image. Hashing algorithms have been extensively and comprehensively tested by numerous mathematicians and governments around the world. One of the main requirements for a hashing algorithm is that the same hash value needs to be outputted when the same message is inputted. Hashing algorithms look at a file from the binary level therefore it does not discriminate between file types, naming conventions, file contents, and so on. Due to this non-discriminatory aspect,

hashing algorithms will process a file of any format of the same binary length equally. However, because a hashing algorithm looks only at the complete binary string even a small change to the string will create a completely different hash value. For example in relation to objectionable images, if there are two identical images and a user makes a change to one pixel in one of the images, the binary string of the image will change. Therefore when an investigator compares the hash value of the two images, they will be different, even though the image appears to be identical to the human eye and both are objectionable.

The next weakness of hashing algorithms is how research and development are always discovering new techniques to make hashing algorithms more efficient with larger bit lengths. This may appear as an advantage to a hash based system, however it also creates complications. Any hash based system requires the same hashing algorithm to be used for the entirety of the software's life. Therefore, even when a new hashing algorithm is developed the HBIR would not be able to implement as it would create unrelated hash values. Therefore it is critical to the research that a effective hashing algorithm is implemented.

2.9 CONTENT BASED IMAGE RETRIEVAL

The content based image retrieval (CBIR) technique is a different method for the retrieval by allowing the user to search the image population by using an image or drawing, instead of using keywords. CBIR systems are employed in the field of pattern recognition industries such as entertainment, art, fashion design, advertising, history, medicine, law enforcement, copyright and commercial industry (Rashedi, et al., 2012).

CBIR was first demonstrated in *Database architecture for content-based image retrieval* (Kato, 1992). The research approach was to map normalised trademark images to an 8 x 8 pixel grid and calculate shape measurements from each image calculating the pixel frequency distributions (Eakins, 1998, p.53). This was a very primitive and limited method. In 1995 researchers at IBM presented a more complex method which was first documented in the Institute of Electrical and Electronic Engineers (IEEE) journal under the title of *Query by Image and Video Content: The QBIC System* (Hafner, et al., 1995). The Query By Image Content (QBIC) system has gained much attention over the last 15 years as

the leading CBIR technique. The authors first begin to explain how a computer can have difficulty in recognising or identifying a certain object. The authors use the analogy of trying to identify dogs in a children's book. They explain that 3 year old children can easily identify a dog in the book, however computers have great difficulty because they do not know definitively what a dog looks like (Flickner, et al., 1995). Unlike Kato's CBIR system, QBIC looks at significantly more attributes in an image. The QBIC system has two main components, adding an image to the database population and query images in the database population. QBIC can query on colour, texture, shape, multi objects, sketch, location and text. Each of the attributes is given a priority of importance, respectively.

To add an image, QBIC receives the inputted image (or video) from the user, extracts the aforementioned attributes and stores them in a database for indexing. To search, or query the database for similar images, the user inserts the image or drawing and the system extracts the attributes of the image. QBIC then compares the extracted attributes to the stored attributes of the image population. The images are then ranked by how similar they are to the inserted image and listed accordingly.

There is no universal accepted model for CBIR systems. However, QBIC does have a strong following in practice and commercially. Other competing models are Virage, Pichunter, VisualSEEK, Chabot, Exaclibur, Photobook, Jacob and Digital Library Project (Stojanovic, et al., 2007). The challenge with any CBIR model is processing time. The model needs to balance the accuracy of retrieving the correct images with the total processing time. A model that supports large quantities of data extraction will increase the accuracy, but will also require more processing time. On the contrary, if the model does not support adequate data extraction, the accuracy of the model will be poor. Furthermore, it is also important the attributes extracted are the most beneficial to support accuracy and speed. In summary, different combinations and testing of all the different factors needs to be completed to make the CBIR model most effective for users.

In *Improving Response Time by Search Pruning in a Content-based Image Retrieval System Using Invert File Techniques* the authors investigate methods for increasing the speed to query the database (Squire, et al. 1999). The researchers explain Concept Based Image Indexing (CBII) has had 30 years to explore new techniques for reducing the query time (Squire, et. al., 1999) where CBIR is

relatively still new. The researchers propose a different technique for CBIR systems by using Invert File Techniques. Inverting a file is a method for querying a large database by its location and data string. For example #8B0000: {2, 10, 15, 18}. The first section represents the colour and the second section represents the location reference in the image. Using Inverted Files has been proven to be faster when being compared to typical database query structures, but it is still significantly slower than using a CBII retrieval techniques.

One approach to help identify objectionable material using the CBIR technique is documented *Image Processing Techniques to Detect and Filter Objectionable Images based on Skin Tone and Shape Recognition* where the authors use a 3 stage process to identify objectionable images (Drimbarean, et al.,2001). The three stages in order were, "detect skin tone", "shape detection" and "other detection techniques". In the detect skin tone stage, the algorithm looks at the pixels that fall within a specified range colours of skin tones and analyses the surrounding pixels by using fuzzy clustering. Fuzzy clustering is a technique where the image is segmented into groups containing similar pixels. The structure of the group (or cluster) will arrange the pixels where pixels that have similarities with other groups are located on the outside of the cluster, closer to the other similar groups. Therefore the cluster does not just show where the pixel is grouped, but also shows what similarities the pixel has with other groups (Nock & Nielson, 2006).

The system logic works in three stages, the first stage consists of detecting the amount of potential skin coloured pixels. If the number of potential skin coloured pixels detected is higher than a predetermined number, the image is then passed to the next stage. The second stage identifies human faces in conjunction with skin colour. If a human face is detected the image is then passed to the final stage. The final stage uses additional algorithms such as texture analysis to determine if the skin coloured pixels are actually skin or simply an object that falls within the skin colour range. By using a three step process the authors try to segregate the non-objectionable images from the objectionable by slowly refining the number of images. The author's state the overall time for processing is 10 - 100 images per second depending on the pixel size of the image (Nock & Nielson, 2006).

The authors Bala, Aakash, Anand and Chandra (2010) follow with a skin based detection of objectionable images in *Intelligent Approach to Block Objectionable Images in Websites* by demonstrating the accuracy of a skin based system. The authors demonstrate their algorithm by implementing the system in a web portal. The authors explain when a user tries to upload an image through the web portal, it will be determined if the image is objectionable or non-objectionable. If the image is objectionable then an error message will appear. The authors state the time taken to detect the objectionable image was almost "no time" (Bala, et al., 2010) at all. However, the authors did not investigate batch processing of images, for example trying to consecutively upload 100 images and recording the time.

The two approaches above to try and detect objectionable images both relied on face detection. However, it is expected that not all objectionable images will contain the facial characteristics. Therefore, by relying on a critical attribute to identify objectionable images which may not always be present is severely decreasing the accuracy in identification and therefore the accuracy of the entire method.

In *Skin color detecting unite YCgCb color space with YCgCr color space*, the authors take a slightly different approach by combining different techniques that do not rely on facial recognition. Instead the approach uses YCgCb colour space and RGB colour space together in conjunction with texture and white balance (Zhang & Shi, 2009). YCgCb and RGB are abstract mathematical models which describe the range of colours as tuples of numbers (ArcSoft, 2013). The research focuses on accuracy of identification and not processing time. The researchers demonstrate their algorithm by testing with 500 images (300 with complicated backgrounds and 200 with basic backgrounds). The results are tabulated in table 2.3 below.

Table 2.3 showing results of Skin Colour Detection under Complex Background.

TA BLE.1
CONTRASTIVE ANALYSIS OF SKIN COLOR HIT RATE

Objects	Cb-Cr threshold value method [11]	The method by Hus [4]	The method of Gaussian model [13]	The method provided by this paper
normal backgrounds images	82.4%	85.3%	91.0%	92.1%
complicated backgrounds images	80.1%	83.5%	83.6%	90.4%

TABLE.2
CONTRASTIVE ANALYSIS OF SKIN COLOR FALSE RATE

Objects	Cb-Cr threshold value method [11]	The method by Hus [4]	The method of Gaussian model [13]	The method provided by this paper
normal backgrounds images	9.2%	7.9%	6.5%	6.0%
complicated backgrounds images	14.6%	13.1%	13.7%	6.9%

(Zhang & Shi, 2009, p.224)

Table 2.3 illustrates, by using the new technique developed by the authors, they were able to increase the accuracy of identifying images containing skin with normal and complicated backgrounds.

In *Image Retrieval in Forensics: Tattoo Image Database Application*, the authors demonstrate by using previously taken photos by law enforcement they are able to create a database with a large range of images (Jung-Eun, et al., 2012). Recording tattoos worn by criminals for identification purposes first appeared in Miami in 1966 (Hicks, 1966). Now, the images are assisting in helping police associate criminals with other criminals and helping to identify potential suspects (Jung-Eun, et al., 2012).

Overall CBIR is relatively new technique and is starting to gain momentum in the image retrieval industry. However, there is still a significant amount of research that is still required for it to reach the level of CBII.

2.10 CONCEPT BASED IMAGE INDEXING

Concept Based Image Indexing (CBII) is the second method of image retrieval. CBII focuses on identifying an image with metadata and keywords. CBII is a very quick and where the accuracy of system is completely dependent on the accuracy of the metadata that is extracted from the image. The metadata includes the filename, file size, date taken, dimensions, bit depth, name, item type (file format), date created, date modified, size, file owner and custom key words. CBII

was the first technique that could be used to search and retrieve images on behalf of the user. In comparison with CBIR, there has been little advancement in CBII techniques in the last decade.

In contrast to CBIR, CBII is a very simple system. However, both systems require the same two main functions. The first function is having the ability to add an image to the image population index and secondly, to query the population. To add an image to the population, the image is selected or inserted into the indexing algorithm (which is often automated by a dynamic indexer). The indexing algorithm will identify and extract all metadata from the image and store it in a database. The indexer will also weigh the different metadata depending on its importance to maximise the accuracy. This can be shown in the Table 2.4 below.

Table 2.4 showing traditional CBII database structure

File Name	File Size	Data Taken	Dimensions	Bit Depth	Key words
Image_001	336.35KB	12.01.11	2200x2590	16	House, green, expensive
Image_002	256.87KB	12.01.11	2200x2590	16	Null
Image_003	567.32KB	12.01.11	2200x2590	16	Expensive, nice, car, blue
…	…	…	…	…	…

The second function is querying the database by using keywords. For example, if a user wanted to search for images of red hats, the user would simply type red hats. The system would receive the keywords and query the database to search for any images that contain that word. A list of images would then be displayed showing the image with the highest accuracy at the top. Researchers and developers found even though the system was simple, fast and accurate, the invention of digital cameras allowing users to take millions of images containing similar or identical metadata decreased the accuracy of the image retrieval.

In the online environment, website administrators started to abuse this system for advertising purposes by adding incorrect keywords to the image. For instance, the website administrator would insert the keyword ‘kitten’ into a picture of a dog. This resulted in the algorithm incorrectly indexing the image of a dog as a kitten. When a user would search for pictures of dogs, they would then

be presented a picture of a kitten. Large companies such as Google, Yahoo and AskJeeves quickly became aware of this problem causing Google to release a game called Google Image Labeller. Google Image Labeller was a game where two players competed with one another to enter tags or keywords for images (Saini, 2008) in the hope it would remove incorrect keywords and metadata from an image. The game also created a source of free labour where users enjoyed competing and secondly where Google could refresh their databases with new associated words to increase accuracy of image retrieval. If Google had to pay for the labour on the other hand, it would be very expensive. Therefore the main reason developers have moved away from CBII is the related costs (Lehmann, et al., 1999).

One advantage of using a CBII system is that it does not restrict the user to search only for images. The technique is very universal and can used to retrieve any file.. However, a problem arises when a user searched for words that are synonyms ('different words, same meaning') or homonyms ('same word, different meaning'). For example, searching for *cars* would not display images that have the keyword *vehicles*. Traditionally, there would not be a relationship between the two words in a CBII system.

In *Implicit Concept-based Image Indexing and Retrieval,* the authors look at a method to increase the accuracy of retrieval by structuring the database on the hierarchical Ternary Fact Model (TFM). The TFM relies on a visual entity relationship index representation (Azzam, et al., 2004). The authors explain that the TFM model has been well researched and defined to be efficient and has a good performance. It functions by structuring the database differently (refer to Table 2.4). The authors use a three level process where the first fact level is very specific in explaining an object in the image. The second level is automated by associating the first level objects to larger categories. The third level is associating the objects in the second level to larger categories again. This can be demonstrated in Figure 2.6 below.

Figure 2.5 showing example of an image used in TFM

(Funwallz, 2014)

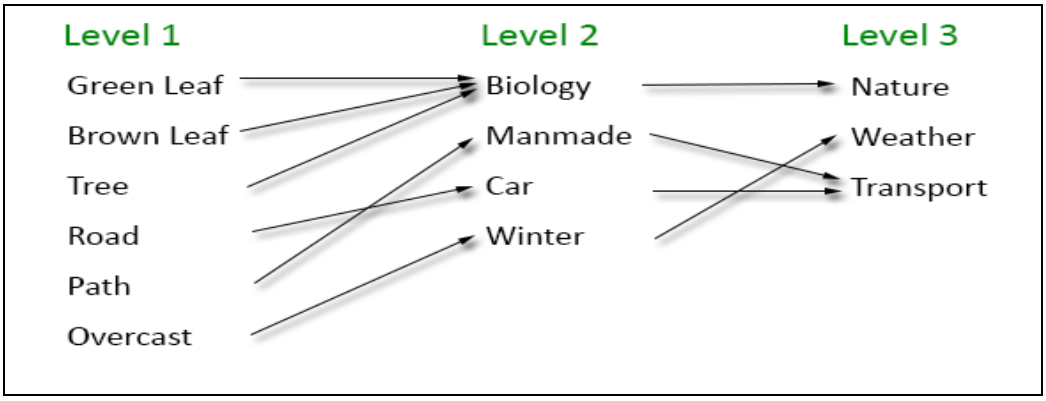


Figure 2.6 showing an example of a TFM break down of Figure 2.5

The main advantage of using this technique is that the system can create the key associated words automatically without interrupting the user, with the benefit of increasing the accuracy of the image retrieval. The database structure is demonstrated below.

LOW LEVEL TABLE:

Table 2.5 showing example of Low Level Table for Figure 2.5

Low Level Fact ID (LFID)	Low Level Fact Name (LLFN)
001	Leaf
002	Dirty
003	Tree
004	Road
005	Path
006	Overcast

INCLUSION TABLE:

Table 2.6 showing example of an inclusion table for Figure 2.5

LFID	MFID	Image	Weight	Available	Inclusive	Value
001	001	001.jpg	0.408	1	0.98	0.499
002	001	001.jpg	0.417	1	1	0.417
003	001	001.jpg	0.175	1	0.42	0.073

MID-LEVEL TABLE:

Table 2.7 showing example of a Mid-Level Table for Figure 2.5

Mid-Level Fact ID	Mid-Level Fact Name
001	Biology
002	Manmade
003	Car
004	Winter

(Azzam, et al., 2004, p.58)

The authors did not document the process time. However, as the CBII is naturally a very fast technique, it is expected the main objective was to solve the main challenge with CBII of increasing image accuracy.

In *Concept Based Indexing of Annotated Images Using Semantic DNA* the authors propose a different CBII technique which the authors refer to as "semantic chromosomes" (Fadzli & Setcha, 2012). Unlike the hierarchical Ternary Fact Model that relied on just one model. The authors decided to use a combination of hierarchical models which contain a detailed flat structured model.

The main objective in their research was to remove the problems related to synonyms ('different words, same meaning') and homonyms ('same word, different meaning'). Using the previous example of each of the Low Level Fact Identifiers (LFID) the authors also include the words that have the same meaning. For instance, *tree* might be referred to as *bush* or *shrub* and *road* might be referred to as *street* or *place*. The database would be structured in similar to table 2.8.

Table 2.8 showing keyword with single DNA representation

ID (DNA)	Key words
1	Craftiness

2	Leaf
3	Unclean
4	Road
5	Pretence
6	Cloudy
7	Tree
8	Winter
9	Dirt
10	Mud
11	Ground
12	Muddy
13	Overcast
14	Circumvention
15	Fraud
16	Cheat
17	Path
18	Untruth
19	Street
….	….

Table 2.9 showing key word examples with multiple DNA associations

Semantic DNA	Concept	Sense	Paragraph Content
19 -17 – 11 - 9	Road	Physical Road	Something we can physically walk on
8 – 13 - 27	Cloudy	Physically cloudy	Related to how the weather is
10 – 12 - 11	Dirty	Physical dirt	You boots have dirt on them

For the keyword of *dirty* there is also synonyms such as 16-15-14-18. However, these words do not mean the same thing as a *dirty road*. Therefore, the table will only show the "DNA" of words that have the same meaning. This removes the problem of when a user searchers *dirty car*, they get a picture of a person who has 'done the dirty' (cheater) and sitting in a car. CBII has been a very popular form of retrieval of all forms of files not only images. There are vast numbers of forensic software available that heavily rely on keyword associations for searching and identifying trends, specific files and history.

Overall CBII is a widely known and used technique that has stood the test of time. Due to the history behind this technique it has been developed into an extremely fast, accurate and efficient system when searching by using textual keywords. The majority of recent research in image retrieval is CBIR.

2.11 LEGAL

Unlike many other crimes to society such as murder, rape and theft, which require the person to be in the country at the time to commit the crime. Computer crimes can be committed by residing in one country and committing crimes in one or more other countries simultaneously. The Internet has provided an unprecedented new platform which has made not only the level of available objectionable material increase, but also other cybercrimes such as fraud, theft and money laundering. Consequently, the different laws and different jurisdictions have made it very difficult and expensive for police and government agencies to investigate.

There are four main issues related to Information Technology and Law. The issues are internet access and usage, privacy, freedom of expression and jurisdiction. In New Zealand, the legislation surrounding these factors are regulated by the Criminal Act 1961 including amendments, the Films, Videos, Publications and Classifications Act 1993, the Privacy Act 1993 and the New Zealand Bill of Rights Act 1990.

There are three sections relating to objectionable material. The sections are section 123 and 124 of the *Films, Videos and Publications Classification Act 1993* and section 131A of the *Criminal Act 1961 (including amendments).* Under *the Films, Videos and Publications Classification Act 1993* section 123 *Offences of strict liability relating to objectionable publications* it states;

Section 123
Offences of strict liability relating to objectionable publications

- (1)Every person commits an offence against this Act who—
    - (a) makes an objectionable publication; or
    - (b) makes a copy of an objectionable publication for the purposes of supply, distribution, display, or exhibition to any other person; or
    - (c) imports into New Zealand an objectionable publication for the purposes of supply or distribution to any other person; or

- (d) supplies or distributes (including in either case by way of exportation from New Zealand) an objectionable publication to any other person; or
- (e)has in that person's possession, for the purposes of supply or distribution to any other person, an objectionable publication; or
- (f) in expectation of payment or otherwise for gain, or by way of advertisement, displays or exhibits an objectionable publication to any other person.

(2)Every person who commits an offence against subsection (1) is liable,—

- (a)in the case of an individual, to a fine not exceeding $10,000:
- (b) in the case of a body corporate, to a fine not exceeding $30,000.

(3)It shall be no defence to a charge under subsection (1) that the defendant had no knowledge or no reasonable cause to believe that the publication to which the charge relates was objectionable.

(4)Without limiting the generality of this section, a publication may be—

- (a)supplied (within the meaning of that term in section 2) for the purposes of any of paragraphs (b) to (e) of subsection (1); or
- (b)distributed (within the meaning of that term in section 122) for the purposes of any of paragraphs (b) to (e) of subsection (1); or
- (c)imported into New Zealand for the purposes of paragraph (c) of subsection (1),—

not only in a physical form but also by means of the electronic transmission (whether by way of facsimile transmission, electronic mail, or other similar means of communication, other than by broadcasting) of the contents of the publication.

Section 124
Offences involving knowledge in relation to objectionable publications

- (1)Every person commits an offence against this Act who does any act mentioned in section 123(1), knowing or having reasonable cause to believe that the publication is objectionable.

  (2)Every person who commits an offence against subsection (1) is liable,—

  - (a)in the case of an individual, to imprisonment for a term not exceeding 10 years:
  - (b)in the case of a body corporate, to a fine not exceeding $200,000.

In the Criminal Act of 1961 it states:

Section 131A
Offence to possess objectionable publication

- (1)Subject to subsections (4) and (5), every person commits an offence against this Act who, without lawful authority or excuse, has in that person's possession an objectionable publication.
  (2)Every person who commits an offence against subsection (1) is liable to a fine not exceeding,—
  - (a)in the case of an individual, $2,000:
  - (b)in the case of a body corporate, $5,000.

  (3)It shall be no defence to a charge under subsection (1) that the defendant had no knowledge or no reasonable cause to believe that the publication to which the charge relates was objectionable.
  (4)Nothing in subsection (1) makes it an offence for any of the following persons to be in possession of an objectionable publication, where such possession is for the purpose of and in connection with the person's official duties:
  - (a)the Chief Censor:
  - (b)the Deputy Chief Censor:
  - (c)any classification officer:
  - (d)any person holding office pursuant to clause 2 of Schedule 1:
  - (e)any member of the Board:
  - (f)the labelling body or any person who is carrying out the functions of the labelling body:
  - (g)any Inspector:
  - (h)any constable:
  - (i)any officer of the Customs:
  - (j)any Judge of the High Court, or District Court Judge, Coroner, Justice, or Community Magistrate:
  - (k)in relation to any publication delivered to the National Librarian pursuant to Part 4 of the National Library of New Zealand (Te Puna Mātauranga o Aotearoa) Act 2003, the National Librarian, any other employee in the department responsible for the administration of that Act, or any person employed in the Parliamentary Library:
  - (l)any other person in the service of the Crown.

  (5)It is a defence to a charge under subsection (1) if the defendant proves that the defendant had possession of the publication to which the charge relates, in good faith,—
  - (a)for the purpose or with the intention of delivering it into the possession of a person lawfully entitled to have possession of it; or
  - (b)for the purposes of any proceedings under this Act or any other enactment in relation to the publication; or
  - (c)for the purpose of giving legal advice in relation to the publication; or
  - (d)for the purposes of giving legal advice, or making representations, in relation to any proceedings; or
  - (e)in accordance with, or for the purpose of, complying with any decision or order made in relation to the publication by the Chief

Censor, the Classification Office, the Board, or any court, Judge, Justice, or Community Magistrate; or

- (f)in connection with the delivery of the publication to the National Librarian in accordance with Part 4 of the National Library of New Zealand (Te Puna Matāuranga o Aotearoa) Act 2003.

(6)Nothing in subsection (5) shall prejudice any defence that it is open to a person charged with an offence against this section to raise apart from that subsection.

(7)For the avoidance of doubt, in this section the term proceedings includes proceedings before the Classification Office.

One of the main issues that contributes to the difficulty of prosecuting international crime is that each country and a different law with different penalties if found guilty.

In New Zealand, on the 25th of May 2013 Justice Minister Judith Collins introduced the Objectionable Publications and Indecency Legislation Bill into Parliament. This bill is aimed at increasing the penalties for child and internet related offences. In an interview with media, Collins explains that 400 people had been convicted of the offence of having objectionable material between 2004 and 2011, but only 33% were jailed (Rutherford, 2013). Some of those people not jailed are shown in table 2.10.

Table 2.10 showing New Zealand criminals who received home detention

Name	Age	Crime	Punishment	Reference
Bryce Butler	44	Caught a second time with possession of objectionable material	9 months home detention	(Kidd, 2012)
Rowan Pearce	29	Possession objectionable images	7 months home detention	("Child porn hoarder", 2009)
John McNeil	67	Possession objectionable videos	7 months home detention	(Clarkson, 2013)
Dean Cates	25	Possession of objectionable images	7 months home detention	(Kidd, 2012)
John Howland	38	Possession of thousands of images	8 months home detention	(Carville, 2013)
John Hubbard	66	Possession of objectionable images	4.5 months home detention	(Humphreys, 2011)

In comparison to similar crimes committed in other jurisdictions in other countires are tabulated below.

Table 2.11 showing punishments of foreign countries for possession of objectionable images

Location	Name	Age	Crime	Punishment	Reference
Canada	Kelly John Schoenroth	22	Possession of objectionable images	1 year sentence	(Polischuk, 2014)
England	Richard Lovett	23	Possession of objectionable images	7 months sentence	("Man jailed for", 2014)
America	Jeffrey L. Perry	56	One charge of receiving objectionable images	5+ year sentence	(Leader, 2013)
Australia	Lynton John Moore	30	Possession of objectionable images	15 month sentence	(Bembridge, 2013)

The two tables easily demonstrate the differences in the level of punishment for similar crimes. There are numerous challenges for governments and police enforcement worldwide and there is a disconnect, rapidly widening, between the speed of technological innovation and the laws set up to govern what goes on in cyber space (Yiannopoulos, 2010).

2.12 CONCLUSION

In this chapter, an introduction to hashing and hashing algorithms was introduced. The introduction was followed by Section 2.2 through to Section 2.5 where cryptographic and non-cryptographic hashing algorithms were reviewed and calculated. Section 2.6 focused on collisions generated by hashing algorithms and the two solutions for reducing the number of collisions. Section 2.7 reviewed current technologies where hashing algorithms have been implemented. This was followed by Section 2.8 where strengths and weaknesses of hashing algorithms were clarified. Once the hashing component of the literature review was completed, content based image retrieval (Section 2.9) and concept based image indexing (Section 2.10) were critiqued. Legal issues surrounding objectionable images were discussed in Section 2.11.

Chapter 3, the methodology chapter, will now summarise issues and gaps in the current research (Section 3.1). This will be followed by identifying the research questions and hypotheses (Section 3.2), the research design (Section 3.3), the system design (3.4), the four research phases (Sections 3.5- Section 3.8) and the data requirements of the research (Section 3.8).

Chapter Three

RESEARCH METHODOLOGY

3.0 INTRODUCTION

As discussed in chapter 2, there are two main techniques that investigators are currently employing to identify objectionable images. Both of these techniques have their advantages and disadvantages.

Chapter 3 will begin by discussing gaps in current research and identifying potential areas that may further be investigated in image retrieval. Once the potential research questions have been identified, Section 3.2.1 will identify the main research problem which will be analysed until a main research question is developed. The main question will then be examined until several sub questions can be constructed. Once the main question and sub questions have been developed, Section 3.2.2 will discuss creating logical and robust hypotheses. Section 3.3 will focus on the research design and methodology. Section 3.3 will include how the research will be directed and controlled by the methodology to support quality research. Section 3.4 will discuss the proposed Hash Based Image Retrieval system and the CBIR and CBII system architectures. Section 3.5 will introduce and clearly outline research phase one of the experiment which initiates the procedures for software testing and pilot testing. Section 3.7 will review the experimental testing for research phases two, three and four. Section 3.8 will show the data requirements of the experiment and more precisely how the evidence will be generated, collected and presented. Section 3.9 will identify the expected outcomes and the limitations of the research. Section 3.10 will conclude the research methodology chapter.

3.1 ISSUES AND GAPS IN CURRENT RESEARCH

The literature review in chapter 2 has identified numerous technical issues related with identifying and retrieving objectionable images as well as legal issues surrounding investigation and prosecution. For ease of understanding, the issues have been categorised into five individual groups. The groups are as follows;

image processing, accuracy, legal and political issues, electronic anti-forensics and creating a long term solution.

Image processing time is the total amount of time the system takes to process a single image. The processing time is a critical area of focus because it directly relates to the financial costs of hardware, which in turn reflects on the amount of data that can be processed at any one time. A delay in processing can also delay legal proceedings which may create additional costs in an investigation. The image processing time will differ between the image retrieval techniques. As the CBIR, CBII and the proposed HBIR techniques all retrieve images on different data, it is expected that not all image processing times will be equal. The goal in this area would be to identify a technique that can process images faster with a higher level of accuracy than current techniques.

Accuracy of image retrieval is the ability of the system to perform tasks with precision and exactness. As previously discussed in chapter 2, accuracy is highly dependent on the amount of available resources and the image retrieval technique that has been implemented. For example, the CBIR technique requires significantly more resources per image. However, the extra resources produce higher accuracy. In contrast, CBII techniques require a lower amount of resources, creating significantly less accuracy. Therefore depending on the availability of resources, the forensic investigator has two options, a fast inaccurate technique or a slow accurate technique. There is a large gap in research in identifying a more balanced technique which has the advantages of being accurate, but less resource demanding.

Legal and political issues are all problems related to laws, regulation and politics that include, evidence collection, human rights and criminal activities. The current laws that regulate objectionable imagery in New Zealand (NZ) are the Criminal Act 1961 including amendments, Films, Videos, Publications and Classifications Act 1993, the Privacy Act 1993, the New Zealand Bill of Rights Act 1990 and the Evidence Act 2006. The largest problem domain currently faced in this area is jurisdiction. Jurisdiction affects the access and authority that one government or party has over I.T. infrastructure is controlled by another government or party. In New Zealand it is “the extent of a person's or body's powers or authority” (“A guide to legal”, 2013) which include “the boundaries of any domestic court's influence overseas” (“A guide to legal”, 2013). The second

issue is political motivations or agendas. It is well known that governments often disagree influencing international and state policies and regulating what people can and can't do in different jurisdictions. However, it is less known that political motivations or agendas can also have dire consequences for solving international computer crimes. Historically, a crime was prosecuted where the actual criminal activity took place and not necessarily where the person was physically located. When governments have deportation agreements one government would request the deportation of a criminal from the other government. However, a problem can arise when two governments do not have a deportation agreement in place to force one government to deport a criminal. These political factors contribute to the extreme difficulty of investigating and prosecuting international cybercrime. The third legal issue is how governments distinguish the differences between objectionable images and non-objectionable images as well as the associated level of punishment. This makes it difficult to create one international law with a fair punishment for objectionable image related offences. The fourth issue is even when governments have good political relationships they do not always have the financial or technical resources to investigate highly technical crimes. Until the legalities and politics are centrally managed, forensic research in this area will be limited and have little benefit to the research community.

Electronic Anti-Forensics is a group of techniques that can be applied by criminals to computer crimes to complicate the forensic analysis stage of an investigation. There are numerous different techniques that are available from file encryption, transport encryption, editing file signatures to editing or cropping images. Each of the techniques involves a high level of complexity however the majority of the techniques are already known to forensic analysts and researchers. Forensic Analysts are aware that when certain methods have been deployed by a criminal it is unfeasible to extract evidence. To publish the techniques that are already widely known would not be new research and secondly, publishing new anti-forensic techniques would be anti-productive in solving crime and is ethically challenging. Due to this reason this area will not be investigated.

The last group which is the most difficult aspect of the research is to try and discover a long term solution that can be used in the foreseeable future. Technology has the ability to advance at an incredibly fast pace, this does not just

create a problem from the legal perspective but it also has shown the short life expectancy of software systems. A recognisable example of how software is quickly outdated can be seen at how often software vendors release updated software and remove support for older software. Therefore, creating a long term solution is a complicated task and can only be judged once the technique has been in the industry for numerous years. However, there are still best practices that can be applied to maximise the potential life time of a software system. Longevity is an important goal of this research.

As chapter 2 has demonstrated, users are storing an exponential amount of media in cloud environments. There is a great opportunity to research into areas that have the ability to not just detect images on home devices, but also to detect images stored on the cloud platform. Furthermore, the large cloud platforms are physically controlled by American based companies such as Microsoft, Google and Amazon who are legally bound by the law of the jurisdiction. This makes it easier from a technical and legal aspect to prosecute. From a system processing point of view, some cloud service providers are by default creating hash values of all stored files for backup purposes and error correction. Therefore, the first task of creating the hash value has already been completed by the cloud provider. In addition, the cloud platform is designed to be elastic with resources making it easier to manage processing from an administration point of view.

Therefore, the proposed Hash Based Image Retrieval technique can be accurate, requires low computer resources, the images can be easy shared internationally by the hash value, it can be used in cloud technology and has real potential for reducing the amount of objectionable images stored online. For the aforementioned reasons the proposed research is current, innovative and exciting.

3.2 THE RESEARCH QUESTION AND HYPOTHESIS

The research question and hypothesis is the section where the main research question and hypothesis is constructed for the research. This section also includes the five sub questions and their associated hypotheses that will be used to support the main research question. The main question, sub questions and hypotheses were carefully and critically chosen to maximise the quality of the research.

3.2.1 Research Questions

Table 3.1 showing main Research Question and Main Hypothesis

Main Question	Main Hypothesis
How can the efficiency of image processing be increased to identify objectionable images in a smaller timeframe?	The HBIR technique will be a feasible solution to identifying objectionable images from a population containing objectionable and non-objectionable images.

Table 3.2 showing sub questions and selection justification

Sub Question	Selection Justification
SQ 1: **Will a hash based solution perform faster when compared to CBIR and CBII?**	By comparing the Hash Based Image Retrieval system to other systems it will create a bench mark. This question helps to support the credibility of the research and the technique.
SQ 2: **Which hashing algorithm category performs the fastest?**	To give the HBIR technique the best opportunity to be the most efficient, it is important that the most effective hashing algorithm is used.
SQ 3: **What is the genuine throughput of images per second with the most effective performing hashing algorithm?**	There are two types of throughput, theoretical and genuine throughput. In a genuine environment there will always be a bottleneck for processing. This sub question was selected to investigate the bottleneck to make the process even more efficient.
SQ 4: **How much is the processing of the images affected when the hardware is reduced?**	This question was selected because it is expected that by reducing the system hardware it will reduce the number of images that can be processed at any one time. However will half the RAM reduce the throughput by half?
SQ 5: **What effect does the size of the image have on processing?**	This question was chosen to look at how the file size of images may affect the processing time.

3.2.2 Hypotheses

The corresponding hypotheses to the sub questions are tabulated below.

Table 3.3 showing research hypothesis

Hypothesis 1: **The hash based image retrieval technique will perform faster than content based image retrieval and concept based image indexing techniques.**

Hypothesis 2: **Non-cryptographic hashing algorithms will perform faster than the cryptographic hashing algorithms.**
Hypothesis 3: **The hash based solution will process more than 100 images per second.**
Hypothesis 4: **Reducing the RAM from 8GB to 4GB will reduce the HBIR throughput.**
Hypothesis 5: **Larger images will take significantly longer to process when using the CBIR/CBII/CBIR techniques.**

3.2.3 Research Data Map

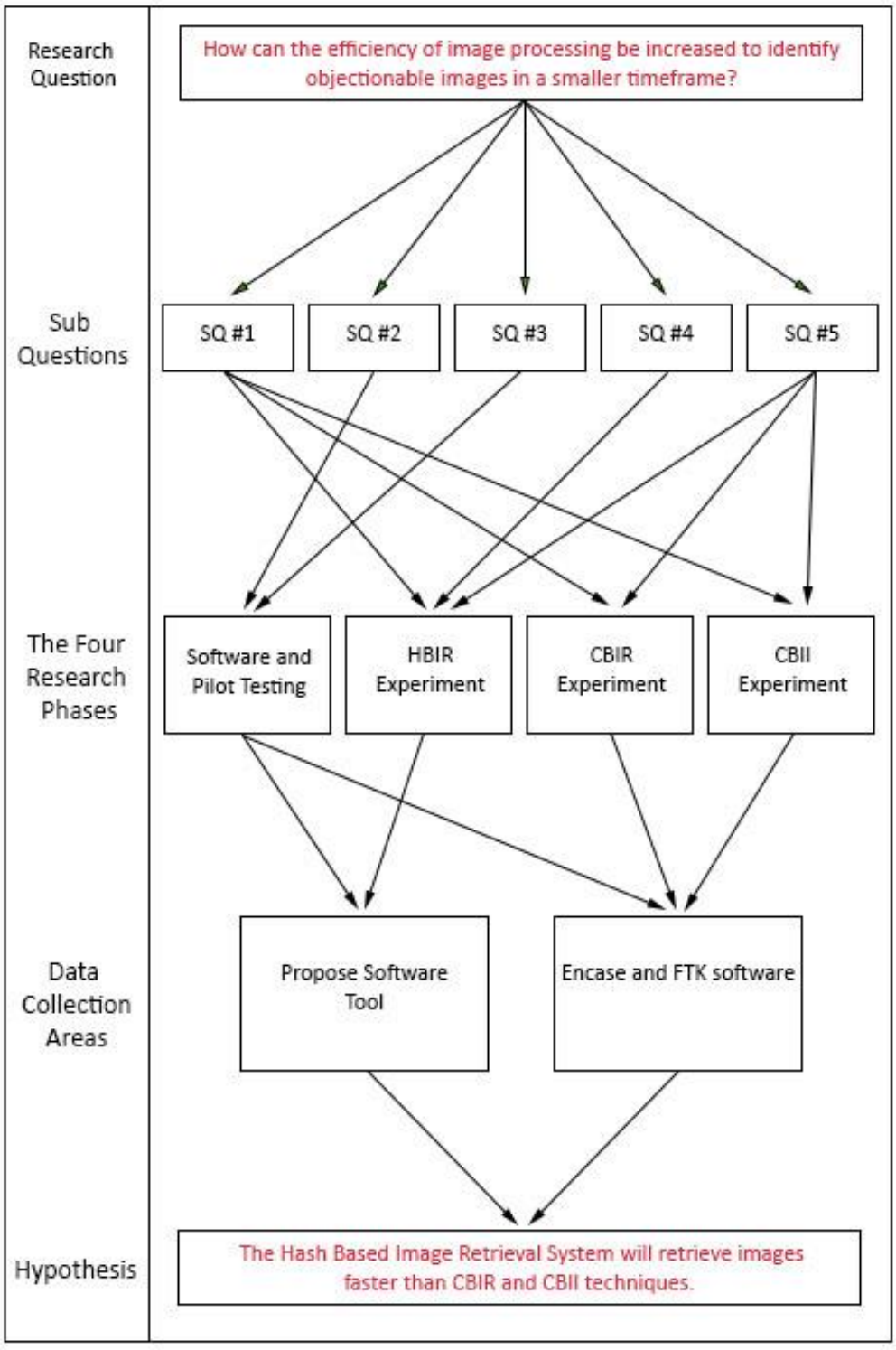


Figure 3.1 showing the research data map.

3.3 THE RESEARCH DESIGN

The research design section will specify the approach and the methodology that will be undertaken in the research. This will be followed by how the methodology is going to be implemented to do the experimental research.

A constructionist approach using the design science methodology (DSM) is to be used to conduct the research. The DSM is a flexible methodology allowing the researcher to slightly redirect the research while it is being undertaken. DSMs main objective is to create an innovative idea, practice, or technical product that can be rigorously analysed and be managed (Hevner, et al., 2004) to serve human purposes (Peffers, et al., 2006).

There are six practical rules or guidelines that must be applied when using the Design Science Methodology (Peffers, et el., 2007; Hevner, et al., 2004).

Table 3.4 showing six rules for DSM

Rule #	Description
1	Artefact created to address a problem
2	Here to resolve an important business problem
3	Utility, quality and efficacy
4	Must be rigorously evaluated (artefact and its evaluation)
5	The artefact should be a search process that draws from existing theories
6	The research must be effectively communicated to appropriate audiences

As Tuunanen (2006) illustrates, the design science methodology can be segmented into six sections as illustrated below in Figure 3.2.

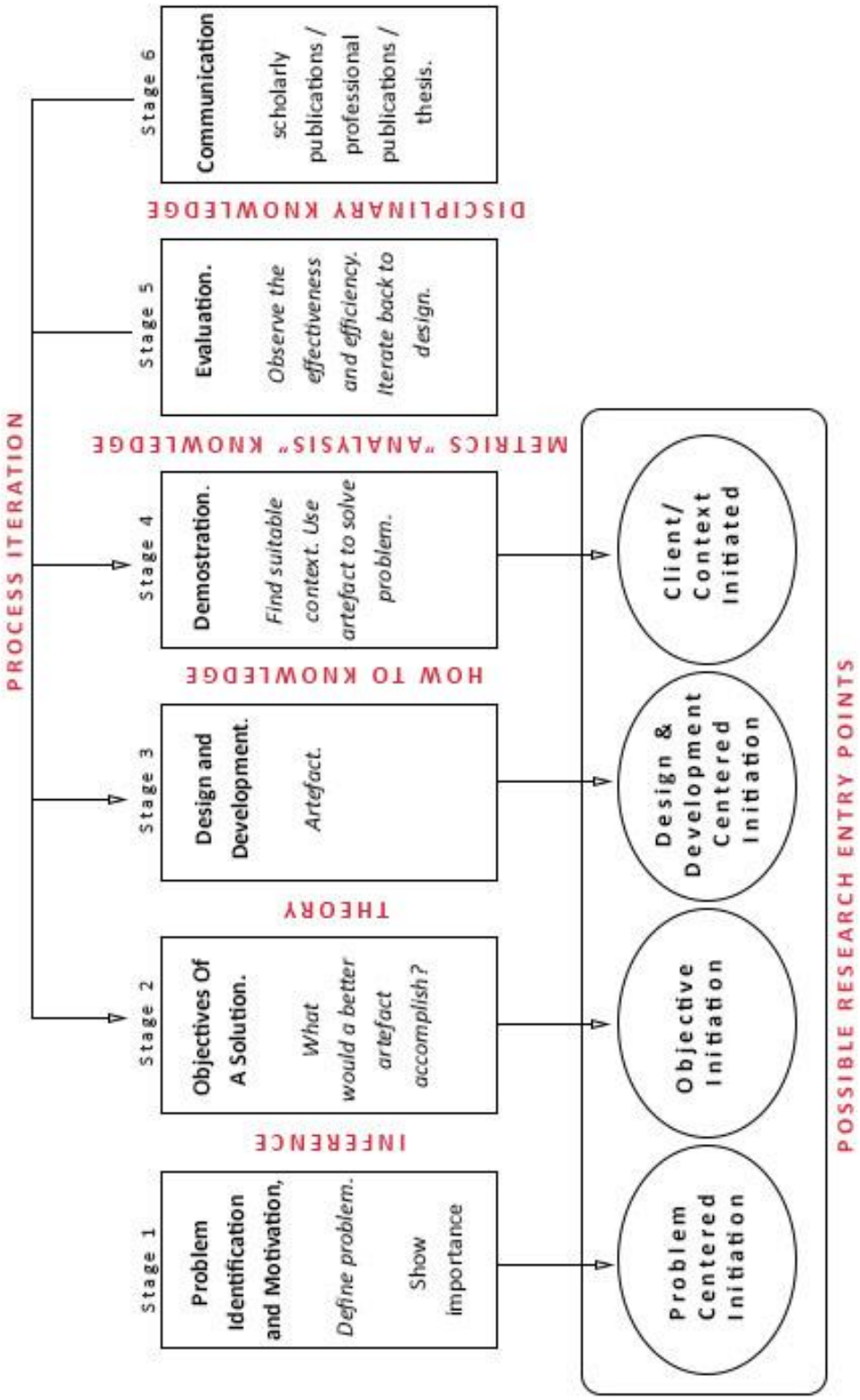


Figure 3.2 showing Design Science Methodology illustration.

(Tuunanen, 2006)

Individual Stage Breakdown

Stage 1:

Identification of the problem and defining of the problem is well documented in Chapter 3.1 and 3.2. The Motivations can be found in chapter 2.

Stage 2:

The objectives of the solution is generalised in Chapter 2 and detailed in Chapter 3.1

Stage 3:

Design and development of the artefact will be accomplished in Chapter 4.

Stage 4:

Demonstration of the artefact will be completed in Chapter 4 by presenting and demonstrating the artefact for peer review. The comments will be noted and discussed with other peer reviewers. The peer reviewers at minimum will have a good knowledge in computer related studies or specialise in forensic information technologies.

Stage 5:

Evaluation of the artefacts effectiveness and efficiencies will be tested in chapter 4 by rigorous testing in order to create an efficient high quality tool. In addition to the rigorous testing, a comparison will take place where the new tool will be compared to two industry standard tools that are used to identify objectionable images. The effectiveness and efficiency of image processing will be analysed and documented in chapter 4.

Stage 6:

Communication and publications will be the presentation of the artefact for peer review in September 2013 and the submitted thesis due by February 2014.

3.4 SYSTEM DESIGNS AND ARCHITECTURES

The system design looks at the three different image retrieval techniques and how the software is structured. It will begin by demonstrating the proposed HBIR technique followed by the CBIR (FTK) and CBII (Encase) techniques.

3.4.1 System Architecture

The proposed HBIR architecture is shown below. It has the same basic structure as CBIR and CBII. The HBIR architecture contains three main components. The three components are the back end database, the software or artefact and the acquired evidence file.

3.4.1.1 HBIR Technique

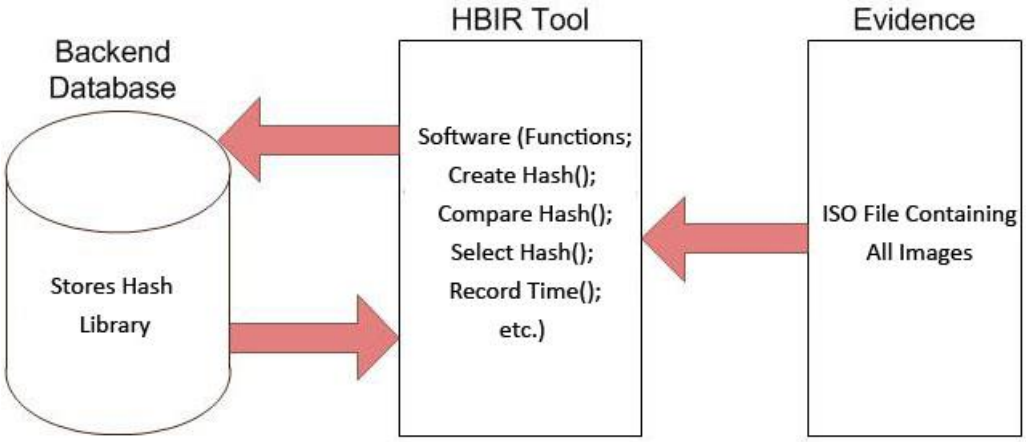


Figure 3.3 showing the proposed HBIR system architecture

3.4.1.2 CBIR Technique

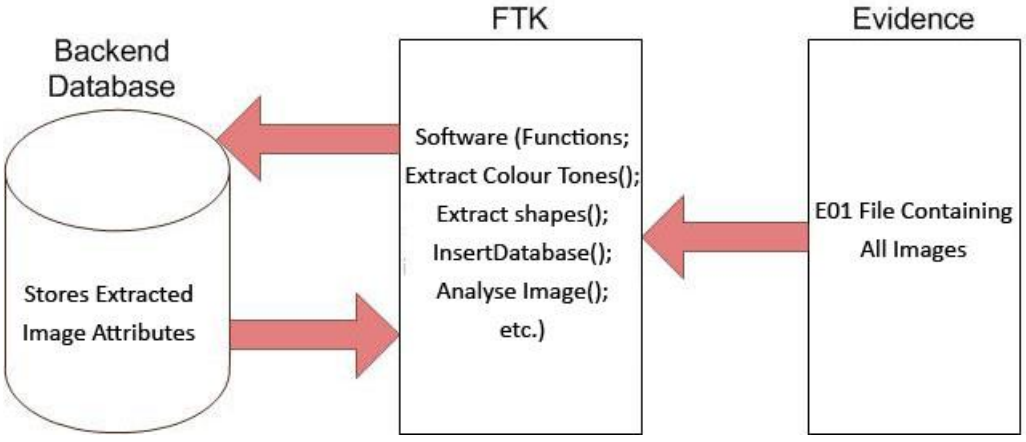


Figure 3.4 showing the proposed FTK system architecture

3.4.1.3 CBII Technique

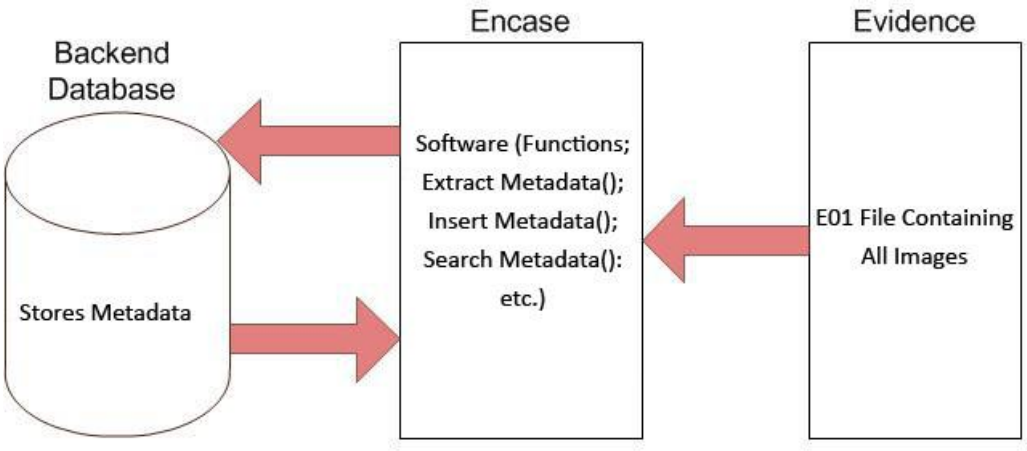


Figure 3.5 showing the proposed Encase system architecture

3.4.1.2 Backend Database

In the HBIR technique the database will be used as a storage mechanism for storing the hash files of known objectionable images. This will enable the software to query the database to search and identify matching hash values. The database will have a simple structure containing a sequential ID number for every row and a second column containing the hash value representing objectionable images. Different database technologies will be tested including Microsoft (MS) SQL, MySQL and PostgreSQL. The CBIR technique will be set up with the default option of a PostgreSQL database. It will be used for storing the extracted image attributes as well as to search for extracted attributes. The CBII technique, using Encase uses an inbuilt database. It will be used for storing the metadata and matching the input keywords to the extracted metadata.

3.4.2 Software Functionality

The HBIR software will be required to perform all user functionalities. The required user functionalities are,

- To provide the investigator an ergonomic interface.
- To enable the investigator to insert evidence files.
- To identify only image files from evidence.
- To support multiple hashing algorithms.

- Send the hash values to the database.
- Query the database.
- Compare database hashes and evidence hashes.
- Notify the forensic investigator of the matching hash values.
- Show the exact time taken to analyse the images.
- Notify the investigator of the location of the objectionable image in the evidence file.
- Visually see the hash values being generated during processing.

The FTK (CBIR) software contains the Explicit Image Detection tool which contains all the required functionality. The Encase (CBII) software supports the functionality to search by keywords. Therefore no further functionality is required for either FTK or Encase.

3.4.3 Evidence

The HBIR software will use evidence that will be inserted into an ISO file. The ISO format is a simple container to control where the software is able to search for images. In the Experiment Stage, the ISO file will contain 10,000 images of a specific size in kilobytes (KB) of 50 KB, 150KB, 250KB, 350KB, 450KB, 550KB, 650KB, 750KB, 850KB and 950KB. For testing, an E01 file will be used as evidence when testing the Forensic Tool Kit and Encase forensic software.

3.4.4 Test Bench Specifications

Below is a list of the hardware and software that will be used in the experimental stage of the testing.

Table 3.5 showing a summary computer components and software

Hardware / Software	Version
Operating System	Windows 7 Enterprise (x64) Service Pack 1 (build 7601

Processor	3.10 gigahertz Intel Core i5-2400 256 kilobyte primary memory cache 1024 kilobyte secondary memory cache 6144 kilobyte tertiary memory cache 64-bit ready Multi-core (4 total) Not hyper-threaded
Main Circuit Board	Board: Intel Corporation DZ68DB AAG27985-104
Memory Modules (RAM)	8100 Megabytes Usable Installed Memory Slot 'DIMM1' has 4096 MB (serial number 8B287A06) Slot 'DIMM2' has 4096 MB (serial number 8B288606)

3.5 RESEARCH PHASE 1: SOFTWARE TESTING

The first stage of the research phase one is preliminary software testing. Preliminary software testing will be performed on the proposed HBIR tool to identify any software glitches. Software testing will only be performed on the HBIR tool as it is expected that FTK and Encase have been thoroughly tested by the vendors.

3.5.1 HBIR Software Testing

This sub section defines the insertion, the identification and the testing of the image extraction techniques.

3.5.1.1 Test 1: Inserting Objectionable Images In To the Database

Inserting images into the database is one of the two main functions of the HBIR tool. To test this functionality nine images will be selected. The nine images will be 3 x JPEG (Joint Photographic Expert Group) format, 3 x PNG (Portable Network Graphics) format and 3 x BMP (Bitmap) format with the option to generate MD5 hash values. Once the hash values have been created by the HBIR tool they will be compared to other hashing software tools to confirm the correct hash value was generated. This test will confirm only images can be accepted by the tool, secondly the hash generator is functioning correctly and lastly the connection to the database is

storing the values correctly. This process will then be repeated for every hashing algorithm. The next test will be to increase the number of images from nine to three thousand. This will check to see if the HBIR tool can manage a large amount of data with stability. Once the first two tests have been completed successfully the third and final test will be using 50 image files and 50 non image files. This test will confirm that only image files can be inserted into the database and non-image files will be rejected.

3.5.1.2 Identifying Objectionable Images in Evidence

Identifying objectionable images in acquired evidence is the second main function of the HBIR tool. This function is as equally important as being able to insert hash values of known objectionable images into the database. This stage of the experiment will test that objectionable images can be detected successfully by the system.

To set up the test environment 1000 objectionable images will be inserted into the database using the MD5 algorithm. Once the hashes have been inserted successfully, an ISO containing nine objectionable images from the 1000 will then be analysed by the tool using the MD5 hash function. The test is successful if all nine images are identified as objectionable. If less than nine images are identified then further testing will be required. This test will then be repeated for all hashing algorithms. Test 6 will be using evidence containing five objectionable images and 5 non objectionable images. If the HBIR tool identifies the correct five objectionable images the test has been successful.

3.5.2 CBIR and CBII Software Testing

As mentioned earlier in Section 3.5, it is expected the software vendors have already performed stringent testing on their own software. Any software glitches with Forensic Tool Kit (CBIR) and Encase (CBII) will become visible when the pilot test is completed on the software.

3.6 RESEARCH PHASE 1: PILOT TESTING

The second stage to research phase one is the pilot testing. The pilot testing will be used to conduct a smaller scale, real life test and to evaluate the feasibility of the experiment. The pilot test will use all three image identification techniques.

3.6.1 The Pilot Testing Scenario

A girlfriend of a potential suspect has advised NZ Police that her boyfriend has a USB drive which contains highly objectionable images, which are (in her opinion) illegal. A warrant was issued to acquire all digital equipment of the suspect including the 8GB USB drive in question. The forensic investigator has acquired a forensic copy of the evidence and has advised the 8GB USB drive contains 8000 files of various file formats. NZ Police would like to know if any of the images on the drive are objectionable including a list of their file locations.

3.6.2 Pilot Test: HBIR

The HBIR Tool has been configured to use a MySQL database that contains MD5 hash values for 30,000 objectionable images. The process will be as follows:

- Extract and insert all the files from the evidence file into an ISO container.
- Insert the ISO file into the HBIR tool for comparison
- Select the hashing algorithm to use, MD5
- Execute the ‘compare function’
- Check to see if the objectionable images have been detected.
- Report on the findings.

3.6.3 Pilot Test: CBIR

The CBIR pilot test will analyse the 3,412 images using the Explicit Image Detection (EID) functionality in the Forensic Tool Kit software. As this technique does not require a dedicated hash database like the HBIR technique it will not be included. The FTK software has been configured with a PostgreSQL backend database. The process will be as follows:

- Insert all suspect images into the CBIR software
- Execute the software
- Record time taken
- Check if all of the objectionable images were identified
- Report on the findings

3.6.4 Pilot Test: CBII

The CBII technique will use the same 3,412 images however 18 images will have the metadata changed so the key word of ‘bad image’ will be inserted. The CBII software will then search through the images metadata to retrieve all of the images that contain the word ‘bad’. The process will be as follows:

- Copy the 3,412 images to a folder
- Edit 18 images metadata with the keyword ‘bad image’
- Open the CBII software and insert the key word ‘bad’ and 1999 other keywords (The other 1999 words would typically be words that relate to objectionable images. However for testing the other words will be unrelated)
- Execute the search function using the 2000 keywords
- Record the time taken
- Report on the findings.

3.7 EXPERIMENTAL TESTING

The following sub sections describe the testing requirements.

3.7.1 Research Phase 2: HBIR

A) Identify Fastest Hashing Algorithm

- Install HBIR software
- Select the ISO file containing 10,000 images of a fixed size
- Execute hashing algorithm over the 10,000 images with the ‘Hash’ option selected
- Record the time

- Repeat steps using 50KB, 150KB, 250KB, 350KB, 450KB, 550KB, 650KB, 750KB, 850KB, 950KB image sizes)

At this stage the fastest algorithm will be identified and how the algorithms manage different image file sizes.

B) Identify Fastest Database

- Connect HBIR tool to Microsoft SQL Database and insert 1,000,000 hash values
- Record time taken
- Connect HBIR tool to MySQL Database and insert 1,000,000 hash values
- Record time taken
- Connect HBIR tool to PostgreSQL Database and insert 1,000,000 hash values
- Record time taken
- Compare times of the three different databases and identify fastest database system.

At this stage the fastest database and hashing algorithm will be identified.

C) Reduction of hardware

- Reduce RAM from 8GB to 4GB
- Run test with fastest algorithm and fastest database
- Record findings and note the differences in processing times.

3.7.2 Research Phase 3: CBIR

A) Test CBIR Processing time

- Install CBIR Software

  Insert the batch of 10,000 50KB images including five objectionable images from evidence into the FTK software
- Execute software
- Record time and number of images found
- Repeat with images batches of 50KB – 950KB.

B) Reduction of Hardware

- Reduce RAM from 8GB to 4GB and run PHASE 2 A).
- Record findings and note the differences in processing times.

3.7.3 Research Phase 4: CBII

A) Test CBII Processing time

- Install Encase Software
- Insert 10,000 images including the 5 objectionable images into software
- Insert keyword list of 2000 words (including the key word "bad")
- In the search parameters select "selected keywords only" and "selected items only".
- Run software.
- Record time and number of images found
- Repeat with images of 50KB – 950KB.

B) Reduction of Hardware

- Reduce RAM from 8GB to 4GB and run PHASE 3A)
- Record findings and note differences in processing times.

3.8 DATA REQUIREMENTS

The data requirements outline the three critical stages of data generation, data collection and data presentation. During the three different stages there are different requirements to manage and control the experiment.

All testing will be performed within a controlled lab environment and all images used will not be objectionable or offensive in any manner. As performance is a fundamental aspect for the validity of this experiment, all testing will be performed on the same hardware, on the same computer system with the same installed operating system. All testing has been developed to create an impartial experiment where no technique will have an advantage over another technique.

The next three sections will define how the data will be generated, collected and presented. The purpose of the following sections is to analyse the requirements of

the research so only necessary testing is performed and secondly the highest quality of data can be generated, collected and presented.

3.8.1 Data Generation

The data generation stage is an important stage in the research. The quality of the data generation will directly influence the quality of the data collected and how the data is presented. It is also important the same batches of 10,000 images are used when testing each of the image retrieval techniques. At this stage of the research it is expected that the file size will be an important factor throughout the experiment and selecting the incorrect file size could have dire consequences for the accuracy of the research.

To create the batches of 10,000 images (randomly selected non-objectionable images found in an acquisition), a search of random key words will be entered into Google Images. Once the images are loaded a data mining tool will be used to download all the images on the page. The images will then be stored on the local hard drive. Once 10,000 images have been downloaded the images will be cropped and optimised to a specific file sizes and formats using the batching functionality in Photoshop. At the end of the data generation stage there will be 10,000 images containing 3334 JPG images, 3333 PNG images and 3333 BMP images of fixed sizes of 50KB, 150KB, 250KB, 350KB, 450KB, 550KB, 650KB, 750KB, 850KB and 950KB. Once the 10 batches of 10,000 images are created they will be inserted in to an ISO container. This ISO will provide the non-objectionable image data for the HBIR tool.

A second copy of the 10 batches will be stored locally which will be used for FTK and Encase.

Table 3.6 showing data collection tools required

Tool	Purpose
Download them all	Data mining tool used for collecting files of any format
Photoshop CS6	Imaging tool for editing and modifying file sizes of images

3.8.2 Data Collection

The data collection is the crucial stage of the experiment. As earlier detailed in chapter 3.2.1 the main objective of the research is to identify a more balanced image retrieval technique that can be used to identify objectionable images more productively. The pilot test will demonstrate the feasibility of the experiment. At present, it is unsure of how much time is required by FTK and Encase to process the batches of 10,000 images. This could however affect the research by either taking far too long, or in contrast, process the 10,000 images so quickly it is difficult to identify the differences between the techniques. In either of the two situations the number of images in each batch may increase or decrease accordingly.

Once the pilot test has completed successfully the experiment will commence. To record the time for processing each batch of images there will be a timing function added to the HBIR tool. This will provide exact times of how long it takes to process each test.

Forensic Tool Kit and Encase include functionality to record the time of processing. However, the accuracy of the time will be tested in the pilot testing stage to confirm it will be a sufficient mechanism for recording.

3.8.3 Data Presentation

Following the data generation and collection stages, the final stage in this section is the presentation stage. Data presentation is a necessary stage to document how the findings of the research will be conveyed. The data presentation will include the recorded processing times to hash the images, insert hash values in to the database and compare hash values. The important points of the research will be illustrated using tables, figures and diagrams.

3.9 LIMITATIONS

There are three main limitations to the research which are time restraints, financial constraints and legal boundaries.

The time period for the entire research including documentation, developing the software, software testing, pilot testing, experimental testing, data analysis,

formal presentation and discussion is 12 months. Due to the relatively short time frame it has been predetermined the experiment must be simulated on a significantly smaller scale in laboratory conditions. In an ideal situation discussions with cloud vendors would be made to integrate the HBIR system within their cloud environment to gather data on a larger, real world scale. This would create a higher quality of data collection. The time restrictions has also predetermined that smaller groups of just 10,000 images will be used for testing as opposed to a real world situation which may contain millions of images.

Financial limitations also restrict the amount processing power that can be used in the experiment. Ideally, server hardware would be purchased and used as a testing bench because it supports multiple CPUs, can store multiple RAM modules and uses high speed SCSI hard drives to fetch data. In addition, in an ideal situation the hardware would be leased from a cloud provider and the experimental testing would include genuine hash files of objectionable imagery so it could record data in real time for analysis. The financial restrictions also limit the variety of forensic CBIR and CBII software that can be used to benchmark the proposed HBIR technique. By using more software for benchmarking it could better demonstrate the effectiveness of the HBIR technique.

The last limitation is the legal issues surrounding the objectionable images. Ideally, a database of objectionable images would be used for testing. However the difficulties and legal issues with requesting the images would be extremely difficult or impossible to achieve. It would also require a significant amount of preparation which could not be accomplished within the12 month time frame.

3.10 CONCLUSION

Chapter 3 began by discussing the issues and gaps in current research (Section 3.1). By analysing the gaps in current research, a main research question was identified including five sub questions and their associated hypotheses (Section 3.2). The research design was next discussed where the Design Science Methodology was introduced and how it was going to direct and control the research. Section 3.3 also included the justification for selecting the Design Science Methodology. Following

Section 3.3 was the review of the system designs of all three image retrieval techniques. Section 3.5 software testing and Section 3.6 pilot testing discussed the first research phase of the experiment. Section 3.7 explained the process of how research phases 2, 3, and 4 were going to be achieved. The data requirements were discussed in Section 3.9 which investigated how the best data was could be generated, collected and presented.

Chapter 4 will now continue on from Chapter 3 by reviewing the data requirements and documenting the variations. This will lead on to documenting of the findings for the software testing and pilot testing and documenting the findings for the experimental stages.

Chapter Four

RESEARCH FINDINGS

4.0 INTRODUCTION

In the previous chapter, the design science methodology was defined as well as how the methodology is going to be implemented to create creditable, high quality research. Chapter 3 also discussed the process of how the experiment would be undertaken. In addition, Section 3.1 outlined gaps in current research by building on the foundations of chapter 2, where a main research question was established. The main question was then closely analysed and divided into sub questions. The five sub questions were created to support and reinforce the main question. Chapter 3 ended with the data requirements of the research, describing how the data will be generated, collected, and presented.

Chapter 4 will now continue on from chapter 3 by first discussing the variations in data requirements (Section 4.1). Variations of data requirements can be defined as the required changes in the data generation, data collection and data presentation stages. These changes may have emerged for technical or non-technical reasons. Variations in data requirements will be followed by the Research Phase 1 including software testing (Section 4.2) and pilot testing (Section 4.3). The software and pilot testing will show more precisely when and where the problem with the data requirement was identified and why it was essential it was modified before the experimental stage (Section 4.4). The experimental stage (Section 4.4) will focus on the results from testing of the HBIR (Research Phase 2), CBIR (Research Phase 3) and CBII (Research Phase 4) techniques as per the research data map. This will be followed by Section 4.5 where a technique comparison will be completed. All findings will then be presented using figures and tables. This is a critical section because the data for the main question and sub questions will be documented for the discussion stage in Section 5.2.

4.1 VARIATIONS IN DATA REQUIREMENTS

The variations in data requirements report the changes from what was proposed in Section 3.8. The following changes may have taken place due to technical restrictions, time restrictions and or financial restrictions which were not apparent during the writing of chapter 3. For ease of reading, this section has been structured similar to Section 3.7 starting with Variances with Data Generation (4.1.1), followed by Variances in Data Collection (4.1.2) and lastly Variances in Data Presentation (4.1.3). Each of the aforementioned sections will then be discussed further by identifying at what stage the requirement changed as well as the reason for making the change.

4.1.1 Data Generation

The data generation stage is the phase that focuses on how the 10 batches of 10,000 images will be created. There are three sections to the data generation stage as specifies in Section 3.7.1.

The first section is to create 10,000 random images of a precise file size of 50 KB, 150 KB, 250 KB, 350 KB, 450 KB, 550 KB, 650 KB, 750 KB, 850KB and 950KB. The second requirement was all the images must be unique in their binary string. This can be easily achieved by using 10,000 visually unique images. The third requirement was each group of 10,000 images must contain 3334 JPG, 3333 PNG and 3333 BMP images.

The first technical problem arose when trying to create 10,000 images of a specified file format and file size. It became apparent there was a significant amount of software available for batch processing of image resolutions, however there was significantly less available software that supported saving of images to a specified file size. The ideal software would perform three main functions.

- Able to save an image to a specified fixed size
- Able to save the image to all three required formats
- Able to process 10 groups of 10,000 images within a feasible timeframe.

Open Source software licensed under the General Public License (GPL) was investigated such as GIMP, Lazpaint and Ultimate Paint. Once it was identified that

the required functionality was not available, proprietary software was then investigated. Testing was performed on Photoshop, Serif PhotoPlus X6 and Aperture. However at this stage, both the proprietary and open source software did not support the required functionality. After a significant amount of research and extensive testing, IrFranView with the Riot plugin was the closest software that could perform batch processing of the total 100,000 images.

However, depending on the colour profile in the image IrFranView had issues saving files to the required file size. In some cases it would save files less than half of the specified size. For example when IrFranView was configured to output a 150KB size image it actually outputted a 75KB image. After extensive testing it showed when a JPG image was used approximately half of the images outputted were within 10KB of the required file size. Furthermore, only 4-5% of the files were with 1KB of the configured file size. Therefore 2.5 million JPG images would need to be processed to have 100,000 images outputted with the correct file size. BMP and PNG results were significantly worse. Due to the time frame 2.5 million images would not be justifiable. Therefore the data generation requirement was changed so each image in each group had a specified value of between 0 and 10KB below the required file size and only JPG images will be used. For example the batch of 150KB images contain 10,000 images ranging between 140KB to 150KB in file size and the batch of 950KB images contained 10,000 images ranging between 940KB to 950KB in file size.

To summarise, the final data generation will contained 10 batches of 10,000 JPG images of the following file sizes.

- 10,000 JPG images ranging between 40-50KB in file size
- 10,000 JPG images ranging between 140-150KB in file size
- 10,000 JPG images ranging between 240-250KB in file size
- 10,000 JPG images ranging between 340-350KB in file size
- 10,000 JPG images ranging between 440-450KB in file size
- 10,000 JPG images ranging between 540-550KB in file size
- 10,000 JPG images ranging between 640-650KB in file size
- 10,000 JPG images ranging between 740-750KB in file size

- 10,000 JPG images ranging between 840-850KB in file size
- 10,000 JPG images ranging between 940-950KB in file size

Table 4.1 showing data collection tools required.

Tool	Purpose
Download them all	Data mining tool used for collecting JPG images
IrFanView with Riot plugin	Batch Imaging tool for editing and modifying file sizes of images

4.1.2 Data Collection

Data collection is a crucial stage of the experiment. However there have been three changes to how the data will be collected. The three changes have become apparent during pilot testing of the HBIR technique. The four main issues are variances in time when repeating the same test using the same constants, FastHash algorithm outputting hash values of zero, RAM not releasing images after processing and reduction of RAM from 8GB to 4GB did not affect processing.

The first problem is the variances in the processing time when the identical test was rerun multiple times. This was identified in the final stage of the HBIR pilot test when 1,000 images ranging in file size from 50-950KB were inserted into the HBIR Tool. The first test showed the time taken was 24.1 seconds. The second test showed the time of 18.1 seconds and the third test showed 17.9 seconds. Even though the differences do not appear to be large, it will have dramatic consequences for the results in the experimental stage when the number of images is increased. By troubleshooting the fault it was determined that multiple applications were running in the background at different times during the test. For example Windows would check for updates, resulting in removed resources from the HBIR Tool which in turn would increase the processing time of the HBIR.

To resolve this issue the test machine hard drive was formatted and Windows 7 SP1 (last updated at 28/10/2013) was installed. The only additional software that was installed was:

- Java SE Development Kit 7
- Microsoft SQL 2008 R2

- MySQL 5.6.14
- PostGreSQL 9.3
- MySQL Workbench 6.0 CE

In addition to using a fresh installation with minimal software installed all database services were disabled unless required. It was expected by only using a fresh operating system it would resolve the fluctuation in processing times. However this assumption was disproved. The fluctuations had decreased severely on average from 2-3mins for the larger batches to just 2-8 seconds. To minimise the effect of the fluctuations each test will be performed five times and an average will be calculated as the official processing time.

The second technical problem faced was that the FashHash algorithm was only generating hash values of '0' (zero) (as illustrated in Figure 4.1). After investigating, the fault could not be identified within a reasonable time frame. However it was expected the algorithm had difficulties processing large files and was not a fault with the tool. This resulted in the FastHash algorithm being removed as a potential hashing algorithm for the final HBIR tool.

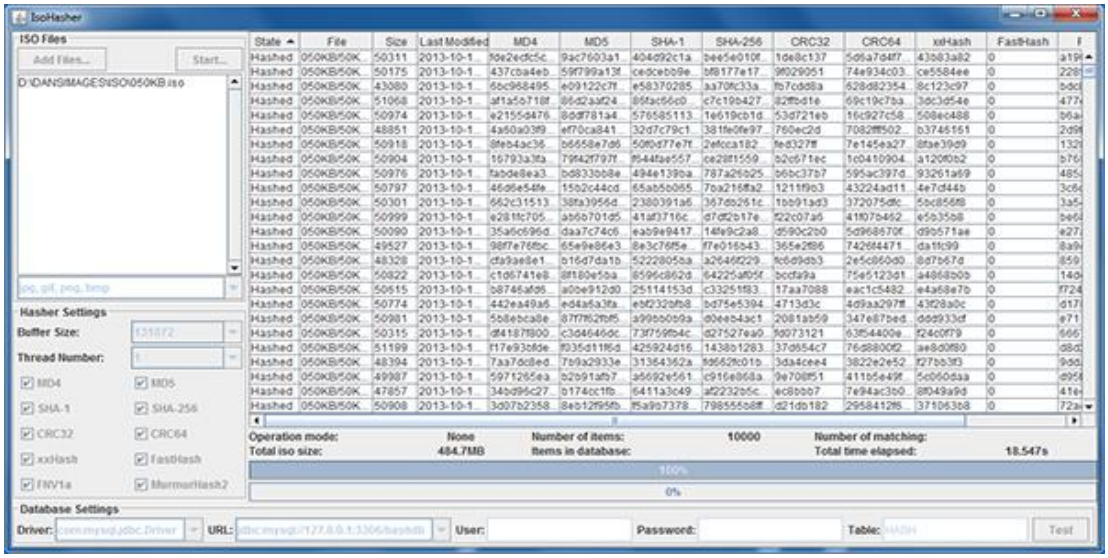


Figure 4.1 showing 10,000 50KB images with FastHash all showing a hash value of 0

The third technical problem was the difficultly of confirming that all images were released from RAM before the next test was to begin. During the pilot test it was discovered the second test always produced a faster processing time than the first. To identify the fault, the same test was run multiple times over the same batch of images.

Testing showed when any algorithm was run over the same batch of images the second test was always faster than the first. To test the assumption that this was a RAM issue, the HBIR Tool was closed and the HBIR process was manually killed via the task manager. The first test was run using the MD4 hashing algorithm and recorded a time of 3min 42seconds. The HBIR tool was then closed and the background processes killed. A second test was run using the same configuration and a time of 3min 45seconds was recorded. To solve the problem on a permanent basis each new algorithm was issued a new process identifier. After extensive testing this solution proved to be an excellent resolution to the problem. In addition, it also further reduced the fluctuations in the processing times identified earlier.

The last technical problem was with the RAM reduction resulting in no effect on processing. This problem was identified during the experiment stage (Section 4.1) of the research. It was expected that reducing the RAM would decrease the productivity (as per Sub Question 4 in Section 3.2.1). However, testing discovered that all three tools required CPU resources significantly more than RAM. Due to the test machine only having one physical CPU it made it difficult to reduce. It was suggested to disable multiple cores of the CPU through virtualisation software, however as the problem was identified at such a late stage of the research, it was not feasible to restart the testing. In addition, it would not have a significant effect in answering the main question. Therefore this question will be noted and discussed in Section 6.3 Future research.

Table 4.2 showing data generation tools required

Tools	Purpose
Netbeans 7.4	Programming IDE
MS SQL Server 2012 R2	Microsoft SQL Server for database testing
MySQL 5.6.15	Alternate SQL server for database testing
PostgreSQL 9.2.6	Alternate SQL server for database testing
Encase 6.01	Content Based Image Indexing software
FTK 5.0.1	Content Based Image Retrieval software
Java SE 6	Software platform suite includes (JDK, Server JRE, JRE)

4.1.3 Data Presentation

After a close analysis of the data presentation stage the proposed techniques of using tables, graphs and figures will display the data presentation stage appropriately.

Table 4.3 showing required tools for data presentation

Tool	Purpose
Microsoft Word 2010	Word processing software for documentation
Microsoft Excel 2010	Data analysis and graph software

4.2 HBIR SOFTWARE TESTING

Software Testing (Research Phase 1) will be performed on the HBIR Tool as a preliminary stage to identify any software glitches with adding images to the database and checking evidence for potential objectionable images.

4.2.1 Inserting Objection Images in to The Database

Section 4.2.1 comprises of three different tests in relation to inserting hash values into the database. The first test will insert nine hash values in to the database. The second test will check that the tool can manage higher quantities of images by inserting 3000 hash values at once. The third test will check that only jpg images are able to be inserted into the database.

4.2.1.1 Test 1

Insert 9 images into the database and confirm the MD5 hash values are correct.

1. Select 9 images to be inserted in to ISO file.

Name	Date modified	Type	Size
950KB_00009.jpg	9/10/2013 7:42 p.m.	JPG File	950 KB
950KB_00001.jpg	9/10/2013 7:41 p.m.	JPG File	948 KB
950KB_00002.jpg	9/10/2013 7:41 p.m.	JPG File	949 KB
950KB_00003.jpg	9/10/2013 7:41 p.m.	JPG File	949 KB
950KB_00004.jpg	9/10/2013 7:16 p.m.	JPG File	944 KB
950KB_00005.jpg	9/10/2013 7:09 p.m.	JPG File	942 KB
950KB_00006.jpg	9/10/2013 7:09 p.m.	JPG File	949 KB
950KB_00007.jpg	9/10/2013 7:42 p.m.	JPG File	949 KB
950KB_00008.jpg	9/10/2013 7:09 p.m.	JPG File	941 KB

Figure 4.2 showing nine images that will be inserted in to the database

2. Insert the images into an ISO and select the option to add to database

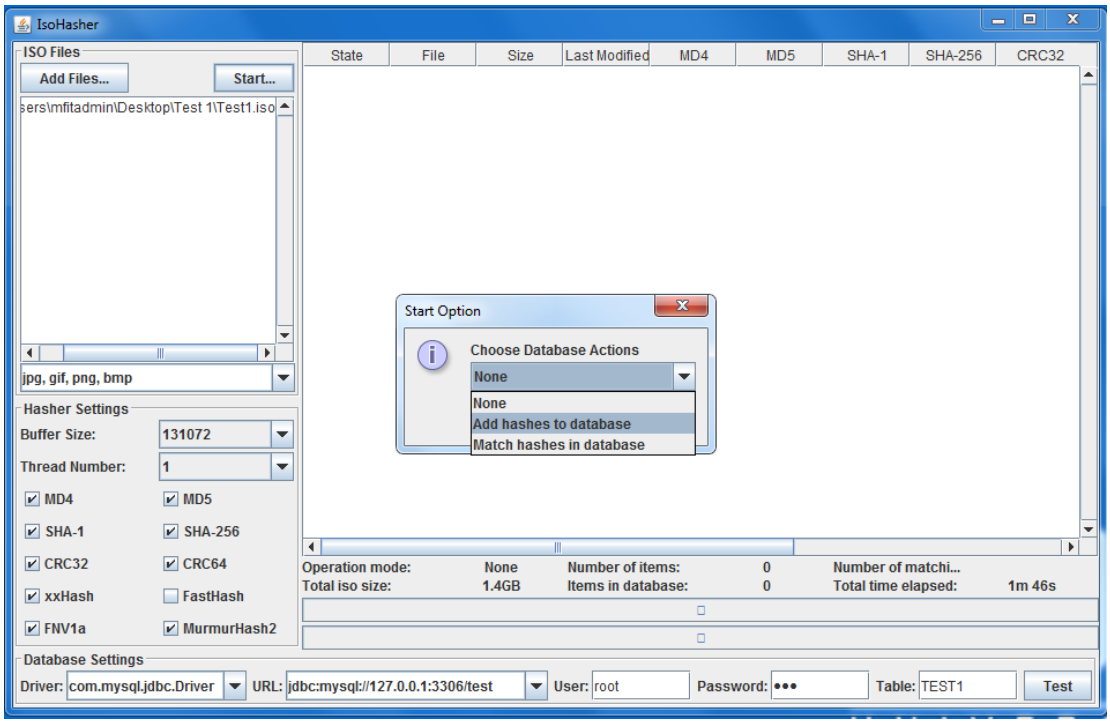


Figure 4.3 showing HBIR Tool with option list to "Add hashes to database"

3. Execute the Test

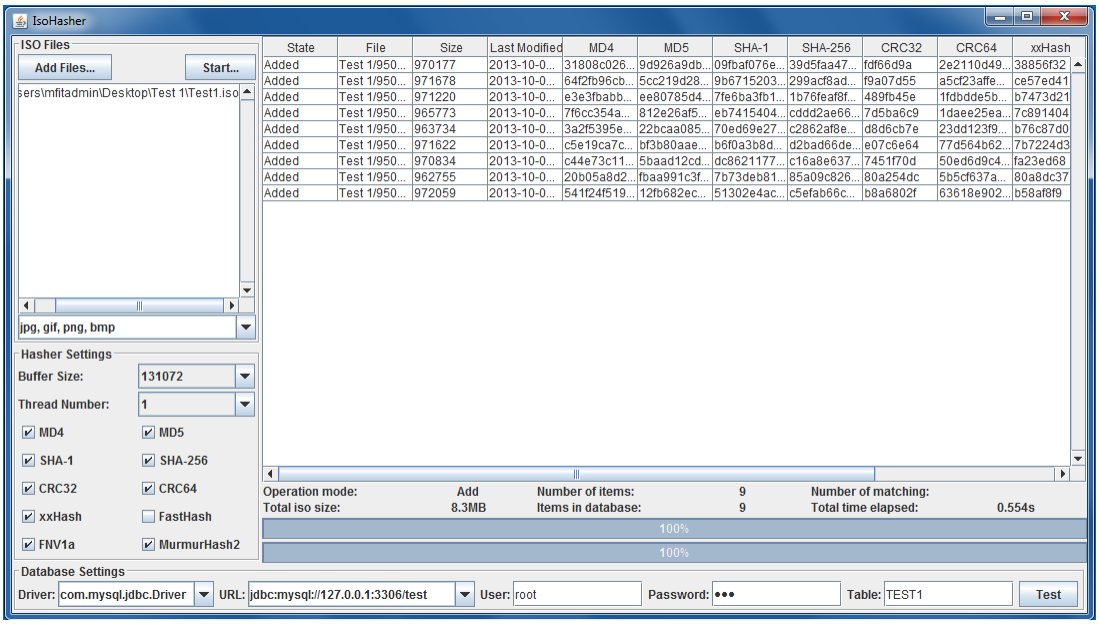


Figure 4.4 showing hash values were added to the database (State: Added)

4. View the database to confirm hash values were **Added**.

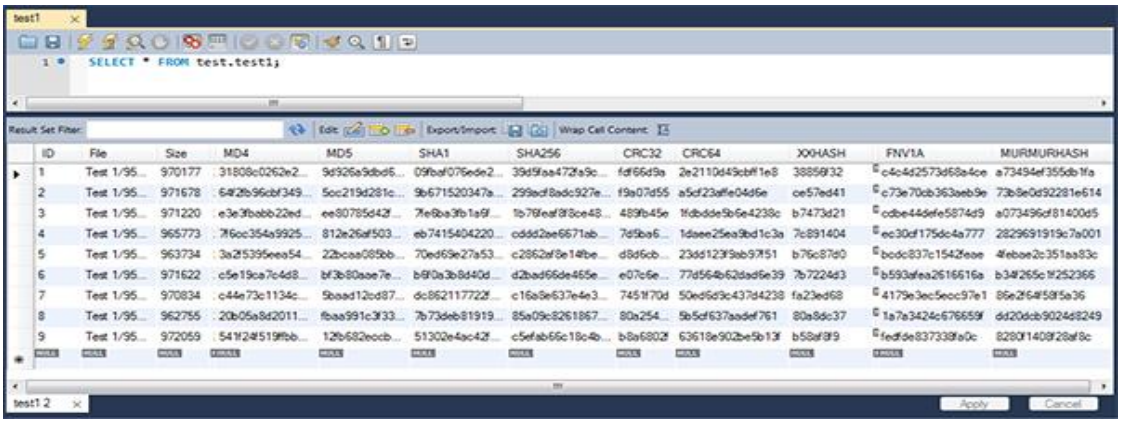


Figure 4.5 showing view of hash values inserted into the database

5. Compare the hash values that were generated by the HBIR tool that matches the hash values generated by the alternative hashing tools.

Table 4.4 showing hash values created by the HBIR Tool compared to hash values of other tools

Image Name	Hashing Algorithm	HBIR Hash Value	Alternate Software Hash Value	Match
Test 1/950KB_00001.jpg	MD4	31808c0262e29b33e9b497e11c04b66	31808c0262e29b33e9b497e11c04b66	YES

Test 1/950KB_00002.jpg	MD5	5cc219d281cd267319843ed087134277	5cc219d281cd267319843ed087134277	YES
Test 1/950KB_00003.jpg	SHA-1	7fe6ba3fb1a6fe9189e6b6c518f789f1d0b0e075	7fe6ba3fb1a6fe9189e6b6c518f789f1d0b0e075	YES
Test 1/950KB_00004.jpg	SHA-256	cddd2ae6671ab55252c4a2a45e1b1e5a31f405ac42a64bfabb73c6baa74c961	cddd2ae6671ab55252c4a2a45e1b1e5a31f405ac42a64bfabb73c6baa74c961	YES
Test 1/950KB_00005.jpg	CRC32	d8d6cb7e	d8d6cb7e	YES
Test 1/950KB_00006.jpg	CRC64	77d564b62dad6e39	77d564b62dad6e39	YES
Test 1/950KB_00007.jpg	xxHash	fa23ed68	fa23ed68	YES
Test 1/950KB_00008.jpg	FNV1a	1a7a3424c676659f	1a7a3424c676659f	YES
Test 1/950KB_00009.jpg	MurmurHash2	8280f1408f28af8c	8280f1408f28af8c	YES
Test 1/950KB_00001.jpg	SHA-1	09fbaf076ede2adcd74bfe6f11174c9b72587	09fbaf076ede2adcd74bfe6f11174c9b72587	YES
Test 1/950KB_00002.jpg	SHA-256	299acf8adc927e91f37e34d7beb4f1a224d02e2a60e6afd94688dadd32944c	299acf8adc927e91f37e34d7beb4f1a224d02e2a60e6afd94688dadd32944c	YES
Test 1/950KB_00003.jpg	CRC32	489fb45e	489fb45e	YES
Test 1/950KB_00004.jpg	CRC64	1daee25ea9bd1c3a	1daee25ea9bd1c3a	YES
Test 1/950KB_00005.jpg	xxHash	b76c87d0	b76c87d0	YES
Test 1/950KB_00006.jpg	FNV1a	b593afea2616616a	b593afea2616616a	YES
Test 1/950KB_00007.jpg	MurmurHash2	86e2f64f58f5a36	86e2f64f58f5a36	YES

Test 1/950KB_00008.jpg	MD4	20b05a8d201175247e9f70f9e7ba281	20b05a8d201175247e9f70f9e7ba281	YES
Test 1/950KB_00009.jpg	MD5	12fb682eccb7eeb7a2c127a662c298	12fb682eccb7eeb7a2c127a662c298	YES
Test 1/950KB_00001.jpg	CRC32	fdf66d9a	fdf66d9a	YES
Test 1/950KB_00002.jpg	CRC64	a5cf23affe04d6e	a5cf23affe04d6e	YES
Test 1/950KB_00003.jpg	xxHash	b7473d21	b7473d21	YES
Test 1/950KB_00004.jpg	FNV1a	ec30cf175dc4a777	ec30cf175dc4a777	YES
Test 1/950KB_00005.jpg	MurmurHash2	4febae2c351aa83c	4febae2c351aa83c	YES
Test 1/950KB_00006.jpg	MD4	c5e19ca7c4d894de1eb805f5b439c47	c5e19ca7c4d894de1eb805f5b439c47	YES
Test 1/950KB_00007.jpg	MD5	5baad12cd873085f39f9e5c53ed52f2	5baad12cd873085f39f9e5c53ed52f2	YES
Test 1/950KB_00008.jpg	SHA-1	7b73deb819199b4ba5a034b17692bf2e9be4	7b73deb819199b4ba5a034b17692bf2e9be4	YES
Test 1/950KB_00009.jpg	SHA-256	c5efab66c18c4b94c3a5703556e721b5c5042f6ecc082d01b6de48fe75649	c5efab66c18c4b94c3a5703556e721b5c5042f6ecc082d01b6de48fe75649	YES

4.2.1.1 Test 2

1. Check HBIR can manage higher quantities of data by testing with 3000 images.

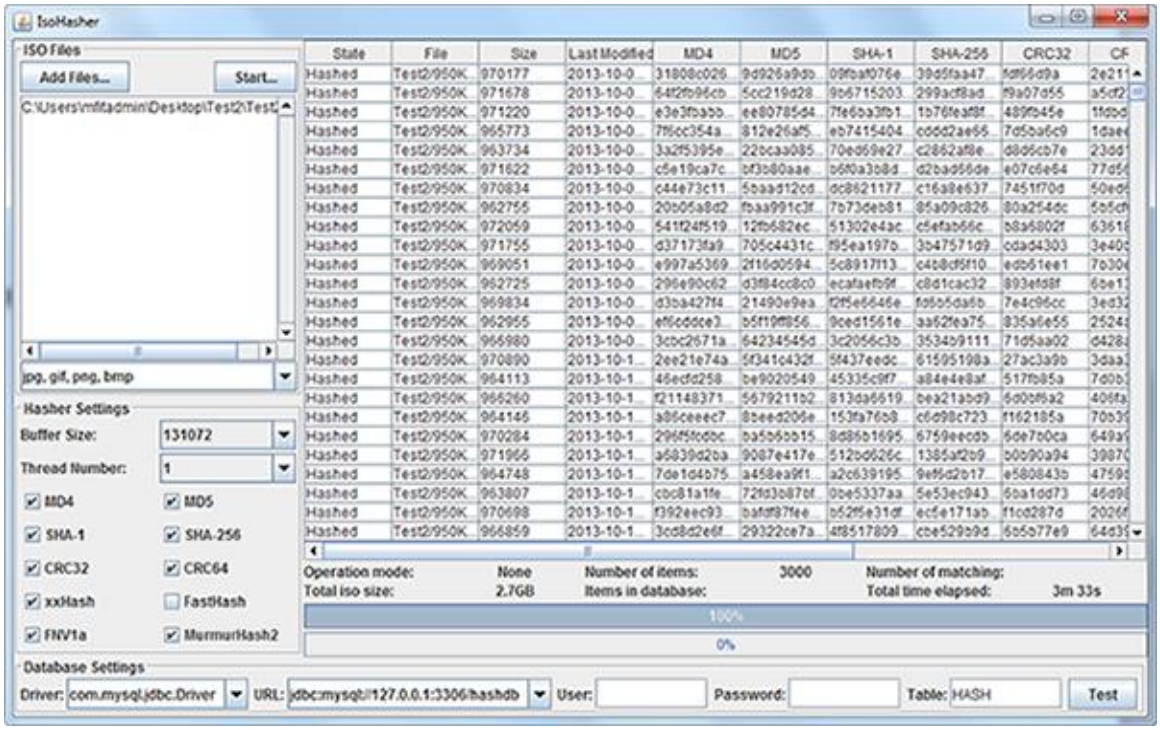


Figure 4.6 showing successful hashing of 3000 image (2.7GB) files using all 9 algorithms

4.2.1.1 Test 3

1. Test 50 non JPG image files and 50 JPG images to confirm the tool will only process JPG images.

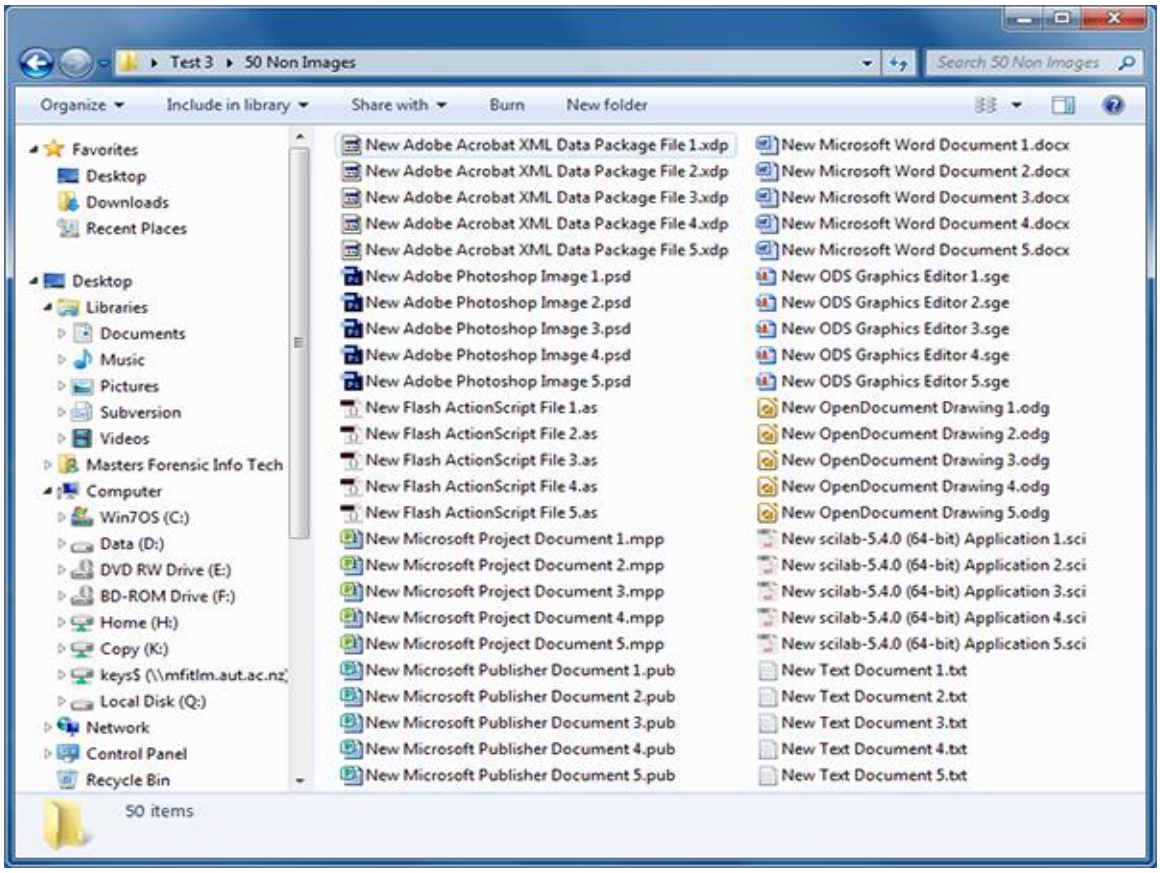


Figure 4.7 showing 50 non-image files used for testing

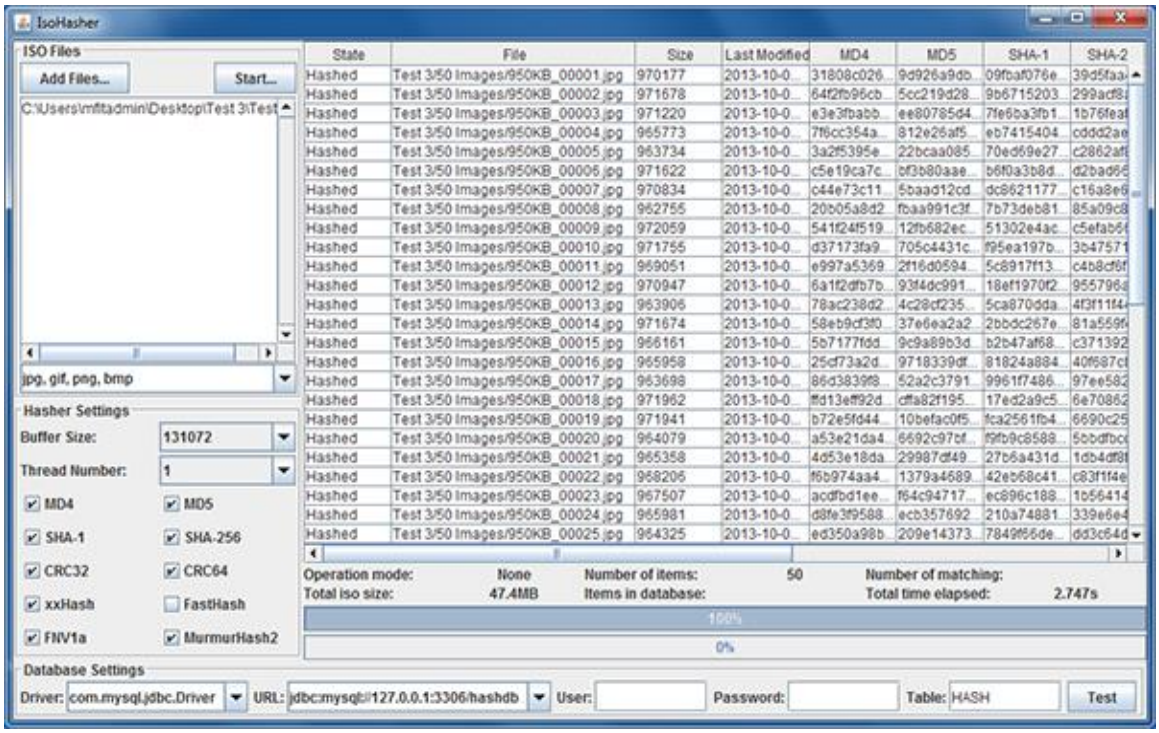


Figure 4.8 showing successfully hashing 50 image files and excluding the 50 non-image files (Number of items: 50)

4.2.2 Identifying Objection Images

Section 4.2.2 comprises of two different tests in relation to identifying objectionable images. The first test will check to see if the tool can detect nine images as objectionable in a database containing 1000 hash values. The next test will check the accuracy of the HBIR by confirming only objectionable images are detected and non-objectionable images are not detected.

4.2.2.1 Test 4

1. Insert 1000 images in to the objectionable hash database.

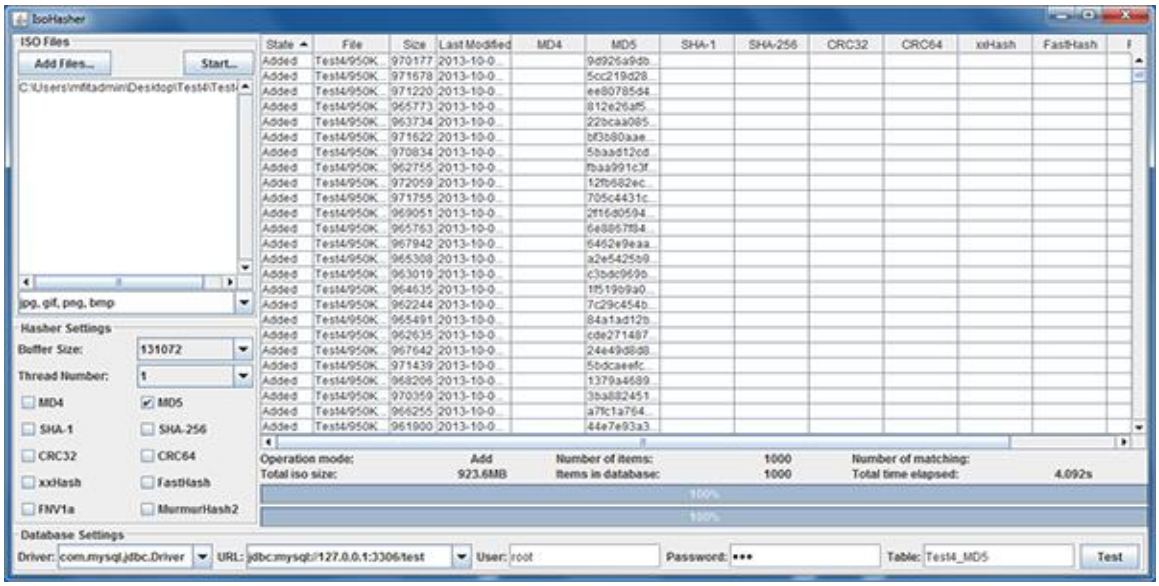


Figure 4.9 showing successful insertion of 1000 images in to the database using the MD5 hashing algorithm

2. Analysing nine known objectionable images with hashes in the database.

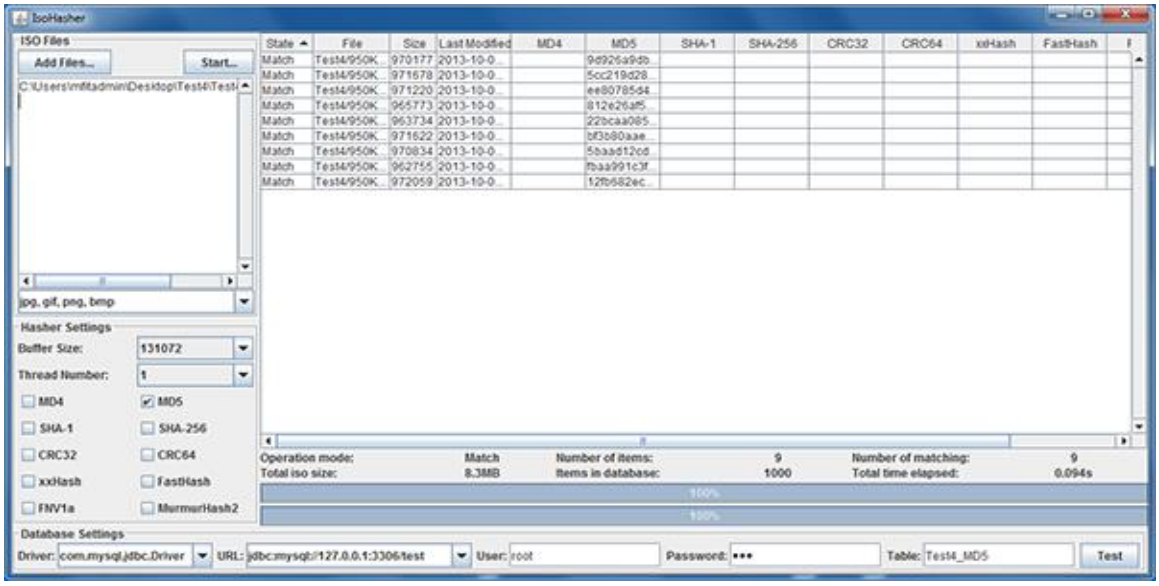


Figure 4.10 showing successfully identifying nine known objectionable images (State = Match)

4.2.2.2 Test 5

1. Test to show when inserting images from evidence containing five objectionable images and five non objectionable images only objectionable images are detected.

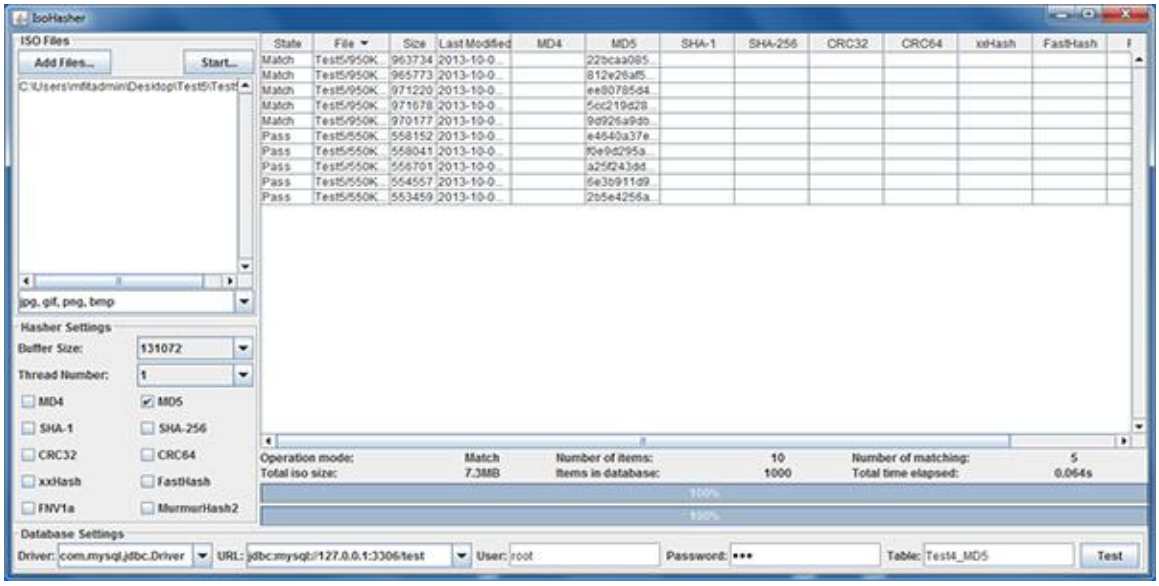


Figure 4.11 showing successful detection of five objectionable (Matched) images, and 5 non-objectionable (Pass) images

4.3 PILOT TESTING

The Scenario:

An 8 Gigabyte (GB) Universal Serial Bus (USB) drive has been acquired by a police forensic investigator. The girlfriend of the suspect has advised New Zealand (NZ) Police that the USB drive contains highly objectionable images, which are (in her opinion) illegal. The forensic investigator has acquired a forensic copy of the evidence. He has advised the 8GB USB drive comprises of 8000 files of various formats. NZ Police would like to know if any of the images on the drive are objectionable including a list of their file locations.

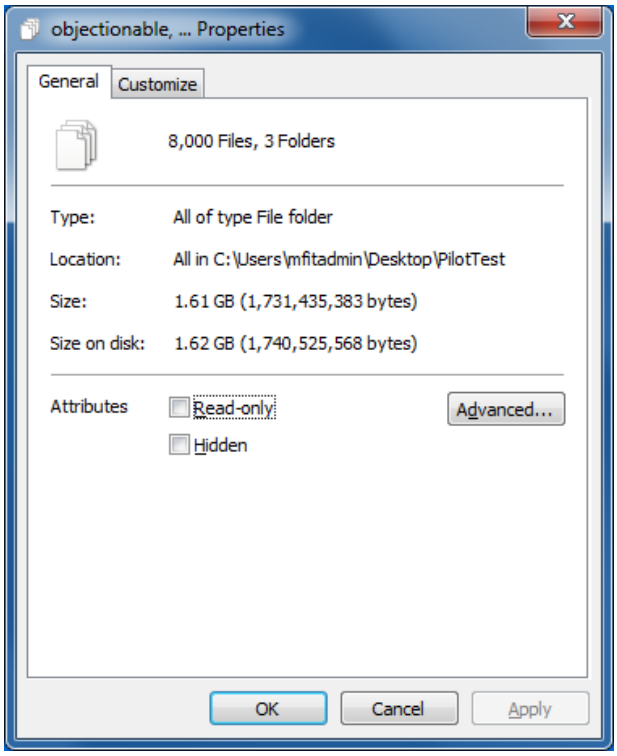


Figure 4.12 showing the folder properties of the 8GB USB Drive

Expected Outcome:

The expected outcome is that all three image retrieval techniques will retrieve 18 objectionable images.

4.3.1 Hash Based Image Retrieval

The objectionable database is contains 30,000 hash values.

HBIR Results:

Process time: 8.589 seconds

Number of images detected: 18

Overall Result:

Satisfactory

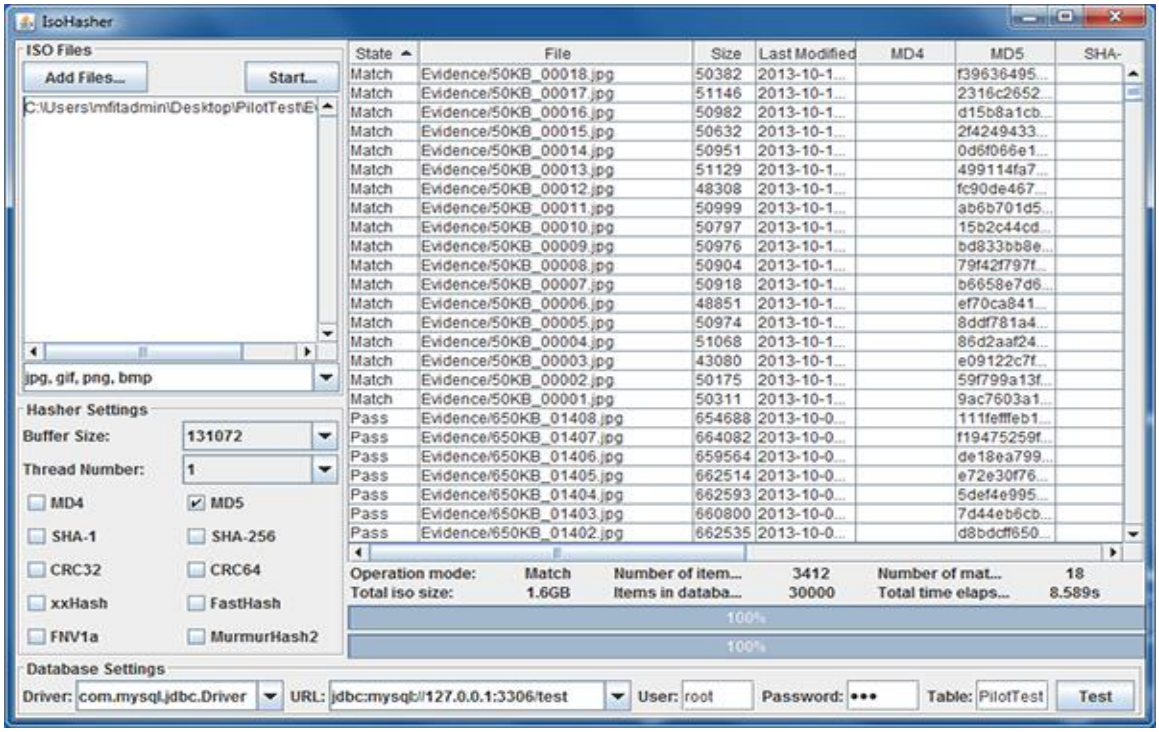


Figure 4.13 showing pilot test results using HBIR Tool

4.3.2 Content Based Image Retrieval

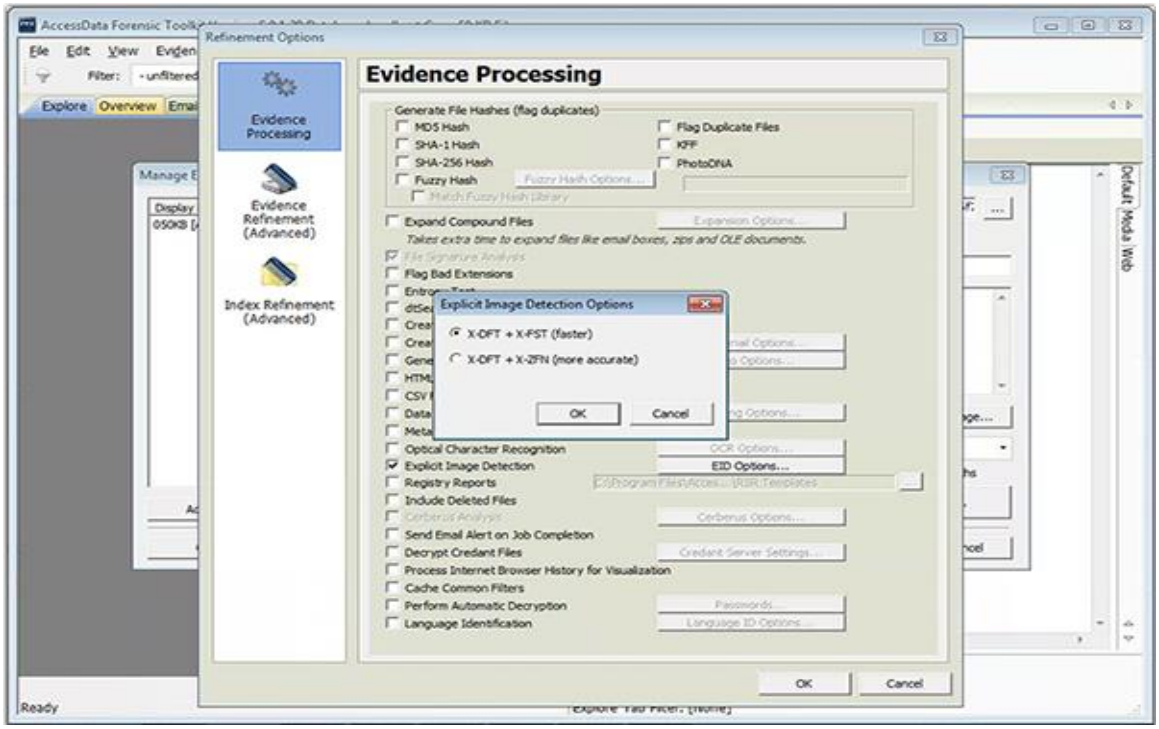


Figure 4.14 showing the FTK evidence configuration settings

FTK Results:

Process time: 23 min. 56 seconds

Number of images detected: 18

Overall Result: Satisfactory

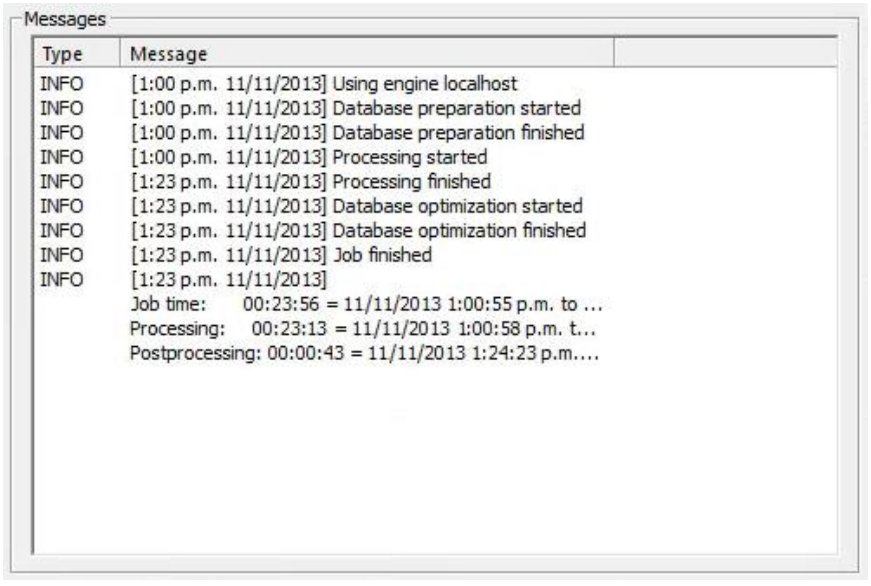


Figure 4.15 showing the pilot test results using FTK

4.3.3 Concept Based Image Indexing

The keyword list comprises of 2000 keywords (random keywords have been selected for testing however, in an industry environment the list would contain words surrounding objectionable images)

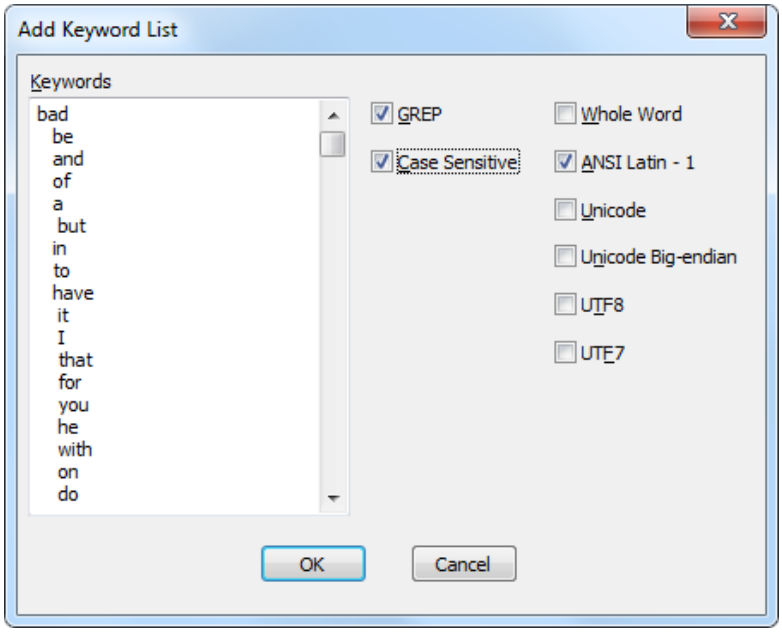


Figure 4.16 showing the keyword list for CBII (Encase)

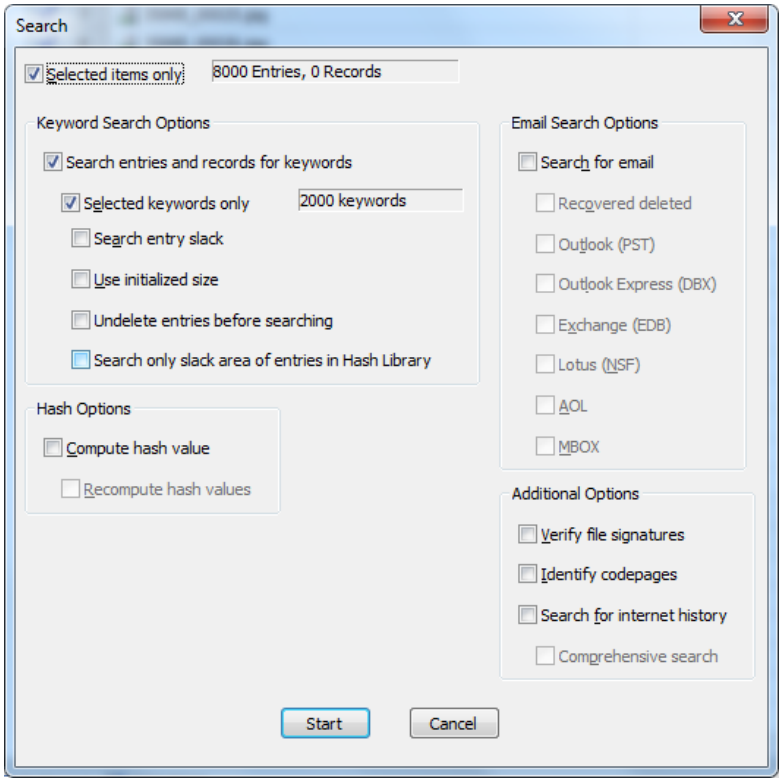


Figure 4.17 showing the CBII Search parameters

CBII Results:

Process time: 17 seconds

Number of images detected: 18 unique images (97 including duplications)

Overall Result: Satisfactory

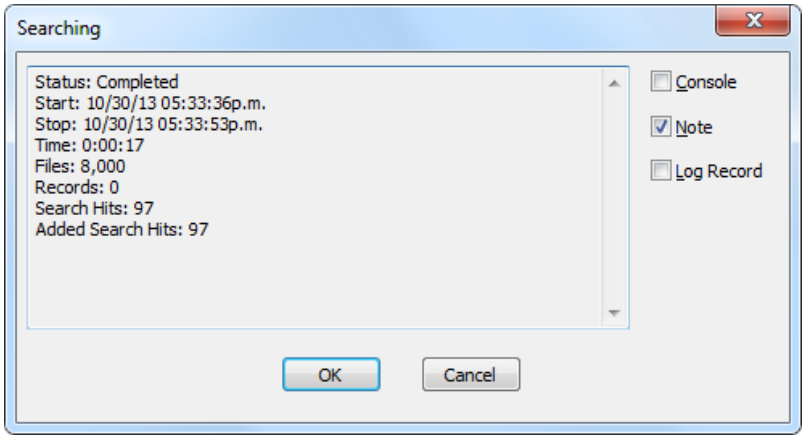


Figure 4.18 showing the pilot test results using CBII

4.4 EXPERIMENT

The Experimental stage is where the actual experiment will take place. This has been sectioned into the three research phases including hash based image retrieval, content based image retrieval and concept based image indexing. Each of the sections will be thoroughly tested and the data will be clearly presented. The presentation will include the use of tables and figures.

4.4.1 HBIR Experiment Testing

The HBIR Experiment (Research Phase 2) is broken in two sub sections. The first section is experimenting to identify the fastest algorithm followed by experimenting to identifying the fastest database. As earlier discussed in Section 3.5.1, the main two functions of the proposed HBIR tool is the selection of the hashing algorithm and the selection of the backend database. Therefore, the first section of the experiment will test using multiple hashing algorithms including MD4, MD5, SHA-1, SHA-256, CRC-64, CRC-32, XXHash, FNVa1 and Murmurhash2. The second section is to identify the fastest database. The most efficient database will be selected to maximise the productively of the HBIR tool thus maximise its overall effectiveness at identifying objectionable images. This section will then be concluded with the final recommendations for the tool.

4.4.1.1 Identify Fastest Algorithm

Nine hashing algorithms were extensively tested to identify the best algorithm for the HBIR tool. There are two main quantitative aspects that required in depth analysis. The first aspect is the amount of time required to process each batch of 10,000 images using each of the selected hashing algorithms. The second aspect is calculating the time it took to process 50KB of data, generating one hash value, compared to processing 50KB of data in a large image when no hash was generated. This data has been illustrated on the two graphs below. The first graph displays the processing time of each batch of images. The processing time of each batch was required to discover the fastest hashing algorithm. As earlier discussed in Section 4.2, the data collection involved recording 10 batches containing 10,000 images of a specified size (50KB,

150KB, 250KB, and so on.) being processed 5 times through the HBIR Tool and then averaged accordingly. In addition to illustrating the processing times, a line of best fit was calculated for the purpose of predicting values of images of a larger file size that were not tested. Lastly, a R^2 value was calculated to determine how well the data fit to the model.

The second graph below illustrates the time taken to process 50KB of data in a small image and 50KB of data in a large image. For example to process 10,000 images at 50KB, using MD4, generating one hash per file requires 21.17 seconds to process. In comparison, processing 10,000 images at 250KB using MD4, generating one hash per file requires 104.81 seconds. If the time taken to process the 250KB is divided by 5, it shows how long it is required to process 50KB of data. In this example that equals 20.96 seconds per 50KB in the 250KB batch of 10,000 images. The time also shows the difference of only processing the data, compared to processing the data as well as generating a hash value. This will be required in the future research section in Section 6.

MD4

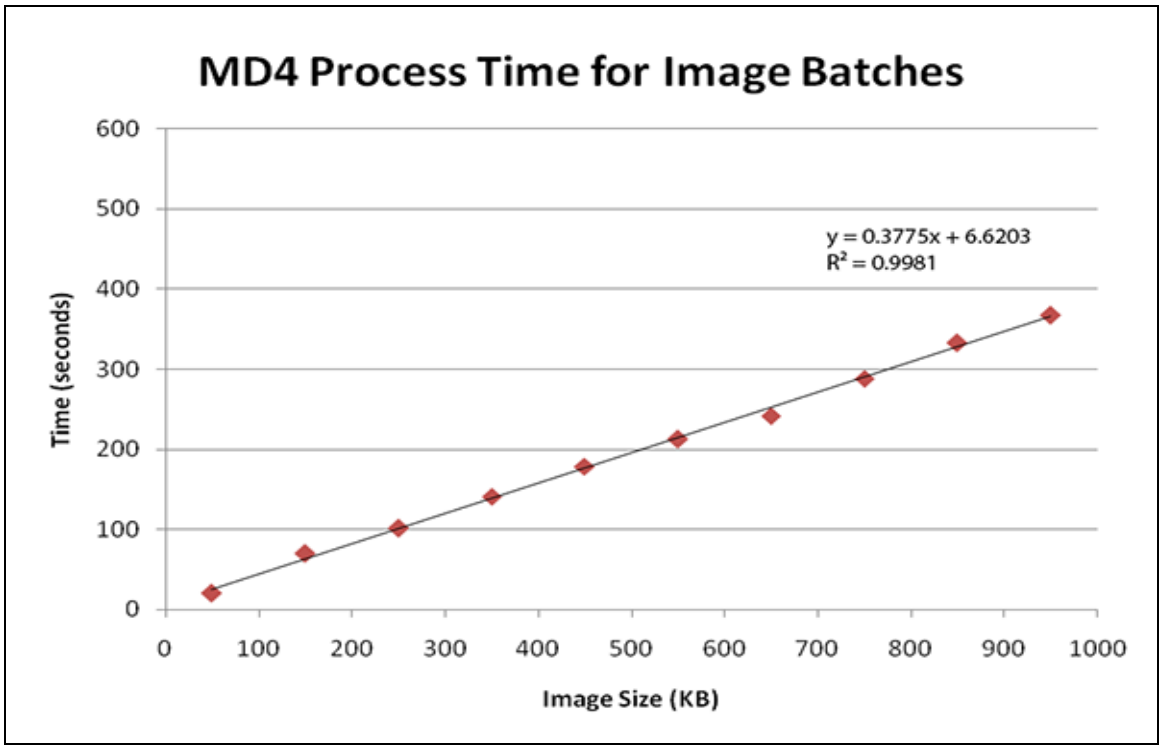


Figure 4.19 showing the time required to process 10 batches of 10,000 images using MD4 hashing algorithm

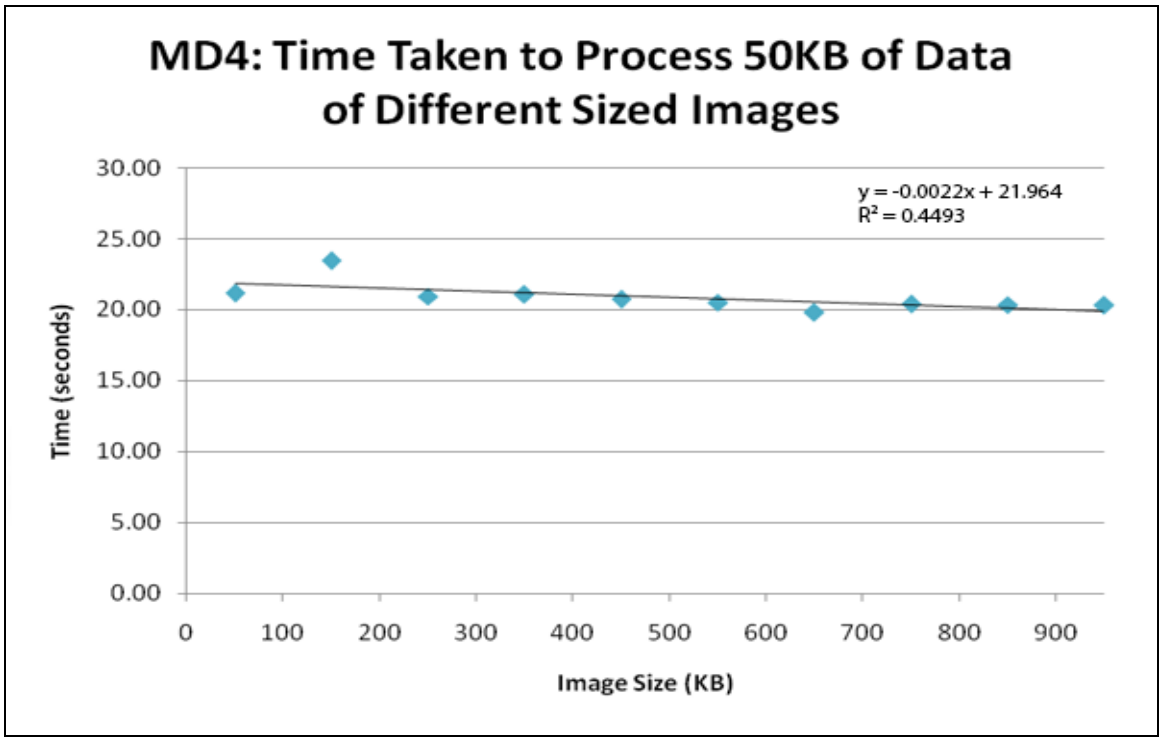


Figure 4.20 showing the time required to process 50KB of data in batches of different sized images using MD4

MD5

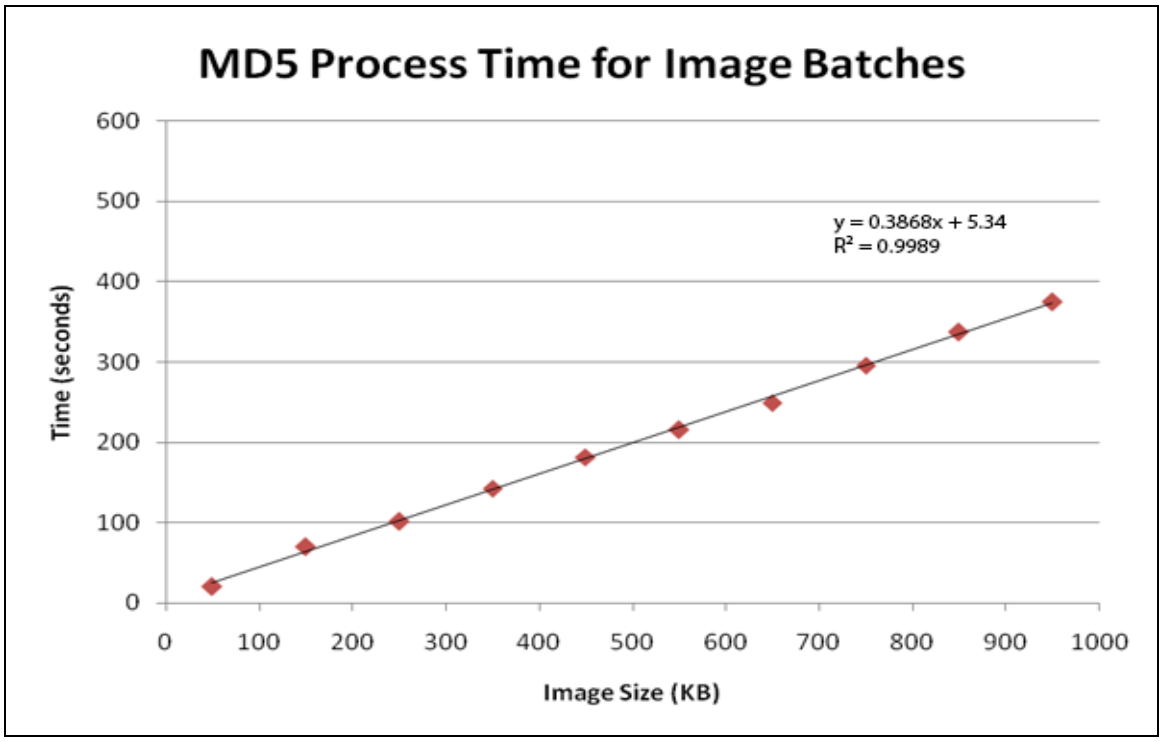


Figure 4.21 showing the time required to process 10 batches of 10,000 images using MD5 hashing algorithm

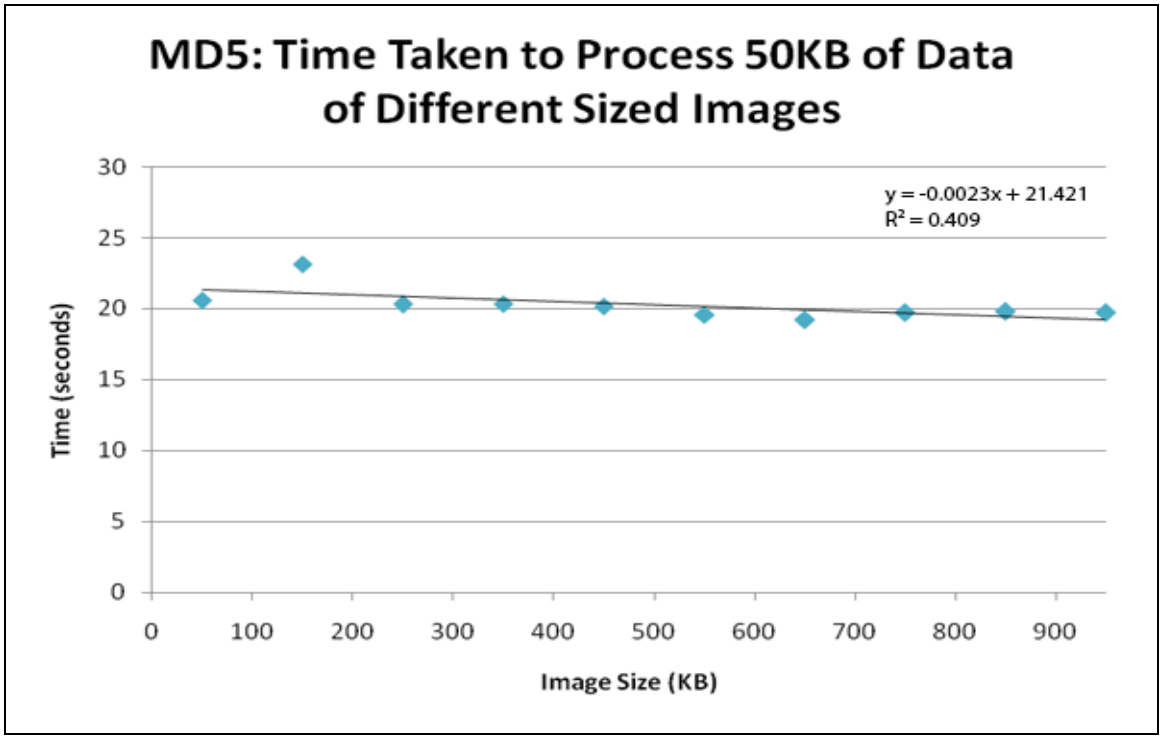


Figure 4.22 showing the time required to process 50KB of data in batches of different sized images using MD5

SHA1

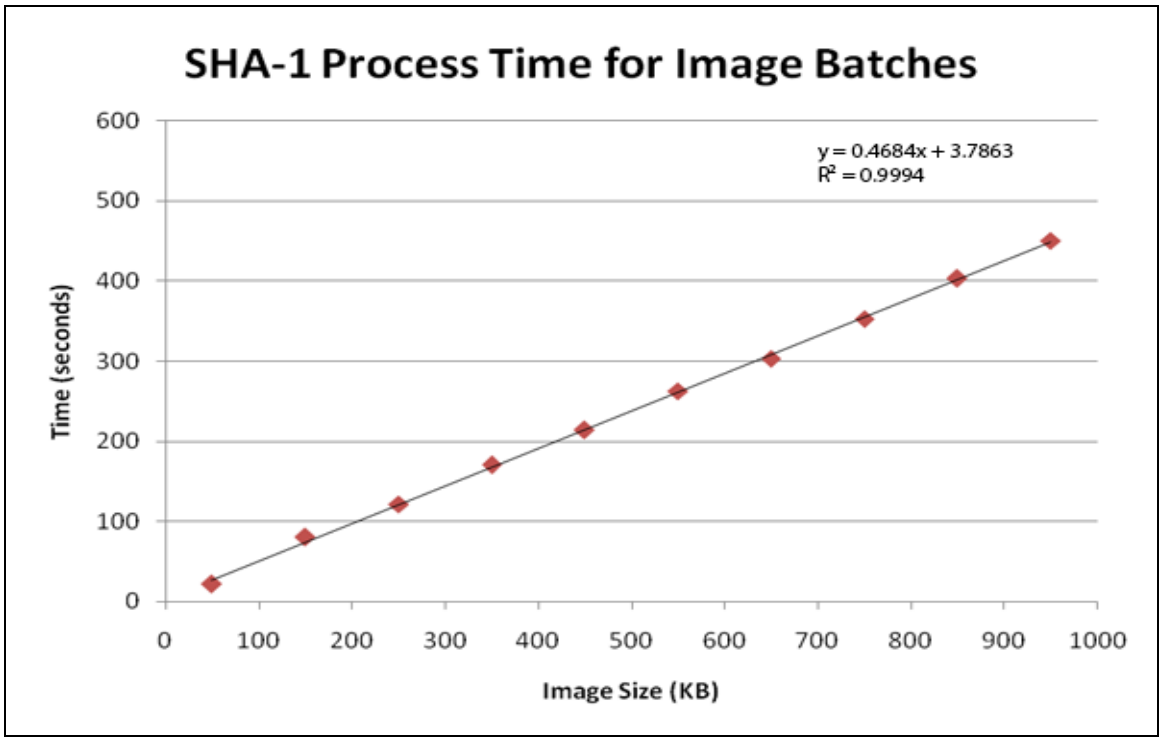


Figure 4.23 showing the time required to process 10 batches of 10,000 images using SHA-1 hashing algorithm

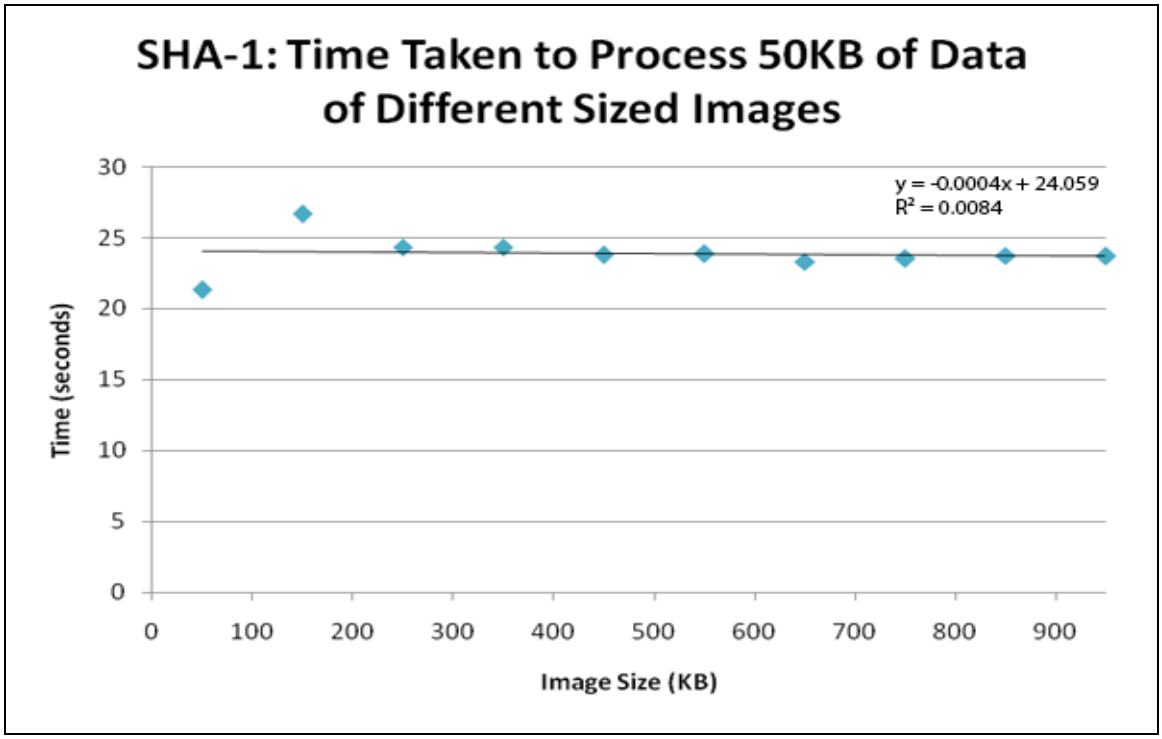


Figure 4.24 showing the time required to process 50KB of data in batches of different sized images using SHA-1

SHA-256

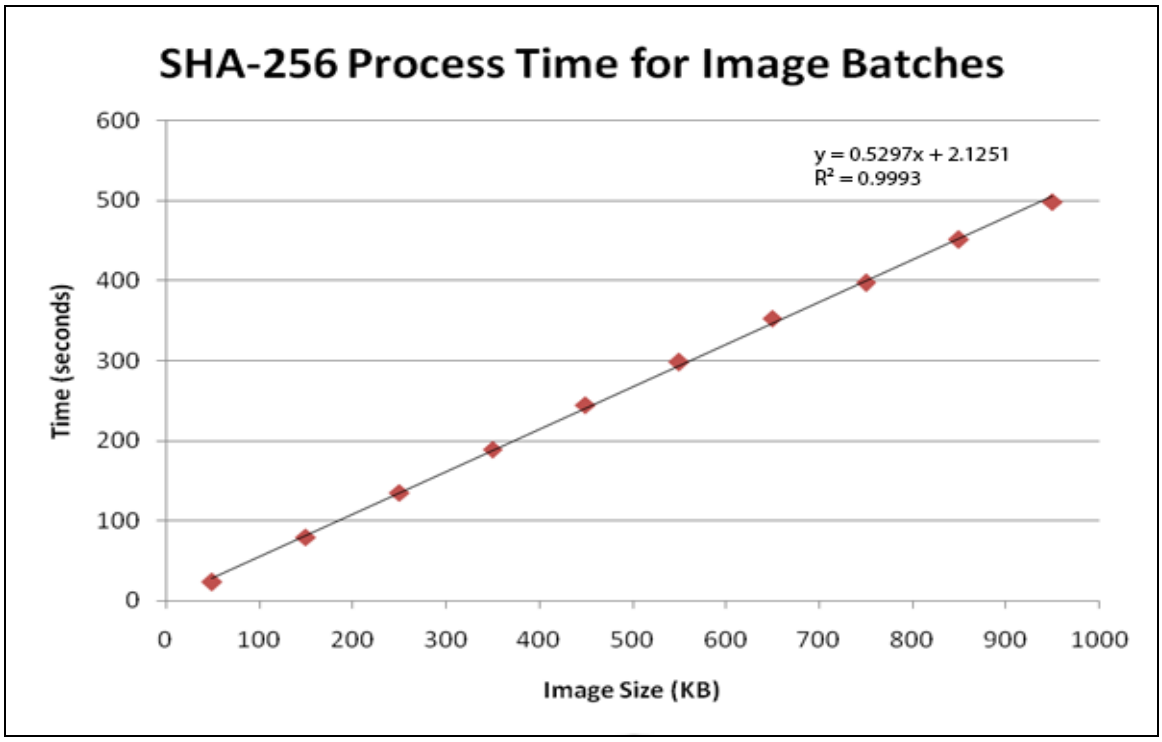


Figure 4.25 showing the time required to process 10 batches of 10,000 images using SHA-256 hashing algorithm

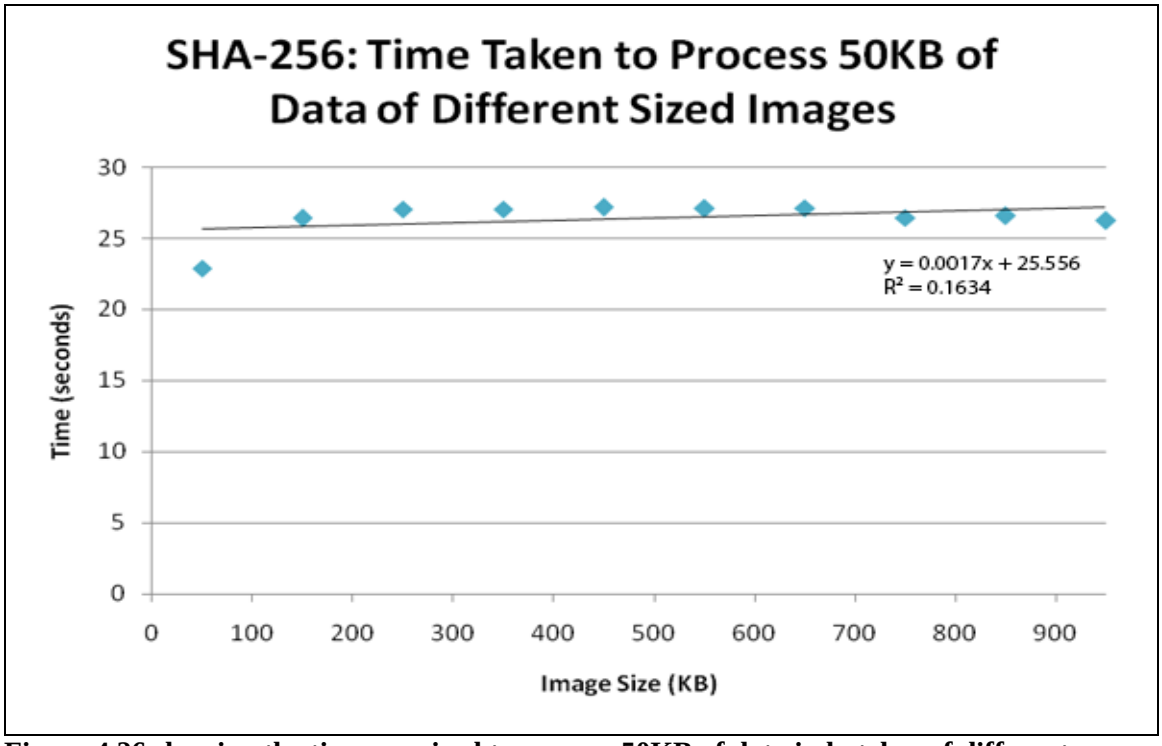


Figure 4.26 showing the time required to process 50KB of data in batches of different sized images using SHA-256CRC-

64

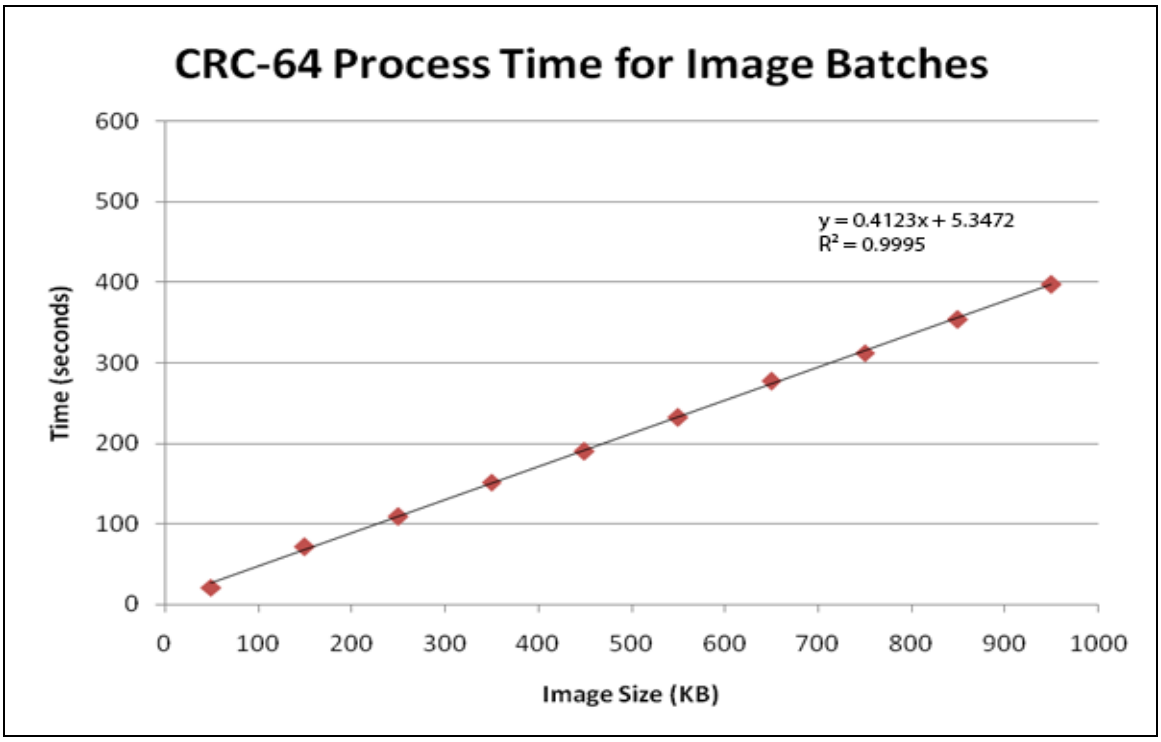


Figure 4.27 showing the time required to process 10 batches of 10,000 images using CRC-64 hashing algorithm

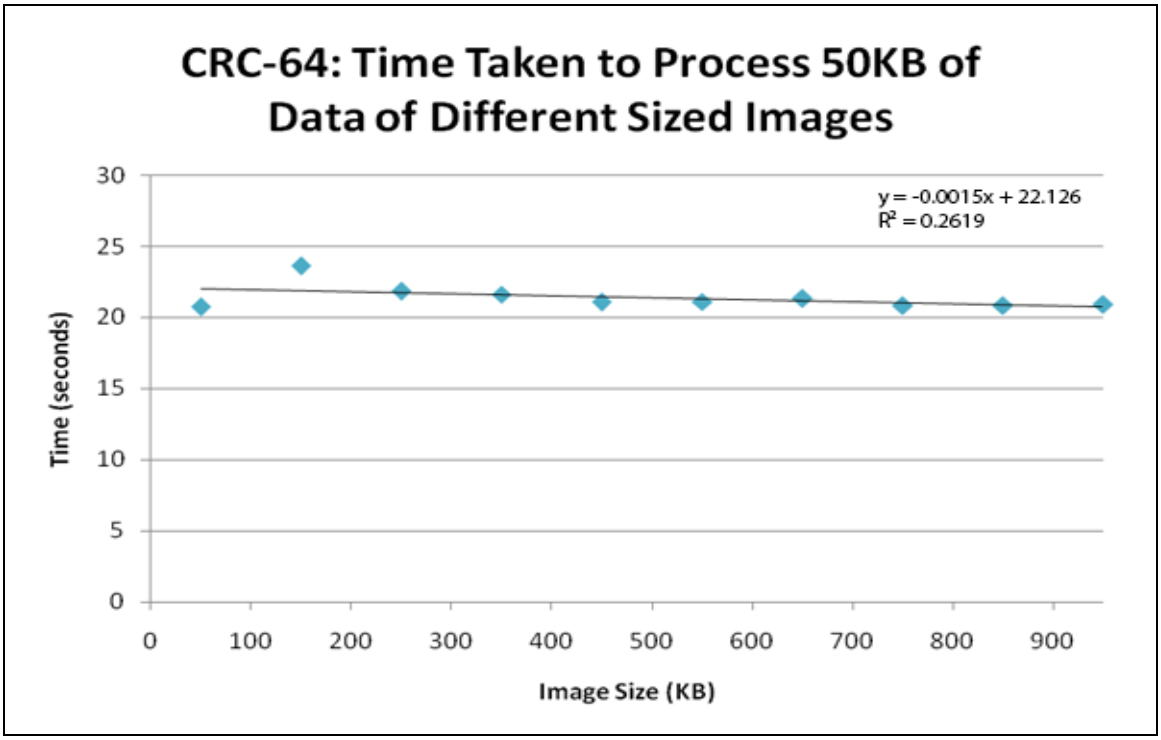


Figure 4.28 showing the time required to process 50KB of data in batches of different sized images using CRC-64

CRC-32

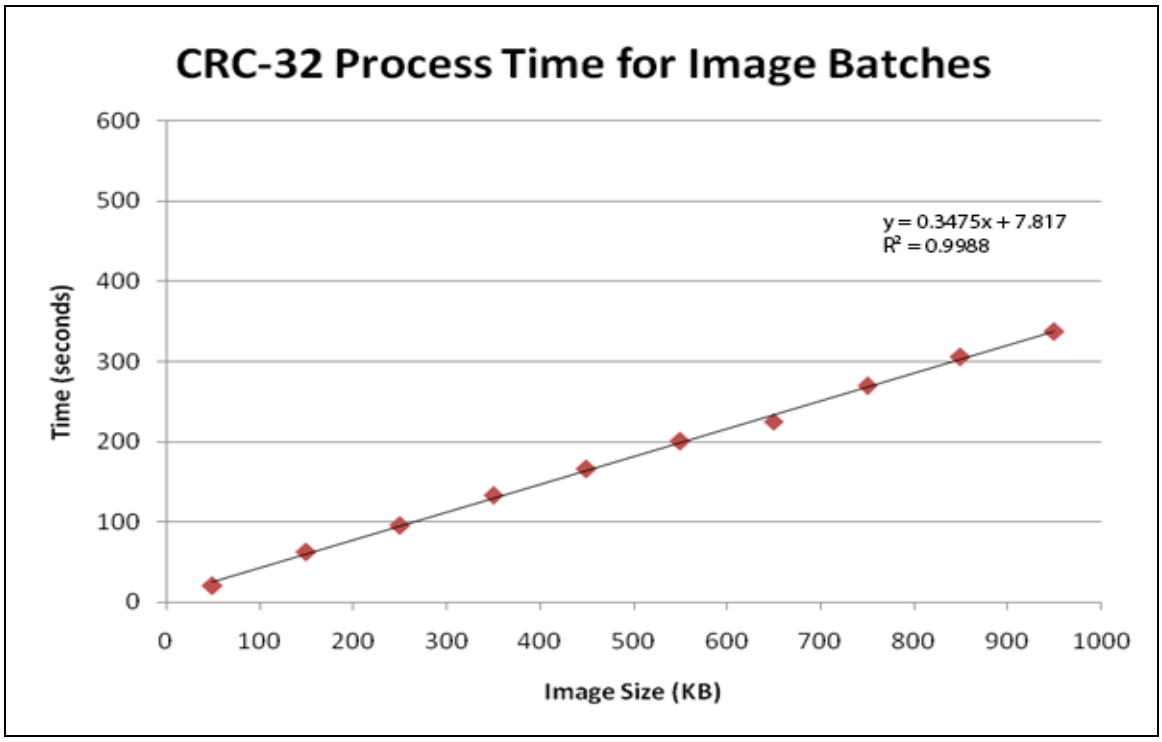


Figure 4.29 showing the time required to process 10 batches of 10,000 images using CRC-32 hashing algorithm

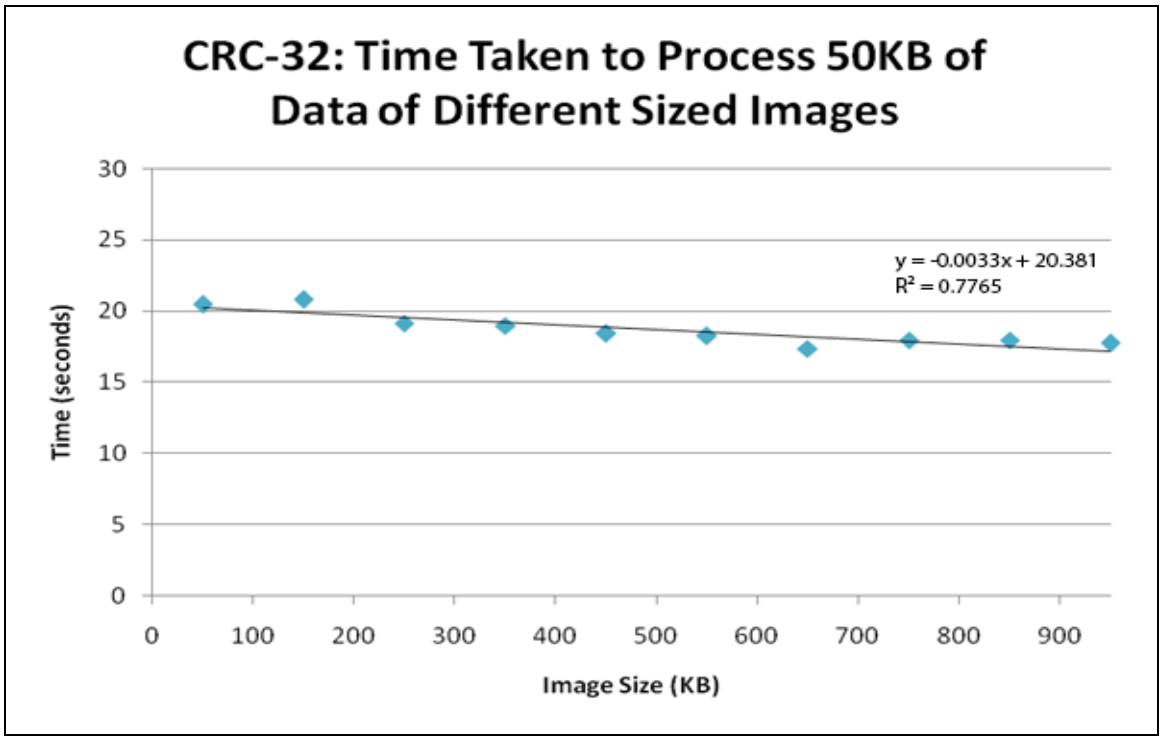


Figure 4.30 showing the time required to process 50KB of data in batches of different sized images using CRC-32

XXHash

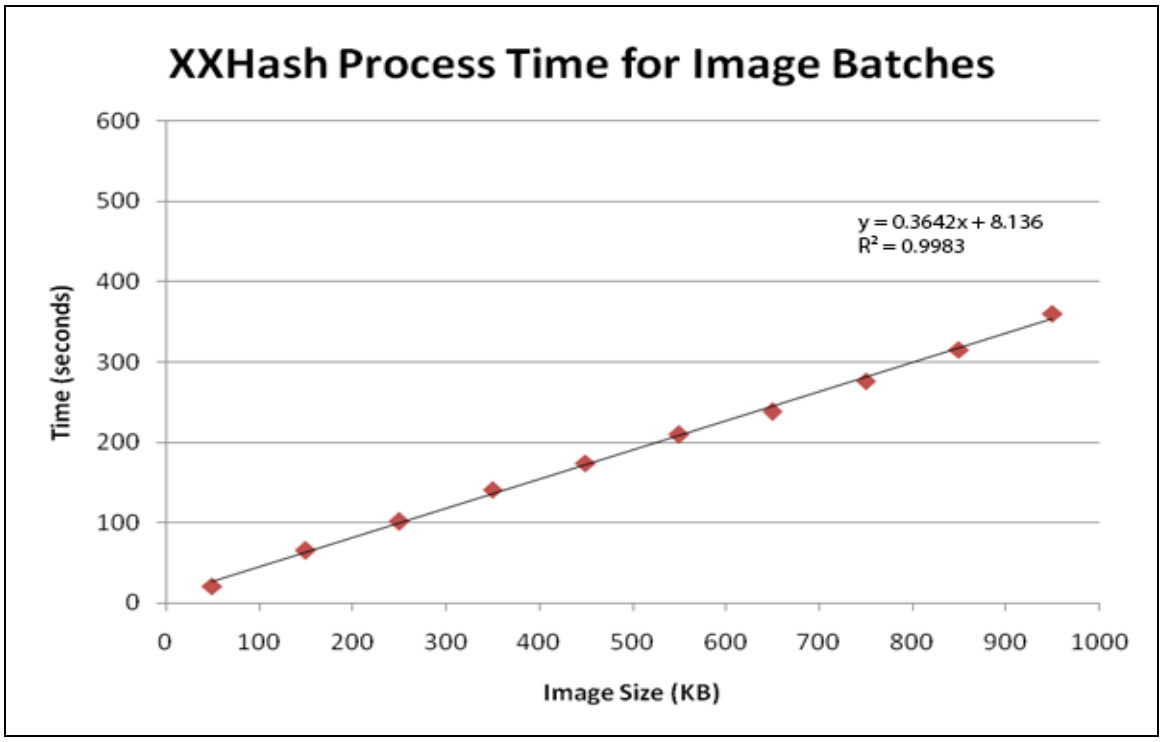


Figure 4.31 showing the time required to process 10 batches of 10,000 images using XXHash hashing algorithm

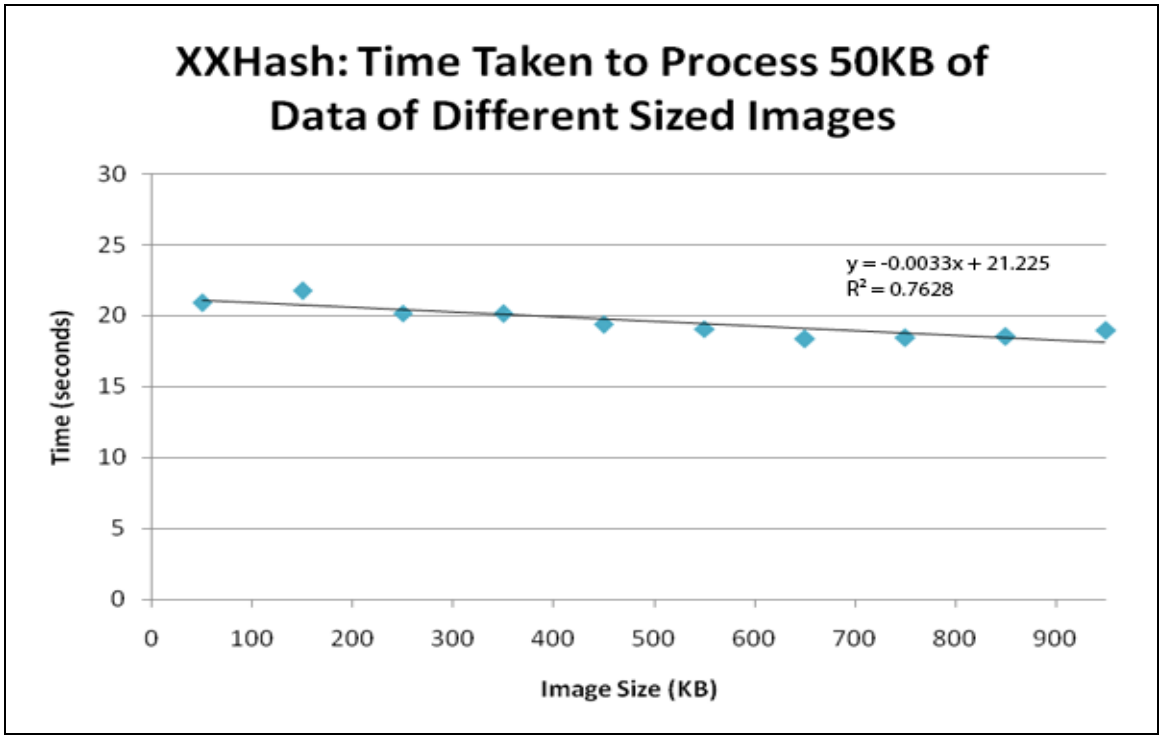


Figure 4.32 showing the time required to process 50KB of data in batches of different sized images using XXHash

FNV1a

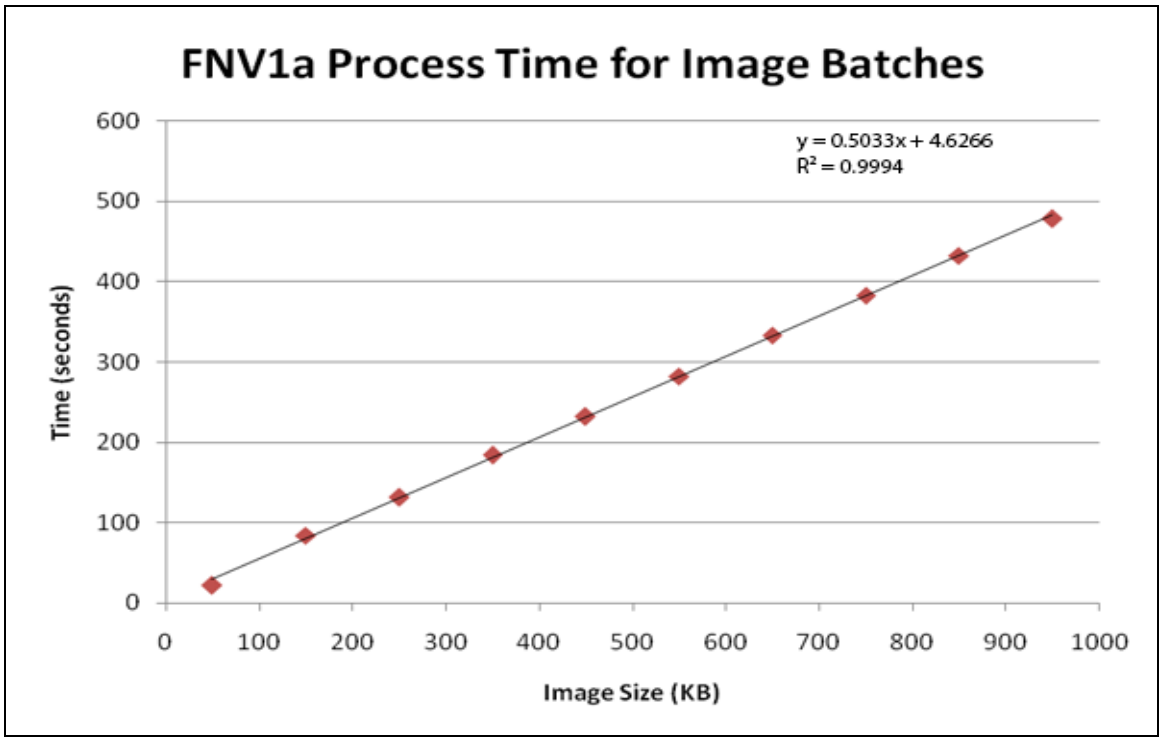


Figure 4.33 showing the time required to process 10 batches of 10,000 images using FNV1a hashing algorithm

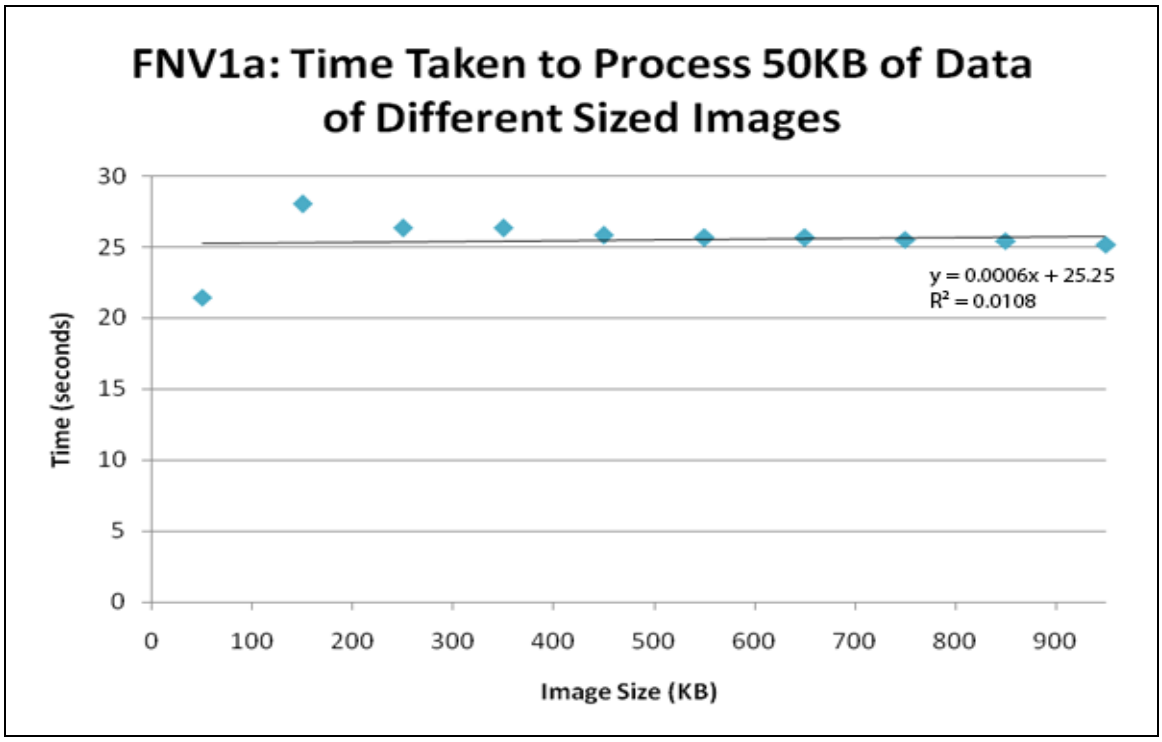


Figure 4.34 showing the time required to process 50KB of data in batches of different sized images using FNV1a

MurmurHash2

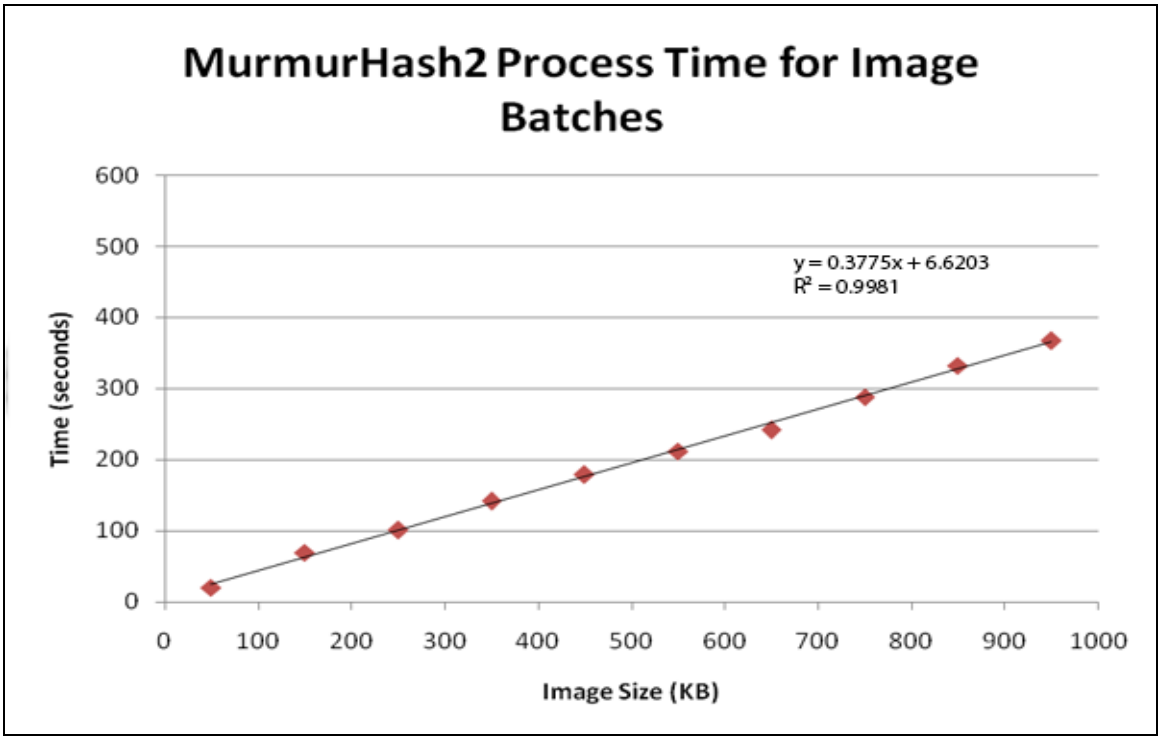


Figure 4.35 showing the time required to process 10 batches of 10,000 images using MurmurHash2 hashing algorithm

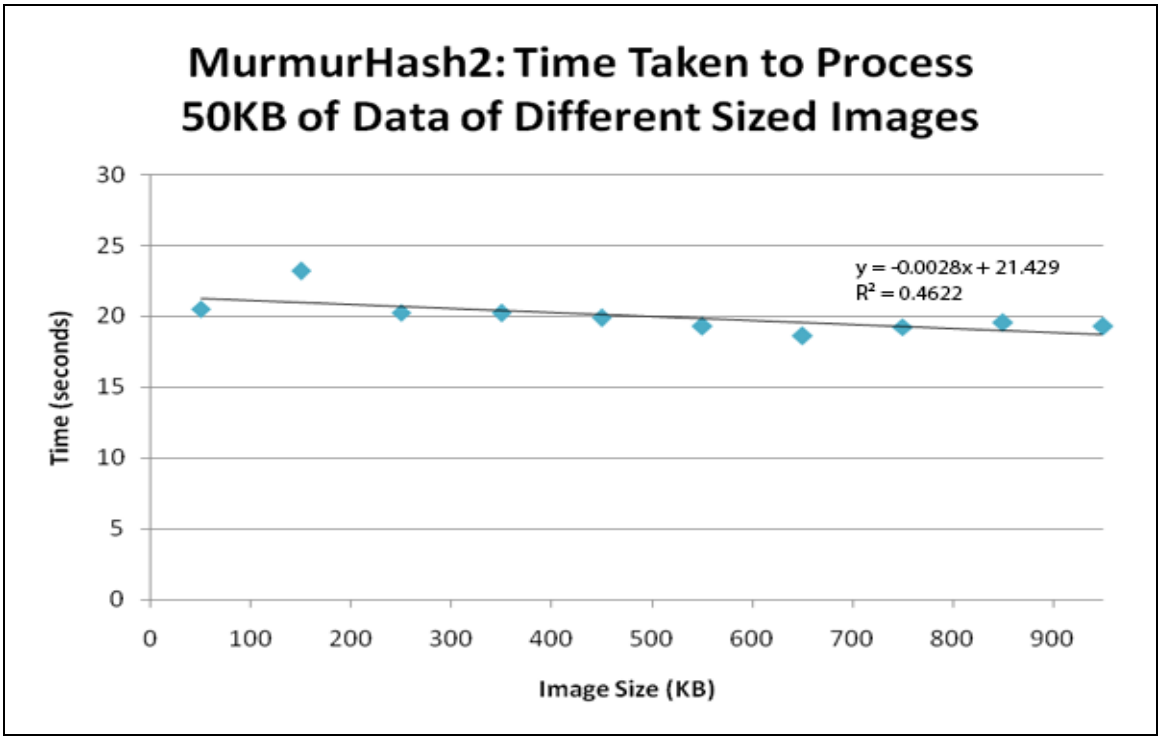


Figure 4.36 showing the time required to process 50KB of data in batches of different sized images using Murmurhash2

Tabularised Data:

The tabulated data below is the corresponding graphed data above. A full copy of the raw data can be found in Appendix B.

Table 4.5 showing the total hash process time in seconds for each batch, with each hashing algorithm (Part A)

File Size	MD4	MD5	SHA-1	SHA-256	CRC-64
50 KB	21.17	20.60	21.35	22.89	20.73
150 KB	70.28	69.47	80.04	79.29	70.90
250 KB	104.81	101.43	121.75	135.24	109.23
350 KB	147.35	142.50	170.32	189.46	151.43
450 KB	186.59	181.50	214.02	244.53	189.78
550 KB	225.82	215.27	262.47	298.25	232.16
650 KB	257.52	249.40	303.07	352.75	277.42
750 KB	305.83	295.77	352.73	397.09	312.25
850 KB	346.23	336.96	403.37	451.57	354.05
950 KB	385.83	374.25	450.58	498.56	396.82
Total	2051.43	1987.15	2379.7	2669.63	2114.77

Table 4.6 showing the total hash process time in seconds for each batch, with each hashing algorithm (Part B)

File Size	CRC-32	XXHash	FNV1a	Murmurhash2
50 KB	20.45	20.93	21.44	20.52
150 KB	62.44	65.26	84.03	69.64
250 KB	95.72	100.92	131.67	101.27
350 KB	132.88	140.86	184.23	141.39
450 KB	165.86	174.44	232.32	178.96
550 KB	200.76	209.74	282.02	212.21
650 KB	225.50	238.35	333.26	241.73
750 KB	269.57	276.31	382.61	288.49
850 KB	305.27	315.23	432.61	332.59
950 KB	337.40	360.37	478.34	366.95
Total	1815.85	1902.41	2562.53	1953.75

Table 4.7 showing the average time taken to process 50KB of data in 10 batches of different sized images (Part A). Value = Batch Time / (Image Size / 50)

File Size	MD4	MD5	SHA-1	SHA-256	CRC-64
50 KB	21.17	20.60	21.35	22.89	20.73
150 KB	23.43	23.16	26.68	26.43	23.63
250 KB	20.96	20.29	24.35	27.05	21.85
350 KB	21.05	20.36	24.33	27.07	21.63
450 KB	20.73	20.17	23.78	27.17	21.09
550 KB	20.53	19.57	23.86	27.11	21.11
650 KB	19.81	19.18	23.31	27.13	21.34
750 KB	20.39	19.72	23.52	26.47	20.82
850 KB	20.37	19.82	23.73	26.56	20.83
950 KB	20.31	19.70	23.71	26.24	20.89

Table 4.8 showing the average time taken to process 50KB of data in 10 batches of different sized images (Part B). Value = Batch Time / (Image Size / 50)

File Size	CRC-32	XXHash	FFNV1a	Murmurhash2
50 KB	20.45	20.93	21.44	20.52
150 KB	20.81	21.75	28.01	23.21
250 KB	19.14	20.18	26.33	20.25
350 KB	18.98	20.12	26.32	20.20
450 KB	18.43	19.38	25.81	19.88
550 KB	18.25	19.07	25.64	19.29
650 KB	17.35	18.33	25.64	18.59
750 KB	17.97	18.42	25.51	19.23
850 KB	17.96	18.54	25.45	19.56
950 KB	17.76	18.97	25.18	19.31

4.4.1.2 Identifying the Fastest Database

Identifying the fastest database is the second critical function to the performance of the HBIR tool. To identify the fastest database two functions needed to be tested. The first is to record the time for the known objectionable hashes to be inserted into the database. The second is the recorded time to compare the new hashes generated by the HBIR Tool to the hashes currently stored in the database. Below is the tabulated data of time recorded for Inserting Hash Values into the Database and Comparing Hash Values. Both functions were tested using the PostgreSQL, Microsoft SQL, and MySQL relational databases..

Table 4.9 showing summaries of database testing (does not include time to hash file)

Test	PostgreSQL	Microsoft SQL	MySQL
Inserting 250,000 hash values into the Database (excluding hash time)	6233 seconds	6347 seconds	11780 seconds
Average Insertion Time (AIT) of one hash value	2.4932x10-6	2.5388x10-6	4.712*10-6
Placing for Inserting Hash Values	1st	2nd	3rd
Comparing 10,000 Hash Values to 250,000 stored hash values (excluding hash time)	446 seconds	557.2 seconds	1147 seconds
Average Comparison Time (ACT) for comparison between one hash value one database stored hash value	1.784 x 10^-7	2.2288x1-^-7	4.588x10^-7
Placing for Comparing Hash Values	1st	2nd	3rd

Calculations:

Using the simple formulas below one can calculate the time required to either insert values into the database or by comparing values from the tool to the values in the database.

Time required inserting *x*-hash values in to database:

➔ Time required = number of values * Average Insertion Time

Time required comparing *x*-hash from tool and *y*-hash from database:

➔ Time required = (num. *x-hashes* * num. *y-hashes*) * Average Comparison Time

Using a combination of Table 4.5 and Table 4.7 estimations can be calculated to model the time required to hash the image only, hash the image and insert the hash value into the database, and hash the image then compare hashes. Furthermore this can be calculated using any combination of any hashing algorithm and any database platform.

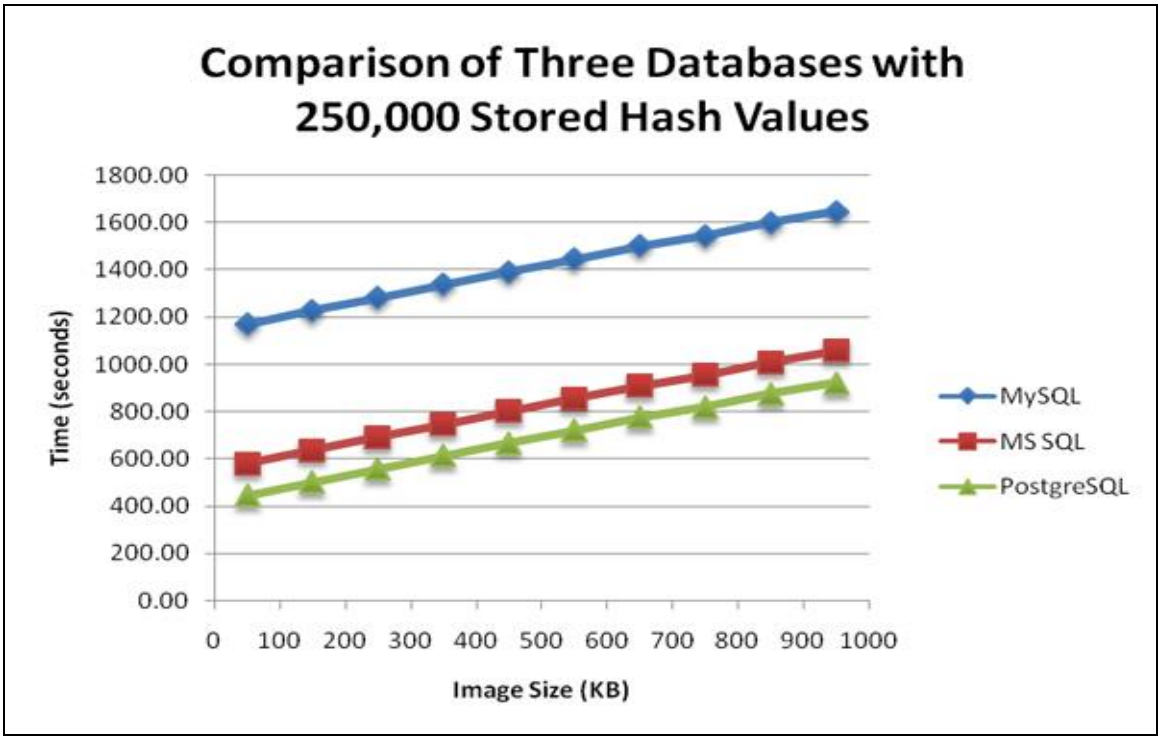


Figure 4.37 showing a display of process times when each batch of images is hash and compared to 250,000 stored hash values (using three different databases)

Table 4.10 showing corresponding data to Figure 4.37

Batch Size	MySQL	MS SQL	PostgreSQL
50 KB	1170.00	580.00	446.00
150 KB	1226.40	636.40	502.40
250 KB	1282.35	692.35	558.35
350 KB	1336.57	746.57	612.57
450 KB	1391.64	801.64	667.64
550 KB	1445.36	855.36	721.36

650 KB	1499.86	909.86	775.86
750 KB	1544.20	954.20	820.20
850 KB	1598.68	1008.68	874.68
950 KB	1645.67	1055.67	921.67

4.4.1.3 HBIR Recommendations

The HBIR Recommendations is the section where the final combinations of software previously tested will be evaluated and selected. The selection of the best combination of software will reflect back on the goals of the research that were previously discussed in Section 3.1, Issues and Gaps in Current Research. This section begins by summarising the goals listed in chapter 3.1. It will continue by using the goals as justification for the selection of the hashing algorithms and database. In Issues and Gaps in Current Research the main goal was to find a technique that balances accuracy and processing time to identify objectionable images. The second goal was to identify a long term solution. It was noted that finding a long term solution was going to be a difficult aspect to measure within the available research period. However, best practices could be followed to maximise the prospects. Therefore the hashing algorithm that has been selected is SHA-256. It has been selected for four reasons. The first reason is SHA-256 supports a very large bit size of 256bits. This large bit size gives any image the possibility of having 1 of a possible 1.1579209×10^{77} hash values. The large bit size will also create fewer collisions and will support the second goal to maximise the software's life expectancy. Thirdly, SHA-256 is a NIST approved cryptographic hashing algorithm, so it has been mathematically designed and proven to minimise the number of collisions unlike non-cryptographic algorithms. Furthermore, Table 4.5 illustrates the total time required to process all 10 batches of images using MD5 (128 bit hash value) is 1987.15 seconds compared to 2669.63 seconds for SHA-256. The benefit of having twice the bit length outweighs the 34.2% increase in processing time required by SHA-256.

In Section 4.4.1.2 MS SQL, PostgreSQL, and MySQL were selected and tested thoroughly to identify the fastest database for inserting hash values and

comparing hash values. As Figure 4.37 clearly illustrates, the best performing database is PostgreSQL. Therefore PostgreSQL will be selected as the second main component for the HBIR tool.

To conclude the above experiments have shown that testing was performed on multiple hashing algorithms with multiple different database platforms. It has also shown that the SHA-256 hashing algorithm using the PostgreSQL database is the best combination of software for the HBIR Tool.

4.4.2 Content Based Image Retrieval

The CBIR experiment (Research Phase 3) investigates and measures the amount of time that is required to process each of the batches using the Explicit Image Detection function developed by Forensic Tool Kit

To execute the tool, the location of the individual batch images is selected and the configuration settings are configured. The experiment stage has replicated the same configuration settings as were run in the pilot test. Therefore only two settings were enabled. The settings were *Explicit Image Detection* with the sub setting of *X-DFT + X-FST (faster)* and *File Signatures*. The *File Signatures* parameter is a requirement for the Explicit Image Detection function. Below is a graph in figure 4.37 and a table 4.9 showing the processing times of the EID function for each batch of images. These are the final results that will be used in Section 4.4.4 Technique Comparison. . A full copy of the raw data can be found in Appendix C.

4.4.2.1 CBIR Image Processing Times

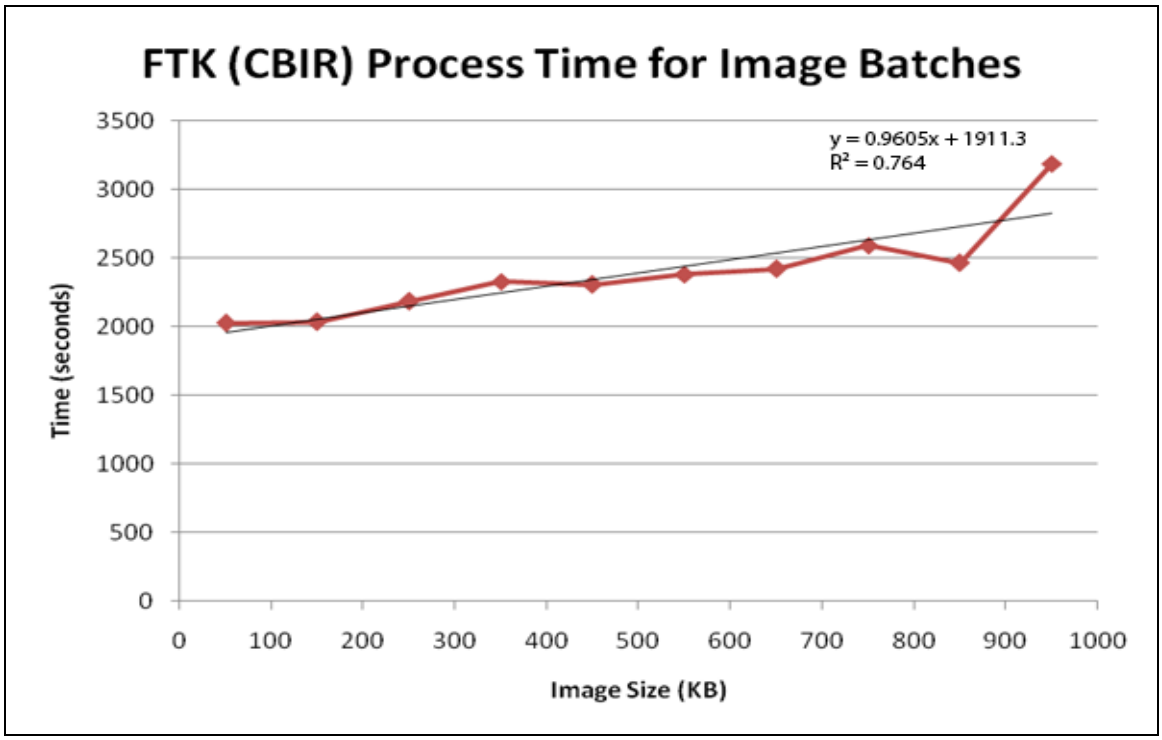


Figure 4.38 showing process time of batches of images using CBIR technique

Table 4.11 showing total process time in seconds for each batch of images using Forensic Tool Kit (CBIR) Software

Image Batch Size	Time (seconds)
50 KB	2023.5
150 KB	2036.0
250 KB	2183.8
350 KB	2327.0
450 KB	2305.8
550 KB	2379.4
650 KB	2418.8
750 KB	2591.0
850 KB	2462.4
950 KB	3187.8

4.4.3 Concept Based Image Indexing

The CBII experiment (Research Phase 4) investigates and measures the amount of time that was required to process the 10 batches of 10,000 images using the CBII technique. Encase has been selected as the tool for testing because it is widely used, accepted in courts globally and has an excellent reputation for being extremely efficient and thorough.

To use the tool, the 10,000 images were acquired by Encase for analysis. Once the images had been acquired a keyword list containing 2000 key words was entered with the selection of *GREP* and *Case Sensitive ANSI Latin -1* enabled. All other settings were not selected as this would increase the processing time (refer to Figure 4.16: Keyword list for CBII). After entering the keyword list, the next step was to enter the search criteria. Again, to maximise the performance only the minimal requirements were selected. Those requirements were to only search the selected 10,000 images and only use the previously entered 2,000 keywords as search parameters (refer to Figure 4.17 CBII Search parameters).

The time started once the search function began, therefore the time did not include the acquisition of the images. This decision was made because during the experiment of the HBIR acquisition was also not included. Below is a table 4.10 and a figure 4.38 graph that show the processing times of each batch of images using the CBII technique. A copy of the testing data can be found in Appendix D.

4.4.3.1 CBII Processing Times

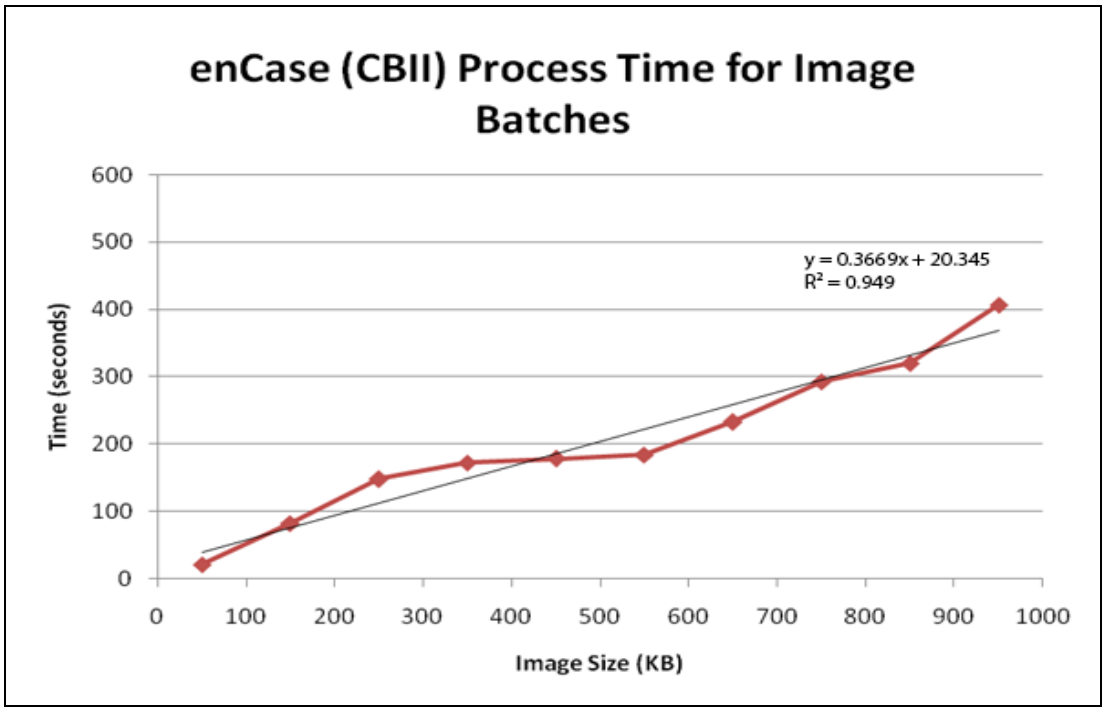


Figure 4. 39 showing the process time of batches of images using CBII technique

Table 4. 12 showing the total process time in seconds for each batch of images using Encase (CBII) Software

Image Batch Size	Time (seconds)
50 KB	21
150 KB	82
250 KB	148
350 KB	172
450 KB	178
550 KB	184
650 KB	233
750 KB	293
850 KB	320
950 KB	407

4.5 TECHNIQUE COMPARISON

The technique comparison shows the final data collection and presentation of all three image retrieval techniques. The total processing times were recorded for each of the 10 batches of 10,000 images and are illustrated on Figure 4.39 below. The raw processing times can be found on the Table 4.11. The HBIR Tool used an objectionable database listed with 250,000 hash values.

4.5.1 Graphical Comparisons

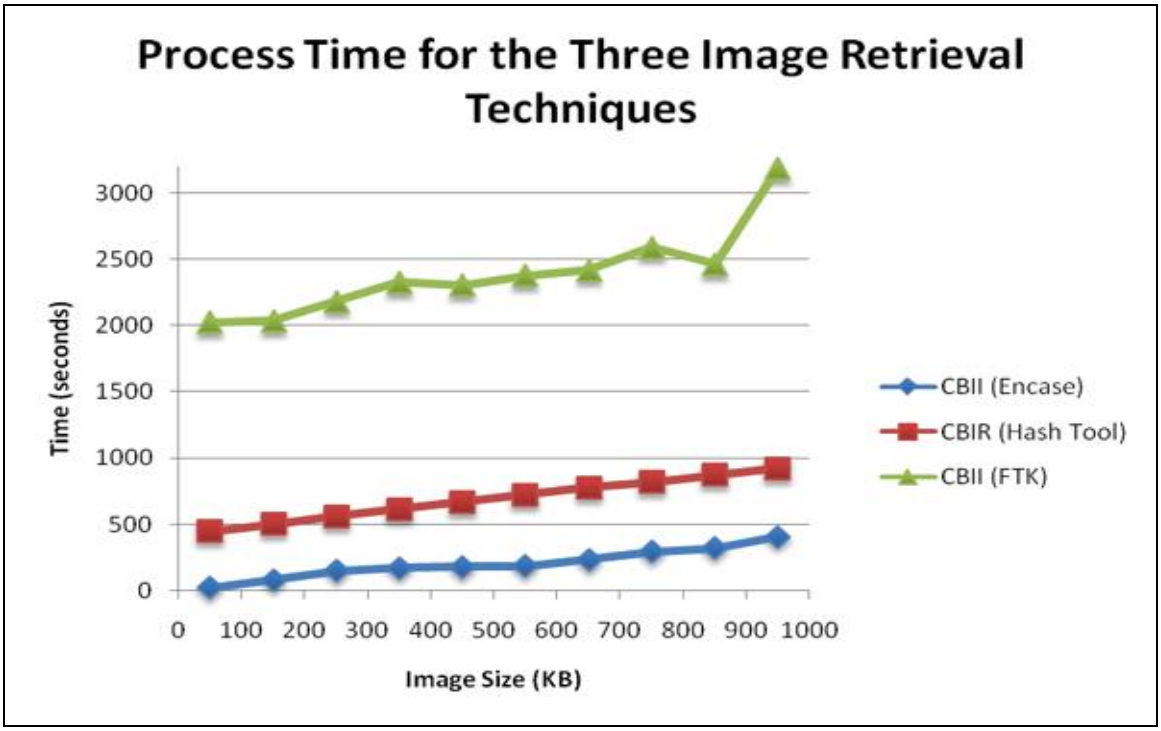


Figure 4.40 showing the process times of all three image retrieval techniques (HBIR, CBIR and CBII)

Table 4.13 showing the process times for all three image retrieval techniques (HBIR, CBIR, CBII) in seconds.

Image Batch Size	CBII (Encase) Time	HBIR (Hash Tool) Time	CBIR (FTK) Time
50 KB	21	446	2024
150 KB	82	502	2036
250 KB	148	558	2184
350 KB	172	613	2327
450 KB	178	668	2306

550 KB	184	721	2379
650 KB	233	776	2419
750 KB	293	820	2591
850 KB	320	875	2462
950 KB	407	922	3188

4.5.2 Summary

Section 4.4.4 shows and compares the processing times of the three different image retrieval techniques. Figure 4.39 shows that the CBII technique requires the least amount of time to process the 10 batches of images, followed by the proposed HBIR technique, and finally the CBIR technique.

4.6 CONCLUSION

Chapter 4 began by discussing the variations in data requirements from what were first discussed in chapter 3, including the explanation of why the variations were necessary. The variation in data requirements was followed by the Research Phase 1, Software Testing. Software testing (Section 4.2) was the section where preliminary software testing was performed on the HBIR Tool to identify any glitches with the software. The second stage of Phase 1 included pilot testing (Section 4.3) of all three image retrieval techniques. Pilot testing was performed on a hypothetical real world example of where an 8GB USB drive was identified by police of possibly containing objectionable images. This stage was necessary to identify if the current experiment methodology was feasible. This was followed by the Experiment stage (Section 4.4). In the Experiment stage, research phases 2, 3 and 4 was completed and the findings were appropriately illustrated using tables and figures. Once the experiment was completed the three techniques were analysed and compared in Section 4.5 Technique Comparison.

In chapter 5, the Research Discussion, will now review the findings documented in Section 4.4 and Section 4.5. It will answer the research questions and hypotheses that were identified in chapter 3. A reflection section will follow by reflecting on the testing processes, strengths and weaknesses of the research and the

bench marking software used to evaluate the HBIR Tool. Once the critical reflection has been completed, the next section will involve theory building (Section 5.6) by analysing the findings in Section 4.4 and Section 4.5. Section 5.7 will review the recommendations.

Chapter Five

RESEARCH DISCUSSION

5.0 INTRODUCTION

Chapter 5 will discuss the research findings that were previously presented in chapter 4. Chapter 4 first began by discussing the variations in data requirements (Section 4.1). Variations of data requirements can be defined as the required changes in the data generation, data collection and data presentation stages. These changes impacted to results and required clarification. Variations in data requirements were followed by the Research Phase 1 including software testing (Section 4.2) and pilot testing (Section 4.3). The experimental stage (Section 4.4) focused on the results from testing of the HBIR (Research Phase 2), CBIR (Research Phase 3) and CBII (Research Phase 4) techniques as per the research data map. This was followed by Section 4.5 where a technique comparison was completed.

Chapter 5 will now respond to the research questions and hypotheses that were documented in Section 3.2. It will then continue with a discussion of the findings (Section 5.2) where each technique will be thoroughly discussed including how effective it would be in a real world environment. Section 5.3 critically evaluates and reflects on the testing procedures and Section 5.4 evaluates the strengths and weaknesses of the research. A review of the benchmarking software is explained in Section 5.5 and this section is concluded with Section 5.6 for conceptualising and theory building.

5.1 DISCUSSION OF RESEARCH QUESTIONS

The literature review in chapter 2 identified numerous gaps in research in regards to identification of objectionable images. The gaps in current research were summarised in Section 3.1 as potential research areas. These research areas were categorised into five groups; image processing, legal and political issues, accuracy, anti-forensics and

creating a long term solution. Section 3.2 analysed the research areas until a main research question, five sub research questions and their corresponding hypotheses could be developed. The sub questions were carefully and critically selected from 10 questions to assist the main question. Section 5.1 will now answer these questions from Section 3.1 in the tables below. The tables will contain the research question, hypothesis, arguments for the hypothesis, arguments against the hypothesis and a summary as to whether the hypothesis will be proven true or false.

5.1.1 Tables

Table 5.1 Main Question and Hypothesis

MAIN QUESTION: ***How can the efficiency of image processing be increased to identify objectionable images in a smaller timeframe?***	
MAIN HYPOTHESIS: **The HBIR technique will be a feasible solution to identifying objectionable images from a population containing objectionable and non-objectionable images.**	
ARGUMENT FOR: **The first point of argument is that the HBIR Tool did detect the objectionable images from a population of unknown images. Testing has shown that in every experiment the objectionable image was successfully detected.** **The second point to confirm the feasibility is the speed of being able to insert, generate, or compare hash values. This will become faster as better database software, hashing algorithms, and hardware is developed. Researchers and engineers are always striving to push the limits and expand current technology to be able generate faster and smaller electronics. This can help in processing times for the future.** **Thirdly, a new, more efficient hashing algorithm developed can be easily adapted to the HBIR Tool. The time taken to insert 250,000 hash values into the database only required 6233 seconds; therefore 10 million images could be inserted in less than 3 days. Furthermore, that time is only using a desktop computer, not a high end server where the time could easily be halved. Therefore as new hashing algorithms are**	**ARGUMENT AGAINST:** The first point against the feasibility of the HBIR tool is the accuracy. Currently CBIR techniques can identify images even when small changes are made to the visual component of the image. In contrast, if the image is altered in any way the HBIR Tool will overlook the image as objectionable. Secondly, by using the SHA-256 hash value it has already predetermined the life expectancy of the software by only be able to create a limited number of hash values. Other techniques do not have such limitations. Thirdly, as more hash values are entered into the database, it will take longer to compare hash values of unknown images. Testing was only performed on 10,000 images at a time with a database containing 250,000 hash values. A more realistic value might be 10,000 images with a database of 10 million hash values. Lastly, the fourth issue is that the system is susceptible to forensic investigators inserting non objectionable images either by accident or because definitions of what is objectionable is not standardised. As chapter 2 discussed each country has a different understanding of

developed it is very feasible to rehash all the objectionable images, even if there are millions of images. **Fourthly, testing showed the recorded times for the HBIR Tool appeared to be very linear across all 10 batches of images. Therefore a model can easily be developed to demonstrate how long it will take to process images of larger sizes. The calculations show the HBIR Tool will have no issue with processing larger images.** **Lastly, the cost of maintaining a system is minimal. The hashing algorithms are freely available and development of the tool is a one off charge. The experimental databases are free. Therefore the only ongoing cost is hardware related.**	what is objectionable. Therefore legal images may be inserted into the database resulting in false positives.
SUMMARY: **In summary, the hypothesis states the HBIR technique will be a feasible solution to identifying objectionable images from a population containing objectionable and non-objectionable images. Extensive testing has shown that the HBIR technique is an excellent technique for identification of objectionable images. It is fast at identification, uses efficient algorithms, is accurate with minimal false positives and is feasible solution for the foreseeable future. Therefore the hypothesis has been proven to be true.**	

Table 5.2 Sub Question 1 and Hypothesis

SUB QUESTION 1: ***Will a hash based solution perform faster when compared to CBIR and CBII?***	
HYPOTHESIS 1: **The hash based image retrieval technique will perform faster than content based image retrieval and concept based image indexing techniques.**	
ARGUMENT FOR: **As expected the Hash Based Image Retrieval could process images very quickly. The experiment results showed the proposed HBIR technique could process the images in approximately 25% of the time that the CBIR technique required. Furthermore, the CBIR technique created false positives creating further delay as the images required to be manually reviewed. Moreover, the CBIR technique was configured with the fastest settings. FTK recommends performing one scan on the fast X-DFT + X-FST settings which will reduce the number of images. Then rerun the scan on X-DFT + X-ZFN (slow but more accurate) setting which will reprocess the**	**ARGUMENT AGAINST:** As Figure 4.40 demonstrates CBII was the fastest technique to process all 10 batches of images. CBII processing times ranged from 21 seconds for the smaller batches and 407 seconds for the largest batch. In second place was the HBIR technique, followed by the CBIR technique. The HBIR technique is also reliant on a hashing database which effects the processing time. The experiment used a database containing 250,000 hash values however if that value was increased to 500,000 the processing time for the HBIR would be doubled. Therefore with a large enough databases it could even be

images. Following the recommendations of FTK the number of false positives would be reduced but the processing time could take twice as long. Unlike the proposed HBIR technique which will be incredibly rare to have a false positive. **Even though the CBII technique did record the fastest time, Encase has had extensive amounts of financial and human resources to develop the CBII search engine, unlike the HBIR technique. Therefore it should be expected with further development on the HBIR Tool it could potentially process images significantly faster.**	slower than the CBIR technique.
SUMMARY: **The experiment could have been manipulated by the proposed HBIR technique using a smaller database of hash values. However keeping consistent with the Design Science Methodology it was important to keep the experiment fair across all three techniques. Testing has showed that CBII is the fastest technique and not the proposed HBIR technique. Therefore, the hypothesis has been proven to be false.**	

Table 5.3 Sub Question 2 and Hypothesis

SUB QUESTION 2: ***Which hashing algorithm category performs the fastest?***	
HYPOTHESIS 2: **Non-cryptographic hashing algorithms will perform faster than the cryptographic hashing algorithms.**	
ARGUMENT FOR: **In total five non-cryptographic and four cryptographic hashing algorithms were tested and documented in Section 4.4.1.1. The results showed on average the three fastest hashing algorithms were non-cryptographic. CRC-32 was the fastest at processing all 10 batches of images with a total time of 1815.85 seconds. XXHash was the second fastest with a total time of 1902.41 seconds. Lastly, Murmurhash2 was the third fastest with a total time of 1953.75 seconds.**	**ARGUMENT AGAINST:** MD4 and MD5 cryptographic hashing algorithms were fastest cryptographic algorithms with total process times of the 10 batches of images with 2051.43 seconds and 1987.15 seconds, respectfully. The CRC-64 non-cryptographic hashing algorithm was able to process the 10 batches of images in 2114.77 seconds.
SUMMARY: **Testing showed when comparing the two categories of algorithms there was very little difference in processing times. The most influential factor for the speed of the algorithm was actually the size of the hash value. As non-cryptographic hashing algorithms are designed for speed and less accuracy the bit size is generally smaller. However the three fastest algorithms were CRC-32bit, XXHash-32bit, and MurmurHash-64bit. The three slowest were SHA-1-160bit,**	

FNV1a-64bit and SHA-256bit. **Therefore the hypothesis is proven to be true. Non cryptographic hashing algorithms do perform faster when compared to cryptographic hashing algorithms.**

Table 5.4 Sub Question 3 and Hypothesis

SUB QUESTION 3: ***What is the genuine throughput of images per second with the most effective performing hashing algorithm?***	
HYPOTHESIS 3: **The hash based solution will process 100 images per second.**	
ARGUMENT FOR: **The number of images that can be processed per second is proportional to the speed of the system's hardware. Therefore even though the number of images processed per second was not more than the expected 100 on the test machine, it could be easily accomplished by using server hardware.**	**ARGUMENT AGAINST:** The fastest algorithm is CRC-32 which hashed all 10 batches of images in 20.45 seconds for the 50KB batch and in 337.40 seconds for the 950KB batch. In addition CRC had a combined hashing time for all 10 batches of 1815.85 seconds (as per Table 4.5). However, these times only include the hashing time and not the comparison time. The comparison time needs to be added to the hashing time to calculate the total time. Furthermore, the time that needs to be added is dependent on the number of hash values that are stored in the database. For instance, to compare 10,000 hash values requires 446 seconds with a database containing 250,000 hash values. Using the calculation of (Num. Images /(hash time + compare time) the total process time (hash and compare) can be easily calculated. The images per second for the 50KB batch of images with a database containing 250,000 hash values using the CRC-32 hashing algorithm is 21.4 images per second. In addition using the batch containing 950KB in file size, the tool processed the images at 12.8 images per second.
SUMMARY: **The experiment showed in the current testing environment processing of images using the CRC hashing algorithm ranged between 21.4 images per second for the 50KB images and 12.8 images per second for the 950KB images. To be able to process 100 images per second the number of hash values in the database would need to be at an unrealistic level or different hardware of hardware would need to be used. Consequently, the HBIR Tool has not reached the hypothesised number of 100 images per second. Therefore the hypothesis has been proven to be false.**	

Table 5.1 Sub Question 4 and Hypothesis

SUB QUESTION 4: *How much is the processing of the images affected when the hardware is reduced?*	
HYPOTHESIS 4: **Reducing the RAM from 8GB to 4GB will reduce the HBIR throughput by 50%.**	
ARGUMENT FOR: **Reducing the RAM from 8GB to 4GB had little to no effect on the processing of the images. However if the RAM was reduced below the operating system requirements it is expected it would cause stability and processing issues.** **Furthermore, the hypothesis is only looking at the RAM and no other component of the computer. Testing showed that all three techniques were heavily dependent on the CPU for processing throughput.**	**ARGUMENT AGAINST:** Testing showed there was no differences in image processing when reducing the available RAM from 8GB to 4GB.
SUMMARY: **Testing showed that reducing RAM from 8GB to 4GB had no effect on the processing throughput. On the other hand, the CPU was strong factor. Therefore further testing is required before the sub question can be substantially answered with definitive results. However the hypothesis that reducing the RAM from 8GB to 4GB will reduce the HBIR throughput by 50% has been proven to be false.**	

Table 5.2 Sub Question 5 and Hypothesis

SUB QUESTION 5: *What effect does the size of the image have on processing?*	
HYPOTHESIS 5: **Larger images will take significantly longer to process when using the CBIR/CBII/CBIR techniques.**	
ARGUMENT FOR: **Figure 4.40 illustrates that the trend of all three image retrieval techniques increased as the batch of images increased.** **The CBII technique provided by Encase shows the value of y = 0.3669x +20.345 with a R2 = 0.949** **The CBIR technique provided by FTK shows y = 0.9605x + 1911.3 however with a lower R2 value of R2= 0.764** **Lastly the HBIR technique provided the variable of hash time of y = 0.5297x +2.1251 with a strong R2 value of R2 = 0.9993. (The insertion of hash values is a constant value for all batches therefore does not affect the trend).**	**ARGUMENT AGAINST:** There is a downward line on Figure 4.38 showing the processing time for the batch of 850KB images was actually faster than the processing time for the 750KB batch of images. Table 4.9 shows the time required to process the 50KB batch of images using the CBIR technique was 2023.5 seconds, however to process the 150KB batch required only 12.5 seconds (0.62%) longer, even though the amount of data was 300% larger. Table 4.9 also shows to process the 50KB batch of images required 2023.5 seconds and to process the batch of 950KB images required 3187.8 seconds. This shows the batch is 1900% larger, but there was only a 57.5% increase in

The extremely strong R2 values of the CBII and HBIR techniques show the data is constant and very linear with a significant upward trend. **Table 4.10 shows to process the 50KB batch of images using the CBII technique required 21 seconds to process and 450KB batch of images took 178 seconds. This shows when the size of the images was increased by 900% the process time was also increased significantly by 850%. This is a significant increase.** **Table 4.5 shows the process time to hash the 50KB batch was 22.89 seconds and 397.09 for the 750 KB batch of images. This shows when the size of the images was increased by 1500% the process time also showed significant increase of 1734%.**	processing.
SUMMARY: **The HBIR and CBII techniques show a significant processing increase as the images tested became larger. Furthermore, all three trend lines showed a positive increase from the smaller batch of images to the larger batch of images. However, Figure 4.38 shows the CBIR technique only showed a relatively small proportional increase between the different sized batches. To conclude, all three techniques did not show a significant increase. Therefore the hypothesis that all three techniques would take significantly longer to process larger images has been proven to be false.**	

5.2 DISCUSSION OF FINDINGS

The discussion of the findings is where the results of experiments in Section 4.4 will be discussed. This section will be structured in the following way. It will be begin by discussing the findings of the proposed HBIR experiment including the selection of the hashing algorithm and the database. It will be followed by explaining the findings for the CBIR and CBII experiment. It will then continue by comparing the three different techniques and how each technique would function in a real world environment. This will include how forensic investigators would use each of the tools to identify objectionable images. The last section will conclude the findings.

5.2.1 The HBIR Experiment

The HBIR experiment contains three main stages. The first stage is to identify an efficient hashing algorithm, the second stage is to identify the efficient database and the third stage is to investigate how reducing hardware would affect image processing

throughput. This section will first look at all the hashing algorithms and how they performed followed by the selection of the SHA-256 hashing algorithm.

The HBIR experiment included the testing of nine hashing algorithms. The literature reviews in chapter 2 identified that the hashing times should be proportional to the file size because they are directly calculated from the file's binary value. For example when two different files have the exact same file size, the hashing time will be identical. Therefore it is expected that all hashing algorithms would create a linear relationship with a proportional increase in processing time as the file sizes increased. Figures 4.19, Figure 4.21, Figure 4.23, Figure 4.25, Figure 4.27, Figure 4.29, Figure 4.31, Figure 4.33 and Figure 4.35 showed this assumption was correct. The figures also list the R^2 which shows the accuracy of the data was excellent. The R^2 values ranged between 0.9981 and 0.9995. SHA-256 was selected as the most efficient hashing algorithm, even thought it was the slowest to process all 10 batches of images. However the effectiveness of the hash value having a large bit size of 256, predetermined the rate of collisions and would be severely low compared to the other algorithms that use a smaller bit size. Furthermore it would support one of the underlying goals mentioned in Section 3.1 for creating a long term solution. The recorded times for processing using the SHA-256 algorithm ranged between 22.89 seconds for the 50KB batch of images to 498.56 seconds for the 950KB batch of images with a total time for processing all 10 batches of images of 2669.63 seconds. To demonstrate the longevity of the 256 bit hashing algorithm, if the HBIR Tool hashed 100,000 images for every person on earth the collision rate would be:

- Number of hashes requires 7.2×10^{15}
- Number of hashes available 1.1579209×10^{77}
- Result (red) is illustrating the percent of a collision is approximately 2×10^{-74}

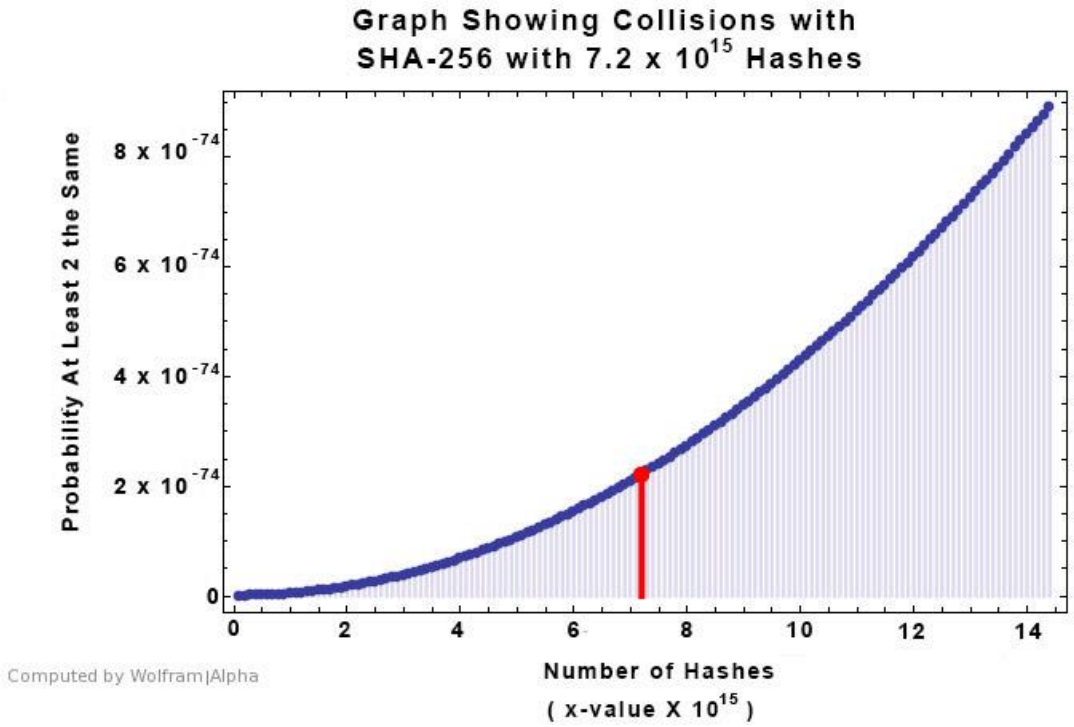


Figure 5.1 showing the chance of a collision using the SHA-256 hashing algorithm

The above example illustrates the likelihood of a hash collision would be 2 x 10 $^{-74}$ when there are 7.2 x 10^{15} hashes being generated.

The next stage is identifying the most effective and efficient database that could be used for the backend functionality of the HBIR Tool. The three databases that were selected for testing were MS SQL, PostgreSQL and MySQL. The testing in this area was straight forward and conclusive. Each of the databases was connected to the HBIR Tool and hash values were either inserted into the database or hashes were compared to the database. The fastest database to insert 250,000 hash values was PostgreSQL with a time of 6223 seconds, followed by MS SQL with 6347 seconds and MySQL with nearly double the time of PostgreSQL of 11780 seconds. In contrast, to compare 10,000 hash values to the database (does not include hash time) of 250,000 hashes was 446 seconds for PostgreSQL, followed by MS SQL with 557.2 seconds and MySQL with more than twice that of MS SQL with 1147 seconds. The testing was conclusive showing that PostgreSQL was the fastest in both areas of inserting and comparing hash values (Refer to table 4.7 and Figure 4.37).

The final stage involves a discussion on how reducing hardware specifications would affect image processing throughput. It was expected that reducing the RAM would change the image throughput. This would demonstrate how RAM was an important factor in the processing of the images. However when the RAM was reduced from 8GB to 4GB there was no change in the processing time. Resource monitoring showed the HBIR technique was more reliant on CPU resources instead of RAM.

5.2.2 The CBIR Experiment

The CBIR experiment involved two main stages. The first stage is to record the time FTK required for processing the 10 batches of images. The second stage is to record the processing throughput when the RAM is reduced from 8GB to 4GB.

The processing times for FTK to analyse the 10 batches of images ranged from 2023.5 to 3187.8 seconds (Table 4.9). The positive trend line of Figure 4.38 demonstrates that as the size of the batches of images increased, the time for processing also increased. It was expected that the recorded data would be proportionally similar to the HBIR. However the R^2 value was comparatively low with 0.764. In addition there were noticeable outliers from the trend line when testing the 850KB and 950KB batch of images. The 850KB batch of images was 250 seconds faster than what was expected and the 950 batch of images required an additional 250 seconds to complete processing. All other recorded data appears to be in close relation with the trend line.

The final stage involves a discussion on how reducing the hardware would affect image processing throughput. This would demonstrate how RAM was an important factor in the processing of the images. However when the RAM was reduced from 8GB to 4GB there was no change in the processing time. Resource monitoring showed the CBIR technique was more reliant on CPU resources instead of RAM.

5.2.3 The CBII Experiment

The CBII experiment involves two main stages. The first stage is to record the time Encase required for processing the 10 batches of images. The second stage is to

experiment by decreasing the RAM from 8GB to 4GB and recording how the decrease would affect processing throughput.

The processing times for Encase to analyse the 10 batches of images ranged from 21 seconds to 407 seconds (Refer to Table 4.10). Figure 4.38 illustrates there was a significant increase in processing time as the size of the images increased. It was expected that the data would be linear. Calculations showed the CBII technique had a R^2 value is 0.949. Overall the data collected was consistent and with the exception of minimal fluctuations between tests.

The final stage involves a discussion on how reducing the hardware would affect image processing throughput. This would demonstrate how RAM was an important factor in the processing of the images. However when the RAM was reduced from 8GB to 4GB there was no change in the processing time. Resource monitoring showed the CBII technique was more reliant on CPU resources instead of RAM.

5.2.4 Global Investigations

Currently there are two methods of identifying objectionable images. As the testing has shown the CBIR technique is extremely resource demanding. To have an efficient CBIR analysis system would be very expensive to purchase the required hardware and software and secondly maintaining the system would be time consuming. Forensic investigators using the CBIR technique globally and they have to limit the amount of processing because of the costs involved. In contrast testing has shown that the CBII is relatively cheap technique to analyse images on a global scale, however it is extremely inaccurate and would be of little benefit to an investigation. To identify an image the suspect would have needed to previously manually enter keywords into the metadata. The likelihood of a suspect doing this is extremely low. Therefore on a global scale, even though this technique is relatively inexpensive to implement, it is also extremely inaccurate. The proposed HBIR technique is a more balanced technique, by balancing cost with accuracy. It is less resource demanding than CBIR and therefore is cheaper to implement. Furthermore, it has significantly higher

accuracy than CBII. The HBIR technique could be used efficiently by a forensic investigator to identify objectionable images on a global scale.

5.3 CRITICAL REFLECTION ON TESTING PROCEDURES

The testing procedure was divided into four research phases to help simplify answering the sub questions. The sub questions then would answer the main research question. The discussion of the testing phases will begin by summarising the four phases and demonstrating how the testing phases were required to address and answer each of the research sub questions. It will also highlight the important findings of each phase.

Phase one included software and pilot testing where preliminary software tests and pilot tests were performed. Software testing was performed on the HBIR Tool to identify any software glitches. Pilot testing is where a hypothetical scenario was presented and each technique was tested to prove the feasibility of the experiment. This critical stage was important in identifying issues surrounding each of the image identification techniques so they could be resolved before the experiment stage. The pilot testing was performed on an 8GB USB drive containing 8000 files, 3412 of which were image files. The pilot test identified four main issues with the HBIR Tool (as discussed in chapter 4.1.2). The issues were variances in time when repeating the same test using the same constants, FastHash algorithm outputting hash values of '0' (zero), RAM not releasing images after processing and lastly a reduction in the RAM having no effect on processing. The solutions were applied to the HBIR Tool and further testing commenced until the testing was completed successfully. It was important that this stage was completed successfully and any changes were documented accordingly because once the experiment stage has begun there would not be enough time for retesting. Overall more time could have been spent on quality control before the software testing stage. Nonetheless the problems were relatively small and were easily resolved.

The HBIR Experiment (Phase 2) is where the most effective and efficient tool can be identified by using a combination hashing algorithms and databases. This stage comprises of three main stages. The first stage is to identify the most efficient

algorithm. The second stage is to identify the fastest database and the third stage is to identify how the HBIR Tool would process images using fewer resources by decreasing the amount of available RAM. During the testing of the hashing algorithms it became clear that the most effective hashing algorithms might not necessary be the fastest performing algorithm. Instead it could be a combination of the algorithms performance and the bit size. For example, even though using a CRC-32 bit hash value may perform the fastest, it would be ineffective because there will be potentially hundreds of trillions of unique hash values generated. The system would be deemed ineffective because the collision rate would be so high. Therefore it was identified that the SHA-256 bit hashing algorithm is the preferred option because of its combination of performance and its large bit size. Ideally, testing should have only included hashing algorithms that support 128+ bit sizes.

Identifying the fastest database consisted of testing the speed the database can store hash files and the speed the database can execute the compare queries. The primary goal is to identify the fastest database for comparing hash values because this function will likely to be used the majority of the time. It was identified after testing was completed on all three databases that PostgreSQL was the best performing database for both functions. The third stage of phase 2 is to record how the HBIR tool would process images when the RAM size was reduced. Testing showed reducing the RAM from 8GB to 4GB has no effect on processing. In addition, it was identified that the CPU was the susceptible component that should have been reduced for testing. Overall the procedure for the HBIR experiment was very thorough and well executed.

The CBIR experiment (Phase 3) was where the CBIR technique was tested using Forensic Tool Kit software. The batches of 10,000 images were inserted into the FTK software with the Explicit Image Detection function enabled. Five images were used as objectionable (woman in bikinis) and the time to identify the images was recorded. The FTK software is a well-defined tool and therefore as expected, no problems developed. Similar to the HBIR Tool, when the RAM was reduced from 8GB to 4GB there was little to no change in processing throughput. Overall phase 3

of the experiment was thorough and no negative aspects could be identified with the process.

The CBII experiment (Phase 4) is where the CBII technique is tested using the Encase software. The batches of 10,000 images were inserted in to Encase with a 2000 keyword list. The search was executed where the parameters of the search was set to identify the keyword "bad". The processing times were recorded and analysed in Chapter 4. The main problem encountered in this stage was determining the correct number of keywords for testing. It was estimated that in a real world situation approximately 2000 keywords would be used. However increasing or decreasing the number of keywords slightly had very little effect on the processing time.

To conclude all three techniques relied on different data for image retrieval. It was interesting to see how slow the CBIR technique is when being compared to CBII and HBIR. Overall, all three techniques performed well and identified all the images. To give each image retrieval technique a fair trial each of the techniques were compared to a real world scenario. The HBIR tool had 250,000 hash files to compare with, FTK used the fastest EID settings and Encase also was set to the fastest search settings.

5.4 STRENGTHS AND WEAKNESSES OF THE RESEARCH

The strengths and weaknesses of the research in this section provide critical evaluation of research as presented in the previous chapters. This section will first begin by reflecting on the positive aspects of the research followed by the negative aspects and a conclusion will be drawn. The first section of the positive reflection will focus on the experimental procedure, followed by the methodology and then the data generation.

An important aspect to the research was that the experimental procedure for all three techniques needed to be unbiased. Each of the techniques required the same opportunity to perform most efficiently. It was expected in chapter 3 that the HBIR Tool would only require one test for each scenario. However the software testing reported in chapter 4 illustrated the processing times were inconsistent when the same test was rerun multiple times. It was also identified in chapter 4 the inconsistent times

were due to numerous background processes running at random times which removed computational resources from the HBIR Tool. Due to the flexibility of the Design Science Methodology and the requirement to be rigorously tested the artefact I was able to resolve the issue by using a fresh installation of Windows 7 with only the minimum installed software. This resulted in an excellent resolution however there were still small differences in the times. To further decrease the unexpected inconstancies the same test was run multiple times and an average was calculated as the final processing time. By implementing the two solutions it proved to be an excellent overall result to the problem.

Another positive aspect of the research was to produce an unbiased experiment by using constant hardware and software throughout each phase of the experiment. It was expected in chapter 3 that the specified hardware would be suitable for all three techniques. It was mentioned that in the ideal situation server hardware would be used. Using server hardware could have produced results that are a closer representation of real world environment. Furthermore, it would enable more images to be processed in the same amount of time creating data of higher accuracy. However, during the testing in chapter 4 it became clear that using the desktop environment was very suitable as it was a comparison between the techniques, not in the hardware.

Another positive aspect of the research was that extensive testing was performed on all three image retrieval techniques to identify the minimum configuration requirements. By investigating the minimum requirements of each technique it helped to support the goal of creating a fair benchmarking comparison. During the very first testing of the FTK and Encase software different configuration settings were tested to identify what were the minimal settings required to identify the objectionable images (refer to Figure 4.14, Figure 4.16, and Figure 4.17). In addition, the HBIR tool used multiple hashing algorithms so the most efficient and effective hashing algorithm could be identified. The experiment testing in chapter 4 showed that every test identified the correct number of objectionable images. Rigorous testing of the three techniques was time consuming however it was imperative to the research that it was completed thoroughly and with the highest level of accuracy.

The second aspect of this section will analyse the positive aspects in relation to the methodology. As earlier stated it was expected that each test with the HBIR Tool was only required to be run once per test. However due to inconsistences the same test was required to be run multiple times.

One of the underlying requirements of the Design Science Methodology is that the artefact is rigorously tested and evaluated until a robust artefact can be identified (as demonstrated in Figure 3.2). Therefore the methodology not only enabled but required the iteration between design, demonstration and evaluation multiple times until a truly high quality artefact could be produced. The advantage of the Design Science Methodology is having this flexibility. It enables multiple different versions of the software to be created, demonstrated and evaluated. This did not just help to identify software issues, but also to identify and implement additional functionality such as batch processing so the tool could be run the same test multiple times without human intervention. This freed up human resources and enabled better time management where the newly freed up resources were put to use in other areas of the research. It also enabled the continual addition of new hashing algorithms to the HBIR Tool for experimenting with in chapter 4.

The Design Science Methodology also required peer review for evaluation (refer to Figure 3.2 and Stage 4 in the Individual Stage Break down). The peer review helped to identify software bugs and optimised programming to enable faster processing. An example is when similar times were being recorded during the testing of the MD4 hashing algorithm. It was expected that the tool would require more time for the larger batches of images than the smaller batches. After the peer review it was identified the time being outputted was incorrect because a variable was not being over written due to a syntax error. Once the fault was identified it was easily fixed and the correct times were being outputted. In addition, the peer reviews also helped by discussing different logic techniques for the multitude of functions in the HBIR Tool. For example to identify which logic was faster, either the image was hashed while sending the hash to the database, or all the images were hashed first and then send all the hashes were sent to the database at once. Testing showed that it was faster to send the hash value to the database as they were generated instead of waiting

until all the hash values were generated, then sending them all to the database at once. Unlike other methodologies such as the Traditional Waterfall Methodology, the Design Science Methodology enables iterations between each of the steps. The iteration was imperative to producing a high quality tool. In addition, I have personally used the Design Science Methodology in previous projects which made me feel more comfortable of its strengths and weaknesses.

The third section to reflect on was the data generation stage. As earlier discussed in chapter 4.1.1 there were numerous unforeseen challenges with generating the 10 batches of 10,000 images. It was expected and documented in chapter 3.8.1 that 10 different sized batches of 10,000 images would be used to as evidence files. To test the feasibility of this expectation one batch of images containing 10,000 images was inserted into each tool. The processing time to analyse the images was then recorded. Testing showed there was a good balance between the amount of data being recorded and the time it took to record the data. I believe using the batch of 10,000 images was an optimal sample size which demonstrated the feasibility, testability and comparability of the three different techniques. Another positive aspect in relation to the generated data was the size of each individual image. Sub Question 5 was to investigate what the effect of different sized images had on the processing time. In chapter 3, Hypothesis 5 states "larger images will take significantly longer to process when using the CBIR/CBII/CBIR techniques". By using the different sized images of 50KB to 950KB it was shown a clear trend for all of the techniques. Even though digital cameras can produce images that are far in excess of 950KB, using a range of smaller images created an optimal trend line where larger images could be closely approximated. The last positive aspect on the research that will be discussed was bench marking the HBIR Tool to other techniques. At the beginning of the research it was proposed to only research new techniques to identify objectionable images. However as the Design Science Methodology underlying goal is to create a robust artefact, it was suggested this robustness could be achieved by benchmarking or comparing the proposed HBIR Tool to CBIR and CBII tools. This decision was an excellent assessment as it helped to measure the effectiveness and efficiency of the HBIR Tool. It also helped to measure the goal that was discussed in

Issues and Gaps of Current Research (Section 3.1) of finding a new balanced technique that was faster than CBIR and more accurate the CBII (refer to Figure 4.40). Overall there were many positive aspects to the research. The next section will discuss the negative aspects of the research.

The first negative aspect will focus on the experimental process, followed by the data generation and then Sub Question 4 (Table 3.2). In the early stages a combination of both cryptographic and non-cryptographic hashing algorithms were randomly selected. It soon became apparent in Section 4.4.1.3 that none of the non-cryptographic hashing algorithms would be effective because they only supported a small bit size. In Section 2.6 it discusses that the bit size is an imperative factor to consider when producing an effective tool. However, when selecting the hashing algorithms this factor was overlooked. Even though the hashing algorithms with the smaller bit size were faster at generating hash values, in a real world environment there would be too many collisions causing the tool to be ineffective. Therefore the data collected by using the smaller hashing algorithms created was interesting to analyse, it would however, not greatly assist in the overall objective of creating a robust artefact. In hindsight, testing should have only included hashing algorithms of which support hash values of 128bit or larger as this would have created an acceptable number of possible hash values.

As earlier discussed, to create an accurate data collection each test was run multiple times and an average was calculated. When testing the CBIR technique with the batch containing the 850KB group of images it was faster than processing the 750KB batch. However the trend line clearly shows a steady increase as the image sizes get larger. Ideally, this should have been investigated and tested further. The next aspect to review is the testing of the databases. It was expected at the beginning of the research that the database would have a significant role in the image processing times. Unfortunately only three relational databases were tested. The testing of the databases was minimalistic and testing should have included different types of databases such as Column-Orientated Database Management Systems (CO-DBMS).

The next aspect is that only one programming language was used to create the tool. Java was used as the programming language for the development of the artefact.

It was expected that Java could perform all the required functionality for the HBIR Tool. However, the selection of the programming language will affect the processing time significantly. Therefore by developing the artefact with a different language such as python, it may have created a faster processing time. Due to time restraints and expertise, other programming languages were not investigated. Ideally, multiple programming languages would have been investigated where three or more tools would be created and compared. Testing with multiple programming languages would have potentially created an even faster tool for comparison.

The next negative aspect of the research was FTK and Encase were solely used and other software tested. Even though Encase and Forensic Tool Kit are leading software in the forensic industry, ideally three or more different software tools demonstrating each technique should have been used for testing. The main reason why only two pieces of software were tested was based on three reasons: difficulties in gaining access to the software, financial restrictions of being able to purchase the software and time to learn the software. Forensic software is highly specialised and targets a niche market and in some situations will only be sold to government agencies, financial firms or law firms. Therefore gaining access to the software is difficult. To purchase the software is very expensive depending on the manufacture and number of licenses required. Lastly, forensic tools can be more complicated to learn than standard software, therefore requiring more time to learn and then test the software. Even though the two leading forensic software used is globally recognised, it has been noted that using open source forensic software would have brought additional data for analysis which would have strengthened the research.

The next negative aspect of the research was that FTK time combined the acquisition time and the analysis time together. This would have created a delay to the processing of the explicit detection function which could not easily be calculated. This was one of the main issues that were encountered during the research. Encase acquired the evidence first then the time started recording when Encase started analysing the evidence. Similar to Encase, the HBIR Tool had the images inserted into an ISO file, and the time started recording when the HBIR started analysing the

evidence. However, the time started recording before FTK acquired the evidence. Ideally all techniques should have used the identical file format for the evidence, not just the same batch of images. In hindsight, the HBIR should have been developed to enable the insertion of E01 files. Therefore all techniques would be using the exact same evidence file and not just contents. It is expected the results would have been similar even if testing was completed using an E01 file.

The next aspect to review was how the HBIR Tool was not utilising hyper threading for processing of the images, unlike the Encase and the FTK software. Testing showed the CPU was a critical component for the processing of all three image techniques. The reason hyper threading was not enabled for the HBIR Tool is because hyper threading creates different hash values with the Murmurhash2 hashing algorithm. Research showed this was due to the block processing of the hash because the seed value was changing (Trmač, 2013) resulting in different hash values. A goal that was discussed in Section 3.1 was to create a long term solution that could be used globally on all hardware. It was then decided that hyper threading would be disabled.
The next aspect was that the research should have included analysis of the CPU resources instead of RAM. It was expected during chapter 2 that RAM would have an impact on processing. However after testing it showed the CPU was the most critical component. When the RAM was reduced there was no change to the processing throughput. This test did prove that the hardware was an important factor to the processing. However by investigating the wrong component, it would have generated supplementary data. In Section 6.3 investigating how the CPU impacts processing has been documented for a future research recommendation.

One aspect that was not investigated was the method of data collection. It was expected that recording the time to process the images was sufficient. However only during the critical reflection stage it was realised that recording the time as well as the amount of resources used during the processing time would have been beneficial to the research. This could have been completed using a resource monitor. It could have created a completely different and exciting aspect to the research if both the processing time and the amount of resources were collected.

To conclude there were many positive and negative aspects of the research. If the research was repeated certain negative aspects will be implemented in to the research to create a stronger set of raw data. Overall the experimental process was fair to all three techniques so the highest quality of data could be produced.

5.5 REVIEW OF BENCH MARKING SOFTWARE

Software tools were used as benchmarking software for measuring and comparing the performance between the three forensic tools. The software used was Forensic Tool Kit using the Explicit Image Detection functionality (CBIR) and Guidance software's Encase (CBII).

Forensic Tool Kit is a widely used software suite supporting multiple different tools that can assist forensic investigators. FTK is used on a global scale by law enforcement, federal governments, state governments and corporations around the world. FTK is also widely accepted in courts as a suitable tool for analysis of digital evidence. The Explicit Image Detection (EID) functionality is a relatively new function in the FTK suite. Testing showed that the EID is a well-developed function that gives the investigator the ability to adjust the accuracy of the results. The EID function provides an accurate method of identification based on skin tone, textures, and objects that appear in the image. Each image is given a ranking on how objectionable the image is based on the visual contents of the image. The investigator has the ability to then view the different ranked images to distinguish if the image is objectionable or non-objectionable. To test the accuracy of the software I inserted an RGB image (colour image) of a woman in a bikini to see how the software would identify the image. FTK gave the score of 63. I then converted the image to black and white and repeated the test. The result was 47. This proved that FTK was not solely dependent on the colour aspects of an image but would detect images of all colours. Overall testing showed the EID function was accurate, the technique had the ability to be semi-automated but it was very resource demanding. It also showed the EID function was very slow at analysing images when compared to the CBIR and HBIR techniques. Overall, the FTK software was used to demonstrate a comparison between the CBIR and HBIR technique. FTK brought a strong component to the

benchmarking stage of the research that was reinforced by level of prestige that FTK holds.

Encase, developed by Guidance Software, is globally used, equally popular, forensic software available to governments, law enforcement and corporations. Similar to FTK, it is also widely accepted in courts as an appropriate tool for forensic analysis in digital forensics. Encase was selected because it is a market leading software and is well known for its thorough and efficient analysis of data. Encase was used to search through the 10 batches of evidence to identify the key word "bad". Testing showed it was 100% accurate every time and it was extremely efficient at processing the images. Overall Encase performed extremely well, it was easy to use and presented the investigator with numerous searching parameters with a well-developed search engine that is highly efficient. It was a correct decision to use both FTK and Encase as bench marking software for the HBIR Tool.

5.6 CONCEPTUALISING AND THEORY BUILDING

Section 5.6 will now introduce conceptualising and theory building. This section will begin with introducing the three concepts that were used in this research. It will then discuss how the methodology enabled and restricted the conceptual development in the research.

The theory behind the hash based image retrieval tool was developed by understanding the overall objective of the research as well as concepts derived from three industries. The first industry is the police enforcement and digital finger printing. The second is the legal industry and time related difficulties in adapting to change. The third concept was derived from the research paper "Database architecture for content-based image retrieval" (Kato, 1992) which used mapping to identify trademarks for copyright infringement.

It is well known that police enforcement have been recording finger prints of suspects for identification purposes. The recording of the finger aids police enforcement identifying a repeat offender. The benefits of this system is that finger print can be cheaply recorded, it is a smaller and unique identifier of the person that can hold up in court, it can be easily recorded, stored and searched for by police

enforcement. The hash based system works precisely the same. It extracts a unique identifier from the image that can be easily stored and compared to known objectionable images. In addition, the finger print system is so effective that police are using the same technique by recording a suspect's tattoos.

The second concept is provided by the law industry and how the writing of law is delayed by politics. As discussed in Section 2.11 and Section 3.1 there appears to be a disconnect between the speed that criminals are finding new ways to perform criminal acts and the amount of time it takes for the law to identify the act as illegal. Therefore the HBIR technique has removed the complications by sharing only digital representations of the images and not the images themselves. Therefore the sharing potential on a global scale can be easily accomplished.

The third concept is a consequence of the research paper "Database architecture for content-based image retrieval" (Kato, 1992). In the research paper Kato describes a CBIR technique of mapping an image to an 8 x 8 pixel grid and calculating a shape measured from each image calculating the pixel frequency. This concept will be further discussed in Chapter 6 future research, inspired this research to investigate using the same concept by mapping the image of a quadrants and hashing the binary value of that quadrant. Thus not just creating one hash per image but creating multiple hashes per single image. Therefore it could potentially create a solution to the problem of when a suspect edits the image by 1 pixel and the image not being identified. Instead the Forensic Investigator will be presented a percent of how many hashes in the image match to the number of hashes which are in the row in the database.

Unfortunately the Design Science Methodology structure does not include a phase where it directs the researcher to search externally for potential solutions to the problem. Instead it starts with a solution, then it iterates until the solution is robust. Researching externally has identified an important solution to a problem that would not have been found if external research was not completed. Furthermore it also helps to support the goal of the Design Science Methodology by helping to create an innovative idea, practice, or technical product.

Overall three concepts from other industries were adapted to create a theory that was developed, thoroughly tested, and proven to be a strong technique for identifying objectionable images on a global scale. It has also proved by researching in to other technologies and adapting their theories and concepts an image retrieval system has been developed and rigorously tested.

5.7 CONCLUSION

Chapter 5 began by discuss the responses to the research question, sub questions and hypotheses. It was identified that the HBIR technique is an excellent technique for identification of objectionable images. It is fast at identification, uses efficient algorithms, is accurate with minimal false positives and is feasible solution for the foreseeable future. A discussion on findings (Section 5.2) followed where the results of each research phase in chapter 4.4 was discussed. The next section critically evaluated the testing procedure and Section 5.4 explained the weaknesses and strengths of the research. Section 5.5 reviewed the benchmarking software of FTK and Encase and finally Section 5.6 discussed conceptualising and theory building.
Chapter 6 will now complete the thesis project and present a summary of the research conducted and the significant results that have been found. It will continue by reviewing the limitations that restricted the research and will finally conclude with a discussion on the future research for the HBIR technique.

Chapter Six

CONCLUSION

6.0 INTRODUCTION

Chapter 6 will now present the final chapter of the thesis, the conclusion. Chapter 5 began by reintroducing the research questions and constructing a response to the main question, sub question and hypotheses. Chapter 5 continued with Section 5.2 where the image retrieval techniques were reviewed, including a short discussion on how well the technique would perform in a real world environment. The testing procedure was reviewed in Section 5.3, followed by the strengths and weaknesses of the Research (Section 5.4). Section 5.5 reviewed the benchmarking software, FTK and Encase, and discussed the performance of the tools. The last section of chapter 5 was conceptualising and theory building.

Chapter 6 will begin by concluding each of the chapters in the thesis followed by the research limitations (Section 6.2). Chapter 6 will be finalised by describing future research directions (Section 6.3).

6.1 FINAL CONCLUSION

A new image identification and retrieval technique was proposed to discover a balanced technique for identifying objectionable imagery that can assist police enforcement and forensic investigators worldwide.The globalisation and the development of legal anonymity software have provided cyber criminals with a new virtual world where images can be traded on a global scale with complete anonymity. These two challenges have made all cybercrimes an immensely complicated and expensive crime to investigate and prosecute.

Chapter 2 reviewed academic literature relating to research and developments of current techniques that are currently used to analyse and detect objectionable images. Chapter 2 also included a thorough discussion on hashing algorithms and

how they have been adapted in other technologies to assist and resolve technical challenges. Furthermore, chapter 2 encompassed how CBIR and CBII are being used to identify and retrieve images including the advantages and disadvantages of each technique. Chapter 2 was concluded by investigating the New Zealand law that protects children from exploitation.

Following chapter 2, chapter 3 begins by summarising the issues and gaps in current research. It then continues where a main question, five sub questions, and their corresponding hypotheses are constructed. The research methodology was discussed in Section 3.3, where the Design Science Methodology was selected as the framework to control and direct the undertaking of the research. In Section 3.4 a research design for the proposed tool was analysed including a comparison between architectures of CBIR, CBII and the proposed HBIR techniques. The four research phases were introduced in Section 3.4 to Section 3.7 where a procedure was constructed for the testing of the three image retrieval techniques. Section 3.8 discussed the data requirements of the research, where the particulars of the data generation, data collection and data presentation were constructed and documented. Chapter 3 was concluded with an assessment of the expected outcomes and limitations of the research.

Chapter 4 reported on the findings of the proposed four research phases. The findings were analysed and sequentially presented. The four research phases included, software and pilot testing, hash based image retrieval testing, content based image retrieval testing, and concept based image indexing testing. Software and pilot testing was undertaken in order to identify technical issues with the tools and secondly to ascertain the feasibility of the experimental stages (phases 2, 3 and 4) of the research. The software testing and pilot testing identified numerous technical challenges that required alteration to the HBIR Tool. The objective of phase one was to identify and resolve any issues with the three techniques before the experiment stage. This was a critical phase because if a significant issue arose during the experiment stage, it could have dire consequences for the entire research. All technical concerns were resolved once the pilot testing on the all three image techniques were completed successfully.

Phase 2 was the beginning of the experiment stage with testing of the proposed hash based image retrieval tool. Phase 2 consisted of thorough testing of the "insert" and "compare" functions of the HBIR Tool by experimenting with different combinations of hashing algorithms and database technologies. It was recognised during the experiment that approximately half the algorithms would not actually be suitable for the final tool due to their short bit length. Nevertheless, testing revealed that the Secure Hashing Algorithm with a 256 bit hash (SHA-256) was the most suitable choice for the final tool. It was also revealed from testing multiple different database technologies that the PostgreSQL performed the strongest by outpacing Microsoft SQL and MySQL.

Phase 3 was the second stage of the experiment and was where testing began on the CBIR technique by using the Explicit Image Detection tool developed by Forensic Tool Kit. Phase 3 entailed analysing 10 batches of 10,000 images of a specific size and recording the processing times. The Forensic Tool Kit software was configured to the minimal requirements to reduce delay and maximise image throughput. The times to process each of the batches of images was recorded and analysed accordingly. In every test the Forensic Tool Kit software identified all the objectionable images accurately.

Phase 4 was the third and final stage of the experiment and was where testing was performed on the CBII technique by using the Encase software provided by Guidance Software. Phase 4 involved analysing the same 10 batches of 10,000 images and recording the processing times. To create a fair testing environment Encase was also configured with the minimal requirements to decrease delays and to maximise image throughput. The processing times of each batch of images was record and analysed accordingly. Once the test completed, the accuracy of the tool was checked to confirm that the correct number of images was detected. Encase detected the correct amount of images in every test.

To summarize, each of the testing phases was completed to assist in answering the main question and hypothesis. Therefore, to answer the main question, a hash based technique of image retrieval is a technique that can be applied to identify objectionable images in a smaller time frame. The findings also revealed that

the proposed hash based technique provides a balanced approach to identifying objectionable images and it is more efficient than content based techniques and has higher accuracy than concept based image indexing. Due to the hash based technique's balanced approach it has a strong potential to function as a global detection tool to support police enforcement and forensic investigators in identifying objectionable images. However, there are still a number of potential issues regarding a hash based solution. The first issue is determining the level of effectiveness that technique has in a cloud based environment and identifying a solution to images that are slightly different and therefore are not detected by the hash based system. Recommendations include investigating the number of objectionable images being stored in cloud environments and also testing using the exact same evidence file for all three tools for a more definitive evaluation.

6.2 LIMITATIONS

As acknowledged earlier in the research methodology (Section 3.9) there was going to be limitations to the research. This section will now begin by summarising the issues, followed by discussing any new identified issues that were realised during the research and lastly, how the issues affected the study.

In Section 3.9, three issues were identified that may have a negative effect on the undertaking of the research. The issues were time restraints, financial constraints and legal boundaries.

As Section 3.9 acknowledged that there was a significant workload that was required within the 12 month period of time. Due to the relatively short time frame to complete the research it resulted in limiting the amount of forensic software that could be learnt and used for a comparison and benchmarking the HBIR Tool. Only one piece of software was used to measure the effectiveness of the CBIR and CBII techniques. Ideally, multiple proprietary and open source software should have been used and tested extensively to create a supplementary level of accuracy for comparison.

The next limitation was financial restraints on the research. The financial constraints affected the amount of hardware that could be used in the lab

environment. By using limited hardware it determined that a smaller sample size was necessary for the data generation stage, which then rolled on to affect the accuracy of the data. Also a sample size of 10 groups containing 10,000 was used as unknown images that required analysis. Preferably 15-20 groups of 10,000 images could have been used as it would have provided a more accurate average for the CBIR tool.

The third limitation was the legal boundaries in researching data. There was a high level of caution when researching data on the topic of objectionable images and current techniques for identification. As earlier stated, Google and Yahoo have only recently blocked certain websites from being displayed in their search results. Therefore the limitation was that the research into current techniques had to be minimal as to not overstep the legal boundary accidentally. The legal boundary probably stopped important research from being discovered.

A limitation that was later identified in the research was the lack of publically available research into how to reduce objectionable images using technology. Furthermore the prior research that did exist, only related to content based image retrieval techniques such as using colour, texture and objects for identification. It is assumed the lack of public material available is due the sensitivity of the topic and publicising techniques could complicate the task of investigating even further.

The second limitation that was identified through the research was gaining access to a cloud provider to detect objectionable images on a global scale. It was expected that this limitation would exist for a multitude of reasons from privacy, legal complications and risk management. Furthermore, New Zealand companies are relatively small and the request could be expensive in down time and demanding. Ideally, any data that a cloud provider could provide would be beneficial to the research however the financial benefit for the cloud provider would be insignificant.

The final limitation is the researcher's epistemology. The epistemology is a branch of philosophy concerning how the solution to the problem was selected and how the background knowledge of the researcher affected the decisions being made. Unlike group research where there are multiple epistemologies, this research had two. Therefore, even though the HBIR tool was of the highest standard, the decisions to get to that standard only used two epistemologies, therefore limiting the level.

Ideally, in research and development multiple researchers would be involved to gain and share the maximum amount of knowledge.

To conclude, in every research and development project there will always be limitations. However, the goal should be to try and minimise the limitations so the research benefits can be maximised.

6.3 FUTURE RESEARCH DIRECTIONS

The research conducted throughout the project has provided additional understanding to the chosen topic area of identifying objectionable images and image retrieval. This project has highlighted that there are still a number of aspects that require further research to take the HBIR technique to the next level. There are four aspects of research that require future exploration. They are to experiment with image mapping and hash values, using multiple programming languages, multiple CBIR and CBII forensic software for better comparison and experimentation with different methods of data collection.

A major area that demands future research is to develop the HBIR Tool so it can perform image mapping of an image. Image mapping is a technique that slices an image into individual quadrants. Hashing can then be performed on each of the quadrants and stored in a database. Therefore, instead of having a single column of hash values that represent one single objectionable image, there is now significantly more data to analyse in the comparison function. Preliminary testing has showed (Figure 4.26) that the time to process 50KB of data in a small image is approximately the amount of time that is required to process 50KB of data in a larger image. Therefore initial analysis shows that by slicing the image into quadrants, the amount of time to process an image to generate 9 hashes (one for each quadrant) will be very similar in processing to and image to generate 1 hash value for the entire image. In addition, it will create significantly more accuracy for the forensic investigator. For example, image_1 has been visually analysed to determine it is an objectionable image. The image has then been inserted into the HBIR Tool which processes the image and stores a single hash value. A criminal has now altered image_1 by placing an additional black pixel in the bottom right hand corner and saved the image as

Image_2. The criminal's computer has been investigated and the HBIR Tool scans all images on the computer and compares them to the hashes that are stored in the database. Image_2 is not detected because the additional pixel has changed the binary value of the image, subsequently changing the hash value, consequently not matching the image_1.

In contrast if the HBIR Tool used the quadrant technique for analysis it would have been different. For example, The image has been inserted into the New HBIR Tool which processes the image. Instead of creating one hash value the tool gets 81 hash values and stores them in the database. Therefore if there are 81 (9 x 9) quadrants, the single pixel will only change the hash value of one quadrant leaving the other 80 quadrants untouched. Subsequently, when Image_2 is being analysed, instead of the tool overlooking the image, it will present the forensic investigator with a result of 80/81 quadrants match. The forensic investigator now has the ability to visually check the image to confirm identification.

The second area of future research is that multiple programming languages need to be tested to identify if one language can perform the functions faster than the other languages. The programming languages should include a variety of types of languages such as scripting, object orientated and iterative languages.

The third area where future research could be undertaken is investigating multiple CBIR and CBII tools. There is a magnitude of open source software available that can retrieve images based on visual characteristics or by metadata. In addition, indexing software could be used to benchmark the selected CBII software. Researching in this area would further demonstrate the effectiveness of the HBIR Tool.

The final area of future research that is required is to investigate using a different method of data collection. It is well documented in this thesis that the processing time was recorded as the instrument of data collection. However, one of the challenges of the research was to be able to record data that was constant and repeatable. Investigating into other data collection instruments such as using a resource monitor to collect data, could provide a unique insight into the research and computation requirements.

To conclude there are numerous areas that this research could be further developed. As the tools for cyber criminals are advancing, it is critical for society that there is serious research and development for all cybercrimes. At the beginning the research there was two available techniques for Police enforcement to identify and retrieve objectionable images. This research how now presented a third, more balanced technique which police enforcement can use to help reduce objectionable images.

References

A guide to legal language (2013). A guide to legal language. Retrieved 14th June, 2013, from http://www.justice.govt.nz/services/access-to-justice/civics-education-1/a-guide-to-legal-language

Apache Server Foundation, (2012). HTTP Server Project. Retrieved 26th April 2013, from http://httpd.apache.org/download.cgi.

ArcSoft (2013). What is Color Space. Retrieved 9th September, 2014, from http://www.arcsoft.com/topics/photostudio-darkroom/what-is-color-space.html

Australian Institute of Criminology (2011). Definitions and general information. Retrieved 8th January 2014, from http://www.aic.gov.au/crime_types/cybercrime/definitions.html

Azzam, I. A., Leung, C. H. C., Horwood, J. F. (2004). *Implicit concept-based image indexing and retrieval*. Paper presented at the Multimedia Modelling Conference, 5-7 January, Brisbane, Australia.

Bala, M. R., Aakash, K., Anand, S., Chandra, S. A. (2010). *Intelligent Approach to Block Objectionable Images in Websites.* Paper presented at the 2010 International Conference on Advances in Recent Technologies in Communication and Computing. 16-17 October, Kottayam, India.

Banzal, S. (2007). *Data and Computer Network Communication*. Firewall Media.

Barnat, J., Brim, L., Edelkamp, S., Sulewski, D., Šimeček, P. (2007). *Formal Methods for Industrial Critical System.* Germany: Springer-Verlag.

Bembridge, C. (2013). Former police officer Lynton Moore jailed for possessing thousands of child exploitation images. Retrieved 17th January, 2014, from http://www.abc.net.au/news/2013-11-22/lynton-moore-jailed/5110942

Bonneau, J. (2012). The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords, *2012 IEEE Symposium on Security and Privacy (SP)*, 20-23 May 2012, California.

Carville, O. (2013). Child sex acts were on computer. Retrieved 18th April, 2013, from http://www.stuff.co.nz/the-press/news/8192821/Child-sex-acts-were-on-computer

Child porn hoarder gets home detention (2009). Retrieved 16th March, 2013, from http://www.stuff.co.nz/nelson-mail/news/1403540/Child-porn-hoarder-gets-home-detention

Clarkson, D (2013). Journalist admits child abuse charges. Retrieved 16th March, 2013, from http://www.stuff.co.nz/national/crime/8274987/Journalist-admits-child-abuse-charges

Convention on the Rights of the Child, Global, 2 September 1990, United Nations, *Treaty Series*, No. 27531.

Crawford, A. (2013). Computer-generated 'Sweetie' catches online predators. Retrieved 10th December, 2013, from http://www.bbc.com/news/uk-24818769

Drimbarean, A. F., Corcoran, P. M., Cuic, M., Buzuloiu, V. (2001). *Image processing techniques to detect and filter objectionable images based on skin tone and shape recognition.* Paper presented at the International Conference on Consumer Electronics, 19-21 June, California.

Dunlop, J., & Smith, D. G. (1994). *Telecommunications engineering*. CRC Press.

Eakins, J. P., Boardman, J. M., Graham, M. E. (1998). Similarity retrieval of trademark images. *MultiMedia, IEEE, 5*(2), 53-63.

Fadzli, S. A., Setchi, R. (2012). Concept-based indexing of annotated images using semantic DNA. *Engineering Applications of Artificial Intelligence, 25*(8), 1644-1655.

Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B., Yanker, P. (1995). Query by image and video content: The QBIC system. *Computer*, *28*(9), 23-32.

Funwallz (2014). Haunted Forest Scary Road HD Wallpaper [Photograph]. Retrieved 16th March, 2013, from http://www.funwallz.com/haunted-forest-scary-road-hd-wallpaper-88656.html

Gersting, J. L. (2007). *Mathematical structures for computer science*: Macmillan.

Hafner, J., Lee, D., Petkovic, D., Steele, D., Yanker, P. (1995). Query by image and video content: the QBIC system. *Computer*, *28*(9), 23-32.

Hevner, A. R., March, S. T., Park, J., Ram, S. (2004). Design science in information systems research. *MIS quarterly*, *28*(1), 75-105

Hicks, W. (1968) Video Recording in Police Identification. *The Journal of Criminal Law, Criminology and Police Science.* 59 (2), 295.

Horbury, A. (2013). Cybercrime Takes Its Toll. Retrieved 6th January 2014, from http://www.symantec.com/connect/blogs/cybercrime-takes-its-toll

Humphreys, L. (2011). Technician caught with child porn on computer. Retrieved 18th April, 2013, from http://www.stuff.co.nz/taranaki-daily-news/news/5351090/Technician-caught-with-child-porn-on-computer

Hyesook, L., Ji-Hyun, S., Yeo-Jin, J. (2003). High speed IP address lookup architecture using hashing. *Communications Letters, IEEE, 7*(10), 502-504.

Jung-Eun, L., Rong, J., Jain, A. K., Tong, W. (2012). Image Retrieval in Forensics: Tattoo Image Database Application. *MultiMedia, IEEE, 19*(1), 40-49.

Kato, T. (1992). Database architecture for content-based image retrieval. *SPIE/IS&T*. Paper first presented at Symposium on Electronic Imaging: Science and Technology, 9 February, California.

Kidd, R. (2012). Home detention for child porn offences. Retrieved 16th May, 2013, from http://www.stuff.co.nz/national/crime/6440189/Home-detention-for-child-porn-offences

Kidd, R. (2012). Child-sex offender sentenced. Retrieved 18th April, 2013, from http://www.stuff.co.nz/waikato-times/news/7175294/Child-sex-offender-sentenced

Koopman, P., Chakravarty, T. (2004). *Cyclic redundancy code (CRC) polynomial selection for embedded networks.* Paper presented at the International Conference on Dependable Systems and Networks, 28 June – 1st July, Florence, Italy.

Leader, B. (2013). Former Continental man gets five years for receiving child pornography. Retrieved 14th January 2014, from

putnamsentinel.com/Content/News/Local-News/Article/Former-Continental-man-gets-five-years-for-receiving-child-pornography/1/1/17401

Lehmann, T. M., Wein, B. B., Dahmen, J., Bredno, J., Vogelsang, F., Kohnen, M. (1999). *Content-based image retrieval in medical applications: a novel multistep approach*. Paper presented at the Storage and Retrieval for Media Databases, 22nd January, California.

Leiner, B. M., Cerf, V. G., Clark, D. D., Kahn, R. E., Kleinrock, L., Lynch, D. C., Postel, J., Roberts, L., Wolff, S. (2009). A brief history of the Internet. *ACM SIGCOMM Computer Communication Review*, *39*(5), 22-31.

Longfei, M., Dengpan, Y., Shiguo, L., Lina, W. (2009). *Computer-Based Copyright Control System in Social Network and an Ordinal Measure Based on Gray Scale Value.* Paper presented at the Multimedia Information Networking and Security, 18-20 November, Hubei, China.

Man jailed for second child porn offence. (2014). Retrieved 17th January 2014, from http://www.wigantoday.net/news/local/man-jailed-for-second-child-porn-offence-1-6375276

Merkle, R. C. (1979). Secrecy, authentication, and public key systems.

Monteiro, F., Dandache, A., M'Sir, A., Lepley, B. (2001). A fast CRC implementation *on FPGA using a pipelined architecture for the polynomial division.* Paper presented at the Electronics, Circuits and Systems, 2001. 2-5 September, Malta.

NIST (1994). FIPS 186 - (DSS), Digital Signature Standard. Retrieved from National Institute of Standards and Technology website: http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2-change1.pdf

NIST (1995). FIPS PUB 180-1 Secure Hash Standard (180-1). Retrieved from National Institute of Standards and Technology website: http://www.itl.nist.gov/fipspubs/fip180-1.htm

NIST (2012). FIPS PUB 180-4 Secure Hash Standard (SHS) (180-4). Retrieved from National Institute of Standards and Technology website: http://csrc.nist.gov/publications/fips/fips180-4/fips-180-4.pdf

Nock, R., Nielsen, F. (2006). On weighting clustering. *Pattern Analysis and Machine Intelligence, IEEE Transactions on, 28*(8), 1223-1235

Peffers, K., Tuunanen, T., Gengler, C. E., Rossi, M., Hui, W., Virtanen, V., Bragge, J. (2006). The design science research process: a model for producing and presenting information systems research. In Proceedings of the first international conference on design science research in information systems and technology (DESRIST), 24-25 February, California.

Peterson, W. W., Weldon, E. J. (1972). *Error-correcting codes*. Massachusetts: MIT press

Polischuk, H. (2014). Man gets jail term for sharing child porn images. Retrieved January 17, 2014, from http://www.leaderpost.com/news/gets+jail+term+sharing+child+porn+images/9397111/story.html

Preneel, B. (2010). The First 30 Years of Cryptographic Hash Functions and the NIST SHA-3 Competition. In J. Pieprzyk (Ed.), *Topics in Cryptology - CT-RSA 2010* (Vol. 5985, pp. 1-14): Springer Berlin Heidelberg.

Rabin, M. O. (1978). Digitalized signatures. *Foundations of secure computation, 78*, 155-166.

Rashedi, E., Nezamabadi-pour, H., Saryazdi, S. (2012). *A gradient descent based similarity refinement method for CBIR systems.* Paper presented at the Telecommunications (IST), 2012 Sixth International Symposium, 6-8 Nov, Tehran, Iran.

Ridge, E., Curry, E (2012). *Emerging Principles for Guerrilla Analytics Development*. Retrieved from http://www.deri.ie/sites/default/files/publications/ridge_bic3_guerrillaanalytics_0.pdf

Rutherford, H. (2013). New laws to tackle online sex crimes. Retrieved 20th June, 2013, from http://www.stuff.co.nz/dominion-post/news/politics/8723927/New-laws-to-tackle-online-sex-crimes

Saini, A. (2008). Solving the web's image problem. Retrieved 18th April 2013, from http://news.bbc.co.uk/2/hi/technology/7395751.stm

Song, H., Dharmapurikar, S., Turner, J., Lockwood, J. (2005). Fast hash table lookup using extended bloom filter: an aid to network processing. *SIGCOMM Computer Communnications Review*, *35*(4), 181-192.

Squire, M. D., Muller, H., Muller, W. (1999). *Improving response time by search pruning in a content-based image retrieval system, using inverted file techniques.* Paper presented at the IEEE Workshop on Content-Based Access of Image and Video Libraries, 22-22 June, Colorado.

Statistics (2013). Retrieved 26th April 2013, from http://www.youtube.com/yt/press/statistics.html

Stojanovic, I., Bogdanova, S., Bogdanov, M. (2007). *Content-Based Image Retrieving Improved by Pixel-Based Search.* Paper presented at the Systems, Signals and Image Processing, 2007 and 6th EURASIP Conference focused on Speech and Image Processing, Multimedia Communications and Services. 27-30 June, Maribor, Slovenia.

Trmač, M. (2013). *Miloslav Trmač*. Retrieved 15th October, 2013, from http://carolina.mff.cuni.cz/~trmac/blog/

Tuunanen, T. (2006). Is It Time to Re-Evaluate the Connection Between Bounded Rationality and Requirements Elicitation,Paper presented at the 12th Americas Conference on Information Systems, 4-6 August, 2006, Acapulco, Mexico.

Vitter, J. S. (2001). External memory algorithms and data structures: dealing with massive data. *ACM Computing surveys*, *33*(2), 209-271.

Ward, M. (2013). Google and Microsoft agree steps to block abuse images. Retrieved 16th December, 2013, from http://www.bbc.com/news/uk-24980765

Yiannopoulos, M. (2010). The law must learn to keep up with technology - Telegraph. Retrieved 15th October, 2013, from http://www.telegraph.co.uk/technology/twitter/8128252/The-law-must-learn-to-keep-up-with-technology.html

Yuval, G. (1979). How to swindle Rabin. *Cryptologia*, *3*(3), 187-191.

Zhang, Z., Shi, Y. (2009). *Skin color detecting unite YCgCb color space with YCgCr color space.* Paper presented at 2009 International Conference on the Image Analysis and Signal Processing (IASP), 11-12 April, Taizhou, China.

Zhiruo, C., Zheng, W., Zegura, E. (2000). *Performance of hashing-based schemes for Internet load balancing.* Paper presented at the INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies, 26-30 March, Tel Aviv, Israel.

Appendices

APPENDIX A

http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf

APPENDIX B

HBIR Tool (Hash Based Image Indexing)

Times are recorded in seconds to the 2 d.p.

MD4

Table 8. 1 HBIR MD4 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	20.92	21.33	21.86	21.27	20.47	21.17	49.11	21.17
150 KB	70.55	70.52	69.82	70.76	69.76	70.28	34.53	23.43
250 KB	104.44	104.68	104.33	105.17	105.43	104.81	42.54	20.96
350 KB	147.52	147.96	147.17	148.68	145.39	147.35	39.24	21.05
450 KB	185.23	185.25	187.84	190.64	183.97	186.59	39.23	20.73
550 KB	226.77	227.35	224.36	227.30	223.31	225.82	31.70	20.53
650 KB	257.76	257.67	257.68	257.34	257.13	257.52	48.31	19.81
750 KB	304.42	303.92	309.49	310.77	300.54	305.83	40.41	20.39
850 KB	346.36	345.91	348.83	348.62	341.46	346.23	39.60	20.37
950 KB	386.95	386.88	385.85	386.83	382.65	385.83	NA	20.31

MD5

Table 8. 2 HBIR MD5 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	20.73	20.24	20.88	20.92	20.24	20.60	48.86	20.60
150 KB	70.38	68.14	70.22	70.21	68.39	69.47	31.97	23.16
250 KB	101.97	100.58	101.88	102.00	100.73	101.43	41.07	20.29
350 KB	143.83	140.48	143.97	143.94	140.30	142.50	38.99	20.36
450 KB	182.79	178.92	183.70	183.24	178.84	181.50	33.77	20.17
550 KB	216.36	217.43	213.73	214.21	214.63	215.27	34.13	19.57
650 KB	251.34	246.62	251.58	251.20	246.27	249.40	46.37	19.18
750 KB	298.27	291.51	298.23	298.94	291.89	295.77	41.19	19.72
850 KB	340.72	331.20	340.78	341.43	330.69	336.96	37.28	19.82
950 KB	379.40	365.46	380.90	379.68	365.80	374.25	NA	19.70

SHA-1

Table 8. 3 HBIR SHA-1 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	21.43	21.43	21.46	21.56	20.87	21.35	58.69	21.35
150 KB	80.41	80.50	80.29	80.36	78.62	80.04	41.71	26.68
250 KB	121.62	121.24	123.37	122.13	120.38	121.75	48.57	24.35
350 KB	171.42	171.74	169.87	169.43	169.14	170.32	43.70	24.33
450 KB	215.66	215.98	213.93	213.73	210.79	214.02	48.45	23.78
550 KB	264.77	265.24	262.45	262.90	256.97	262.47	40.61	23.86
650 KB	302.76	303.28	302.70	302.49	304.14	303.07	49.65	23.31
750 KB	355.80	355.68	351.44	351.00	349.73	352.73	50.64	23.52
850 KB	406.97	406.81	401.88	401.29	399.90	403.37	47.21	23.73
950 KB	452.18	453.78	446.99	446.56	453.40	450.58	NA	23.71

SHA-256

Table 8. 4 HBIR SHA-256 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	22.81	23.88	22.13	23.52	22.14	22.89	56.40	22.89
150 KB	78.26	80.56	78.81	80.37	78.47	79.29	55.95	26.43
250 KB	134.29	136.31	134.68	136.64	134.28	135.24	54.22	27.05
350 KB	188.91	188.10	189.44	192.17	188.70	189.46	55.07	27.07
450 KB	248.15	248.25	241.23	243.66	241.38	244.53	53.72	27.17
550 KB	296.52	303.15	296.58	297.85	297.17	298.25	54.49	27.11
650 KB	350.54	356.53	350.54	354.86	351.25	352.75	44.35	27.13
750 KB	395.35	397.93	396.31	400.47	395.40	397.09	54.47	26.47
850 KB	449.25	455.20	450.54	447.59	455.25	451.57	46.99	26.56
950 KB	495.86	501.40	498.37	502.25	494.91	498.56	NA	26.24

CRC-64

Table 8. 5 HBIR CRC-64 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	20.80	20.43	20.84	20.81	20.78	20.73	50.17	20.73
150 KB	70.99	70.58	70.89	71.10	70.95	70.90	38.32	23.63
250 KB	108.40	107.81	110.15	110.33	109.44	109.23	42.21	21.85
350 KB	150.97	149.92	152.31	152.28	151.68	151.43	38.35	21.63
450 KB	188.24	188.96	190.95	190.62	190.13	189.78	42.38	21.09
550 KB	230.99	229.17	233.77	233.16	233.69	232.16	45.26	21.11
650 KB	276.91	275.32	277.53	278.59	278.75	277.42	34.83	21.34
750 KB	312.82	311.73	312.64	312.49	311.56	312.25	41.80	20.82
850 KB	351.36	349.11	356.66	356.45	356.67	354.05	42.77	20.83
950 KB	396.46	397.00	396.99	397.67	395.97	396.82	NA	20.89

CRC-32

Table 8. 6 HBIR CRC-32 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	20.60	20.38	20.51	20.41	20.34	20.45	41.99	20.45
150 KB	62.74	62.82	62.58	62.18	61.86	62.44	33.29	20.81
250 KB	96.14	95.55	96.13	95.47	95.33	95.72	37.16	19.14
350 KB	133.47	132.74	133.38	132.15	132.66	132.88	32.98	18.98
450 KB	167.41	165.66	166.86	164.87	164.51	165.86	34.90	18.43
550 KB	201.42	200.85	201.77	200.10	199.67	200.76	24.74	18.25
650 KB	225.14	225.42	225.69	225.33	225.93	225.50	44.06	17.35
750 KB	272.11	269.61	270.31	268.93	266.87	269.57	35.70	17.97
850 KB	305.16	306.67	306.74	304.17	303.59	305.27	32.13	17.96
950 KB	338.16	336.76	338.78	337.28	335.99	337.40	NA	17.76

XXHash

Table 8. 7 HBIR XXHash Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	21.36	20.56	20.66	21.12	20.93	20.93	44.34	20.93
150 KB	66.39	66.38	64.45	64.52	64.58	65.26	35.66	21.75
250 KB	102.62	102.26	98.33	100.61	100.81	100.92	39.94	20.18
350 KB	144.31	140.96	136.77	140.93	141.36	140.86	33.57	20.12
450 KB	174.21	171.61	172.49	176.99	176.88	174.44	35.31	19.38
550 KB	211.27	206.34	207.60	211.96	211.54	209.74	28.61	19.07
650 KB	239.35	237.86	237.57	237.52	239.45	238.35	37.96	18.33
750 KB	278.77	273.85	273.47	277.25	278.22	276.31	38.92	18.42
850 KB	314.68	311.41	310.75	319.12	320.18	315.23	45.14	18.54
950 KB	361.57	358.21	360.23	359.73	362.13	360.37	NA	18.97

FNV1a

Table 8. 8 HBIR FNV1a Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	20.65	22.17	20.67	21.85	21.86	21.44	62.59	21.44
150 KB	82.24	85.22	82.10	85.11	85.47	84.03	47.65	28.01
250 KB	130.97	132.40	130.74	132.27	131.99	131.67	52.55	26.33
350 KB	183.57	184.43	184.55	184.38	184.21	184.23	48.09	26.32
450 KB	231.14	232.11	231.16	233.99	233.20	232.32	49.70	25.81
550 KB	277.45	285.14	276.97	285.13	285.41	282.02	51.24	25.64
650 KB	335.94	331.59	335.37	331.88	331.51	333.26	49.35	25.64
750 KB	381.47	388.52	379.63	378.52	384.94	382.61	49.99	25.51
850 KB	421.82	439.26	422.38	439.39	440.19	432.61	45.74	25.45
950 KB	470.31	483.76	469.83	483.67	484.15	478.34	NA	25.18

Murmurhash2

Table 8. 9 HBIR Murmurhash2 Hashing Algorithm Raw Data

Size	Test 1	Test 2	Test 3	Test 4	Test 5	AVG. Time	Difference I2 - I1	AVG per 50KB
50 KB	20.65	20.16	20.82	20.25	20.73	20.52	49.12	20.52
150 KB	70.14	68.33	70.95	68.62	70.16	69.64	31.63	23.21
250 KB	101.72	100.63	101.79	100.56	101.62	101.27	40.13	20.25
350 KB	143.15	139.51	142.82	139.21	142.29	141.39	37.56	20.20
450 KB	181.10	176.12	181.26	176.46	179.84	178.96	33.25	19.88
550 KB	211.72	212.31	213.40	212.66	210.97	212.21	29.52	19.29
650 KB	242.79	244.88	242.72	245.40	232.85	241.73	46.76	18.59
750 KB	286.44	287.20	286.51	285.73	296.59	288.49	44.10	19.23
850 KB	335.93	325.50	338.61	324.40	338.51	332.59	34.36	19.56
950 KB	367.62	355.99	365.87	369.62	375.67	366.95	NA	19.31

APPENDIX C

Table 8. 10 FTK 50KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
50KB	1	33	44	2024
50KB	2	NA	NA	NA
50KB	3	33	43	2023
50KB	4	33	49	2029
50KB	5	33	38	2018
			Average	**2023.5**

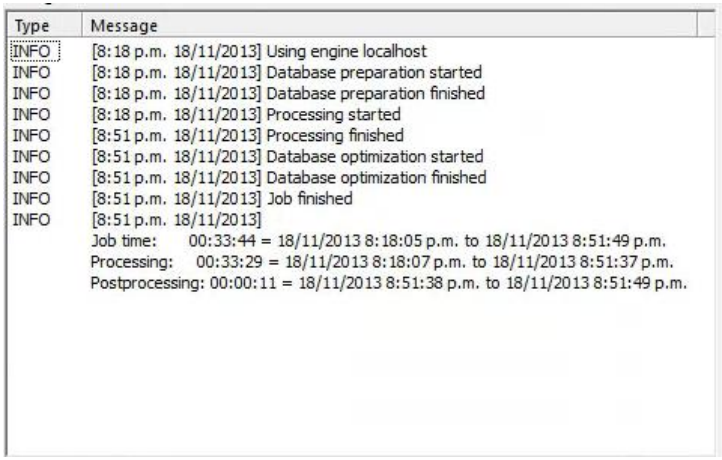


Figure 8. 1 FTK 50KB Test 1

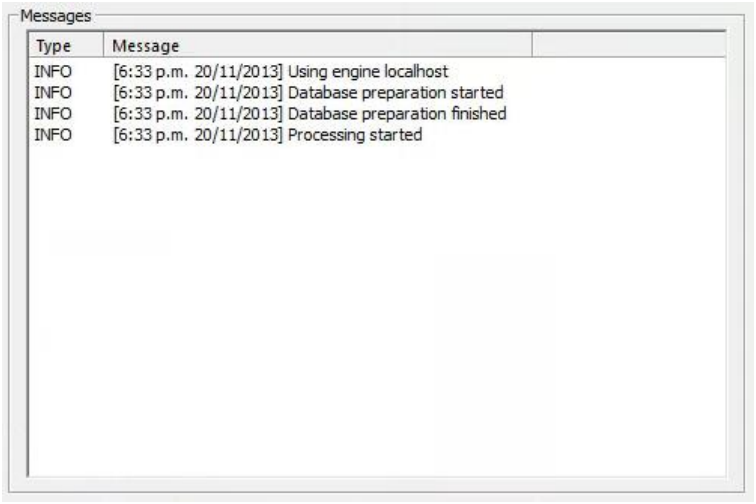


Figure 8. 2 FTK 50KB Test 2

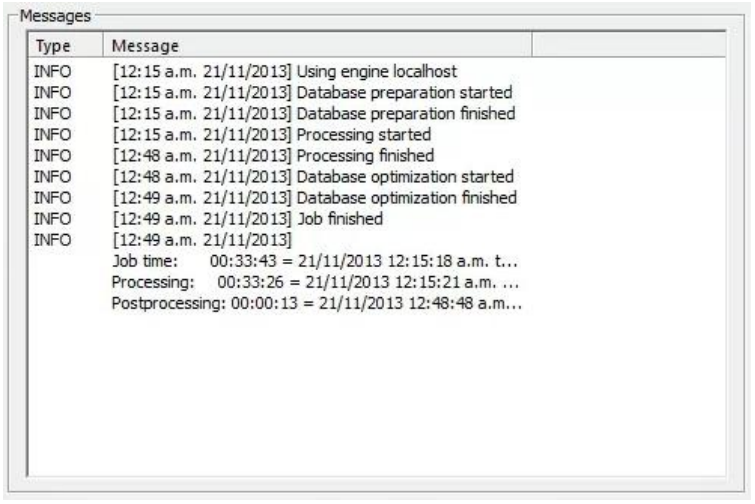


Figure 8. 3 FTK 50KB Test 3

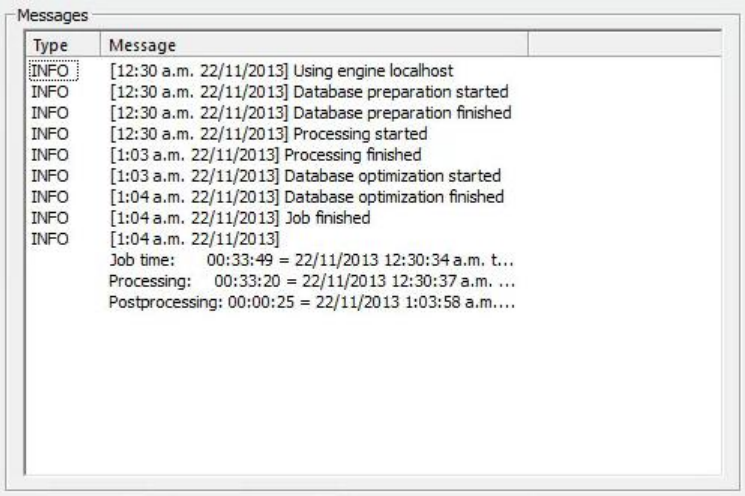


Figure 8. 4 FTK 50KB Test 4

Table 8. 11 FTK 150KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
150KB	1	34	5	2045
150KB	2	34	5	2045
150KB	3	32	25	1945
150KB	4	34	29	2069
150KB	5	34	36	2076
			Average	**2036**

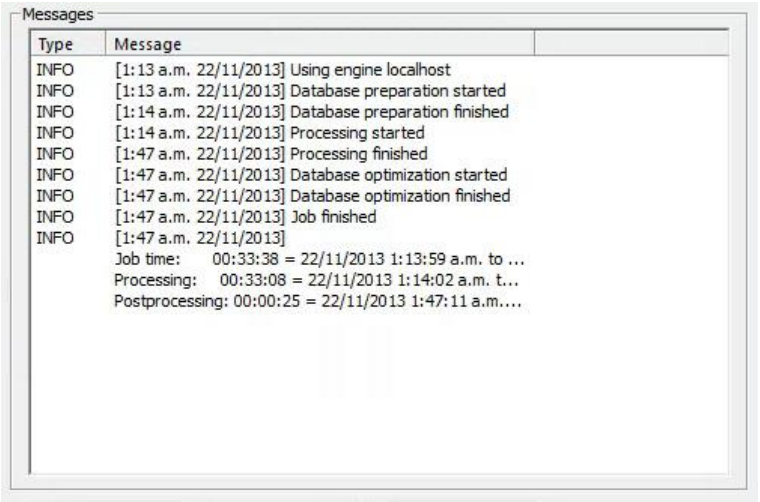


Figure 8. 5 FTK 50KB Test 5

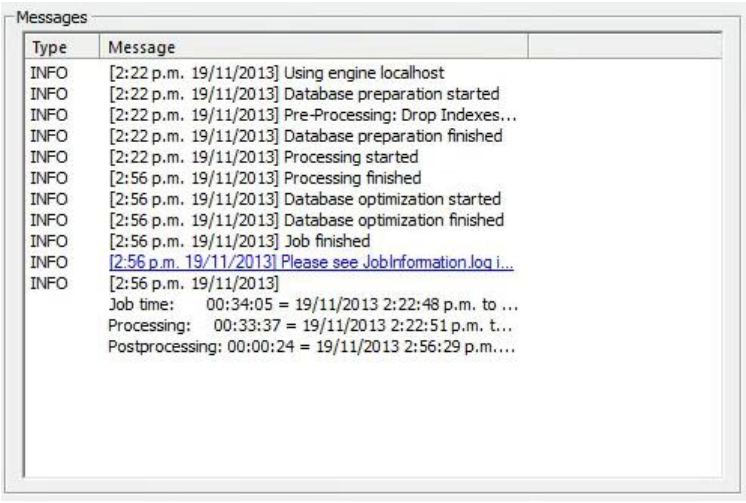


Figure 8. 6 FTK 150KB Test 1

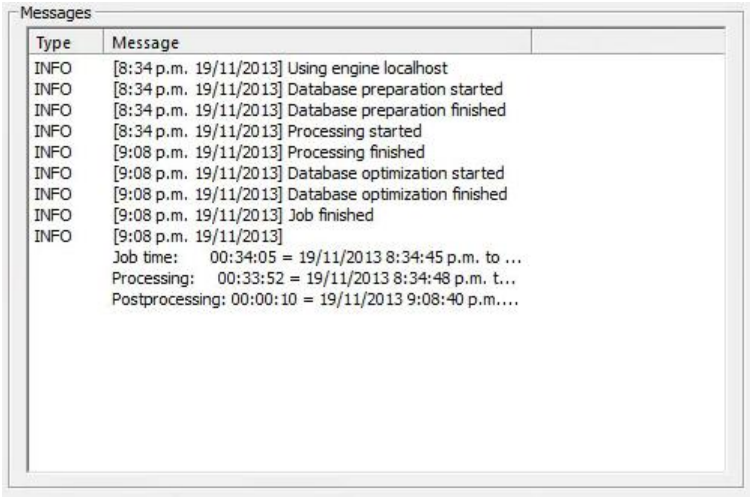


Figure 8. 7 FTK 150KB Test 2

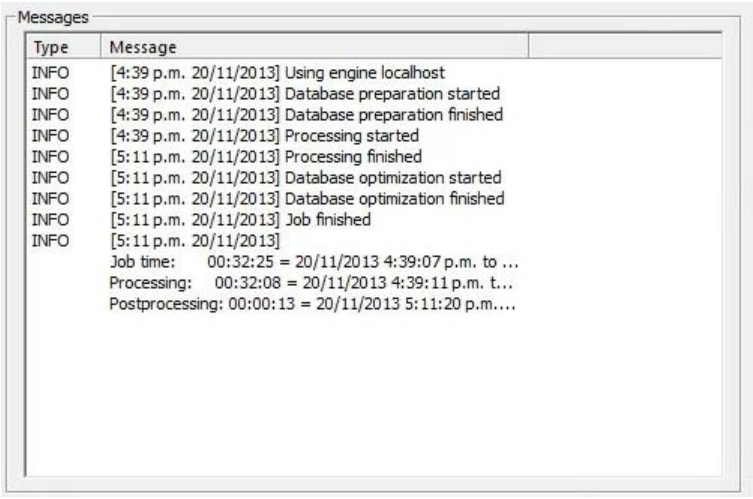


Figure 8. 8 FTK 150KB Test 3

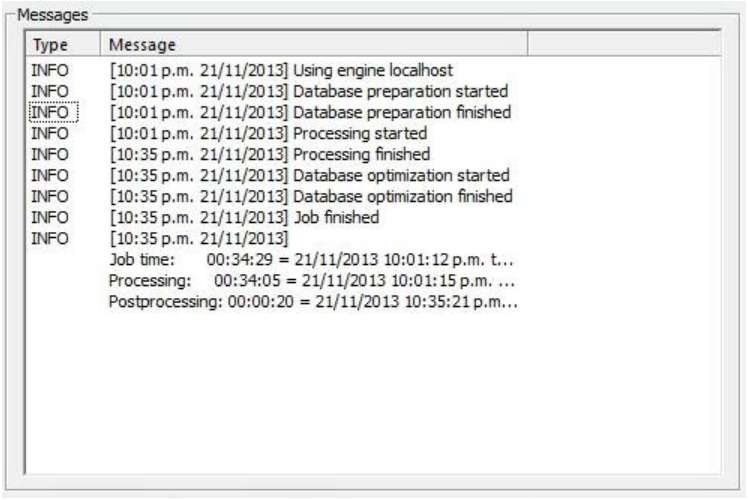


Figure 8. 9 FTK 150KB Test 4

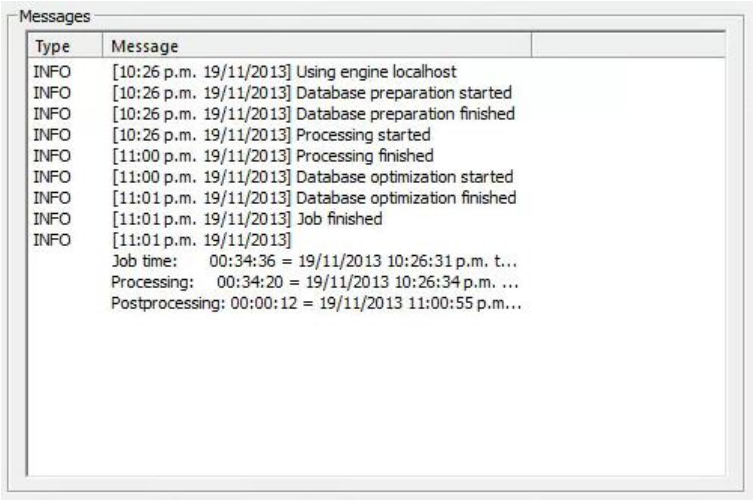


Figure 8. 10 FTK 150KB Test 5

Table 8. 12 FTK 250KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
250KB	1	39	19	2359
250KB	2	35	22	2122
250KB	3	37	23	2243
250KB	4	35	18	2118
250KB	5	34	37	2077
			Average	**2183.8**

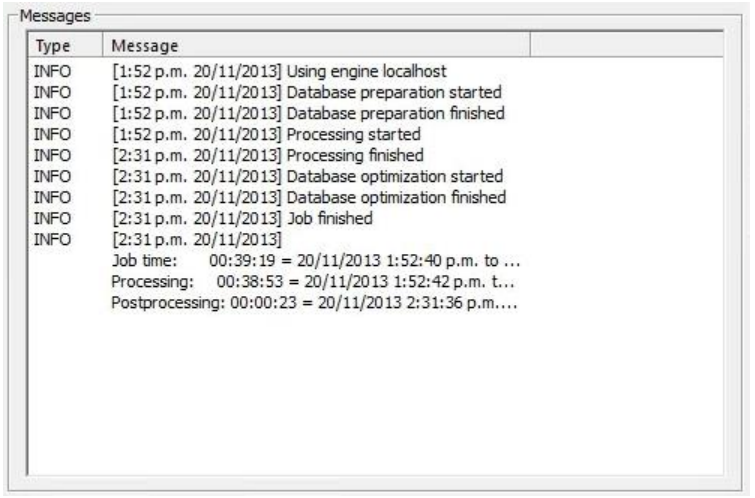


Figure 8. 11 FTK 250KB Test 1

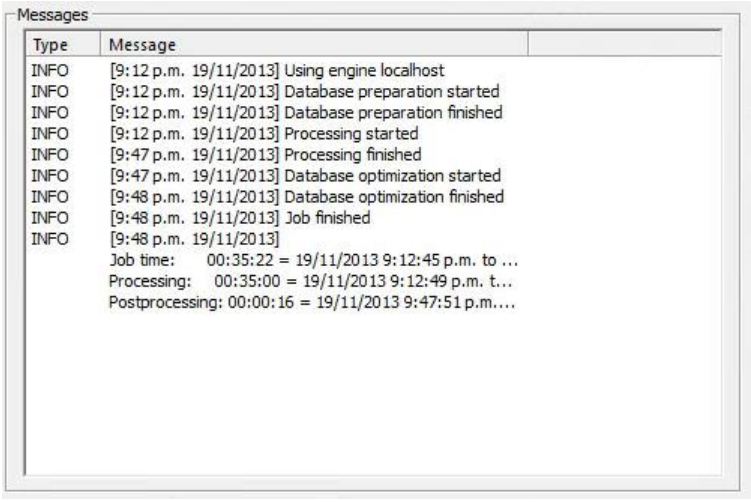


Figure 8. 12 FTK 250KB Test 2

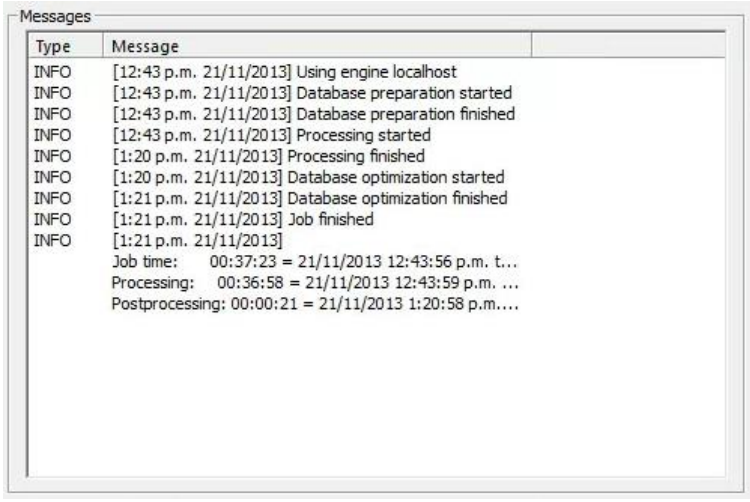


Figure 8. 13 FTK 250KB Test 3

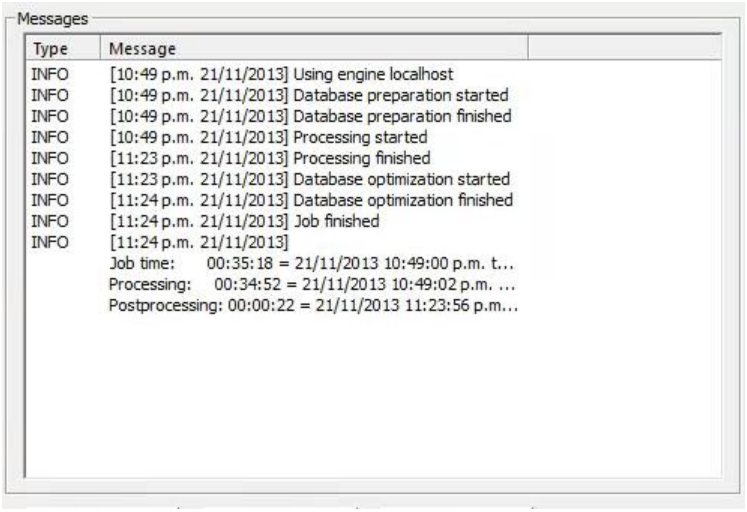


Figure 8. 14 FTK 250KB Test 4

Table 8. 13 FTK 350KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
350KB	1	38	24	2304
350KB	2	39	14	2354
350KB	3	39	13	2353
350KB	4	38	28	2308
350KB	5	38	36	2316
			Average	**2327**

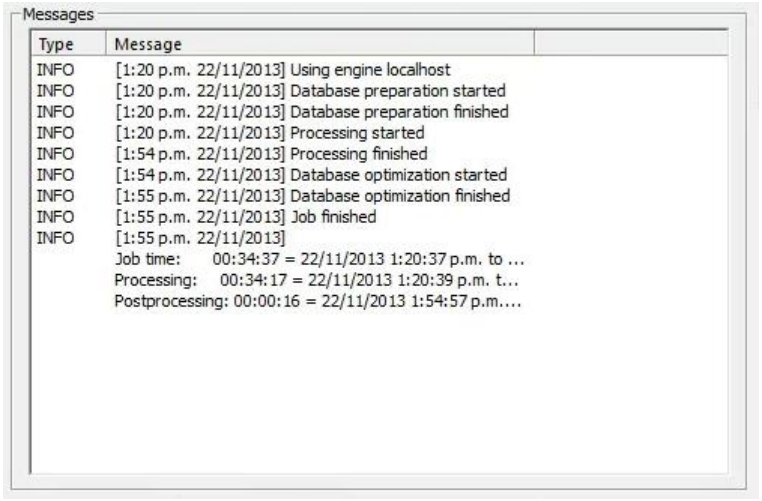


Figure 8. 15 FTK 250KB Test 5

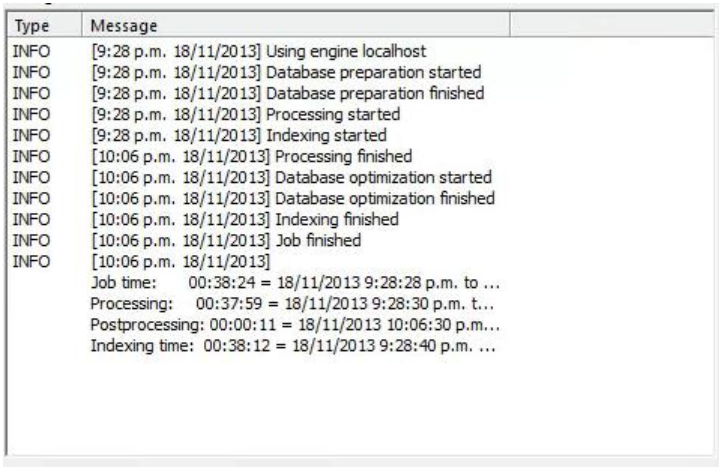


Figure 8. 16 FTK 350KB Test 1

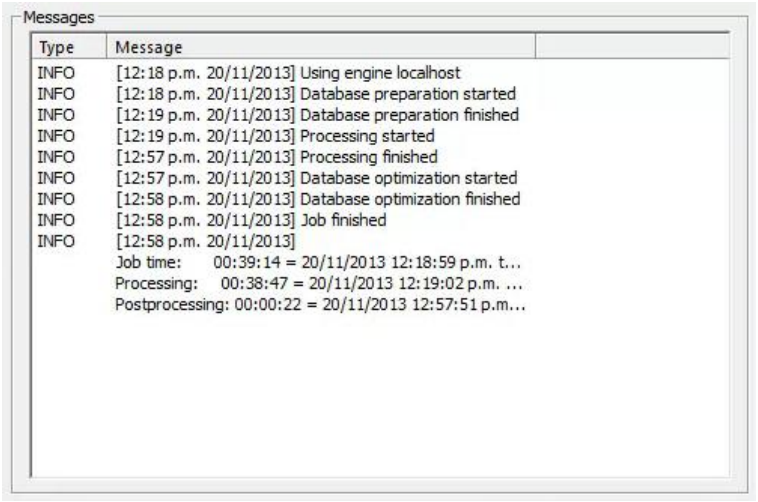


Figure 8. 17 FTK 350KB Test 2

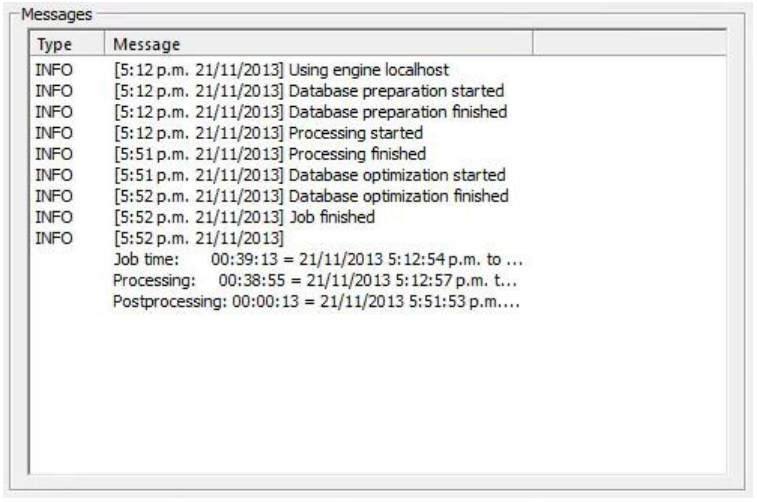


Figure 8. 18 FTK 350KB Test 3

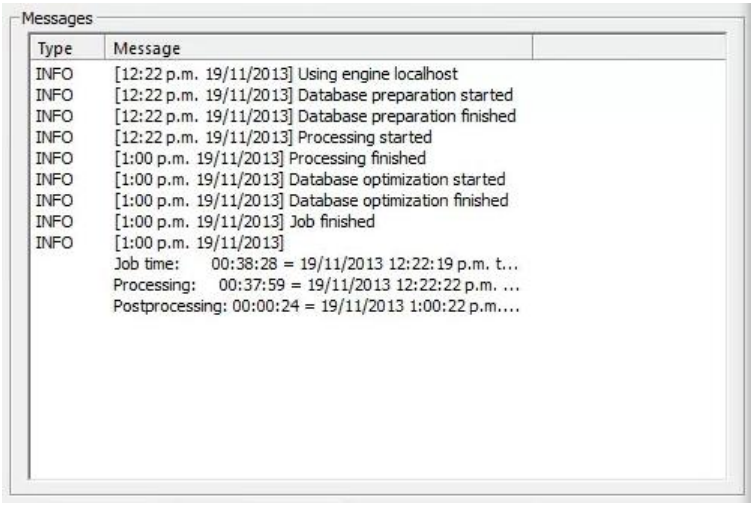


Figure 8. 19 FTK 350KB Test 4

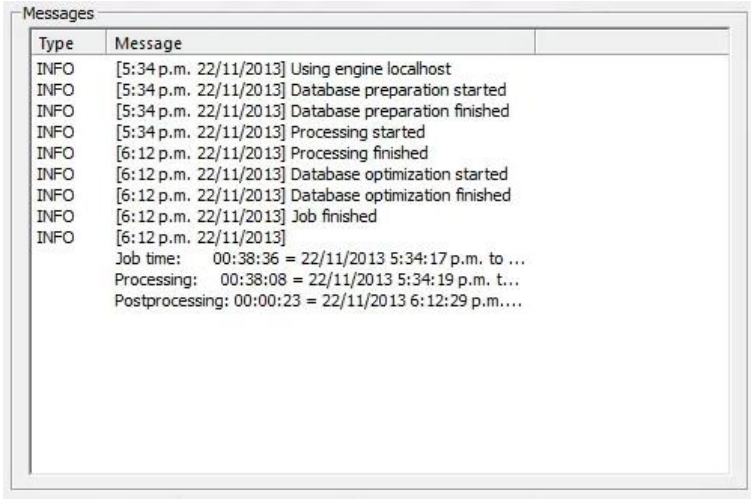


Figure 8. 20 FTK 350KB Test 5

Table 8. 14 FTK 450KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
450KB	1	39	19	2359
450KB	2	39	20	2360
450KB	3	39	22	2362
450KB	4	34	43	2083
450KB	5	39	25	2365
			Average	**2305.8**

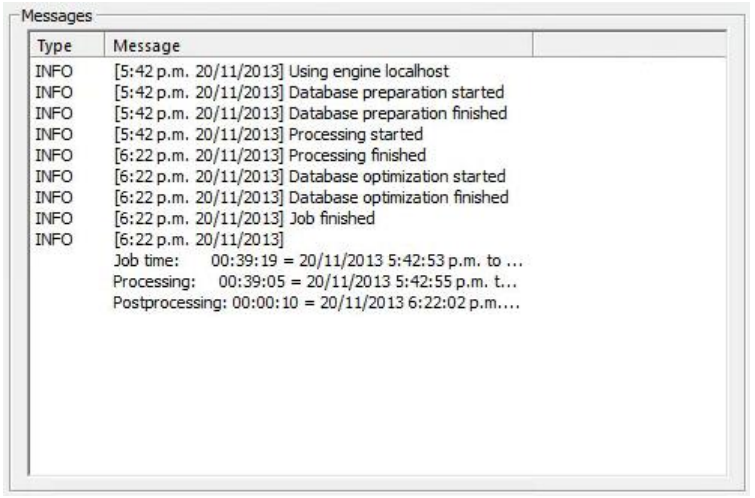


Figure 8. 21 FTK 450KB Test 1

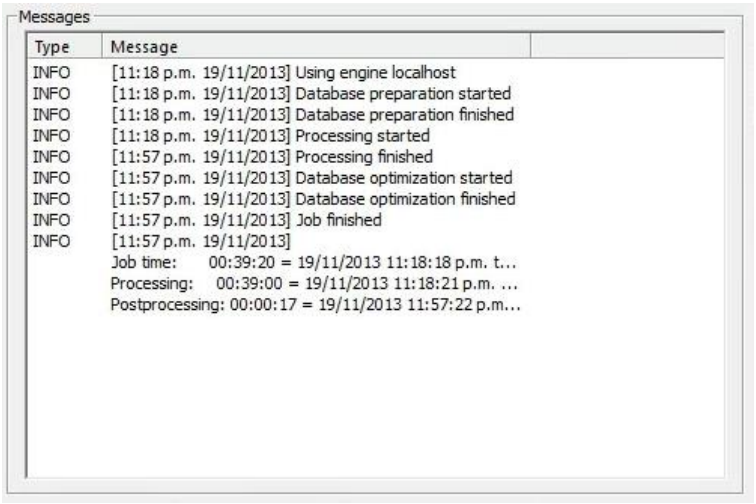


Figure 8. 22 FTK 450KB Test 2

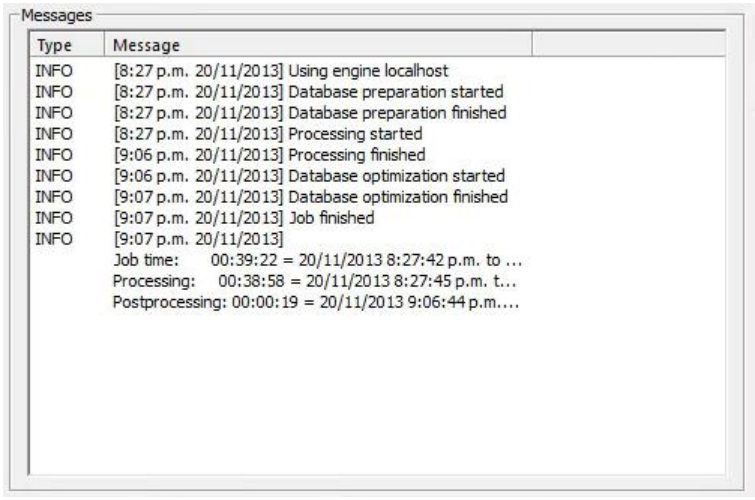


Figure 8. 23 FTK 450KB Test 3

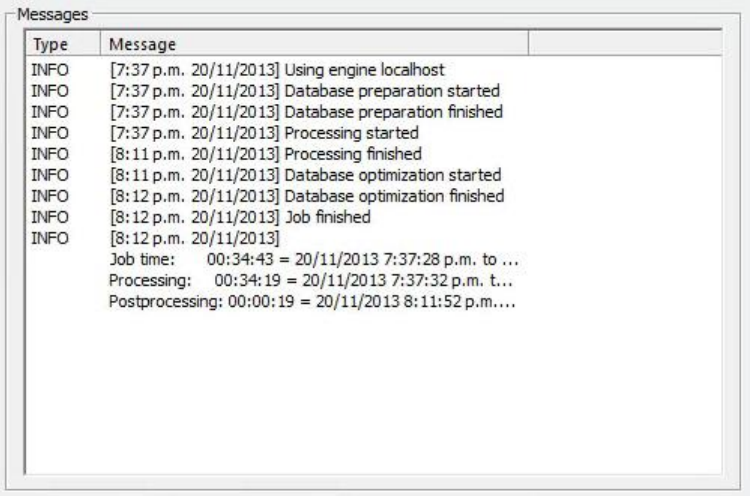


Figure 8. 24 FTK 450KB Test 4

Table 8. 15 FTK 550KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
550KB	1	39	30	2370
550KB	2	39	28	2368
550KB	3	39	56	2396
550KB	4	39	33	2373
550KB	5	39	50	2390
			Average	**2379.4**

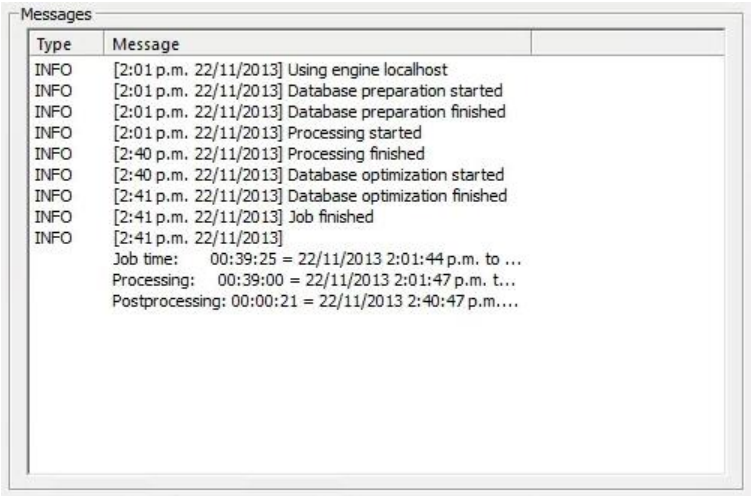


Figure 8. 25 FTK 450KB Test 5

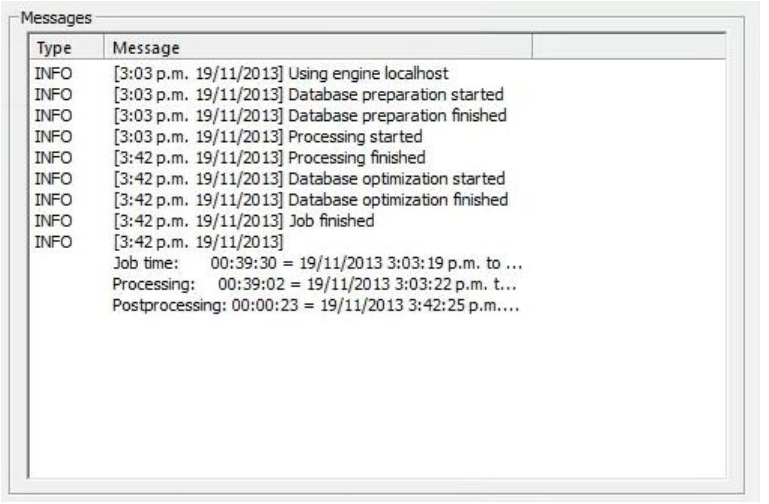


Figure 8. 26 FTK 550KB Test 1

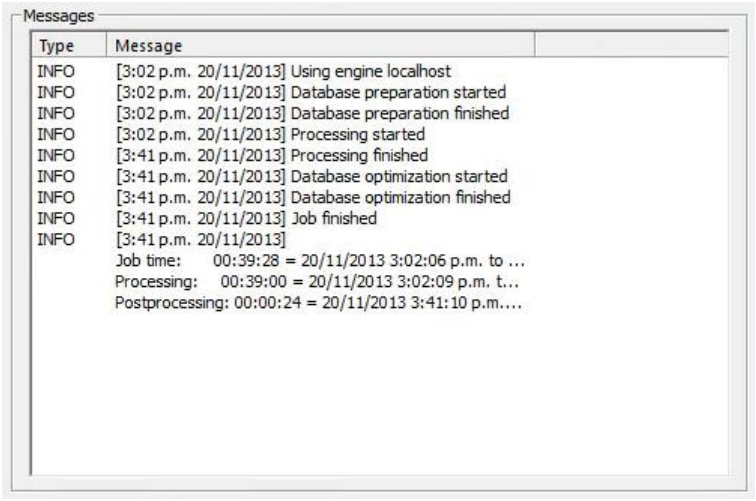


Figure 8. 27 FTK 550KB Test 2

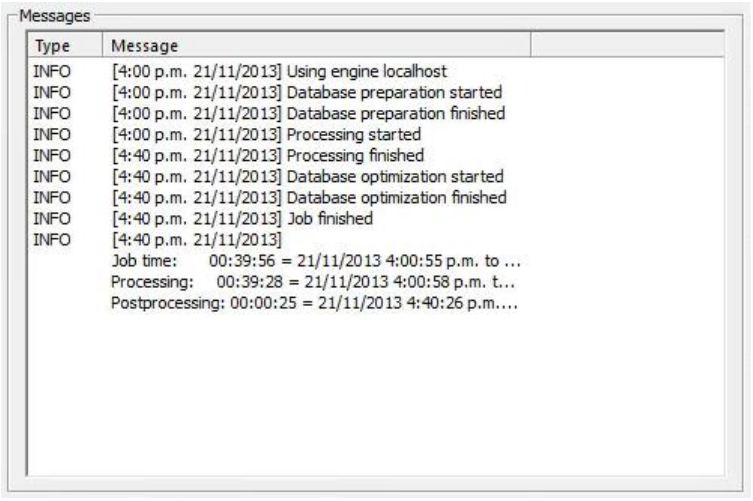


Figure 8. 28 FTK 550KB Test 3

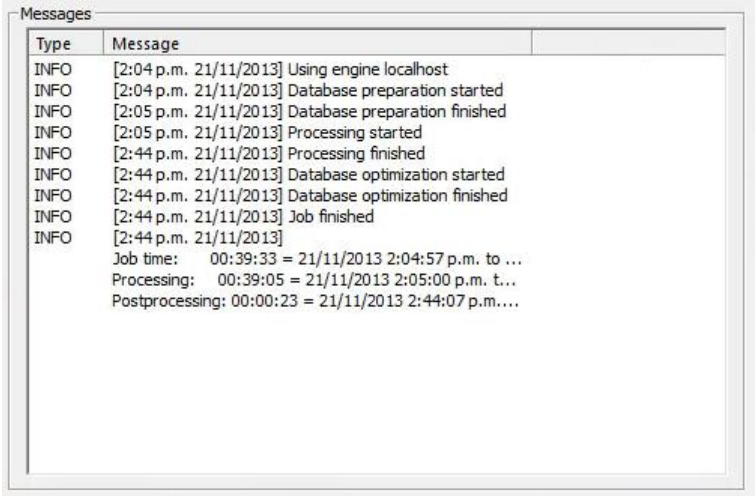


Figure 8. 29 FTK 550KB Test 4

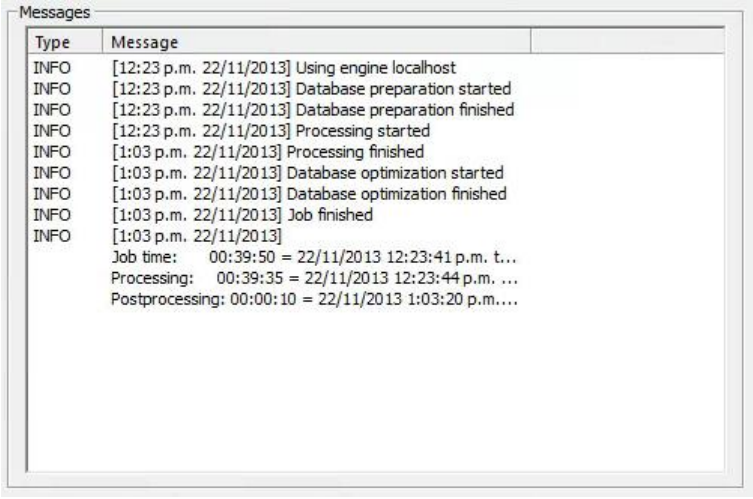


Figure 8. 30 FTK 550KB Test 5

Table 8. 16 FTK 650KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
650KB	1	40	32	2432
650KB	2	40	3	2403
650KB	3	40	12	2412
650KB	4	40	27	2427
650KB	5	40	20	2420
			Average	**2418.8**

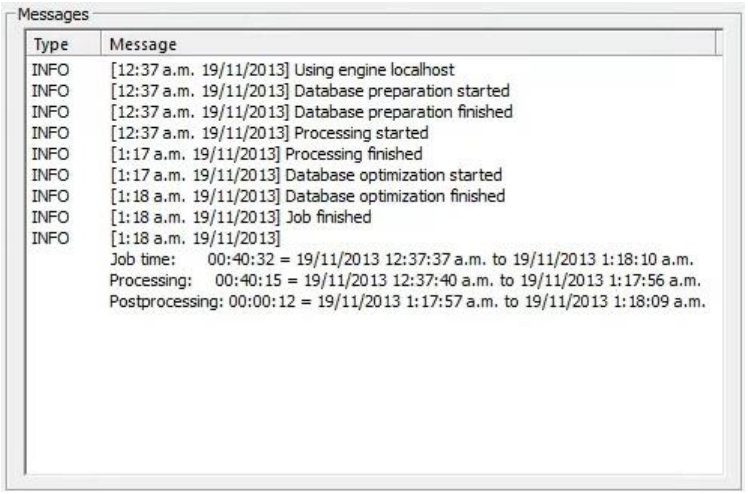


Figure 8. 31 FTK 650KB Test 1

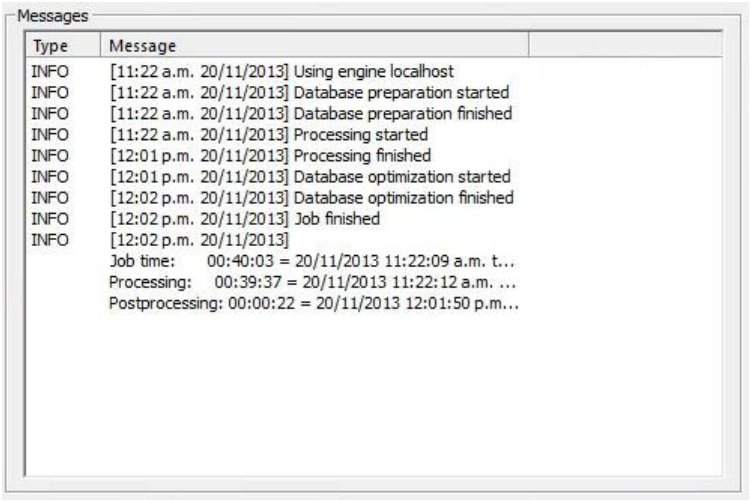


Figure 8. 32 FTK 650KB Test 2

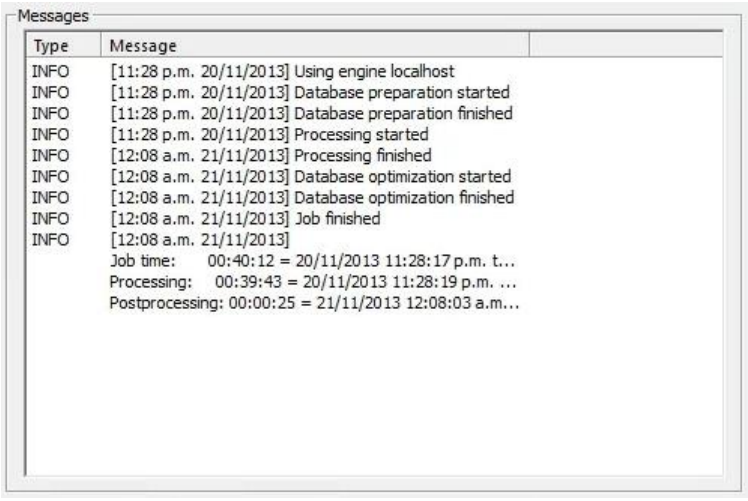


Figure 8. 33 FTK 650KB Test 3

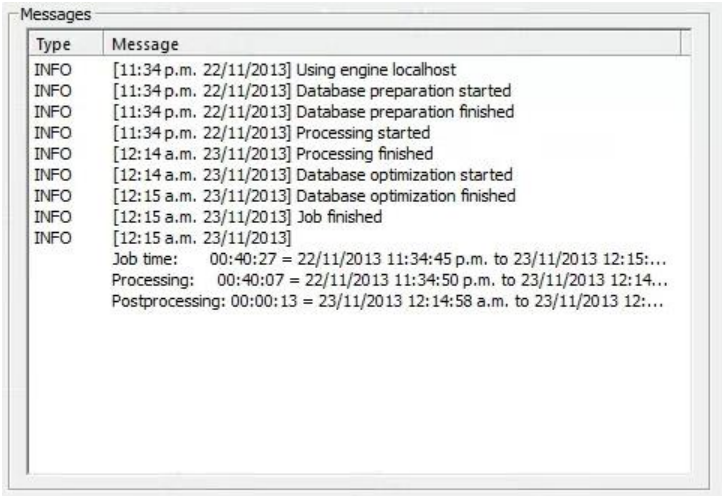


Figure 8. 34 FTK 650KB Test 4

Table 8. 17 FTK 750KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
750KB	1	40	50	2450
750KB	2	40	46	2446
750KB	3	53	3	3183
750KB	4	40	43	2443
750KB	5	40	33	2433
			Average	**2591**

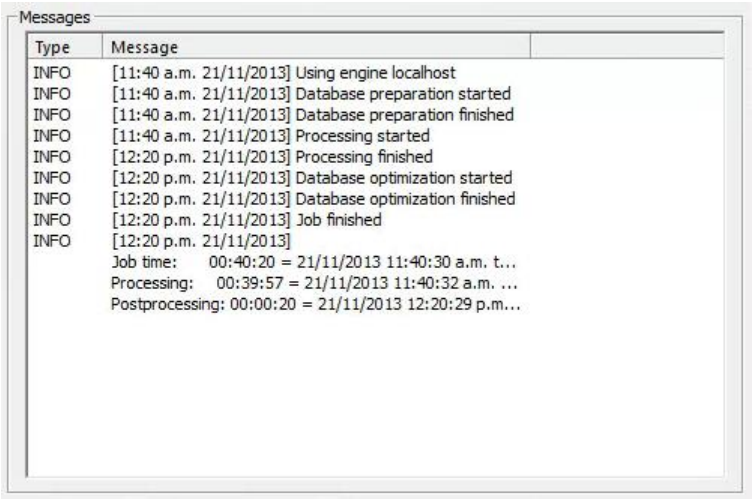


Figure 8. 35 FTK 650KB Test 5

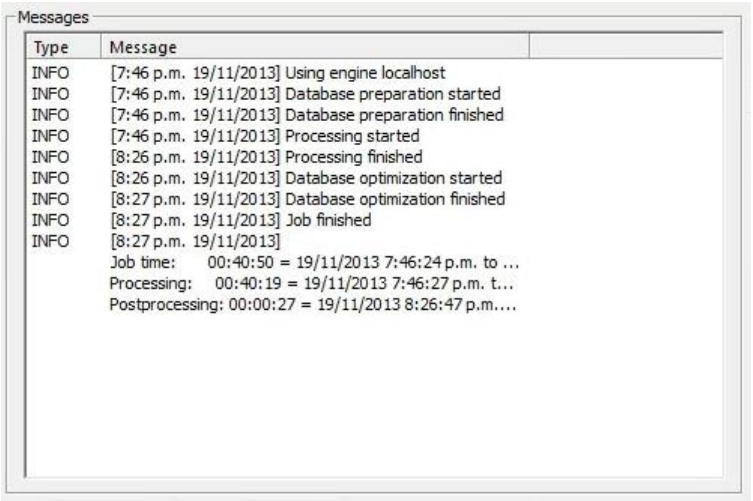


Figure 8. 36 FTK 750KB Test 1

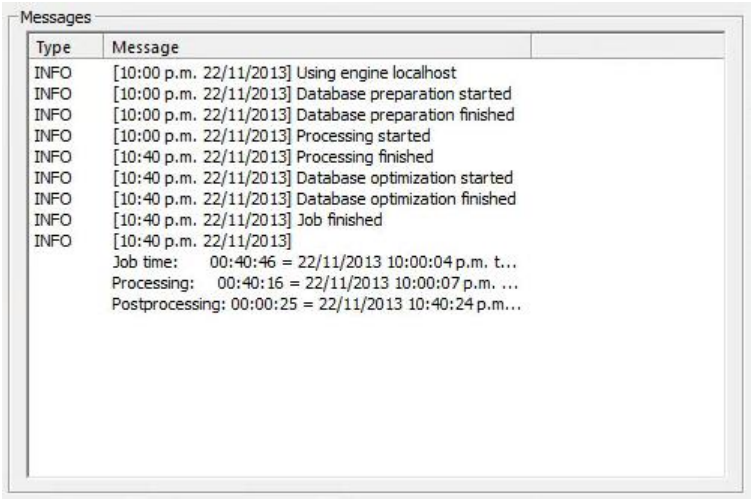


Figure 8. 37 FTK 750KB Test 2

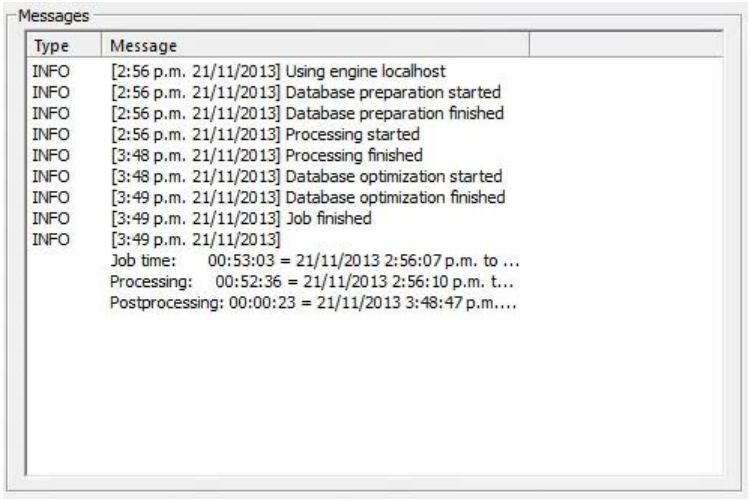


Figure 8. 38 FTK 750KB Test 3

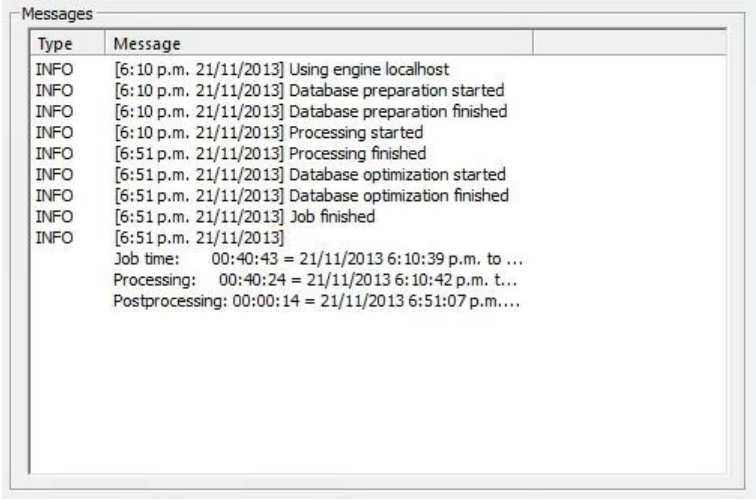


Figure 8. 39 FTK 750KB Test 4

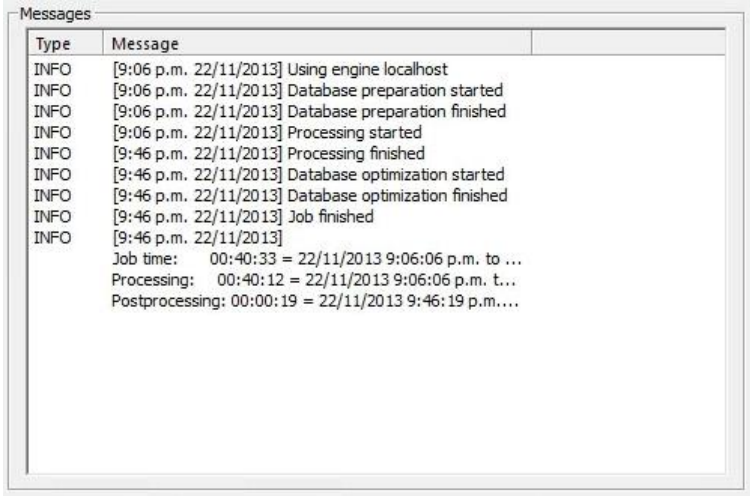


Figure 8. 40 FTK 750KB Test 5

Table 8. 18 FTK 850KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
850KB	1	40	52	2452
850KB	2	41	18	2478
850KB	3	41	3	2463
850KB	4	40	36	2436
850KB	5	41	23	2483
			Average	**2462.4**

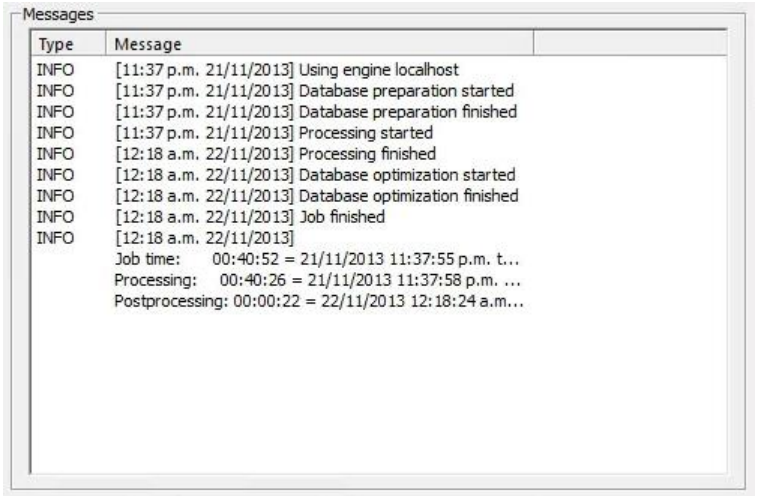


Figure 8. 41 FTK 850KB Test 1

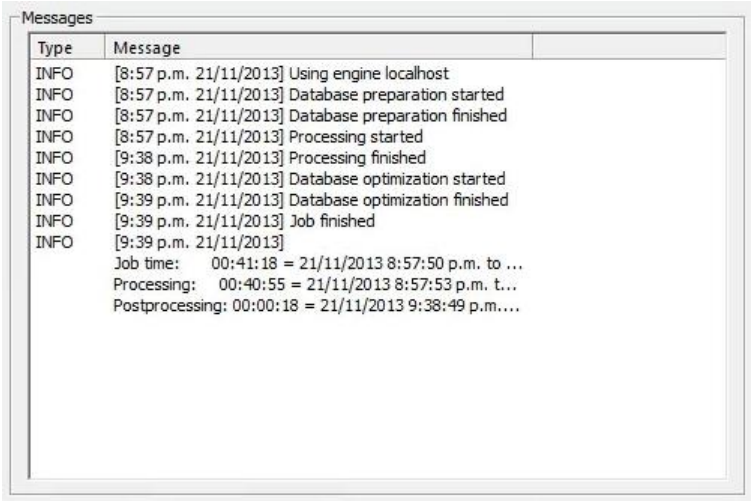


Figure 8. 42 FTK 850KB Test 2

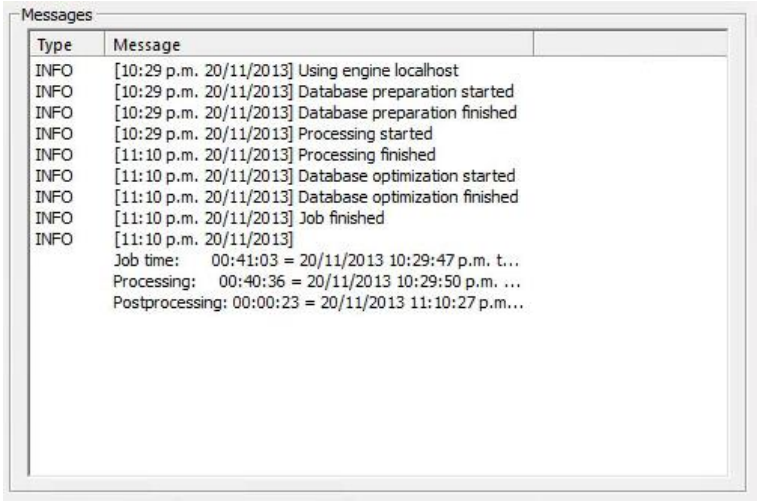


Figure 8. 43 FTK 850KB Test 3

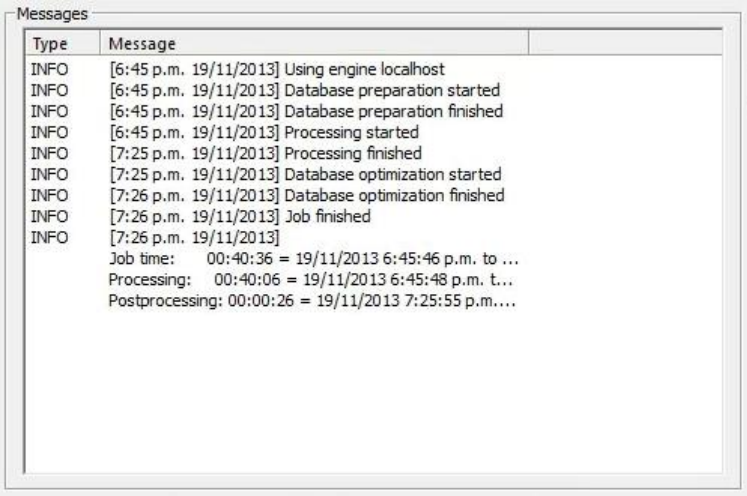


Figure 8. 44 FTK 850KB Test 4

Table 8. 19 FTK 950KB Summary Data

Size	Test Num.	Min	Sec	Total (sec)
950KB	1	52	9	3129
950KB	2	53	50	3230
950KB	3	53	29	3209
950KB	4	50	48	3048
950KB	5	55	23	3323
			Average	**3187.8**

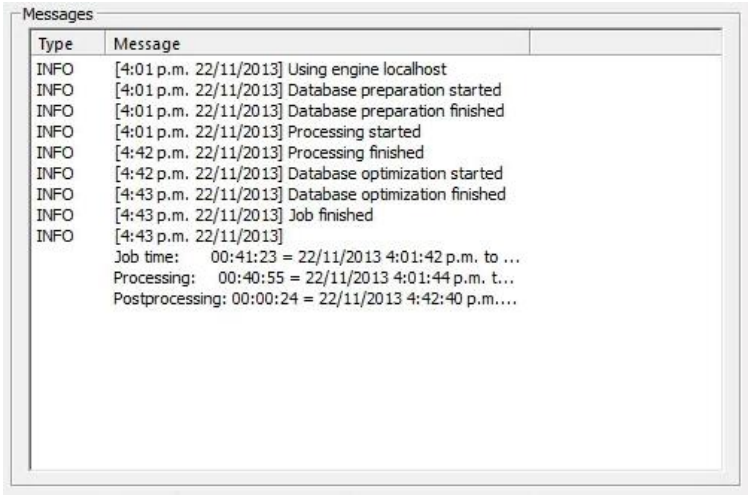


Figure 8. 45 FTK 850KB Test 5

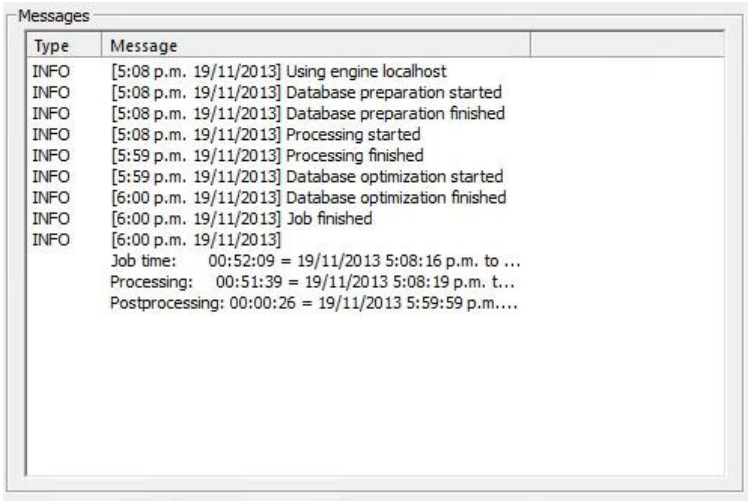


Figure 8. 46 FTK 950KB Test 1

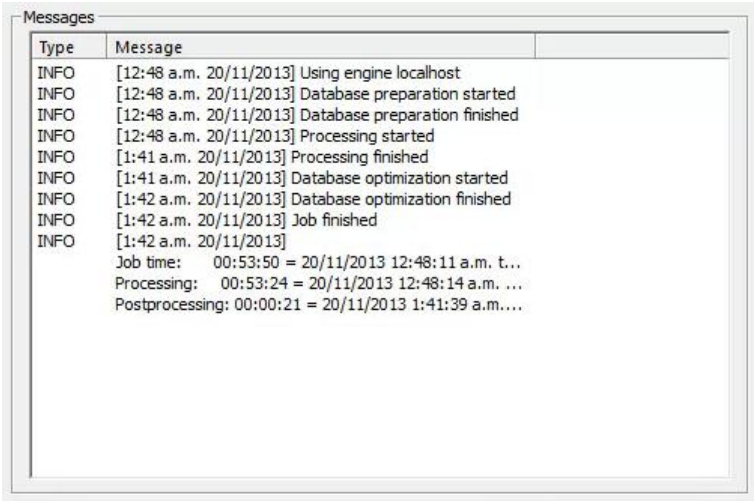


Figure 8. 47 FTK 950KB Test 2

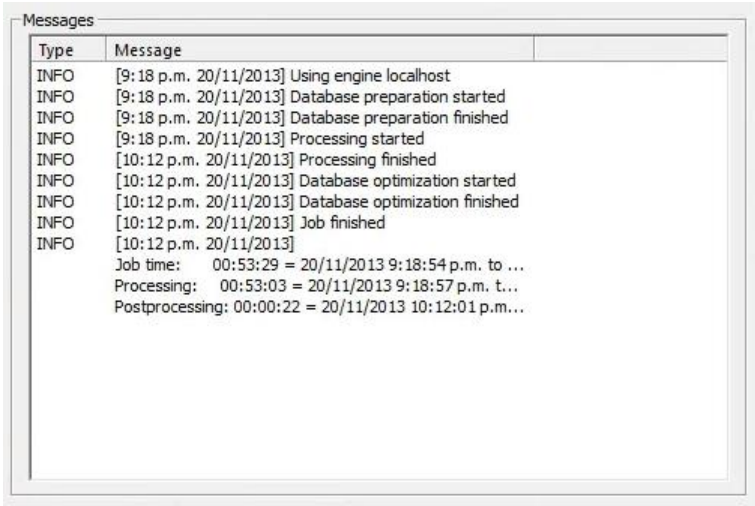


Figure 8. 48 FTK 950KB Test 3

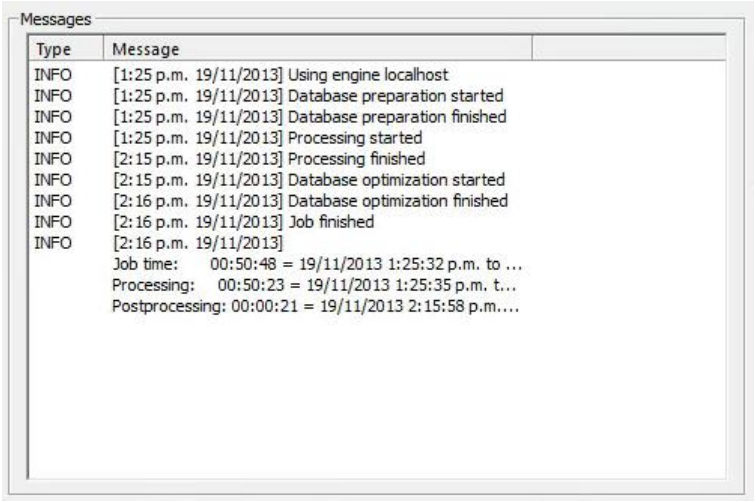


Figure 8. 49 FTK 950KB Test 4

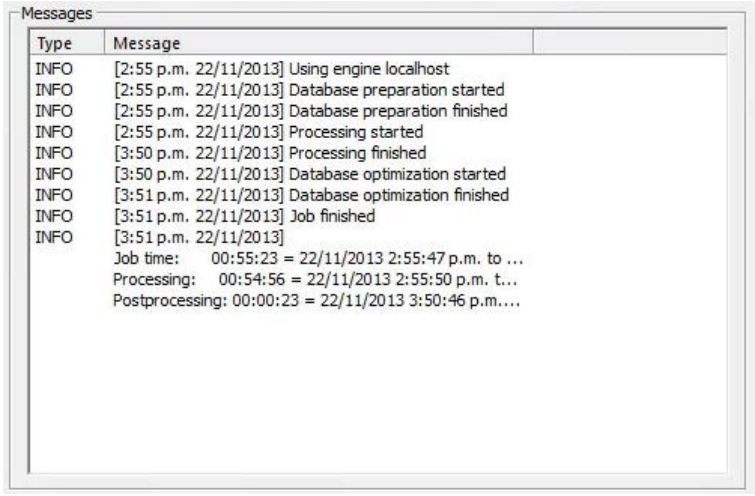


Figure 8. 50 FTK 950KB Test 5

APPENDIX D

Encase (Concept Based Image Indexing)

#multiple tests were not required as the data did no fluctuate.

Size	Number of Images	Number of Keywords	Objectionable Images	Start Time	Stop Time	Time Taken
50KB	10,000	2000	5	12:59:02	23:59:23	21s
150KB	10,000	2000	5	1:07:04	1:08:26	1m22
250KB	10,000	2000	5	1:11:29	1:13:57	2m28s
350KB	10,000	2000	5	1:17:55	1:19:47	2m52s
450kB	10,000	2000	5	1:21:28	1:31:51	2min58s
550KB	10,000	2000	5	1:35:10	1:18:14	3m04s
650KB	10,000	2000	5	1:41:35	1:45:28	3m53s
750KB	10,000	2000	5	1:48:04	1:52:57	4m53s
850KB	10,000	2000	5	1:54:14	1:59:34	5m20s
950KB	10,000	2000	5	2:03:46	2:10:33	6m47s