An Improved Randomization of a Multi-Blocking JPEG based Steganographic system

Peter Dawoud Shenouda Dawoud

B.Sc. Engineering (Computer)

University of KwaZulu-Natal, South Africa

Submitted in fulfilment of the academic requirements

for the degree of M.Sc. in Engineering

in the School of Electrical, Electronic and Computer Engineering

at the University of KwaZulu-Natal, South Africa

June 18, 2010

As the candidate’s supervisor I have approved this dissertation for submission.

Signed: ___________________________________

Name: **Prof. Roger Peplow**

Date: June 18, 2010

As the candidate’s co-supervisor I have approved this dissertation for submission.

Signed: ___________________________________

Name: **Mr. Bashan Naidoo**

Date: June 18, 2010

DECLARATION

I.. declare that

I. The research reported in this dissertation/thesis, except where otherwise indicated, is my original work.

II. This dissertation/thesis has not been submitted for any degree or examination at any other university.

III. This dissertation/thesis does not contain other persons’ data, pictures, graphs or other information, unless specifically acknowledged as being sourced from other persons.

IV. This dissertation/thesis does not contain other persons’ writing, unless specifically acknowledged as being sourced from other researchers. Where other written sources have been quoted, then:

   a. Their words have been re-written but the general information attributed to them has been referenced;

   b. Where their exact words have been used, their writing has been placed inside quotation marks, and referenced.

V. Where I have reproduced a publication of which I am an author, co-author or editor, I have indicated in detail which part of the publication was actually written by myself alone and have fully referenced such publications.

VI. This dissertation/thesis does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the dissertation/thesis and in the References sections.

Signed: __________________________________

To my loving family, I thank you for all the love and support you have showed me throughout all my studies.

Acknowledgements

I would like to take this opportunity to thank my supervisor Prof. Roger Peplow for all his time and effort. He has provided me with a constant source of knowledge and has been there to guide me through every major hurdle I have encountered. I would also like to thank him for constantly pushing me to perform better, not just through my MSc but also through my undergraduate studies. He has taught me a lot both technically and personally and I thank him for all he has done.

My thanks are also extended to Mr. Bashan Naidoo, my co-supervisor, for all the time he has spent discussing many different topics both related to my research and other general topics. I thank him from his guidance and encouragement.

I would also like to thank Prof. Stanley Mneney for his assistance at the start of my thesis. I appreciate his guidance as well as all the input he gave me at the start of my research.

I would also like to thank Ms. Tonya M Esterhuizen for her assistance during the course of my research. Her assistance through the designing of my qualitative survey was invaluable and I thank for her taking the time from her schedule to guide me through the process.

Finally I would like to thank Prof. DS Dawoud. He inspired me to pursue my MSc and without him I doubt I would have achieved anything. I thank him for constantly expecting me to perform at my best and helping me throughout my life both as his son and his student. I thank him for all his input and encouragement and for all the lessons he has taught me.

Abstract:

Steganography is classified as the art of hiding information. In a digital context, this refers to our ability to hide secret messages within innocent digital cover data. The digital domain offers many opportunities for possible cover mediums, such as cloud based hiding (saving secret information within the internet and its structure), image based hiding, video and audio based hiding, text based documents as well as the potential of hiding within any set of compressed data.

This dissertation focuses on the image based domain and investigates currently available image based steganographic techniques. After a review of the history of the field, and a detailed survey of currently available JPEG based steganographic systems, the thesis focuses on the systems currently considered to be secure and introduces mechanisms that have been developed to detect them.

The dissertation presents a newly developed system that is designed to counter act the current weakness in the YASS JPEG based steganographic system. By introducing two new levels of randomization to the embedding process, the proposed system offers security benefits over YASS. The introduction of randomization to the B-block sizes as well as the E-block sizes used in the embedding process aids in increasing security and the potential for new, larger E-block sizes also aids in providing an increased set of candidate coefficients to be used for embedding.

The dissertation also introduces a new embedding scheme which focuses on hiding in medium frequency coefficients. By hiding in these medium frequency coefficients, we allow for more aggressive embedding without risking more visual distortion but trade this off with a risk of higher error rates due to compression losses.

Finally, the dissertation presents simulation aimed at testing the proposed system performance compared to other JPEG based steganographic systems with similar embedding properties. We show that the new system achieves an embedding capacity of 1.6, which represents around a 7 times improvement over YASS. We also show that the new system, although introducing more bits in error per B-block, successfully allows for the embedding of up to 2 bits per B-block more than YASS at a similar error rate per B-block. We conclude the results by demonstrating the new systems ability to resist detection both through human observation, via a survey, as well as resist computer aided analysis.

Contents

Table of Figures

1 Introduction

The subject of steganography, although relatively young within the context of the digital domain, has been around for quite some time. The core concept of hiding information can be traced back for hundreds of years and early examples of its use can be found throughout history.

This section introduces the reader to the concept and meaning of steganography. The concept of steganography can best be explained by the following simple scenario:

Assume Alice and her brother Bob are planning to buy their parents a surprise anniversary present. Since this is a surprise present, they aim to keep their scheming and planning as covert as possible. At this point Alice and Bob are faced with a challenge, how are they going to communicate without raising the suspicion of their parents?

The first option the two have is to simply talk about their plans openly in front of their parents. Although this is a rather simple approach, it leaves very little in the way of surprise, since their parents will be fully aware of their plans. A second option for the two could involve them running out the room and making their plans out of their parents view. Although this provides Alice and Bob with the secrecy they require, their constant disappearance will raise a lot of suspicion. A third possibility for our adventurous siblings could involve the writing of notes in a language foreign to their parents. Again this achieves the goal of keeping their plans secret but their parents will surely be suspicious of the heavy use of a foreign language by their children (Note: This is the paradigm of cryptography; communicating using texts that are unknown for any observer other than sender and recipient).

At this point, Alice and Bob still have a problem. On the one hand they require a communication channel that allows them to freely communicate while not raising the suspicion of anyone monitoring their communications. Steganography offers Alice and Bob with a solution.

Now if Alice and Bob both share a passion for drawing, their talent provides them with a perfect carrier for their secret plans. They can easily incorporate their plans into drawings and share these images with each other, all the while; their parents will simply see the two sharing their art homework. This simple solution allows the two to freely communicate with each other without raising any suspicion.

Alice and Bob's simple example presents the major advantage of steganographic systems. By using such a system, two communicating parties are able to communicate through an observed channel without raising suspicion. This is achieved by embedding the secret message into an innocent looking carrier object, normally referred to as a *Cover-Object*. These cover-objects need to be innocent in the context of the channel they are being sent through, for example, if Alice and Bob had never drawn a picture in their lives, their sudden adoption of the fine arts may raise suspicion in itself.

1.1 Selecting a Channel

When considering a steganographic system, any mechanism for communication has the potential to be used as a channel to allow for information hiding. What determines this potential is whether or not the channel contains objects that can be used as carriers for our hidden information. These cover-objects determine the amount of data one can hide, as well as how often one can send hidden messages. Therefore the selection of a suitable channel is of the utmost importance.

Before the advent of the telephone, digital technologies or the internet, communication was referred to as physical in nature. If two people wished to communicate, they would need to physically stand in front of each other, or share an object that needed to be physically transported between the two parties. Examples such as letters, paintings, or photos all required that a third party physically transport the message from sender to receiver. Steganographic systems from that time therefore focused largely on hiding of information into parts of those objects that would raise the least amount of suspicion.

Everything from having the first letter in a word represent a character from your secret message to painting subtle detail into complicated scenes were used as a steganographic approach during an age where technology was limited and communication was slow and laboured.

Since the beginning of the digital era and the creation of the internet, our ability to communicate has been greatly enhanced; everything from emails to social networks has brought everyone in the world closer than ever before. With the development of these new technologies brings many new opportunities to allow for the incorporation of steganographic systems.

Since the internet represents such a huge collection of communication channels and networks, it would make sense to focus our attention at developing a steganographic scheme that allows for covert communications over such a larger network.

Looking through different internet statistics, we find that in 2009 alone, over 90 trillion emails were sent through the internet (The Radicati Group 2009). Granted about 81% of these emails were spam (The Radicati Group 2009), but this represents a huge set of possible cover-objects than can be used as carriers for hidden information. The only problem with using emails as covers for secret communication is the fact that emails, by nature, are private items. Emails sent between two parties, especially on a regular basis require an established relationship to avoid suspicion, and therefore, present a slight problem for anyone wishing to use them to communicate secret information with a stranger.

Another interesting statistic related to internet usage is social networks. These networks by design allow people to search for and connect with other people that search common interests, whether it is through friendship, a love of football, an interest in technology or anything at all. Social networks have grown rapidly over the past few years, none more so than Facebook.

According to Facebook published statistics, it is a network of over 400 million active users. This represents a growth of 100% in the period between April 2009 and February 2010. Further analysis of Facebook usage statistics shows that over 2.5 billion photos and images are uploaded to Facebook a month. This represents over 30 billion images uploaded to Facebook per year alone.

Obviously, this presents us with a huge set of potential cover-objects in the form of images that can be used as covers for secret communication. The added benefit is that social networks such as Facebook, due to their popularity, have indirectly provided a huge network where it is an accepted culture to share and upload photos, private or otherwise, for anyone to see.

This *'open to all'* nature of social networks plus the massive popularity of image based sharing provides us with the perfect cover-object for a digital era, namely: the digital image.

1.2 Images as Cover-Objects

Since images offer us a huge set of potential cover-objects as well as many potential channels for distributing them, (social networks and image sharing sites) they provide the perfect cover-object for our research into steganographic systems.

Images can be stored and transmitted in either natural, uncompressed form or they may be compressed using any one of a variety of image compression techniques. Since data on the internet is usually transferred through bandwidth-limited channels, images, which are naturally rather large files, tend generally to be compressed to save on bandwidth. In fact, the bulk of images presented on web sites are compressed. Uncompressed images tend to only exist when the image is first captured by the camera, and are rarely found uploaded on the internet. Taking this into consideration, we note that it is important to research and develop a steganographic system than can take advantage of how images are compressed and saved. It should also of course be relatively immune to the compression process so as to survive the compression and consequent de-compression with as little damage as possible.

The JPEG compression standard is a widely used and extremely popular compression algorithm. Referring back to the statistics published by Facebook, all images uploaded to their servers are compressed using the JPEG standard. This alone suggests that a huge set of images found on the internet are compressed using the JPEG compression standard and therefore, given the standard's popularity, it would be best to investigate JPEG based steganographic systems and research the effects of the compression algorithm on hidden information.

1.3 Research Problem Statement

The initial aim of this dissertation is to study the currently available research in the field of steganographic systems that hide in JPEG based images and to investigate the weaknesses and shortcomings of these current systems.

The final goal of the dissertation is then to propose new mechanisms for hiding in JPEG based images that avoid the weakness and improve upon the current steganographic schemes. The research will consider achieving these goals by increasing the number of randomization levels in the embedding process as well as designing new embedding techniques for increasing the capacity to hide in JPEG images.

1.4 Research Methodology

The aim of the research being to investigate image based steganographic systems, the approach is broken down into four clear phases:

1. The first stage in the research process is to develop an understanding of the science of steganography and the wider, overarching view of the subject as a whole. Once an understanding of the current terminologies and system structures has been achieved, a full

survey of current JPEG based image steganographic systems will be performed. The aim is to investigate and to determine the possible weakness and strengths of these systems.

2. Once a knowledge base has been gained on the subject of JPEG based image steganography, we shall attempt to design solutions for the weaknesses determined in the first phase of the research process. These solutions shall be then incorporated into a new JPEG based image steganographic system. This new system can build upon a currently available system that was researched or designed from the ground up if no system is currently available that shows promise for development.

3. The third stage of the research process is to prove the effectiveness of the newly designed system. This shall be determined through three major procedures:

    a. A simulation of the embedding performance of the new system shall be performed. The aim of this simulation is to determine the new system's embedding rate, embedding capacity as well as the expected error rates.

    b. The second procedure is to determine the system's performance to resist detection through a series of tests. The first test is a visual distortion survey which shall be run on a normal university campus to determine an average observer's ability to detect if any visual cues were left behind due to embedding. The results of the survey shall be analyzed to determine any statistical trends or conclusions that can be made. The second test that shall be conducted is a sequence of blind steganalysis tests, discussed in Chapter 3. These tests represent computer aided detection algorithms that aim at detecting distortions to higher order images features.

    c. We conclude the testing phase of the research process by performing a comparison in performance of the newly developed system as compared to currently available systems and attempt to present any performance enhancements achieved by the new system as compared to the older systems.

4. The fourth and final stage in the research process is to determine the success of the developed steganographic system based on the aims presented at the start of this section. We shall determine if the goal of improving the embedding capacity was achieved by reviewing the comparisons made during the third stage of the research process. We shall then compare the security performance of the new system compared to other similar steganographic systems.

1.5 Chapter Outline

Chapter 2 presents the history of the science of steganography. It presents a look at the evolution of the science from ancient times up till systems currently in use. The chapter also includes a formal definition of the science, expanding on its properties as well as the different approaches to classifying steganographic systems. The chapter concludes with a discussion of the differences between a steganographic system and a cryptographic system.

Chapter 3 is a survey of the development of image based steganographic systems with an aim of introducing current JPEG based steganographic systems. These systems are introduced and their motivations and processes explained. We then introduce currently available mechanisms for detecting these systems and present their weaknesses.

Chapter 4 details the development of the newly proposed system. It highlights major improvements developed to improve the system's randomness as well as increase its security. We then present a completely new embedding scheme that aims to trade off the possibility for higher errors for increased embedding capacity and security. We detail every stage in the development process presenting clear motivation for each stage.

Chapter 5 presents a performance analysis of the new system. We investigate the blocking performance of the new system as well as similar systems. We then investigate the embedding and error rates of the new system along with similar steganographic systems. We compare the embedding capacities and rates to investigate any potential improvements and areas of potential development. We then perform tests to investigate the new systems ability to resist detection and compare the systems performance to other systems.

We then conclude by presenting a roundup of the improvements achieved and comparing them to the original aim of the research. We also present any potential future work that has been raised through the process of investigating the new steganographic system.

1.6 Published works

In the course of performing our research the following papers have been accepted and published to conferences:

1. P. Dawoud, D. S. Dawoud, S. Mneney, B. Naidoo, "Information Hiding: An Introduction", Proc. Of MICSSA 2009 **(submitted & published)**
2. P. Dawoud, D. S. Dawoud, S. Mneney, R. Peplow, "Proposed Technique for Information Hiding", Proc. Of WMSCI 2009, Orlando, Florida **(submitted & published)**
3. P. Dawoud, D.S. Dawoud, R. Peplow, "A Proposal for Secure vehicular communications", Proc. Of ICCIT2009/ICIS2009, Seoul, Korea **(submitted & published)**

2 History of Steganography

2.1 Steganographic system development

As mentioned in the Introduction, the subject of steganography, although relatively young within the context of the digital domain, has been around for quite some time. The core concept of hiding information can be traced back for hundreds of years and early examples of its use can be found throughout history. This section will introduce two distinct phases in steganographic system development, namely Physical Steganography and Digital Steganography.

2.1.1 Physical Steganography

Before the age of computers, communication was largely physical in nature. From messengers to letters, as technology evolved, so too did the methods used to hide information. The first recorded steganographic systems were all physical in nature since most communication was physical as well.

The first recorded use of steganography was back in 440BC when Herodotus mentions two separate examples where secret messages were passed to warn of imminent attacks of Greece. The first example from Herodotus' *The Histories of Herodotus*, involved Demaratus sending a warning about a looming attack on Greece by inscribing directing on the wooden backing of the wax tablets before the beeswax was applied. These wax tablets were used at that time as reusable writing surfaces and an innocent message was inscribed on the wax covering the secret message.

The other example mentioned in Herodotus' writings involves Histiaeus, who shaved the hair off the head of his most trusted slave and tattooed a message warning against a Persian attack. Once the slave's hair grew back, he was sent to the King of Greece, who shaved to slave's head to retrieve the message.

Looking closer to current times, during World War 2, many examples have surfaced of steganography in use. A famous example is the use of microdots by espionage agents that contained secret information in a space no bigger than a period produced by typewriters. These microdots were mostly embedded into paper and normally covered by an adhesive. Another famous example of steganography used during times of war comes from the Cold War, where in 1968 the crew of the USS Pueblo was held prisoner by North Korea and managed to communicate using sign language during staged photo opportunities, informing the US high command that the crew were not defectors but were being held by the North Koreans.

Other examples include the use of invisible ink over innocent letters and writing secret messages on the back of stamps that were used to send innocent letters. A final, rather interesting example is the use of Morse code on knitting yarn and then knitting into a piece of clothing worn by a courier. By tying a sequence of knots into a yarn as though it was a sequence of Morse code, one could then proceed by using this yarn to knit a simple patch into any item of clothing.

2.1.2 Digital Steganography

With the advent of computers and more importantly the massive growth of the internet, digital steganography has advanced rapidly. In general one can say that information can be hidden in any source that contains redundant information. Based on this knowledge, many original digital steganographic systems hid their information in the least significant bit of an image or audio file, this bit generally adding a precision to the original that is unnecessary.

Other current methods of hiding information include hiding images within video material such that it can only be viewed if the video material is played at a slower speed or even split into frames. Other examples of hiding within images are by concealing information in the transformation domain of popular compression algorithms or by changing the statistical properties of the image to embed information.

Most recently, much research is being done into the hiding of secret information with the cloud and internet based structures. Examples of this including hiding information within unused header fields within Voice Over IP packets as well as hiding within padded bits in the TCP and UDP structure used over the IP structure.

2.2 Overview of a Steganographic system

The scientific study of steganography in open literature began in 1983 when Simmons (Simmons G.J. 1984) stated the problem in terms of communication in a prison. In his formulation, two inmates, Alice and Bob, are trying to hatch an escape plan. The only way they can communicate with each other is through a public channel, which is carefully monitored by the warden of the prison, Ward. If Ward detects any encrypted messages or codes, he will throw both Alice and Bob into solitary confinement. The problem of steganography is then: how can Alice and Bob cook up an escape plan by communicating over the public channel in such a way that Ward doesn't suspect anything “unusual" is going on.

The theoretical bases of information hiding are not yet well established. It is accordingly common to find many conflicting terminologies and notations being used. For this reason we start by giving the general model of hiding a message and use the model to present the terminology used throughout this thesis.

The general scenario for hiding messages is shown in the Figure 2-1.

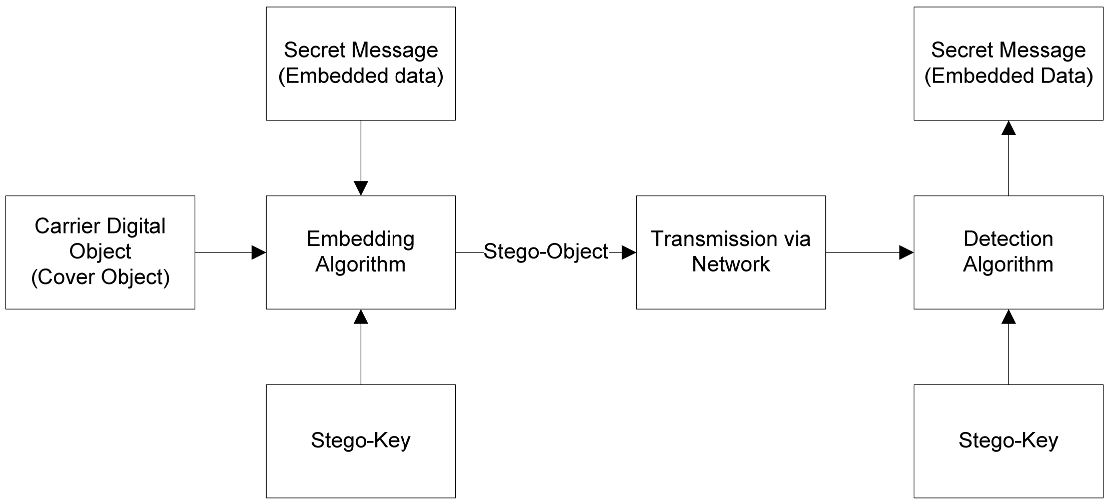

Secret Message (Embedded data)
Secret Message (Embedded Data)
Carrier Digital Object (Cover Object)
Embedding Algorithm
Stego-Object
Transmission via Network
Detection Algorithm
Stego-Key
Stego-Key

Figure 2-1: Model for of data hiding technique

In general, a system in which data is hidden in other data can be modelled as follows. The *embedded data* is the message we wish to secretly send. It is usually hidden in innocuous data referred to as the *cover-object*, such as *cover-text,* or *cover-image,* depending on the type of data used, producing a *stego-object*. A secret key known as a *stego-key* may be used to control the hiding process so as to restrict detection and recovery of the hidden data.

Within information hiding, there are three aspects that contend with each other:

- **Capacity:** refers to the amount of information that can be hidden in the cover-object.
- **Security:** refers to the difficulty an eavesdropper would have in detecting or accessing the hidden information.
- **Robustness:** refers to the amount of modification the stego-object can withstand before the hidden information is destroyed or rendered unusable.

Figure 2-2 shows a classification of Information Hiding (Anderson R.J and Pitzman B. 1996). In a general sense, Information Hiding can be divided into four main categories, namely Covert channels, Steganography, Anonymity and Copyright marking. Of the four mentioned above, Steganography and Copyright marking are very closely related.

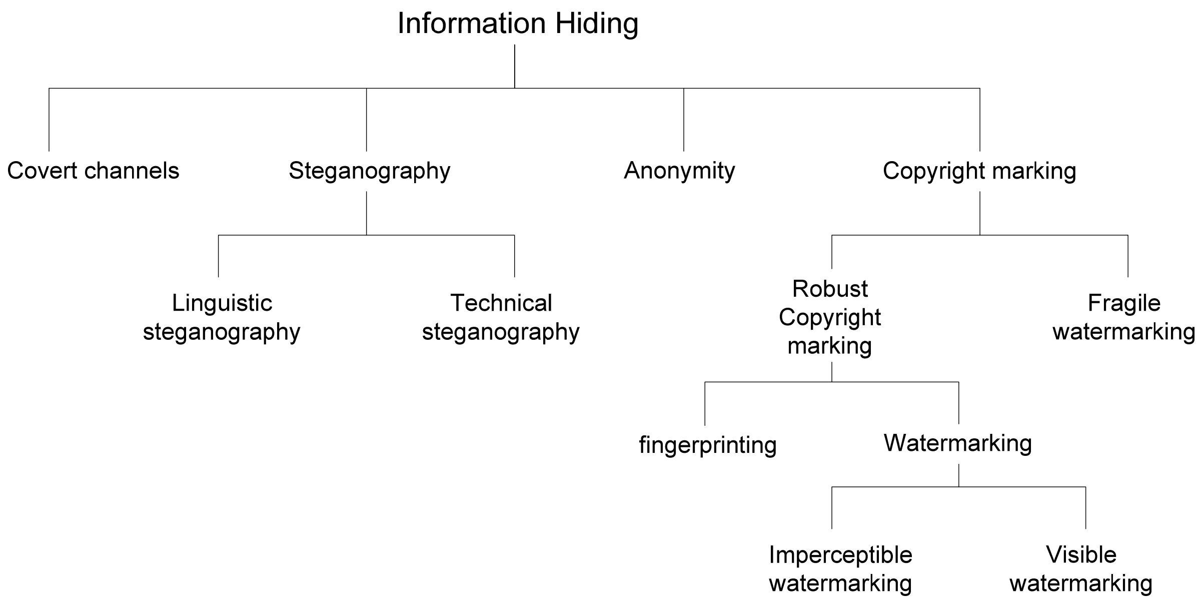


Figure 2-2: A classification of Information Hiding (Anderson R.J and Pitzman B. 1996)

Steganography is the art of hiding information in such a way as to supply covert communication between two parties without a possible attacker ever being aware of the communication's existence. A successful attack would first require that an attacker be able to detect the existence of this communication.

Modern Steganography attempts to be undetectable unless some secret information is known – a stego-key. This is similar to Kerckhoffs Principle in Cryptography (Kerckhoffs A. 1883). Therefore, for steganography to remain undetected, the unmodified cover-object as well as the stego-key must be kept secret.

Copyright marking on the other hand, has all the requirements of Steganography but has the added requirement of robustness against possible attacks. Although most Copyright marking is hidden, not all systems rely on hidden data, such as *visible digital watermarks.* With digital watermarking, the stego-object is normally referred to as the marked object. Different types of marks can be used depending on the application.

Fragile watermarks are destroyed as soon as the object is modified too much, helping to protect against the doctoring of the object, while *Robust marks* have the property that it is infeasible to remove them or make them useless without destroying the object at the same time. Authors also make a distinction between various types of robust marks. *Fingerprints* are likened to hidden serial numbers that allow owners of data to identify copies of their property while *Watermarks* tells us who the owner of the object is.

2.2.1 Definition of a Steganographic System

The word Steganography comes from Greek, meaning "covered writing" and as mentioned earlier, *Steganography* is the art and science of hiding information by embedding messages within another, seemingly harmless message (van Tilborg H.C.A 2005).

The obvious goal of such a system is to hide a message in such a way as to stop a possible attacker from ever discovering the very existence of the message. Therefore one can say a stego-system is perfectly secure if no decision rule exists that can perform better than a random guess at determining whether some image might contain an embedded message or not. As with Cryptography, the counterpart to Steganography is known as *Steganalysis* which has the goal of detecting hidden messages (Zeng W. et al 2006).

Before presenting a formal definition of Steganography, the following notions should be noted (Anderson R.J and Pitzman B. 1996):

- **Cover-object**, *c*: the innocent object in which we wish to embed our data. Objects include text and images.
- **Message**, *m*: this represents the embedded data we wish to secure. It is also *called stego-message*.
- **Stego-object***, s*: The cover object, once the message has been embedded.
- **Stego-key**, *k:* The secret shared between *sender* and *receiver* to embed and retrieve the message.

Similar to Cryptography, Steganography is divided into two separate processes, the Embedding process (analogous to encryption) and the Retrieving process (analogous to decryption). The embedding function *E* is a function that maps the triplet; cover-object *c,* message *m* and stego-key *k* to a stego-object *s*.

$$E(c,m,k) = s. \qquad (2.1)$$

The retrieving function *D* is a mapping from *s* to *m* using the stego-key *k*.

$$D(s,k) = m. \qquad (2.2)$$

In some stego-systems the original cover-object *c* may be used as input for the function *D,* but in that case it can be assumed that *k* = *c*||*k'* where *k'* is the secret key.

This process pair give rise to what is known as a *secret key steganographic system* defined as the quintuple $\mathfrak{S} = (C, M, K, E, D)$, where *C* is the set of possible cover-objects, *M* is the set of messages with $|C| \geq |M|$, K the set of secret keys, $E: C \times M \times K \ \rightarrow C$ and $D: C \times K \ \rightarrow M$ with the property that

$$D(E(c, m, k), k) = \ m, \qquad (2.3)$$

$$for\ all\ m \in M, c \in C\ and\ k \in K\ (Katzenbeisser\ S. et\ al\ 2000).$$

2.2.2 Properties of a Steganographic system

As mentioned earlier, most information hiding techniques have three, mainly conflicting, requirements: namely Capacity, Security and Robustness. In the case of Steganography, Capacity and Security are the primary concerns along with a new requirement called the system's *Transparency.*

In most practical steganographic schemes, the embedding process should produce a stego-object that is perceptually similar to the corresponding cover-object. We define this similarity between cover-object and stego-object as the system's *Transparency.* We define the transparency function of a steganographic system as (Katzenbeisser S. et al 2000):

Let C be the set of cover-objects. A function $tra: C^2 \rightarrow [0,1]$ *is called a transparency function on C, if given* $x, y \in C$

$$tra(x, y) = 1 \ \Leftrightarrow x = y, \qquad (2.4)\,,$$

$$and\ for\ x \ \neq y, tra(x, y) < 1.$$

Taking this definition: $tra(c, E(c, m, k)) \approx 1$ for all $c \in C$, $m \in M$ and $k \in K$.

Since messages can be embedded in different types of cover-objects, such as images and text documents, there are different ways to practically test transparency.

Transparency in the image domain between cover-images and stego-images is often computed using the peak signal-to-noise ratio (PSNR). Such a measure is totally objective and often it does not properly model the "human transparency perception". In the audio domain, PSNR is also commonly used to compute a transparency function. Another measure is the Objective Difference Grade (ODG). ODG is a neural network model of a subjective measure (Katzenbeisser S. et al 2000).

The *Capacity* of a stego-system measures the amount of information that can be embedded in the cover object *c* providing that $tra(c, E(c, m, k)) \approx 1$. Capacity is measured in bits per pixel (bpp) in images and in bits per second (bps) in audio. One can note that the higher the capacity of the stego-system, the higher the robustness of the system since more redundancy and error correcting codes can be used.

In most cases, a system with a high capacity level would be a rather attractive candidate for practical use but with an increase on capacity, there is almost always a decrease in transparency (Katzenbeisser S. et al 2000) hence these are competing design requirements.

The final important property of any stego-system is its *Security/Robustness*. The best way to understand security from a steganographic context is to consider the attacker's objectives, namely:

1. To detect the presence of a message in a stego-object.
2. To retrieve the message from the stego-object.
3. To overwrite the original message in the stego-object with a different message.
4. To disable the stego-message.

Clearly, from a steganographic point of view, if an attacker manages any of the listed objectives, the system would be compromised. Copyright marking, and in particular watermarking, is far more relaxed in its security and robustness requirements, only being concerned with attacks of type 3 and 4.

The four listed objectives can be grouped into one for three types of attacks:

Passive Attacks: the aim of these kinds of attacks is not to modify the stego-object but rather only detect the presence of an embedded message within the stego-object. These attacks represent the main threat to steganographic systems.

Active Attacks: these attacks aim at actively determining the contents embedded into a stego-object. The goal is merely to determine the embedded message but not change it in anyway.

Malicious attack: these attacks aim at manipulating the stego-object in order to affect the embedded message. The attack can aim at disable stego-message or to actively attempt to change the embedded message.

2.2.3 An information-theoretic model for Steganography

Cachin (Cahin C. et al 2004) proposed an information-theoretic model for Steganography for modelling security against passive attackers. The idea is to model the selection of the cover-object as a random variable with probability distribution P_C and the stego-object *s* = *E*(*c, m, k*) with probability distribution P_S (once probability distributions on *M* and *K* are fixed).

In this situation, the measure of relative entropy

$$D(P_C||P_S) = \sum_{q \in Q} P_C(q) log_2 \frac{P_C(q)}{P_S(q)}, \quad (2.5)$$

Where || represents concatenation.

This gives rise to the definition of a perfectly secure stego-system:

Let $\mathfrak{S}$ be a Steganography system, P_C the probability distribution of cover-objects and P_S the probability distribution of the stego-objects. $\mathfrak{S}$ is called ε-secure against passive attackers if

$$D(P_C||P_S) = \varepsilon, \quad (2.6)$$

and **perfectly secure** if $\varepsilon = 0$. (Katzenbeisser S. et al 2000)

The concept of a steganographic system being perfectly secure can be proven through the following proof:

Let C be the set of all bit strings of length n, P_C the uniform distribution of C and m the message to embed $(m \in C)$. The sender selects one $c \in C$ at random and computes $s = c \oplus m$. The resulting stego-covers s are uniformly distributed on C, so $P_C = P_S$ and $D(P_C||P_S) = 0$, and then the scheme is perfectly secure.

2.3 Classification of a Steganographic system

As with most systems available, there are many ways to classify steganographic systems. One such natural way of classification would be to distinguish between techniques based on the type of cover-object they use in their implementation. Cover-objects can range from text to images and even video and audio can all be used as cover-medium for embedding messages.

Another way to differentiate between techniques is based on the modification applied to the cover-object (Katzenbeisser S. et al 2000). Some of the more popular techniques are listed below and looked at in brief detail.

- Substitution techniques
- Transform domain techniques
- Spread spectrum techniques
- Statistical methods
- Distortion techniques
- Cover generation methods
- Public key steganography
- Masking and filtering techniques

2.3.1 Substitution techniques

These techniques substitute redundant parts of the cover-object with the embedded message. This was one of the first types of steganographic techniques used and thus includes many relevant methods including:

- Least significant bit (LSB) substitution
- Pseudorandom permutations
- Image downgrading
- Cover-regions and parity bits

Of the five substitution methods listed above, the least significant bit substitution is the method first associated with the field of Steganography. This system replaces the least significant bit of the cover-object with a bit from the embedded message as shown in Figure 2-3.

```
For $i = 1$ to $l(c)$ do
    $s_i \leftarrow c_i$
End for
For $i = 1$ to $l(c)$ do
    Compute index $j_i$ where to store $i$th message bit
    $LSB(s_{j_i}) \leftarrow m_i$
End for
```

Figure 2-3: Embedding Process for LSB substitution (Katzenbeisser S. et al 2000)

Note: The cover-object, *c* can be represented by a sequence of numbers c_i of length $l(c)$: $c = (c_1, c_2 \dots, c_{l(c)})$. c_i values can be binary (bits) {0, 1} or bytes [0,256]. The stego-object, *s* is also a sequence s_i of length $l(c)$. As before the stego-key will be denoted as *k.* Finally, the message embedded in the stego-object will be $m = (m_1, m_2 \dots, m_{l(m)})$ where we assume that $m_i \epsilon \{0,1\}$ for all $1 \leq i \leq l(m)$ (Katzenbeisser S. et al 2000).

With LSB substitution, the stego-system designer has to decide on how to select the index j_i where

```
For $i = 1$ to $l(m)$ do
    Compute index $j_i$ where the $i$th message bit is stored
    $m_i \leftarrow LSB(s_{j_i})$
End for
```

Figure 2-4: Retrieving process for LSB substitution (Katzenbeisser S. et al 2000)

the message m_i to be embedded. This can be computed in two different ways:

> ***Sequentially***: This method assumes that $l(m) < l(c)$ and message bits are embedded one after another into the cover-object. An example of such a system is JSteg.
>
> ***Pseudorandom***: This method uses a pseudorandom generator with the stego-key *k as the seed* to choose the $l(m)$ indices in *c*.

Finally, we note that LSB substitution, although results in a high capacity level as well as a high transparency, has very poor robustness properties since any sort of lossy compression or filtering would result in the embedded message being destroyed.

2.3.2 Transform domain techniques

To overcome the robustness vulnerabilities of substitution systems and to remain imperceptible for the human sensory system, stego-system designers moved from the spatial domain to the *transform space* to embed their secret messages.

The most common domains currently in use in steganographic systems include

- The discrete cosine transform domain (JPEG).
- The wavelet transform domain (JPEG2000).
- The discrete Fourier transform domain.

Of all the mentioned domains, the discrete cosine transform domain is by far the most popular.

2.3.3 Spread spectrum techniques

Spread spectrum (SS) communication technologies (Marvel L.M. et al 1999), which find a wide use in military applications, spread a signal using a code which is independent of the data being sent. That same code can then be used on the receiver side to de-spread the signal and recover the data. Such technologies provide communications with low probability of interception and anti-jamming properties.

SS technologies applied to embedding schemes provide good robustness due to the difficulty of removing the spread signal. Currently there are two variants of SS that are used in information hiding related applications, they are:

Direct-sequence: the signal is spread by a constant called the chip rate and is modulated using a pseudorandom signal.

Frequency-hopping: the carrier frequency is altered (in a random process) describing a wide range of frequency values.

A general framework for spread spectrum Steganography was presented by Smith and Comiskey (Smith J.R et al 1996):

$$C = \{ c \in (N \times M) - \mathit{greyscale\ image} \}$$

$$S = \{ s \in (N \times M) - \mathit{greyscale\ image} \}$$

$$m = \{ m_1, m_2 \dots, m_{l(m)} \}$$

$$K = \{ \emptyset \in C, \mathit{for}\ 1 \le i \le l(m)\ , \mathit{where}\ \ \emptyset_i\ \ \mathit{are\ orthogonal} \}$$

$$\langle \emptyset_i, \emptyset_j \rangle = \sum_{x=1}^{N} \sum_{y=1}^{M} \emptyset_i(x,y)\, \emptyset_j(x,y) = G_i \delta_{ij}. \quad (2.7).$$

Where, as before, *C* is a set of all possible cover objects, *S* a set of all possible stego-objects and *K*, a set of all possible stego-keys.

From this framework, we present the embedding process for SS (Katzenbeisser S. et al 2000):

Before insertion, codify the message m as follows:

$$M(x,y) = \sum_{i}^{l(m)} m_i \emptyset_i(x,y). \quad (2.8)$$

The next step is to compute the sego-object in the following way:

$$s(x,y) = c(x,y) + M(x,y). \quad (2.9)$$

On the receiver side, the message can be retrieved as follows (Katzenbeisser S. et al 2000):

Every message bit m_i can be recovered without the cover-object:

$$m_i = \frac{\langle s_i, \emptyset_i \rangle}{G_i} = \frac{\langle c_i + M_i, \emptyset_i \rangle}{G_i} = \frac{\langle c_i \emptyset_i \rangle + \langle M_i \emptyset_i \rangle}{G_i}$$

$$= \frac{\langle \sum_j^{l(m)} m_j \emptyset_j, \emptyset_i \rangle}{G_i} = \frac{\sum_j^{l(m)} m_j \langle \emptyset_j, \emptyset_i \rangle}{G_i} = \frac{G_i m_i}{G_i} = m_i$$

2.3.4 Statistical methods

Statistical methods embed information by changing the statistical properties of the cover-object and use hypothesis testing in the extraction process. To embed a 1, the statistical characteristics of the cover-object are significantly changed, and the cover-object is unchanged to embed a 0.

Such methods are 1-bit steganographic schemes in the sense that only one bit can be embedded. However, they can be easily extended to embed $l(m)$ bits once the cover-object is divided into $l(m)$ disjoint blocks $B_1, \ldots, B_{l(m)}$.

The extraction function is the following hypothesis-testing function:

$$f(B_i) = \begin{cases} 1 & block\ B_i\ was\ modified\ in\ the\ embedding\ process \\ 0 & otherwise \end{cases}$$

The problem with using such a function is with how to construct it. We know that if there exists a function $h(B_i)$ which depends on the cover-block B_i and the distribution of $h(B_i)$ (for the unmodified B_i) is known, we will be able to test if $h(B_i)$ exceeds a specific value and then if the block B_i has been modified. The main problem is thus finding a good test statistic $h(B_i)$. (Katzenbeisser S. et al 2000)

2.3.5 Distortion techniques

These techniques store information by distorting the signal and measure the deviation from the original cover-object in the extraction process. Such techniques need the cover-object in the extraction process. Text steganography is often based on distortion techniques, below are a few examples of types of Text steganography.

- Line-space encoding: the line position in the document is moved up in order to encode $m_i = 1$ and not changed if $m_i = 0$.
- Word-space encoding: horizontal spaces between selected words of the cover-text are altered. Right align concerns!

In most cases of distortion based text steganography, retyping any stego-text will remove the message and thus cover generation based methods are far more suited for Text based steganography.

2.3.6 Cover generation methods

Cover generation methods generate an object only for the purpose of being a cover for secret communication. One of the most popular cover generation methods can be traced back to World War I, when the Germans communicated covertly using what is known as a null-cipher (Kahn D. 1996). A null-cipher is a set of predetermined characters and words, along with an accompanying set of rules which are used to read the hidden message. The rules were generally of the type, "Read every third letter in every other word." These null-ciphers therefore required the generation of cover-text that would allow for a particular message to be hidden. An obvious problem with generating cover-text in this fashion is that in most cases it is rather hard to generate a cover-text, using the predetermined set of words, that doesn't raise suspicion.

Another cover generation method is known as *mimic functions*. Mimic functions (Wayner P. 1992) can be used to hide the identity of a message by changing its statistical profile in a way so that it matches the profile of an innocent looking cover-text. The final stego-text traditionally has a statistical profile that fits the original cover-text, thus not raising any suspicion. Since stego-texts generated using mimic functions have a profile identical to the normal text, mimic functions tend to be very resilient against statistical attacks. Since mimic functions effectively produce meaningless sequences of characters or words, they tend to be very suspicious and thus require further processing to achieve a more meaningful stego-text. To produce such a meaningful text, mimic functions can be extended using context-free grammars.

Synonym Based steganography is also an example of cover generation. This system was developed by Chapman M. et al (1997) and involves the use of two functions called NICETEXT and SCRAMBLE. These functions use a huge dictionary and embed data by using synonym substitution on the generated cover-text. What this process manages to do is preserve the meaning of the cover-text every time embedding is done thus solving the obvious problem of generated cover-texts that are linguistically poor, such as mimic functions. The biggest flaw of this approach is the lack of a huge substation base. There are only so many synonyms per word and thus it limits the number of times a given word can be used to hide data. Thus an adversary, observing different messages with similar means will start to suspect the innocence of the messages.

A fourth example of cover generation steganography is what is known as *Noise Based* steganography. The underlining principle of these types of systems is the introduction of seemingly innocent errors or noise into the generated cover-object. A simple example of such systems can be seen in (Topkara et al 2007). What this approach effectively requires is the introduction of typos and ungrammatical abbreviations in the generated cover-text to hide the secret message without raising suspicion. This approach relies heavily on the new, "Internet" or "SMS" speech that people all around the world have started to utilize.

A final system developed and published in 2009 by (Desoky A. 2008) suggests a new system called NoStega, or Noiseless Steganography which relies on the generation of subject specific content. The premise is rather straight forward. Assume Alice and Bob are two spies wishing to communicate. The system proposes that a professional relationship is established between them, such as Alice being Bob's professor and using that to generate innocent cover-text that is legitimate within the context of a teacher-student relationship. Using such cover-text will most likely remove most if not all the suspicion that may exist in the eyes of an attacker, but to protect against even the most scrupulous monitor, secrets are exchanged in the form of test questions or answers to tests, with a rule set, much like null-ciphers, determining which words within the cover represent message data.

The same author (Desoky A. 2008) suggested a further way of hiding information within the context of a teacher-student relationship. He suggests the use of graphs and its data as a way to introducing potential to hide information.

2.3.7 Public key Steganography

According to Lenti J. (2000) *Public Key Steganography* requires the pre-existence of a shared secret key to designate pixels which should be tweaked. Thus both the sender and the receiver must have this secret. The idea of private/public key pair doesn't work since the eavesdropper can use the public key to sabotage the whole affair.

2.3.8 Masking and filtering techniques

In sharp contrast to other embedding techniques, the *masking or image-adaptive techniques* embed information to perceptually significant areas of the image. The use of 'significant parts' make these techniques very robust. Masking refers to the phenomenon where a signal can be imperceptible to an observer in the presence of another signal referred to as the masker (Lin E.T et al 1999). The phenomenon of camouflage is a manifestation of this human weakness. The image must be analyzed in advance for the information to determine appropriate regions to place the message data so that it is *camouflaged* in the environment.

2.4 Difference between Cryptography and Steganography

Although both aim to achieve the same general goal, each approaches this goal with a different approach in mind. This section is a discussion of the two sciences in general terms with a focus of highlighting the major differences between the two sciences.

As mentioned before, steganography is informally defined as the art and science of **hiding information**. In general this is achieved by hiding our secret information in an innocent looking cover-object, whether it is text, images or anything else for that matter. The clear focus of steganography is therefore to hide our secret information in such a way as to not raise the suspicion of anyone monitoring the communication.

With this in mind, one can say that the aim of steganography is the securing of communication between two parties by insuring that a third party will never detect that the communication is taking place.

Similarly, cryptography has the aim of securing a communication between two parties by insuring that a third party will never understand the meaning of the communication. This implies that although the goal of cryptography is to conceal the meaning of what is being sent, the presence of the communication is still present. In other words, when two parties communicate and rely on cryptography to protect their communication, the message's meaning is concealed but any third party is still aware of the communication taking place.

Herein rests the difference between the two sciences. Both aim to protect communications between two parties, but one goes about it by hiding the meaning of the content being sent (cryptography), while the other aims to conceal the communication altogether (steganography). As should be obvious at this point, steganography, by design aims to minimize suspicion, allowing two parties to communicate without drawing unwanted attention. Cryptography on the other hand, almost always raises suspicion if the two communicating parties would not naturally be communicating or if it used within a domain were it raises trust issues.

2.5 Summary

This chapter presented the historical developments of the science of steganography as it evolved through the ages. Before the digital age, steganographic systems focused on the use of physical communication systems to allow for covert communication. Examples such as microdots and invisible ink represented the start of modern steganography.

With computers growing in popularity and the massive expansion of the internet, digital steganography became the focus of study. We introduced the concept of utilizing digital content, such as images, videos and even the cloud itself as a cover for covert communication.

We followed by introducing the concept of a steganographic system and introduced the key components making up a stego-system, namely: the embedded message, referred to as the stego-message; the cover-object, representing the object in which our stego-message is embedded; the stego-object, the object corresponding to the cover-object with our stego-message embedded; and finally the stego-key, a shared secret key between the sender and receiver that is used to embed and decode the stego-message. Following the breakdown of a stego-system, we introduced a formal definition of a stego-system by introducing the quintuple $\mathfrak{G} = (C, M, K, E, D)$.

We continued by introducing the properties of a stego-system, defining the concepts of *Capacity, Robustness, Security* and *Transparency*. Every stego-system is defined by the amount of data it allows a user to embed within a cover-object, the *capacity* of the system; it is also defined by its *transparency*, which is a measure of how comparable a stego-object is to the cover-object used in the embedding process. We then defined the concept of an ε-secure stego-system, which a measure of a stego-systems *security*.

We then finished the chapter by introducing a range of classifications for steganographic systems as well as explaining the core differences between a cryptographic system and a steganographic system.

The chapter to follow presents a review of all currently developed image based steganographic systems with an aim of understanding and analyzing the strength and weakness of current JPEG based image steganographic systems.

3 A Review of Current, Image Based Steganographic Systems

The previous chapter focused on introducing steganography as a whole, introduced the subject and presented its general properties and theory. This chapter now presents the relevant literature which was studied in order to understand the current image based methods available. The first section of introduces the Discrete Cosine Transformation (DCT) process as well as the Quantization process, both used in the JPEG compression scheme. These two processes are used heavily in all JPEG based image hiding techniques and thus are presented to refresh the reader's understanding. The second section of this chapter then covers each of the reviewed image based steganographic systems as well as the current means by which they are detected.

3.1 Review of the DCT and Quantization Processes

The JPEG image compression system is a commonly used lossy compression algorithm. It is normally used to store images on computers and on the internet and due to its rather high compression to quality ratio, images can be compressed to factions of their original size with little loss of overall image quality.

In the author's opinion, it could be argued that one of the dominating forces in the boom of the Internet could be the JPEG compression system. Its relative simplicity and huge saving in image size allow for the use of images on bandwidth restricted networks and enabled the internet to come alive with the colours and sights of photos and images. Although newer compression systems have been developed since the first creation of the JPEG standard, it is virtually impossible to find a website on the internet that does not contain a JPEG compressed image and thus it is of great interest to anyone interested in hiding information within images, whether online or locally.

Although the complete compression algorithm is not important within the context of this study into image based steganographic systems, it's DCT and Quantization stages are. For the purpose of this explanation, we shall present a simple image and work through the DCT process and quantization process as though we were performing a JPEG compression on the image.

Although images are traditionally represented as Red-Green-Blue (RGB) images, the JPEG process requires that we convert any colour image from the RGB domain to what is known as $Y'C_bC_r$ colour space. This represents images as Y', the luminance or brightness of pixels, and$C_b$and C_r; the chrominance or the split between blue and red colour components. Once an image has been converted to this new colour space, the next step is the blocking of each colour component of the image into 8 by 8 pixel wide grids. The remainder of the JPEG compression process is performed on these individual grids.

To explain the concept of the Discrete Cosine Transform and the Quantization process, we refer to the image show in Figure 3-1. Since this is a greyscale image, there is no need to convert from one colour space to another. Therefore the first step is to grid the image into 8 by 8 pixel wide grids.

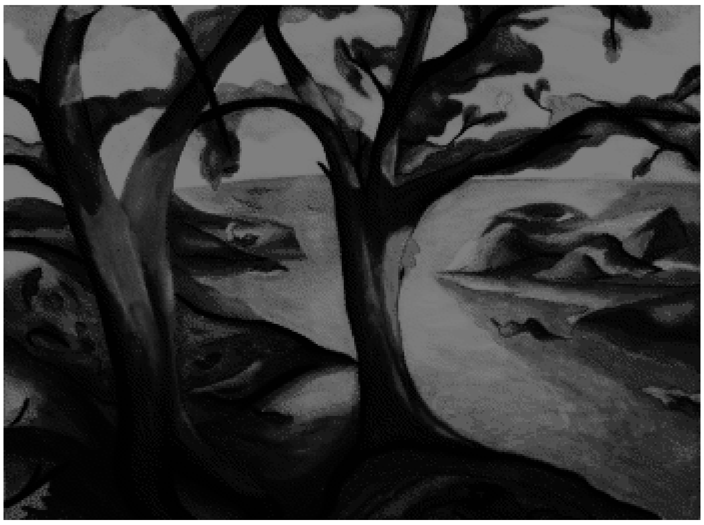
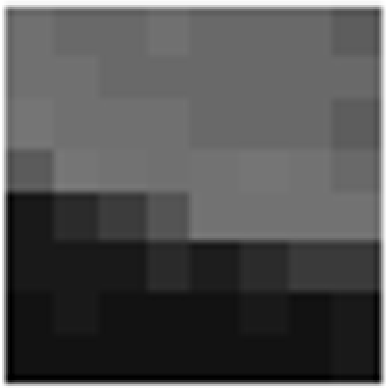


Figure 3-1: Example Image (greyscale)



112	105	105	112	105	105	105	93
112	112	105	105	105	105	105	105
117	112	112	112	105	105	105	93
91	117	114	112	114	117	114	105
25	43	60	84	114	114	114	114
25	25	25	42	29	43	58	58
18	25	18	18	18	25	18	25
18	18	18	18	18	18	18	25

Figure 3-2: 8 by 8 pixel crop from original image (Left: Visual presentation of pixels. Right: Pixel values)

We now present the Discrete Cosine Transformation and Quantization processes used in the JPEG compression algorithm using the cropped 8 by 8 pixel image shown in Figure 3-2.

3.1.1 Discrete Cosine Transform

Although the DCT transform stage is not the first stage in the JPEG compression algorithm, it is the stage responsible for the reduction of the pixel intensity information. Traditionally, the original pixel information is referred to as the perceptual or spatial domain and the DCT process transforms the pixel information into the frequency domain. This transformation process is defined within the JPEG standard as the two-dimensional, discrete cosine transform performed on each 8 by 8 grid of pixel values. These values are first normalized around zero by subtracting 128 from the byte sized pixel values.

The DCT of the original 8x8 signal $F(u,v)$ is defined as:

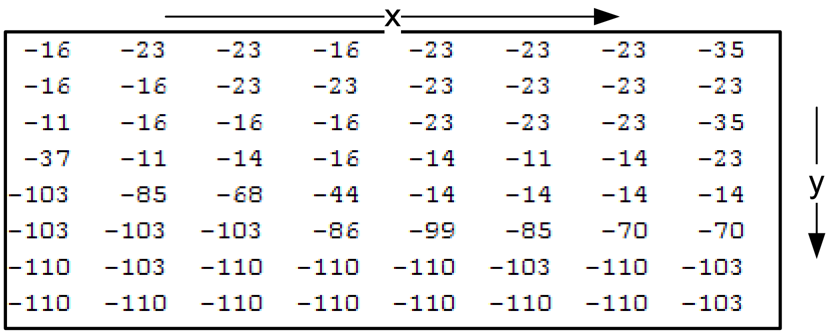
$$F(u,v) = \frac{1}{4}C(u)C(v)\sum\nolimits_{x=0}^{7}\sum\nolimits_{y=0}^{7} f(x,y)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right). \quad (3.1)$$

And the inverse DCT, $f(x,y)$ is defined as

$$f(x,y) = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)F(u,v)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right), \qquad (3.2)$$

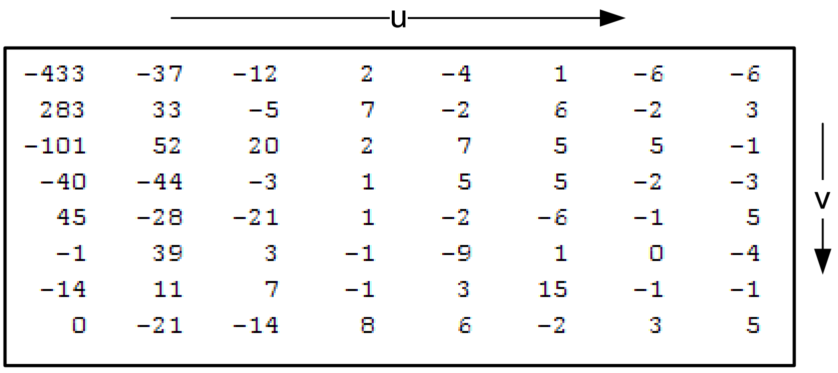
$$where\ C(z) = \begin{cases} \frac{1}{\sqrt{2}} & if\ z = 0, \\ 1 & otherwise. \end{cases}$$

Figure 3-3 shows a normalized 8 by 8 grid before the 2D DCT is applied. Figure 3-4 shows the result of the transformation process. A few elements of the resulting matrix should be noted. The first is that the upper left coefficient of the matrix in Figure 3-4 is referred to as the DC coefficient. Further coefficients represent higher frequency components as we move horizontally to the right and /or vertically down with the lower right coefficient representing the magnitude of the highest frequency component.



-16	-23	-23	-16	-23	-23	-23	-35
-16	-16	-23	-23	-23	-23	-23	-23
-11	-16	-16	-16	-23	-23	-23	-35
-37	-11	-14	-16	-14	-11	-14	-23
-103	-85	-68	-44	-14	-14	-14	-14
-103	-103	-103	-86	-99	-85	-70	-70
-110	-103	-110	-110	-110	-103	-110	-103
-110	-110	-110	-110	-110	-110	-110	-103

Figure 3-3: Example 8 by 8 pixel block before 2D Discrete Cosine Transform is performed



-433	-37	-12	2	-4	1	-6	-6
283	33	-5	7	-2	6	-2	3
-101	52	20	2	7	5	5	-1
-40	-44	-3	1	5	5	-2	-3
45	-28	-21	1	-2	-6	-1	5
-1	39	3	-1	-9	1	0	-4
-14	11	7	-1	3	15	-1	-1
0	-21	-14	8	6	-2	3	5

Figure 3-4: Example 8 by 8 pixel block after performing 2D DCT (results rounded to nearest integer)

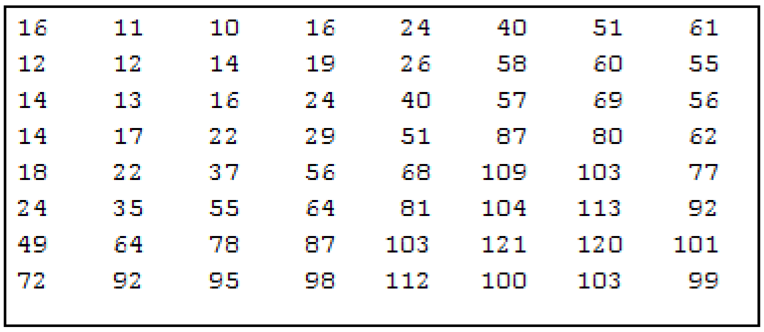
With an understanding of what is meant by the perceptual domain and frequency domain as well as the functions used to move from each domain, we can now look at the quantization process which follows the DCT process and understand the method used in designing the quantization matrix.

3.1.2 Quantization Process

The purpose of the quantization stage in the JPEG compression scheme is to introduce redundancies within the JPEG grid than can be ignored during the encoding stage; all the while making sure that a reasonable image quality is retained. The DCT process results in coefficients that represent the magnitudes of the different frequency components.

Through many tests and general observations is was determined that the human eye is far more sensitive to high frequency changes in images but that changes in low frequency components had a greater overall effect on surrounding pixel detail and colour. It was then decided in the standard that when quantizing the JPEG grid was to occur, the quantization was to be more accurate for lower frequency coefficients and far more aggressive the higher the frequency of the coefficient.

The standard Quantization matrix, as defined within the JPEG standard (JPEG Standard) can be found in Figure 3-5. One can note that on average, the quantization value increases as they move along the main diagonal of the matrix with off diagonal components increasing as the move closer to the main diagonal. The reason for the lack of symmetry shown in Figure 3-5 is due to the average energy distribution of a normal image not being perfectly symmetric (Konrad M. et al 2009). This lack of symmetry was discovered through the sampling of hundreds of thousands of JPEG grids from normal images and their energy distribution being notably symmetric but not perfectly so (Konrad M. et al 2009). Therefore the value of each quantization component in the quantization matrix is not symmetric, i.e. $Q(1,2) \neq Q(2,1)$.



16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

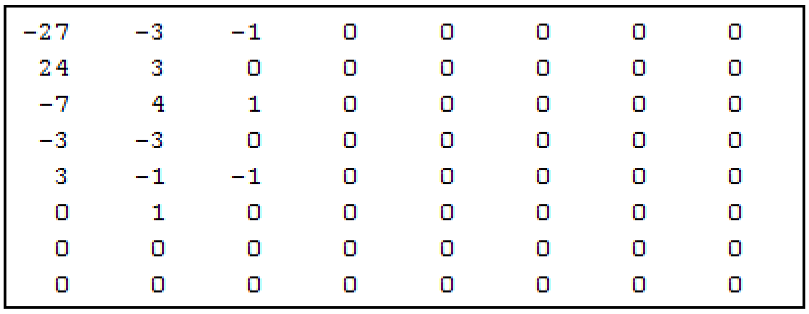
Figure 3-5: Quantization Matrix as defined in the original JPEG standard

With an understanding of the Quantization matrix, we can now look at the quantization process itself. The process takes every coefficient C_{ij}, resulting from the 2D DCT, and divides it by the quantisation integer, q_{ij} with the result being rounded to the nearest integer.

The result of the quantization step B_{ij} is defined by:

$$B_{ij} = round\left(\frac{C_{i,j}}{q_{i,j}}\right) for\ i = 0,1,2, \qquad \ldots,7; j = 0,1,2,\ldots,7 \quad (3.3)$$

Performing the quantization step on Figure 3-4 using the original JPEG standard quantization matrix, we get the result shown in Figure 3-6.



-27	-3	-1	0	0	0	0	0
24	3	0	0	0	0	0	0
-7	4	1	0	0	0	0	0
-3	-3	0	0	0	0	0	0
3	-1	-1	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Figure 3-6: Resulting 8 by 8 blocks after quantization has been performed

With an understand of the function of the Discrete Cosine Transform and the Quantization process used in the JPEG compression, we now present a survey of image based steganographic systems that present the development of JPEG based steganography.

3.2 Review of Image Based Systems

This section introduces some of the more famous and popular, image based steganographic schemes. It covers the dominant literature on steganography and steganalysis discovered during the course of this project. It then concentrates on JPEG based stenographic schemes in particular and provides an in-depth survey of the steganalysis techniques used to detect and attack them.

3.2.1 Simple Hiding

One the earliest image base information hiding schemes is the simple scheme of embedding the hidden message in the image by merely setting the least significant bit (LSB) of each pixel to match the embedded message. This is illustrated in Figure 3-7. The embedded data is first represented in binary form, and once ready, the least significant bit (LSB) of each pixel value is changed sequentially to match the embedded message.

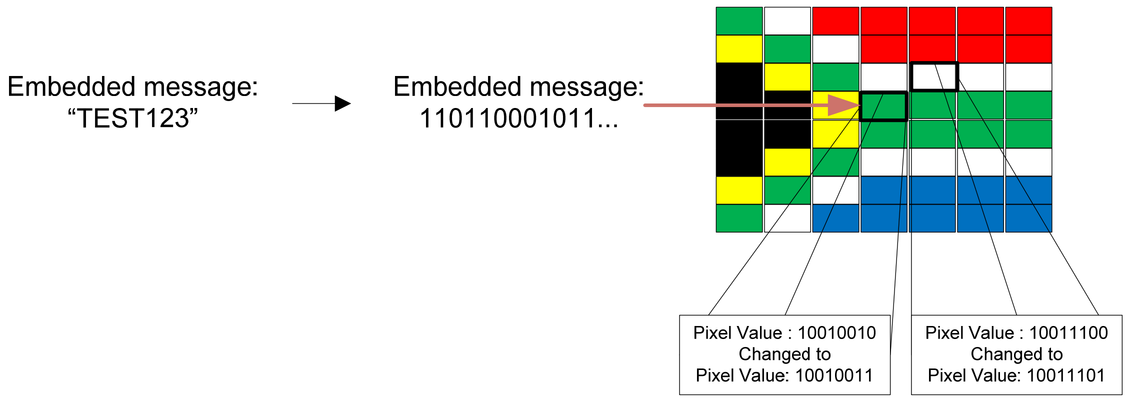


Figure 3-7: Example of embedding use the Simple Hiding Scheme

The use of only the LSB is not a requirement of the scheme and should the user wish to increase the capacity of the scheme, the two (or even three) least significant bits can be used. An obvious drawback of being more aggressive with one's embedding is that the image starts to distort and any attacker monitoring the channel will start noticing artefacts in the images being sent; this will undoubtedly cause suspicion.

Figure 3-8 shows an example of the simple hiding scheme in use where one complete image is hidden in another image of the same size. Figure 3-8a shows the stego-image with the embedded image data saved in the two least significant bits of the cover-image's pixel data. Figure 3-8b shows the extracted data maintained in the same layout as the original cover image. As we are using 8 bits per colour per pixel for these images, the two bit extracted data as shown in image b) is not viewable, being around 85 times darker than it should be. Figure 3-8c remedies this problem by scaling up the values of each embedded data pixel value by 85 resulting in our hidden image being clear to see.

Figure 3-8: Example of hiding using the Simple Hiding Scheme

The simple hiding scheme, although simple and straightforward to implement, has a few draw backs. The first major issue with the scheme is its lack of robustness. If an image was to be compressed or filtered in any way, any information hidden within the image will not survive the process. Another issue is the introduction of visual distortion at high embed-rates resulting in the embedded data bleeding through the cover. Figure 3-9 shows an example of this "bleeding" where the image of the cat is embedded more aggressively in the image of the tree. A final issue to note is that since the embedded information is embedded sequentially, there is no need for a shared secret stego-key. This means that anyone can decode the message from the stego-image and thus understand the message. Within the context of the goals of steganography, this is of no concern, but from a practical sense, it makes the system a lot less suitable for handling sensitive information.

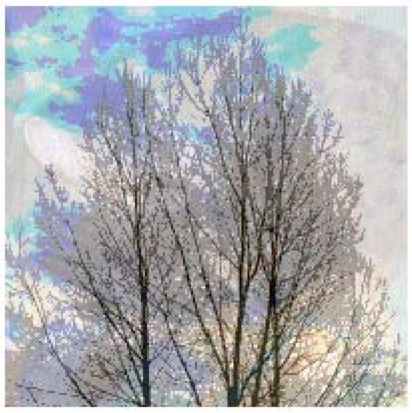


Figure 3-9: Example of bleeding when high embedding rates are used

That said the simple hiding scheme has one major advantage. Assuming the attacker has no access to the original cover-image and the embedding rate is kept low, it is virtually impossible for them to have a better than guessing chance at determining if information has been hidden within the image. This is due to the fact that if the LSB of every pixel is effected, no statistical model can be created to model the original cover image and thus an attacker will have no way of telling if the image has been affected by embedding.

Although the simple hiding scheme is useful for hiding data within images, its lack of robustness really hurts the system. In most cases, images are sent through channels that either require compression or inject noise into the image, both of which will damage the embedded message. Thus the use of this simple system is no longer considered very practical.

3.2.2 JSteg

JSteg was the first publicly available steganographic system for JPEG images (Provos N. et al 2003). Created by Derek Upharm, JSteg aims to solve the robustness problem of the simpler embedding scheme mentioned before. Instead of embedding the secret information sequentially throughout the pixel data, the embedding algorithm sequentially replaces the LSB of the DCT coefficients with the message's data.

By moving from the perceptual domain to the frequency domain, JSteg manages to not only solve the robustness issue of the simple hiding scheme but also manages to remove the concern of the bleeding effect. Figure 3-10 shows the embedding strategy for the JSteg scheme (Provos N. et al 2003).

```
Input: secret message, cover-image
Output: stego-image
While data left to embed do
        Get next DCT coefficient from cover-image
        If DCT≠0 and DCT≠1 then
                Get next LSB from secret message
                Replace DCT LSB with message LSB
        End if
        Insert DCT into stego-image
End while
```

Figure 3-10: JSteg embedding algorithm (Provos N. et al 2003)

However, as JSteg changes the LSBs sequentially, it causes a detectable distortion. Andreas Westfeld and Andreas Pfitzmann noticed that for a given image embedding high-entropy data (like encrypted data) changed the histogram of colour frequencies in a predictable way. This can be shown using a simple case (Provos N. et al 2003). Assume colours are addressed by their index, i within the colour table. We now refer to their respective frequencies before and after embedding high entropy data in their LSBs as $n_i \ and \ n_i{}^*$. Now assuming the embedded message has a uniform distribution, an assumption one can easily make for encrypted data, we note that if $n_{2i} > n_{2i+1}$, then we note that pixels with colour 2*i* are changed more frequently to colour 2*i*+1 than pixels with colour 2*i*+1 being changed to 2*i*. This results in the following relation:

$$|n_{2i} - n_{2i+1}| \geq |n_{2i}{}^* - n_{2i+1}{}^*| \quad (3.4)$$

In other words, assuming we embed uniformly distributed data, the frequency difference between adjacent colours is reduced. This concept can be translated to JPEG based data and to the DCT coefficients generated during the DCT process. We therefore note that embedding encrypted data sequentially has the effect of decreasing the difference between adjacent coefficients and therefore a χ^2- test can be used to determine whether the observed frequency distribution in an image matches a distribution that shows distortion from embedding hidden data.

Although the attacker does not know the cover-image, they do know that the sum of adjacent DCT coefficients remain invariant (Provos N. et al 2003), which leaves the attacker to compute the expected distribution ${y_i}^*$ from the stego-image. Now let n_i represent the DCT coefficient histogram, we can compute the arithmetic mean to determine the expected mean

$$ {y_i}^* = \frac{n_{2i} - n_{2i+1}}{2} \quad (3.5) $$

and compare it against the observed distribution (Provos N. et al 2003)

$$ y_i = n_{2i}. $$

The χ^2- value is defined as

$$ \chi^2 = \sum_{i=1}^{v+1} \frac{(y_i \quad - {y_i}^*)}{{y_i}^*}, \quad (3.6) $$

where v are the degrees of freedom, i.e. one less than the number of different categories in the histogram. With the χ^2-value, we can calculate the probability *p* that the two distributions are equal using the following cumulative distribution function,

$$ p = 1 - \int_0^{\chi^2} \frac{t^{(v-2)/2}\, e^{-t/2}}{2^{v/2}\, \Gamma(\mathrm{v}/2)}, \quad (3.7) $$

where Γ is the Euler Gamma function.

By calculating *p*, an attacker not only has the ability to detect the presence of sequentially embedded data but also the length of the embedded data. This makes JSteg extremely insecure since it is effortlessly detectable. That said, it did present a brilliant stepping stone for the development of JPEG based steganographic systems and started a new type of science.

3.2.3 OutGuess 0.1

OutGuess 0.1 was created by Niels Provos and is a steganographic system that improves the embedding step by using a pseudo-random number generator (PRNG) to select DCT coefficients at random. The LSB of these selected DCT coefficients are replaced with encrypted message data. This randomisation results in the χ^2-test being ineffective in detecting embedded data since changes no long effect sequential DCT coefficients but rather random ones. Figure 3-11 shows the embedding strategy used to embed the secret message into an image.

```
Input: secret message, cover-image, secret shared key
Output: stego-image
Initialize PRNG using secret shared key as seed
While data left to embed do
        Get pseudo-random DCT coefficient from cover-image
        If DCT≠0 and DCT≠1 then
                Get next LSB from secret message
                Replace DCT LSB with message LSB
        End if
        Insert DCT into stego-image
End while
```

Figure 3-11: OutGuess 0.1 embedding algorithm (Provos N. et al 2003)

OutGuess 0.1 improves upon JSteg in two important ways. The first introduces a secret shared key (stego-key).This key is used as a seed for a PRNG which selects random DCT coefficients which combats the use of the previously presented χ^2-test. The second is that unlike JSteg, the embedded message is now protected by the stego-key and can now only be read by a receiver who knows the key. This makes OutGuess 0.1 much more attractive for use with sensitive data.

However, it is possible to extend the χ^2-test to become more sensitive to local distortions in an image. Instead of sampling the entire image searching for a distortion, we use a constant sample size but slide the position of where the samples are taken over the image's entire range. The biggest issue with this extended χ^2-test is how to determine an appropriate sample size to be used in the detecting. As detailed in (Provos N. et al 2003), a threshold value for the χ^2-value is determined beforehand and a binary search is used on the sample size to determine a sample size that matches the previously determined χ^2-value.

The designer of OutGuess, Niels Provos, later showed that by applying corrective transforms to the embedding step, one could completely defeat the extended χ^2-test (Provos N. 2001). It was later suggested by Siwei Lyu and Hany Farid that an approach based on discrimination of two classes, namely stego-images and non-stego images would allow for better detection of pseudo-randomly hiding encrypted data (Lyu S. et al 2002). This represents the first mention of a blind steganalysis approached in literature and is detailed later in the chapter.

3.2.4 F5

With steganalysis successfully managing to detect systems that embed in the LSB of the DCT coefficients, Andreas Westfeld proposed the steganographic system he called F5. Instead of utilizing the LSB of the DCT coefficients, Andreas' F5 proposed the embedding of data by decrementing the absolute value of the DCT coefficients in a process called *matrix encoding* (Westfeld A. 2001). As a result of the new embedding strategy, there is no relation between fixed pairs of DCT coefficients and thus a χ^2-test or even the extend χ^2-test cannot detect F5.

The above mentioned Matrix encoding computes an appropriate $(1, 2^k - 1, k)$ Hamming code by calculating the message block size k from the message length and the number of nonzero non-DC coefficients. The Hamming code $(1, 2^k - 1, k)$ encodes a k-bit message word m into an n-bit code word with $n = 2^k - 1$ (van Lint J.HH. 1992). Since this is a Hamming code, this encoding technique allows for the recovery of a single bit in error in a code word. Figure 3-12 below shows the F5 embedding algorithm.

```
Input: secret message, cover-image, secret shared key
Output: stego-image
Initialize PRNG using secret shared key as seed
Permutate DCT coefficients with PRNG
Determine k from image capacity
Calculate code word length $n \leftarrow 2^k + 1$
While data left to embed do
        Get next k-bit message block
        Repeat
                $G \leftarrow$ {n non-zero AC coefficients
                $s \leftarrow$ k-bit hash $f$ of LSB in G
                $s \leftarrow s \oplus k - bit$ message block
                If s ≠0 then
                        Decrement absolute value of DCT coefficient $G_s$
                        Insert $G_s$ into stego-image
                End if
        Until s = 0 or $G_s \neq 0$
        Insert DCT coefficients from G into stego-image
End while
```

Figure 3-12: F5 embedding algorithm (Provos N. et al 2003)

The full detail of the above mentioned Matrix Encoding can be found in [*6,*7, 11] but for completeness, the decoding scheme is presented below:

> *"F5 uses the decoding function $f(a)$ and the Hamming distance d. With matrix encoding, embedding and k-bit message into any n-bit code word changing it at most by one bit. In other words, we can find a suitable code word a' for every code word a and every message word m so that $m = f(a')$ and $d(a, a') \leq 1$. Given a code word a and message word m, we calculate the difference $s = m \oplus f(a)$ and get the new code word as"* (Provos N. et al 2003)

$$a' = \begin{cases} a & if\ s = 0 \\ (a_1, a_s, \dots, \neg a_s, \dots a_n & otherwise \end{cases} \quad (3.8)$$

Although F5 is not detectable using the χ^2-test, the authors of (Fridrich J. et al 2002) succeeded to break F5. The authors of (Fridrich J. et al 2002) recognized two measures that can be used; one of the measures is directly related to F5 and the second is common for all algorithms that modify the JPEG quantized DCT coefficients to embed data bits.

a. <u>**Number of zero coefficients**</u>: F5 decrements the absolute value of the non-zero coefficients to embed the message. As a result all the coefficients that have an absolute value of 1 will become zero as a result of embedding. This increases, accordingly, the number of zero coefficients and decreases the non-zero coefficients.

b. **Blockiness**: This is a statistical measure of discontinuity at the boundaries of the 8x8 JPEG grids. This measure increases for any steganographic technique that modifies the quantized DCT coefficient to embed any data bits. Blockiness is also used to attack OutGuess 0.1 algorithm as detailed in (Fridrich J. et al 2002). Since F5 uses these 8x8 JPEG grids, accordingly it is expected that it will suffer from the discontinuity at the boundaries. This makes it possible to use blockiness as a measure to attack F5 (Fridrich J. et al 2002).

According to (Fridrich J. et al 2002), the main difficulty with breaking F5 was finding the baseline value of the estimate of the number of DCT coefficients that have zero value as well as the estimated value of blockiness. What follows is the detailed analysis of the approach used in attacking F5.

3.2.4.1 Description of the attack

The authors of (Fridrich J. et al 2002) used the normal philosophy of attacking any scheme;

1. Find a statistical parameter that changes with the hiding of a secret message: In this case the authors selected a statistical parameter β that is correlated to the number of coefficients changed due to embedding the message. β represents the modification rate that indicates if F5 has modified an image.
2. Find the baseline value of the selected statistical parameter: The authors of the attack estimate the cover-image histogram from the stego-image and start to compare the value of β of the estimated cover with that of the actual one to decide whether F5 is used or not.

The following explains how the authors of (Fridrich J. et al 2002) succeeded to break F5. In the explanation we use the same assumptions and symbols used in (Fridrich J. et al 2002).

Let:

$h(d), d = 0,1,\ldots$ be the total number of AC coefficients in the original cover-image with absolute values equal to *d* after the image has been compressed inside the F5 algorithm.

$h_{kl}(d)$ be the total number of AC DCT coefficients corresponding to the frequency (*k, l*), $1 \le k,l \le 8$ in the histogram of the cover-image, whose absolute value is equal to *d.*

$H(d)$ be the total number of AC coefficients in the histogram of the stego-image with absolute values equal to *d* after the image has been compressed inside the F5 algorithm.

$H_{kl}(d)$ be the total number of AC DCT coefficients corresponding to the frequency (*k, l*), $1 \le k,l \le 8$ in the histogram of the stego-image, whose absolute value is equal to *d.*

The total number of non-zero AC coefficient (*P*) is given, accordingly, by the equation:

$$P = h(1) + h(2) + \cdots$$

Assume the F5 embedding process results in changing *n* AC coefficients. Then the modification rate (the probability that a non-zero AC coefficient will be modified) is

$$\beta = n / P$$

Since F5 changes coefficients pseudo randomly, H_{kl} , h_{kl} and β are related by the equations (Fridrich J. et al 2002)

$$\begin{aligned} H_{kl}(d) &= (1-\beta)h_{kl}(d) + \beta h_{kl}(d+1), && for && d > 0, \\ H_{kl}(0) &= h_{kl}(0) + \beta h_{kl}(1) && for && d = 0 \end{aligned} \qquad (3.9)$$

The authors of (Fridrich J. et al 2002) used these estimates to calculate the expected changing rate β from the cover-image histogram. They found that the best value of β is the value that minimizes the square error between the stego-image histogram H_{kl}, and the expected value $\widehat{H}_{kl}$ calculated from the estimated histogram $\widehat{h}_{kl}$ calculated using equation (3.9):

$$\beta_{kl} = \arg\min_{\beta}[H_{kl}(0) - \widehat{h}_{kl}(0) - \beta\widehat{h}_{kl}(1)]^2 + [H_{kl}(1) - (1-\beta)\widehat{h}_{kl}(1) - \beta\widehat{h}(2)]^2 \qquad (3.10)$$

The least square approximation of equation (2) leads to:

$$\beta_{kl} = \frac{\widehat{h}_{kl}(1)[H_{kl}(0) - \widehat{h}_{kl}(0)] + [H_{kl}(1) - \widehat{h}_{kl}(1)][\widehat{h}_{kl}(2) - \widehat{h}_{kl}(1)]}{\widehat{h}^2{}_{kl}(1) + [\widehat{h}_{kl}(2) - \widehat{h}_{kl}(1)]^2} \qquad (3.11)$$

Note: (Fridrich J. et al 2002) used $h(0)$, $h(1)$ in the histogram because the experience the largest change during embedding.

We then calculate the final value of β as an average value over a number of selected values of the frequency (*k, l*), mainly: $(k,l) \in \{(1,2),(2,1),(2,2)\}$

The authors used β to estimate the secret embedded message.

3.2.5 Yet Another Steganographic System (YASS)

YASS, a scheme proposed by (Solanki K. et al 2007) aims to solve the weakness of current JPEG schemes to blind steganalysis methods. This method is based on embedding data in randomized locations so as to disable the self-calibration process popularly used by *blind steganalysis* schemes.

Before discussing the embedding strategy involved with YASS, we present the definition of *Blind Steganalysis* and the goals of such systems. Following these definitions we present the YASS embedding strategy and review how the designers of YASS combat blind steganalysis.

3.2.5.1 Blind Steganalysis

Blind Steganalysis algorithms bank on the fact that steganographic techniques that use matching, restoring or modelling can result in matching marginal statistics but such techniques cannot stop some other image features from being modified during the embedding process. Blind steganalysis algorithms use such modified features as the input of a learning machine to detect the existence of a hidden message and also to determine the steganographic algorithm used.

For such blind algorithms to succeed, it is crucial to select those features that are very sensitive to embedding changes, but at the same time not sensitive to image contents. This needs, in turn, a good model for natural images against which the suspected stego images can be evaluated. Due to the absence of such a universal model, at least until now, the use of self-calibration methods are proposed (Dabeer O. et al 2004). The calibration process uses the stego image to estimate the statistics of the cover image.

As stated in (Solanki K. et al 2007, Dabeer O. et al 2004), the key components to a successful blind steganalysis system are:

> *"1.* ***Self-calibration mechanism: A*** *Calibration process is used by the blind steganalysis schemes to estimate the statistics of the cover image from the stego image. For JPEG steganography, this is typically achieved by decompressing the stego image to the spatial domain followed by cropping the image by a few pixels on each side and compressing the image again using the same compression parameters.*
>
> *2.* ***Features capturing cover memory:*** *Most steganographic schemes hide data on a per-symbol basis, and typically do not explicitly compensate or preserve statistical dependencies. Hence, features that capture higher dimensional dependencies in the cover symbols are crucial in detecting the embedding changes. Cover memory has been shown to be very important to steganalysis (Sullivan K. et al 2006), and is incorporated into the feature vector in several ways, e.g. (Fu D. et al 2006, Shi Y.Q. et al 1974).*
>
> *3.* ***Powerful machine learning:*** *Use of powerful machine learning techniques and training with several thousand images ensures that even the slightest statistical variation in the features is learned by the machine."*

The above discussions lead to the following conclusion: to avoid blind steganalysis the steganographer may use one (or all) of the following approaches:

- To find and use steganographic algorithms that must embed information into cover images in such a way that the embedding process does not result in significantly perturbed image features.
- Instead of trying to preserve the feature vectors of the image, the steganographer can concentrate on finding data embedding schemes that result in distorting the capabilities of the steganalyst from estimating the statistics of the cover image.

Some of the approaches that can be used to achieve the second suggested approach include:

1. **Hiding with high embedding strength:** The idea is to embed the data in such a way as to destroy the cover image as much as possible such that the steganalyst cannot derive any reliable statistics from the stego-image. This approach was tested by Kharrazi et al (Kharrazi M. et al 2006) and proved its effectiveness.
2. **Randomized hiding:** This technique is based on disabling the capabilities of the steganalyst from estimating the cover statistics. Randomizing the embedding approach can achieve that. The spatial location of hiding, the choice of transform domain, the quantization matrix and embedding method are some examples of parameters that may be randomized.

Recently steganographic techniques operating in the Matching Pursuit (MP) domain have been proposed (Cancelli G. et al 2006) as a possible way to reduce the detectability of a stego object against blind steganalysis. The MP approach banks on the fact that blind steganalysis does consider the semantic contents of the host message, accordingly, it is expected that embedding the stego-message at a higher semantic level will tend to improve the un-detectability. The use of such an idea is however currently facing problems of computational complexity and instability of the decomposition.

YASS uses the randomising approach to confound blind steganalysis. In this case, the random approach is as shown below:

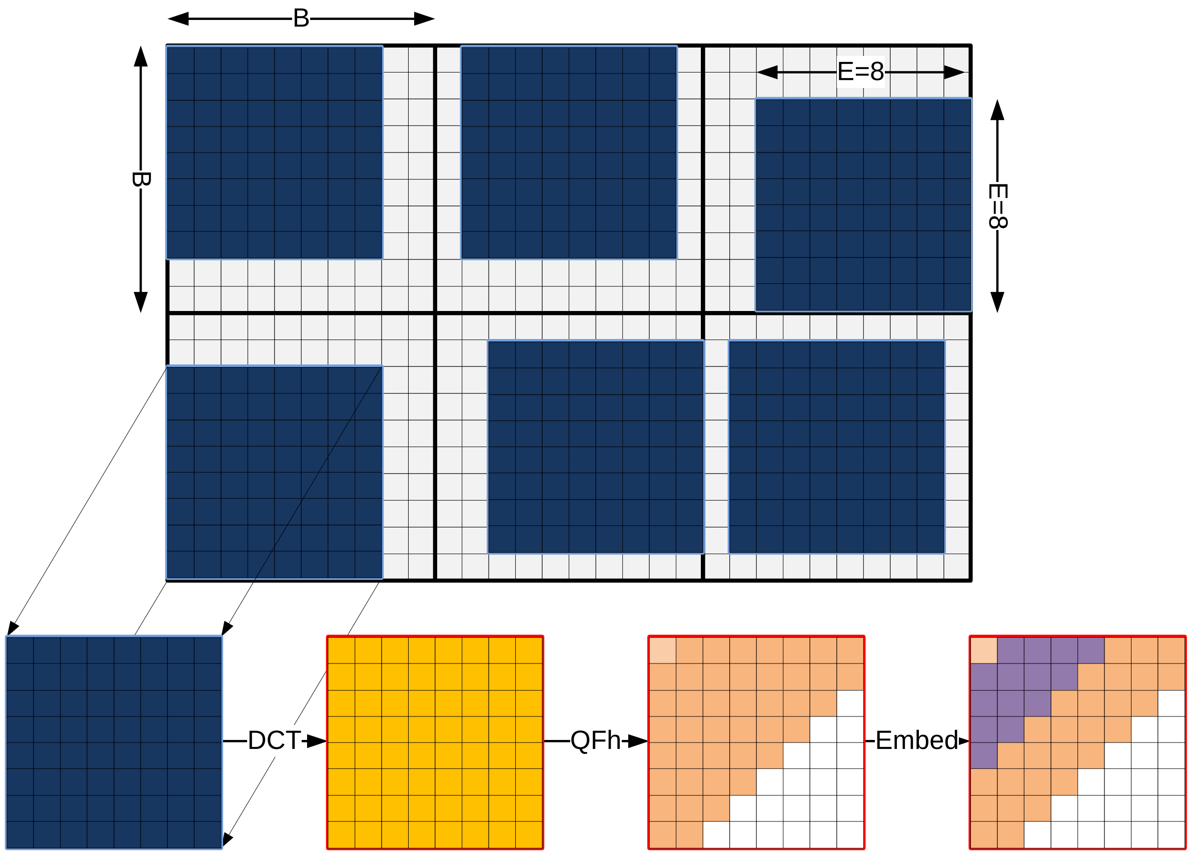
3.2.5.2 Embedding Algorithm

The YASS embedding algorithm is divided into five distinct stages:

1. The entire message is first encoded using a Repeat-Accumulate (RA) error correcting code.
2. The image, in its spatial domain, is divided into big blocks (B blocks) of size B x B, with B >8.
3. Once the B x B blocks have been determined, a 8 x 8 E block is pseudo-randomly placed with each B-block, with the location determined by a secret shared stego-key.
4. The following procedure is then run on each 8 x 8 E block:
    a. A 2D DCT is performed on the E block
    b. Every resulting DCT coefficient is divided by its corresponding quantization number determined from a predefined quantization matrix, Qf_h, normally referred to as the hiding quality factor. No rounding is performed as this point.
    c. A segment of the encoded message is now embedded into the 19 lower frequency AC coefficients using Quantization Index Modulation (QIM). All coefficients that have been quantized to zero are skipped.
    d. The E block is then decompressed back to the spatial domain and returned to its location in the B block.
5. The image is then compressed using JPEG with an advertised quality factor of $Qf_a$to obtain the stego-image.

One should note that as a result of stage 5 in the embedding process, there is a requirement for the aggressive error handling code used in the first stage. Since the E block and thus the embedding do not rest on the JPEG 8 x 8 grids, there is a good chance that information hidden may be damaged when the JPEG compression occurs. (Solanki K. et al 2007)

Figure 3-12 shows the embedding algorithm strategy although the decompression stages have been left out.



🟦	Original Pixel Intensity value	🟪	DCT coefficients changed due to embedding
🟨	DCT coefficients	🟧	Unchanged AC coefficients
🟫	DC coefficient	⬜	AC coefficients quantized to zero

Figure 3-13: YASS embedding Algorithm

A further modification to the YASS embedding algorithm was introduced by the authors of (Solanki K. et al 2007) in a bid to increase the embedding capacity of the system as a whole. These modifications include (Sarkar A. et al 2008):

1. The inclusion of multiple Qf_h used during the embedding stage. The Qf_h used during the quantizing of each E block is selected randomly from a list of available Qf_h or selected adaptively based on the variation or number of DCT coefficients eligible to embed in with the E block.
2. By iteratively re-embedding the secret message, the overall error rate can be reduced and thus allowing for a potentially higher embedding rate.

3.2.5.3 Advantages of YASS

As mentioned before, YASS uses a shared secret stego-key during the embedding process. Much like its use in OutGuess 0.1 and F5, this key allows for more randomization to be introduced into the embedding system while insuring that the hidden message can only be extracted by a receiver knowing the key.

The embedding strategy, aiming to combat detection by blind steganalysis, manages to introduce the two feature sets detailed above. By introducing the random location of the E blocks and with the introduction of the modifications, a higher embed rate, YASS achieves the two suggested approaches to defeating blind attacks.

3.2.5.4 Analysis of possible weaknesses of YASS

As mentioned before, F5 was introduced to increase the embedding capacity while keeping the security high, but steganalysis succeeded in breaking F5. YASS was designed to be very secure but, as mentioned correctly by the authors, reduces the embedding capacity. The two reasons behind the reduction of the embedding capacity are:

- Some real estate of the image is wasted by using bigger blocks to hide only in the DCT coefficients of an 8x8 block
- Secondly, since the embedding grids are not in coincidence with the natural 8x8 JPEG grids, an error detecting code must be used to correct the errors in the received data (Solanki K. et al 2007).

Concerning the security, we expect that breaking YASS will be possible through coefficient analysis due to the following two factors:

3.2.5.4.1 The number of zero DCT coefficients:

YASS uses the QIM technique to hide the message. If the data bit to be hidden is *m,* and the DCT coefficient before modifying it is *x* and after modifying it is *y*, QIM scheme is described by the equations:

$$y = x + \Delta - (x + \Delta) \bmod 2\Delta \quad \text{For } m = 0,$$

and

$$y = x + \Delta - (x) \bmod 2\Delta \quad \text{For } m = 1,$$

Where Δ is the quantization step.

Using the above equations, it is clear that any un-rounded coefficient that has value in the range $-\Delta < x < \Delta$ will be rounded into zero. This means that using QIM for modulating the coefficients will increase the number of zero DCT coefficients. This was one of the main parameters used to break F5.

3.2.5.4.2 Possibility of identifying the positions of the 8x8 E-blocks:

Using fixed size B-blocks to contain the E blocks means that reaching the 8x8 blocks used to hide the data does not add excessively to the steganalysis, especially if the value of B is known. The possible positions of the 8x8 E blocks in any B block of size B x B is (B-8+1) x (B-8+1). The rest of locations are impossible since none of them allow for the accommodation of an E block without overlapping the neighbouring B blocks. This means that the probability of identifying the host E block within a B block is equal to $1/(B-8+1)^2$. This attack in presented in chapter 4 when discussing the proposed system.

This result means there is a high possibility of finding the host locations in the image and accordingly breaking the system if B is known before hand. Since the value of B is determined by the algorithm and not by some unknown key, it is wise to assume that it is known to the attacker.

3.3 Summary

This chapter presented a review of the most popular, image based Steganographic techniques and reviewed their embedding procedures as well as looking at possible ways of detecting those systems. The first, and most basic of the Steganographic schemes was the Simple Hiding scheme. This scheme looked towards hiding stego-messages into the LSB of pixel data within the image. This scheme was not restricted to using the LSB of pixel data but as presented, more aggressive embedding results in clear visual distortion of the cover-image raising suspicion. This simple hiding scheme has a major advantage over other image based schemes as, providing embedding aggression is kept low but every LSB present in the image is effected, no statistical model can be created that can model the original cover-image statistic since the image's complete feature set has been effected (Kharrazi M. et al 2006) by the embedding process.

The next scheme discussed was JSteg, the first JPEG based Steganographic scheme. Although the move to embed stego-messages in the frequency domain resulted in stego-images that showed little distortion, JSteg's lack of a secret stego-key and its sequential embedding strategy resulted in a rather insecure system. OutGuess 0.1 aimed to address JSteg's shortcomings by introducing a shared secret stego-key, used as a seed to a PRNG, resulting in a pseudo-random embedding scheme that combated JSteg's weakness to the χ^2-test. Ultimately though, an extended χ^2-test was devised (Provos N. et al 2003) that is capable of detecting OutGuess 0.1 embedding.

Since the concept of LSB embedding used in previous JPEG based Steganographic systems directly aided in detecting these systems, Andreas Westfeld (Westfeld A. 2001) proposed the steganographic system he called F5. Instead of utilizing the LSB of the DCT coefficients, Andreas' F5 proposed the embedding of data by decrementing the absolute value of the DCT coefficients in a process called matrix encoding. F5 was a rather attractive scheme with a high embedding capacity and resistance to statistical acts such as the extended χ^2-test. J. Fridrich *et al.* (Fridrich J. et al 2002) managed to design a statistical attack that took advantage of F5's embedding scheme. Attempting to predict a statistical model of the cover-image based on only the features of the stego-image allows for the creation a threshold marker that could be used as a baseline for testing for the present of stego-information.

Finally we presented the newly created *"Yet Another Steganographic System"* (YASS) framework. This system is a framework that allows for embedding of steganographic messages in such a way as to remain undetectable to blind steganalysis systems. The introduction of Big blocks (B-blocks) that offset embedding from the JPEG 8 by 8 compression grid that was used in previous JPEG based schemes and the introduction of the pseudo-randomly placed Embedding blocks (E-blocks) allowed for a higher level of diffusion throughout the image resulting in current, blind steganalysis methods having no better than a guessing chance to detect the presences of steganographic material. Through some smart analysis on YASS's embedding strategy and the effect of QIM as the embedding, it s possible to detect whether or not a stego-message was embedded into an image using YASS.

The next chapter will introduce the proposed system and present its structure and strategies as well as focus on the expect gains of using our proposed Multi-level randomization scheme.

4 Proposed JPEG Steganographic System

The proposed Steganographic system presented in this chapter represents an evolution of design. Originally based on the YASS framework presented in Chapter 3, this proposed system titled Multi-Block, has evolved through three distinct phases, each working towards solving the inherent weakness of YASS as discussed in Chapter 3 as well as aiming to improve aspects of the framework to allow for both a higher security in a steganographic context and an improvement in the embedding capacity of the system.

The chapter is divided into three main sections. The first introduces how the Multi-Block system was designed to improve upon YASS by introducing new layers of randomization to the YASS framework. The Multi-Blocking system evolved through two separate phases and the first section details both. The second section in this chapter focuses on addressing the embedding issues of the YASS framework and presents a solution to YASS's embedding problem. The final section summarises the core differences between the proposed system and YASS and further details some of the hypothesised improvements in security and expected embedding capacity.

4.1 Increased Randomization of Blocking

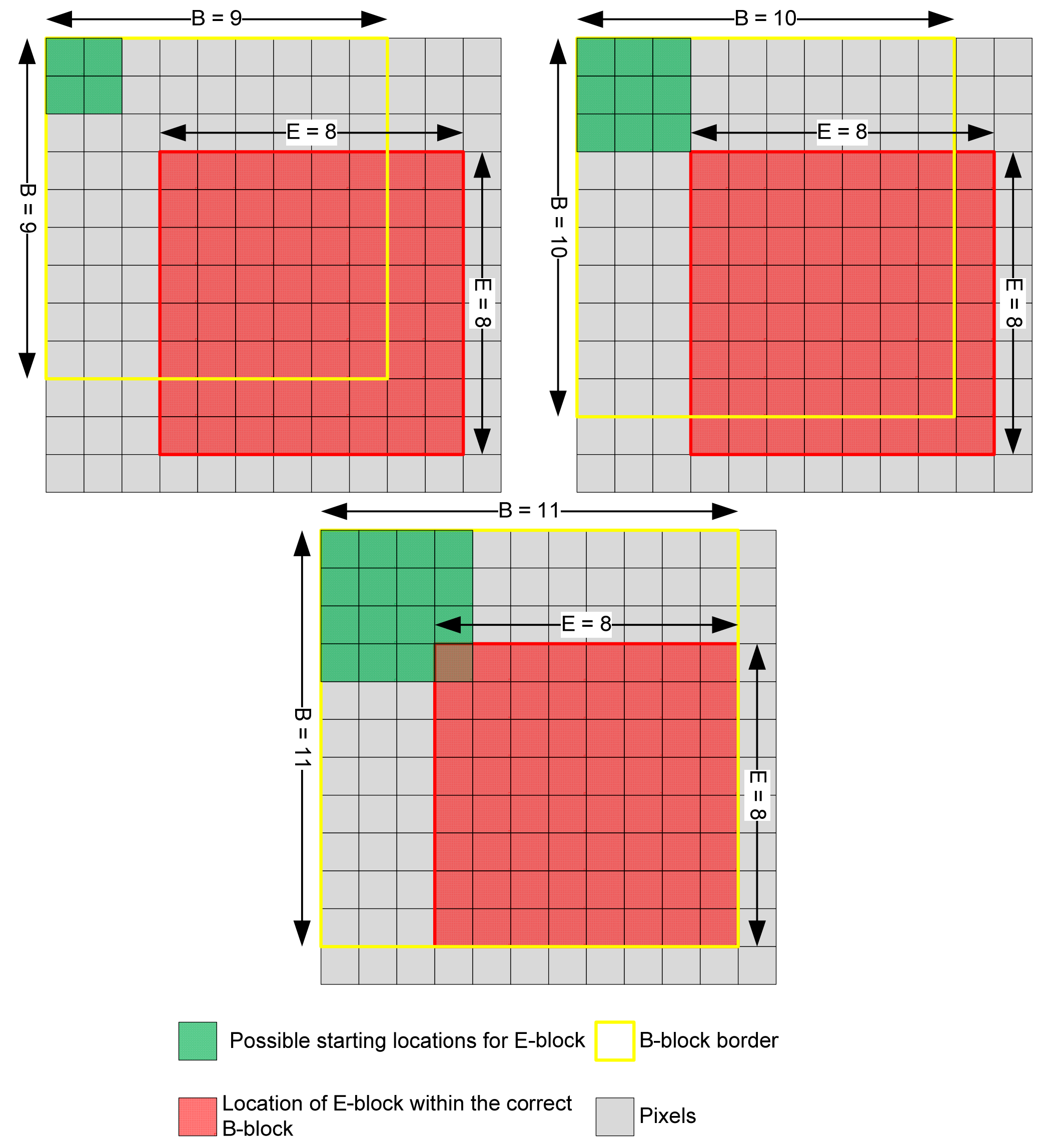
In Chapter 3, the issue of the possibility of identifying the position of the E-blocks within the YASS framework was discussed. The location of these blocks, although determined by a Pseudo-Random Number Generator (PRN), is not impossible to determine. Since E-blocks are not allowed to overlap two separate B-blocks, the possible locations of their position can be determined. Take the example presented in Figure 4-1.

Assuming an attacker has a Steganalysis method that allows for the detection of saved data using QIM as defined in the YASS framework, the challenge for an attacker is to localize the E-block so as to perform this analysis to determine if any tampering can be detected. The fact that an E-block can only be in a small finite set of locations given a B-block size works in the attackers favour, allowing them to crop all possible locations and test for tampering.

In Figure 4-1 the attacker would have to crop all possible locations in which an 8 by 8 grid of DCT coefficients (the E block) can fit within a B-block. At this point, the attacker may have knowledge of the agreed upon B-block size or he may not. Figure 4-1 assumes the attacker does not possess this information and will have to do things the hard way. Starting with a B-block size of 9, the attacker will try and crop all possible 8 by 8 grids that can fit completely within that B-block. After failing with a B-block size of 9, the attacker now tests for a B-block of size 10. This time, the attacker needs to crop a total of nine possible E-blocks, but once again, fails in finding the complete block. Finally, the attacker tests by cropping E-blocks from a B-block of size 11. Cropping all sixteen possible locations, the attacker finally finds the correct E-block with his analysis.

At this point, the attacker has two important pieces of information, namely:

1. He is now aware of the B-block size used throughout the embedding process, whereupon he may, if he so wishes, actively change or destroy the hidden information,
2. More importantly he has detected the presence of the potential of hidden information and thus can confirm his suspicions with further testing of B-blocks size 11.



B = 9
E = 8
B = 9
E = 8
B = 10
E = 8
B = 10
E = 8
B = 11
E = 8
B = 11
E = 8
Possible starting locations for E-block
B-block border
Location of E-block within the correct B-block
Pixels

Figure 4-1: Cropping the B Block to find E Block in YASS

This simple example illustrates one of the two major issues with the YASS framework. Although the location of the E-block is controlled through a seed and a PRNG, the small number of possible locations can aid attackers in detecting the system.

4.1.1 Randomization of B-blocks

The proposed system introduces an added layer of randomization to the YASS framework by using the shared stego-key as a seed to a PRNG which determines a pseudo-random sequence of B-block sizes. This effectively means that B-blocks are no longer the same size throughout the embedding process and can vary within a range of agreed values.

This randomization decreases the possibility of an attacker breaking the system using the cropping attack mentioned previously. The attacker is now faced with the task of determining the B-block size for each and every B-block used and thus increases the number of iterations required to potentially detect tampering. The suggested increase in randomization is show in Figure 4-2.

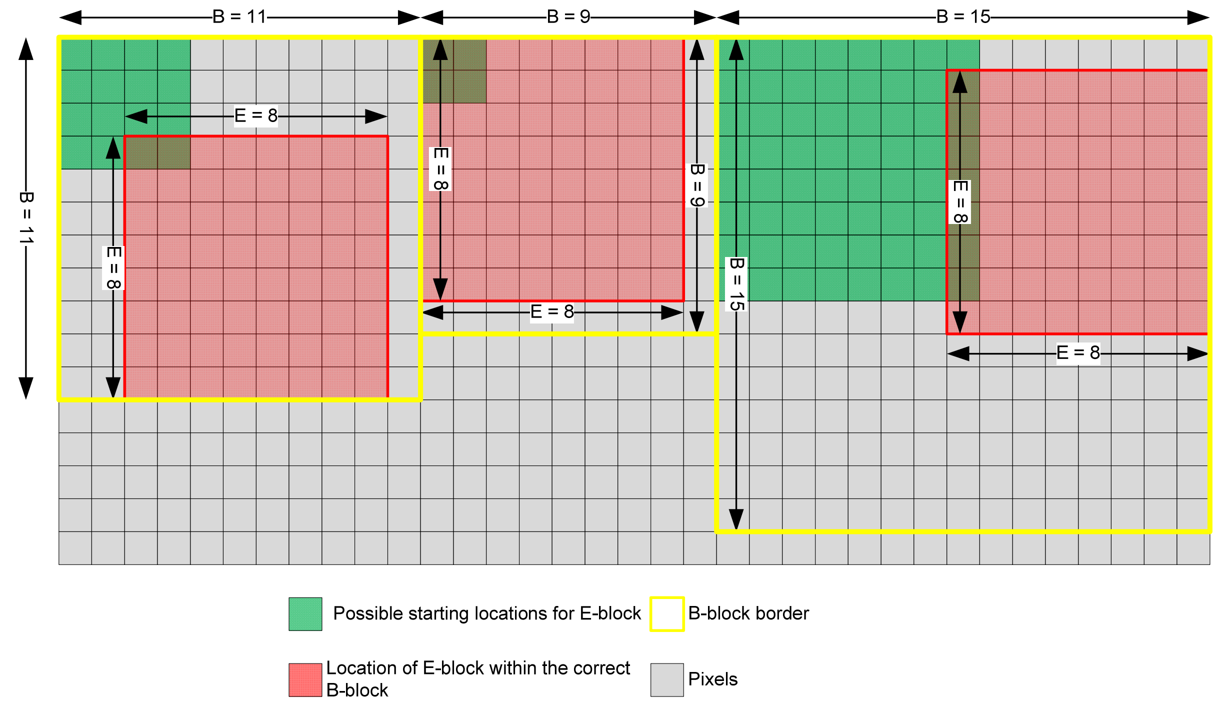

B = 11
B = 9
B = 15
E = 8
E = 8
E = 8
E = 8
E = 8
E = 8
B = 11
B = 9
B = 15
Possible starting locations for E-block
B-block border
Location of E-block within the correct B-block
Pixels

Figure 4-2: Suggested Randomization of B-blocks

The first important aspect to note about the suggested randomization of the B-blocks is that an attacker gains no information on the system as a whole by detecting one of the B-block sizes. With YASS, successfully detecting a single B-block would result in the attacker being able to localize and later affect E-blocks much easier. The pseudo-random nature of the suggested randomization will require the attacker to perform this cropping based attack iteratively, for every single block used therefore an attacker will have a much harder time in breaking the system as a whole.

Let us consider the average time required to detect an E-block using the cropping attack. The assumption here is that the attacker has just guessed the correct B-block size for the YASS framework used in embedding and is now trying to detect if any information hiding E-blocks are present within the B-block.

Figure 4-3 shows a graphical representation of the YASS blocking scheme. We first note that the size of the E-blocks is E = 8. This E-block size is a requirement of the YASS framework and shall be extended later in this chapter. For now, we also note that b can be defined as follows:

$$b = n + 8 , \quad where\ n = 1,2,3, \dots$$

since according to the YASS framework, $b > 8$.

With E = 8, we can define *e* (the width of the matrix representing the possible starting locations of an E-block) as follows:

$$e = b - E + 1,$$

Noting that for YASS, $E = 8$, we have:

$$e = b - 8 + 1,$$

$$e = n + 1.$$

where n is the variable that determines the B-block size.

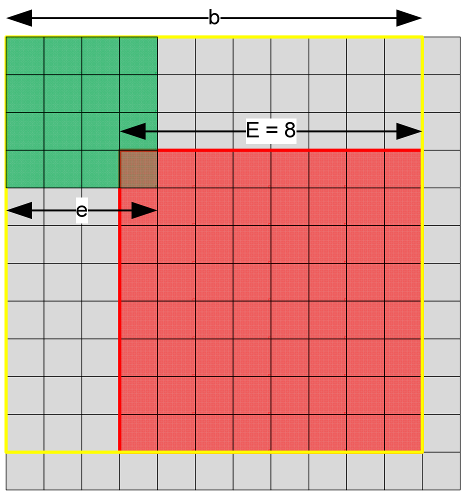


Figure 4-3: Graphically Relation between b and e

We can now note that for a given B-block of size b, there are a total of e^2 possible locations for an E-block of size 8. We also note that since the location of an E-block is determined by a PRNG, the probability of an E-block starting in location i of all possible e^2 locations is,

$$p_i = \frac{1}{e^2}, i \ = 1,2,3, \dots, e^2$$

i.e. the probability of an E-block starting at any of the e^2 positions is the same.

With this information, we can now look at T, the average time required by an attacker to detect whether or not there is an information hiding E-block within a YASS B-block.

Let τ be the average amount of time to analyse an 8 by 8 E-block and determine whether or not information has been hidden within it. Therefore we have:

$$T = 1.\tau.p_1 + 2.\tau.p_2 + 3.\tau.p_3 + \cdots + e^2.\tau.p_{e^2},$$

$$but\ since\ \ p_1 = p_2 = \cdots = p_{e^2} = P = \frac{1}{e^2}\ we\ have$$

$$T = \ P.(1 + 2 + 3 + \cdots + e^2).\tau,$$

$$T = \frac{1}{e^2}.\frac{e^2(e^2 + 1)}{2}.\tau,$$

$$T = \frac{(e^2 + 1)}{2}.\tau,$$

$$but\ e = n + 1$$

$$\therefore T = \frac{((n+1)^2+1)}{2}.\tau\,,$$

$$\therefore T\ is\ O(n^2).\quad (4.1)$$

This result shows that an attacker of YASS will be faced with an average calculation time that is proportional to the square of the number of possible B-block sizes. Now, introducing the randomization of the B-block size, an attacker is not only concerned with detecting a possible E-block within a give B-block as is the case with YASS but they need to determine the correct B-block size at each analysis. Given this added requirement we now show that order of complexity increases due to this added randomization.

Let τ' be the average time required to determine whether a B-block may contain an information hiding E-block within it. Therefore we have T', the average time required to determine the correct B-block size. We now have:

$$T' = 1.p_1'.\tau' + 2.p_2'.\tau' + \cdots + n.p_n'.\tau',$$

$$where\ p_i, is\ the\ probability\ of\ the\ B - block\ being\ of\ size\ i + 8.$$

We now note that once again the size of the B-block is determined by a PRNG which therefore means that

$$p_i' = \frac{1}{n-1}\ ,\ \ i = 1,2,3,\dots,n-1$$

Where *n* is the total number of possible B-block sizes. Take note that since a B-block can take on any value within a range, we have that $9 \leq b \leq n+8$. Therefore we note that the b can take up $n-1$ possible values since$[(n+8)-9] = n-1$.

$$\therefore p_1' = p_2' = \cdots = p_{n-1}' = P' = \frac{1}{n-1},$$

$$\therefore T' = P'.(1+2+3+\cdots+(n-1)).\tau',$$

$$T' = P'.\frac{(n-1)(n)}{2}.\tau',$$

$$T' = \frac{n(n-1)}{2(n-1)}.\tau'\,,$$

$$T' = \frac{n}{2}.\tau',$$

$$but\ since\ \tau'\ is\ O(n^2)\ from\ (4.1) we\ have$$

$$\therefore T\ 'is\ O(n).O(n^2)\ which\ is\ O(n^3).\,(4.2)$$

This result shows that by adding the proposed level of randomization, we have managed to increase the complexity of the steganalysis by an order.

4.1.2 Randomization of E-blocks

As discussed in the previous section, we showed that by introducing randomization to the B-block size, we managed to increase the order of complexity with regard to the average time an attacker requires to detect our embedding. A further extension to the proposed system is to add another layer of randomization by using the PRNG and the shared stego-key, to generate a pseudo-random sequence of E-block sizes.

Just like the introduction of randomization to the B-block size, the introduction of randomization to the E-block size will result in adding complexity to the attacker. The attacker is now tasked with detecting the correct B-block size as well as the correct E-block size for every possible block in the image. Once he has managed this, he is finally tasked with determining the location of the E-block within the B-block.

The average time required to determine the location of the E-blocks within a B-block was shown previously to be of order $O(n^2)$ (4.1). We also showed that the order of complexity for determining the B-block size as well as the location of the E-block size was of an order higher, i.e. $O(n^3)$ (4.2). We now determine the average time required by an attacker to determine an E-block size given a B-block size.

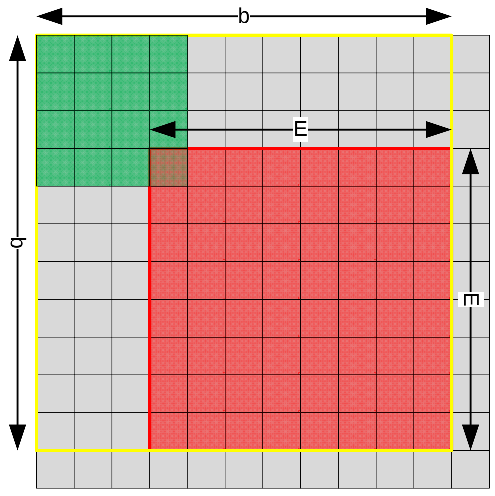


Figure 4-4: Graphical Relationship between b and E for randomized E-block size

Before we start we note that an E-block of size E, Figure 4-4, within a B-block of size b has a size which satisfies the following inequality:

$$8 \leq E \leq (b-1) \quad (4.3)$$

We therefore not that the size of an E-block is constraint to be at least 8 and no bigger than $B-1$. This insures that a given E-block can fit within a B-block and not overlap with any neighbouring B-blocks.

Given this information; we can now determine the average time an attacker requires to verify the correct E-block size.

Let τ'' be the average time required to test if an E-block is present within a B-block of size b. We therefore define T'', the average time required to find the correct E-block size within a B-block. We now have:

$$T'' = 1.p_1''.\tau'' + 2.p_2''.\tau'' + \cdots + n'.p_{n'}''.\tau''.$$

Since the size of the E-block is determined by a PRNG, we note that the probability of any given E-block size been selected is constant, and thus:

$$p_i'' = \frac{1}{n'},\ n' = 0,1,2,3,\ldots$$

Where n' defines possible range of E-block size, i.e. $E = n' + 8$.

$$\therefore p_1'' = p_2'' = \cdots = p_{n'}'' = P'' = \frac{1}{n'},$$

$$\therefore T'' = P''.(1 + 2 + 3 + \cdots + n').\tau'',$$

$$T'' = P''.\frac{n'(n' + 1)}{2}.\tau'',$$

$$\therefore T'' = \frac{n'(n' + 1)}{2n'}.\tau'',$$

$$T'' = \frac{(n' + 1)}{2}.\tau'',$$

At this point we recall from (4.3) that the size of the E-block is limited by the size of the B-block. We then recall that the size of B-blocks is defined as

$$b = n + 8,$$

$$\therefore 8 \le E \le n + 8 - 1,$$

$$8 \le E \le n + 7.$$

But $E = n' + 8$,

$$\therefore 8 \le n' + 8 \le n + 7.$$

We therefore conclude that $O(n') = O(n)$.

With this result, we have that T'' is $O(n).O(n^3)$ since from (4.2) τ'' was shown to be $O(n^3)$. Therefore we have that T'' is $O(n^4)$.

Given these results, we have shown that our proposed system has not only added layers of randomization which allows for a hidden stego-message to be better diffused through its cover-image, but it also increased the order of complexity by a factor of two, as shown in Table 4-1.

Table 4-1: Order of Complexity for Stego-Systems

System	Order of Complexity
Yet Another Steganographic System (YASS)	$O(n^2)$
Hybrid Blocking System	$O(n^3)$
Multi-Blocking System (proposed system)	$O(n^4)$

4.1.3 Full Blocking Strategy

With the analysis in the previous section, we now present the full blocking algorithm employed in out proposed system.

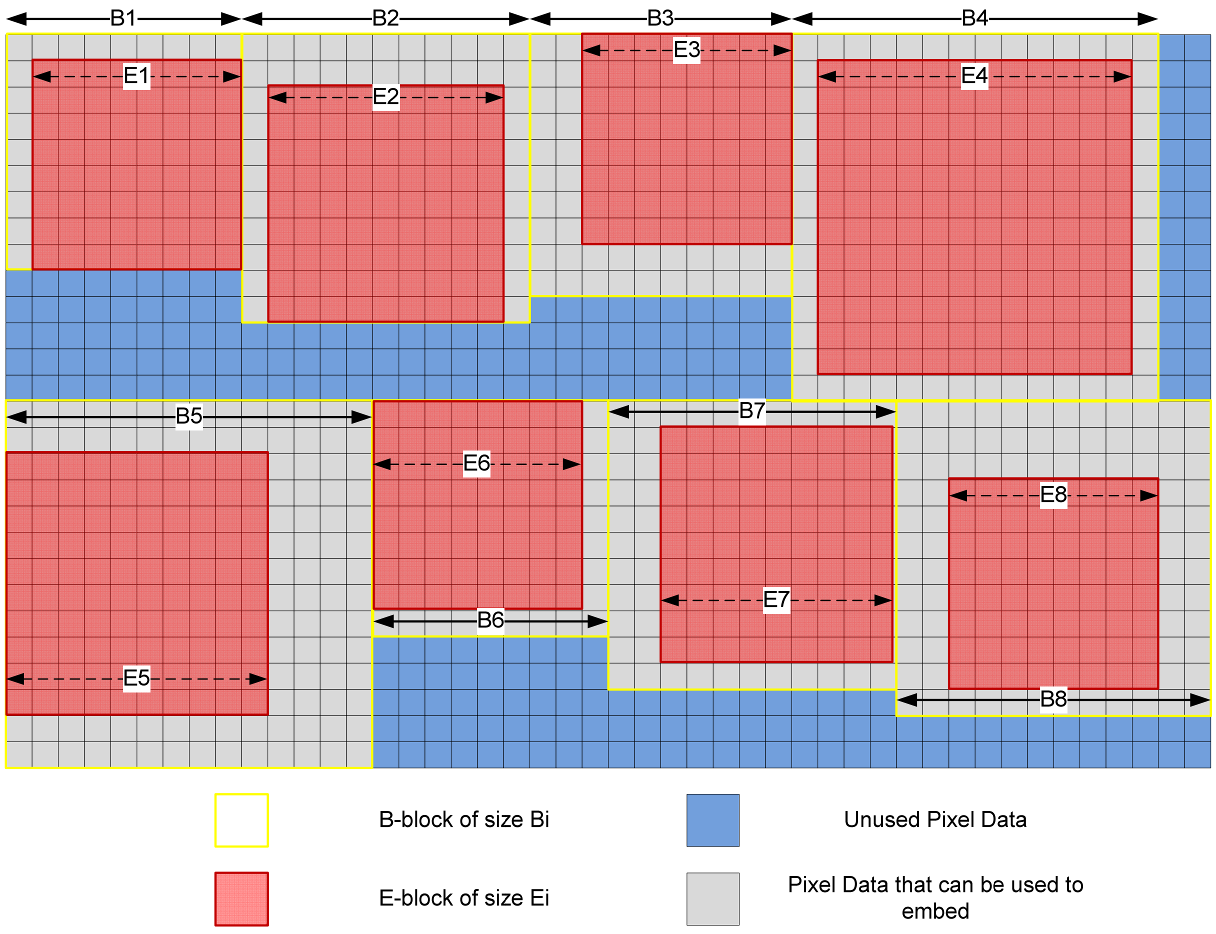


Figure 4-5: Example of proposed blocking

The blocking algorithm is as follows:

1. The image in its spatial domain is divided into $B_i x B_i$ B-blocks, where B_i is determined by a PRNG with the shared stego-key as seed such that B_i is within a set range.

2. For each $B_i x B_i$ B-block, a PRNG is used to determine an $E_i x E_i$ E-block. The size of the generated E-block must meet the condition of: $8 \leq E_i < B_i - 1$.

3. Finally, using a PRNG, we determine a location for the $E_i x E_i$ E-block within its parent $B_i x B_i$ B-block.

Figure 4-5 shows an example of our proposed blocking algorithm as applied to a sample of pixels. Since a greedy blocking algorithm is used to place the B-blocks, one should note a very important result of this approach.

As the strategy performs its job, regions of unused pixels start appearing (blue regions within Figure 4-5). Although these regions can be eliminated by designing an optimum blocking algorithm, using this optimum algorithm will aid an attacker and undo most of what is achieved by randomizing the B-blocks to start with.

With an optimum blocking algorithm, the strategy would require the knowledge of the size of all the B-blocks that will be generated in the embedding process. With this knowledge, the algorithm would then place the blocks in the most optimum way so as to cover as much of the cover-image as possible. The algorithm therefore no longer places blocks randomly but based on size thus allowing the attacker to estimate which size blocks will be clustered in which region of the cover-image. The attacker's estimate can be as follows:

Assuming an attacker is aware of the range of possible B-block sizes that are used in the embedding process. In the case of this example, we assume the range of B-block values is:

$$9 \leq b \leq 13,$$

The attacker can now calculate the average B-block size,

$$b_{average} = \frac{13+9}{2},$$

$$b_{average} = 11.$$

The attacker is now faced with trying to detect if any information is hidden within an innocent looking image of resolution $500x500\ pixels$. He can therefore calculate the average number of B-block that would be generated within the image:

$$Average\ Number\ of\ Blocks = \frac{500x500}{11x11},$$

$$Average\ Number\ of\ Blocks\ \approx 2066\ .$$

Given the average number of expected blocks and the knowledge that $p_i = \frac{1}{5}, i = 9, 10, 11, 12, 13,$ the attacker can determine the average expected count per block size since:

$$B_{average}^{9} = B_{average}^{10} = \cdots = B_{average}^{13} = \frac{2066}{5} \approx 413.$$

The attacker can now run the optimum blocking algorithm using this estimated average blocking numbers and given the nature of the algorithm, he would have a very good estimate of the expected block sizes in each region of the image, counteracting some of the benefits of introducing randomization to the B-block size.

Obviously this approach above can be used to estimate the number of blocks generated by the proposed system as well as determine an estimate of the numbers of each B-block size generated, but one should note that since the blocks are placed in the order of their generation, rather than a position to maximize the space used, we note that an attacker is still tasked with regenerating a pseudo-random sequence of numbers without knowing the shared secret seed used by the generator.

4.2 Embedding Strategy

The previous section introduced the extension to the YASS framework aimed towards counteracting an attacker's ability to determine the B-block size and thus the E-block region used in the embedding process. With this more random approach to determining locations where information can be hidden, we now focus on improving the embedding strategy used in the hopes of improving on YASS by achieving the following goals:

1. Improve the embedding rate and allow for better control of embedding ability within the stego-system.
2. Decrease the effects of the embedding process on visual distortion within the stego-image.
3. Increase the embedding process's robustness against JPEG compression.

The proposed embedding strategy is still based on the YASS framework but the framework has been modified to account for the new, larger E-blocks that our proposed system can generate, as well as refinements aimed at achieving the goals listed above.

4.2.1 Performing the 2D Discrete Cosine Transform

The first stage in the embedding process, once the E-block size and position has been determined is to perform the two dimensional Discrete Cosine Transform (2D DCT). This process is unchanged from the YASS framework but for the fact that the transform is no longer performed on a $8x8$ grid, but rather $E_i x E_i$ grid, where E_i is the size of the ith E-block. Figure 4-6 shows this stage in the embedding process.

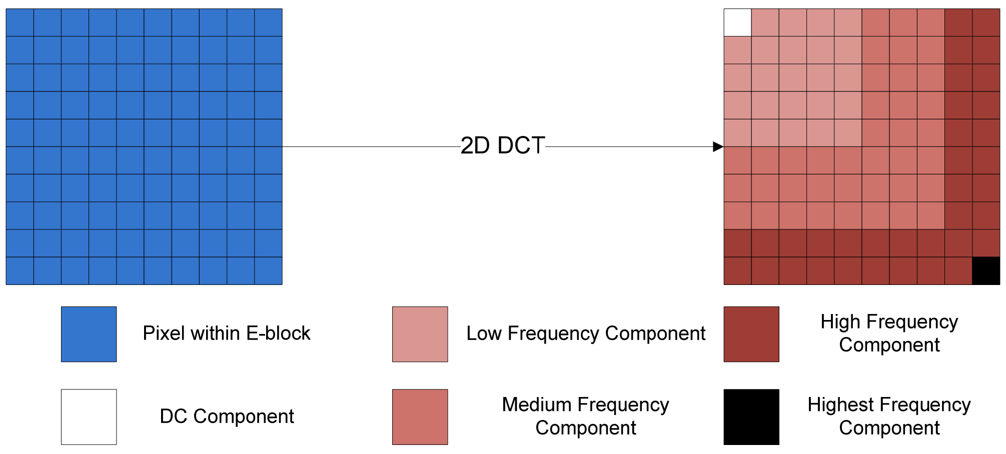


Figure 4-6: Two Dimensional DCT being applied to E-block Pixels

The forward function used to perform this transformation is:

$$F(u,v) = \frac{1}{4}C(u)C(v)\sum_{x=0}^{E_i-1}\sum_{y=0}^{E_i-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right), \quad (4.4)$$

$$where\ C(z) = \begin{cases} \frac{1}{\sqrt{2}} & if\ z = 0, \\ 1 & otherwise. \end{cases}\ and$$

$$f(x,y)\ is\ the\ pixel\ intensity\ in\ location\ (x,y)\ of\ the\ E-block.$$

The reasoning behind the use of transformation mirrors those discussed in Chapter 3. By embedding in the frequency domain, we limit the effect of the embedding on the cover-image thus resulting in far less visual distortion. We also embed in this domain in the hopes of allowing our stego-message to be far more robust to subsequent JPEG compression.

The next stage in the embedding process is the quantization stage.

4.2.2 E-block Quantization

Once the 2D DCT process has been completed, the next stage in the embedding process requires the quantization of the resulting DCT coefficients. In the YASS framework, this stage utilized the quantization matrix from the JPEG compression standard since the E-blocks were the correct size (8 by 8 coefficients). With our proposed extension to increase the size of the E-blocks, we require a new technique to perform this quantization since the grid size of the resulting E-blocks has the potential to be greater than 8 by 8.

Figure 4-7 shows the resulting E-block after the 2D DCT process has been performed. At this junction, we are faced with quantizing the DCT Coefficients. We have the options of still using the 8 by 8 quantization matrix detailed in the YASS framework, but when needed, using padding coefficients to filling in the needed space. Figure 4-7 shows this suggested quantization method (orange bordered regions).

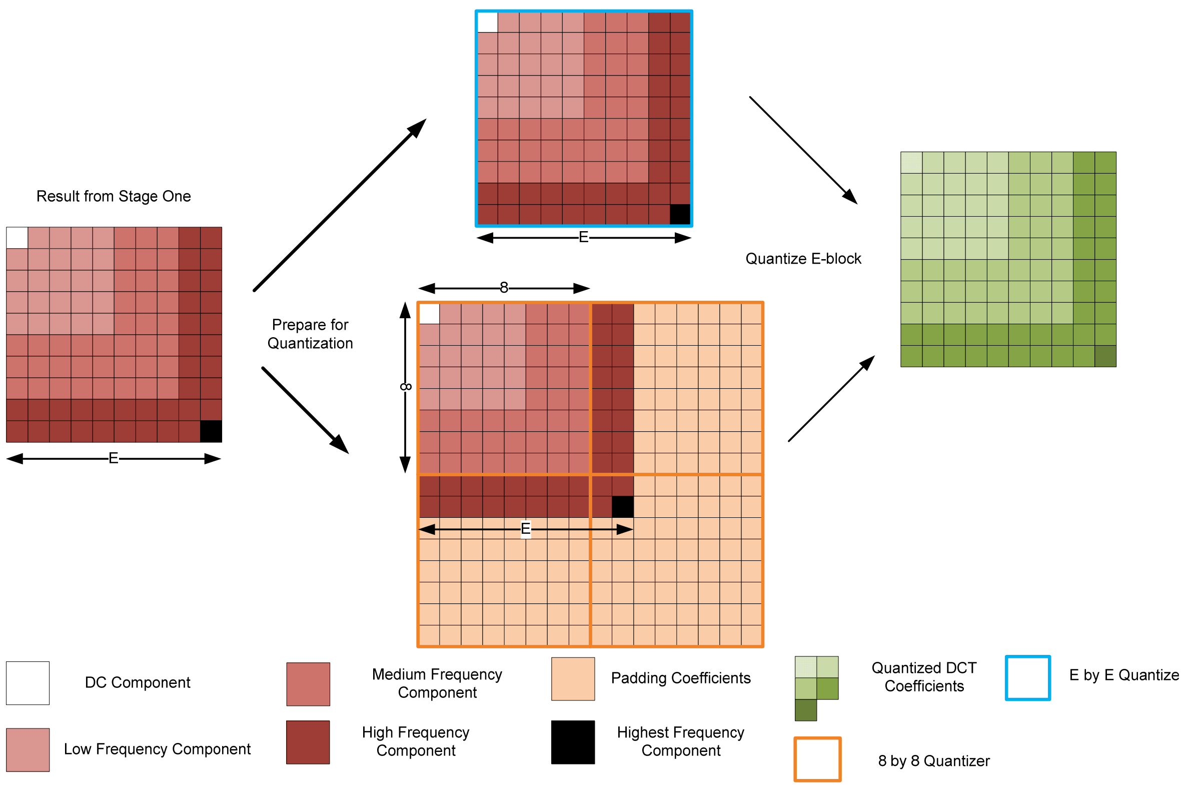


Figure 4-7: Proposed Quantization Methods

The biggest issue with using this method is the discontinuities created when quantizing an E-block greater than 8 by 8. In Figure 4-7, we can clearly see that high frequency coefficients in the E-block are being quantized within three different quantization matrices. Since the 8 by 8 quantization matrix used in the JPEG standard and YASS is designed to compress high frequency by using a large quantization step for those components, and smaller steps for lower frequency components. Using this method of blocking the E-block and the quantizing it along an 8 by 8 grid would result in a huge discontinuity in the resulting quantized E-block.

The biggest issue with this discontinuity is that two neighbouring coefficients, of similar frequencies could be quantized using two varying quantization steps. This would result in one coefficient, which may have been perfectly suited for embedding being quantized to a value that would not allow for embedding and for another coefficient, one which was not suitable for embedding, suddenly becoming attractive to the possibility of embedding. An example of two such neighbouring coefficients is showing in Figure 4-8 by the two coefficients marked with an “X”.

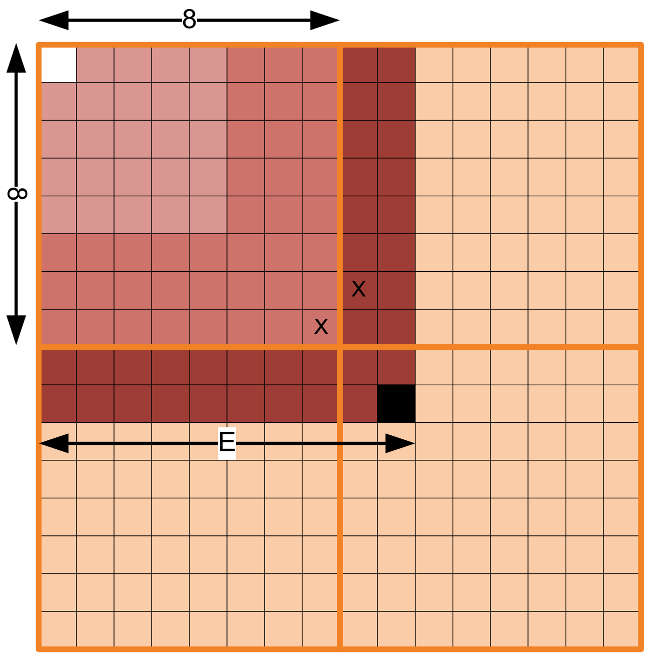


Figure 4-8: Example of neighbouring Coefficients that are quantized at the wrong step value

Since this method of blocking an E-block to perform individual 8 by 8 quantization results in unwanted behaviour, our proposed system will need a new approach towards quantization. Instead of using the 8 by 8 quantization matrix detailed in YASS, we now propose a custom designed quantization matrix that can be generated regardless of E-block size.

4.2.2.1 Custom Quantization Matrix Design

To design a custom quantization matrix that can be generated for a given E-block size, we must first understand the design principles behind the one used in YASS and the JPEG compression standard.

As mentioned in Chapter 3, the purpose of the quantization stage in the JPEG standard is to introduce redundancies that can be later dropped to decrease the amount of information required to represent the image. The Quantization matrix is designed to compress the magnitudes of higher frequency components much more than lower frequency components. This is achieved by using bigger quantization steps for high frequency components as opposed to the smaller steps used for low frequency components.

Figure 4-9 shows the increase of coefficient through an E-block, where a darker shade of blue indicates a higher frequency coefficient position.

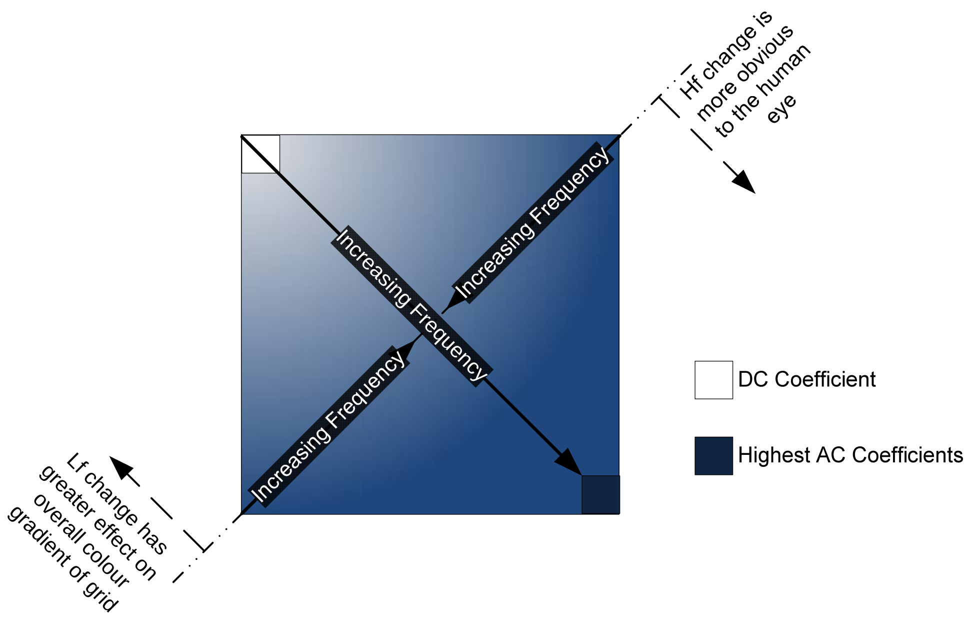

Hf change is more obvious to the human eye
Increasing Frequency
Increasing Frequency
Increasing Frequency
DC Coefficient
Highest AC Coefficients
Lf change has greater effect on overall colour gradient of grid

Figure 4-9: Frequency position in E-block due to 2D DCT

With knowledge of how the differing frequency coefficients are distributed through the E-block and with the knowledge that lower frequency components survive JPEG compression better than higher frequency components, we can model the quantization matrix to follow the frequency distribution expected by the 2D DCT.

Simply, we require that quantization matrix to behave similar to Figure 4-10. This model requires that as the frequency component of the coefficient increases, so does the size of the quantization step applied to it.

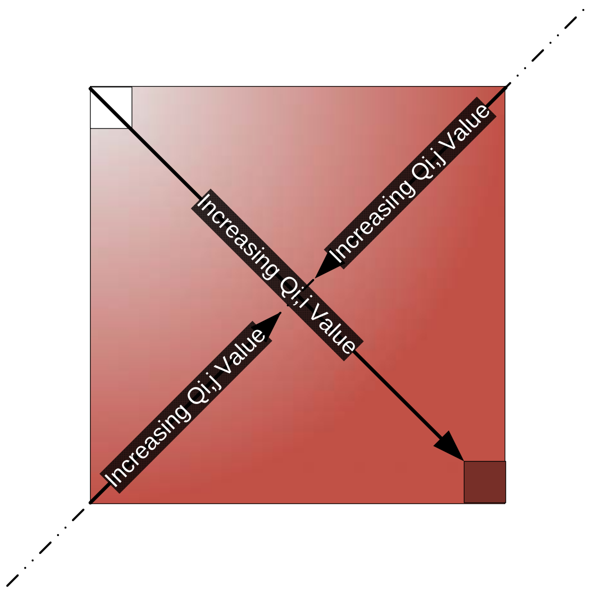


Figure 4-10: Required Quantization model

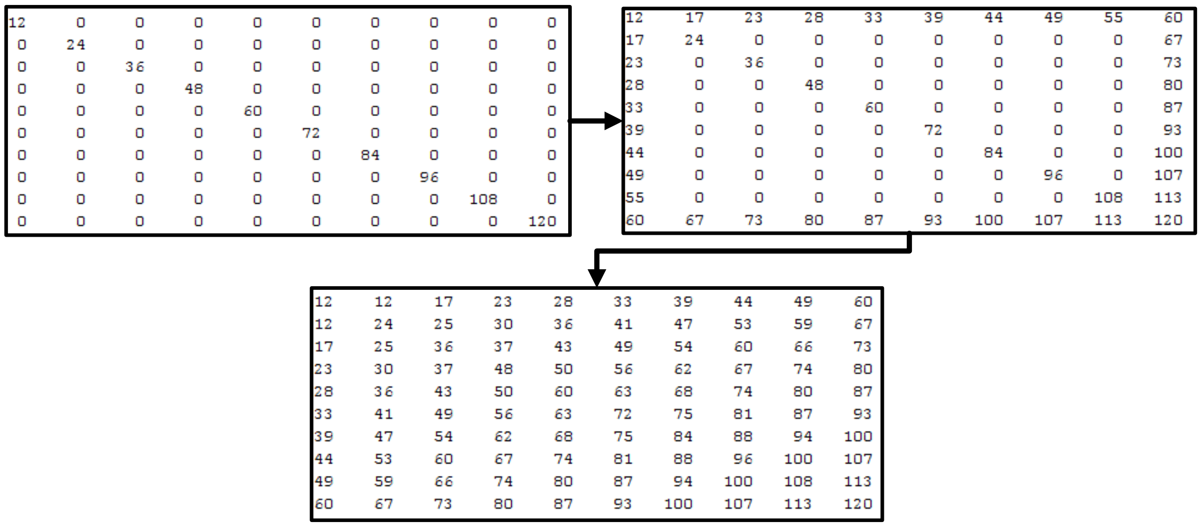
We note from Chapter 3 the JPEG standard quantization matrix shown again in Figure 4-11. The first thing to note is that although the model shown in Figure 4-10 implies a level of symmetry about the main diagonal, this is not true for the JPEG standard matrix.

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Figure 4-11: JPEG Standard Quantization Matrix

For the purpose of the proposed system, the symmetry modelled in Figure 4-10 aids in supplying simplicity in the algorithm used for generating the quantization matrix. Both symmetric and asymmetric quantization matrices can perform as capably as each other when it comes to the task at hand and thus there is little in the way of performance loss by making this compromise.

To match the original JPEG Quantization matrix, we utilize the same minimum step size of 12 along the main diagonal as well as the biggest step size of 120. At this point, we generate a step number of uniform linear steps to fill in the remaining elements along the main diagonal. Once the main diagonal is complete, we then generate the remaining i diagonals. This is done by calculating the values along the first and last row of the matrix as well as first and last column of the matrix. At this point, we generate a linearly spaced number sequence to fill in the remaining off diagonals as shown in Figure 4-12.



12	0	0	0	0	0	0	0	0	0
0	24	0	0	0	0	0	0	0	0
0	0	36	0	0	0	0	0	0	0
0	0	0	48	0	0	0	0	0	0
0	0	0	0	60	0	0	0	0	0
0	0	0	0	0	72	0	0	0	0
0	0	0	0	0	0	84	0	0	0
0	0	0	0	0	0	0	96	0	0
0	0	0	0	0	0	0	0	108	0
0	0	0	0	0	0	0	0	0	120

12	17	23	28	33	39	44	49	55	60
17	24	0	0	0	0	0	0	0	67
23	0	36	0	0	0	0	0	0	73
28	0	0	48	0	0	0	0	0	80
33	0	0	0	60	0	0	0	0	87
39	0	0	0	0	72	0	0	0	93
44	0	0	0	0	0	84	0	0	100
49	0	0	0	0	0	0	96	0	107
55	0	0	0	0	0	0	0	108	113
60	67	73	80	87	93	100	107	113	120

12	12	17	23	28	33	39	44	49	60
12	24	25	30	36	41	47	53	59	67
17	25	36	37	43	49	54	60	66	73
23	30	37	48	50	56	62	67	74	80
28	36	43	50	60	63	68	74	80	87
33	41	49	56	63	72	75	81	87	93
39	47	54	62	68	75	84	88	94	100
44	53	60	67	74	81	88	96	100	107
49	59	66	74	80	87	94	100	108	113
60	67	73	80	87	93	100	107	113	120

Figure 4-12: Generating Custom Quantization Matrix

Now that we have a strategy for the creation of a custom $E_i x E_i$ quantization matrix, our proposed embedding strategy can be updated to Figure 4-13.

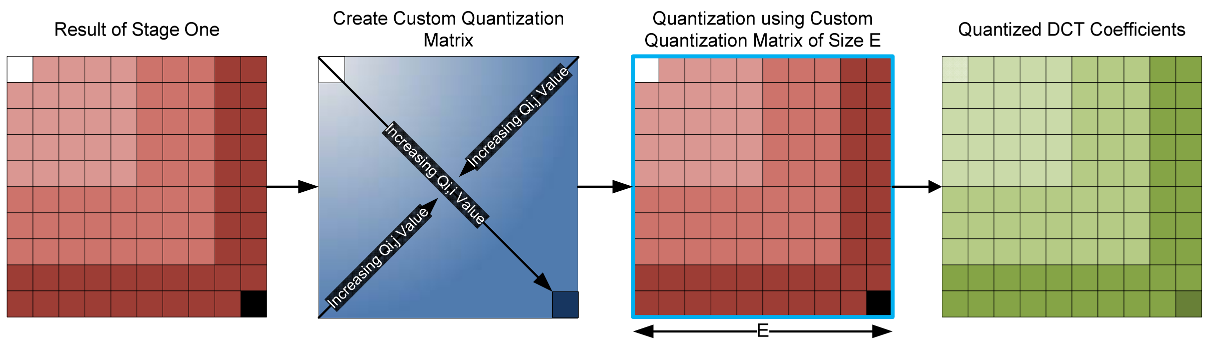

Result of Stage One
Create Custom Quantization Matrix
Increasing Qi,j Value
Increasing Qi,j Value
Increasing Qi,j Value
Quantization using Custom Quantization Matrix of Size E
E
Quantized DCT Coefficients

Figure 4-13: Quantization Stage of Embedding Process

Once the DCT values have been quantized, the next stage in the embedding process is to determine the location of the DCT coefficients that shall be used to embed the stego-message.

4.2.3 E-block Masking

The processing of masking the E-block relates to a set of rules that determine which DCT Coefficients can and will be used for embedding the stego-message. There are a few approaches as to where the best location for embedding is as well as a few approaches as to how to determine them.

The first feasible way to select possible embedding candidates is through a random selection. Much like the OutGuess 0.1 strategy, our system could randomly select a group of quantized coefficients and embed. That said, random select, although far more secure, does not allow for a great level of robustness. The stego-image is more than likely to pass through a compression stage while entering the communication channel, and randomly selecting coefficients may result in the stego-message being corrupted by this compression process. Therefore although random select is attractive for its increased level of security, it is far from practical due to the large amount of error handling required to manage the potential corruption that may occur.

Therefore the next logical step would be to select candidate coefficients based on their likelihood of surviving the compression that will occur as the stego-image enters the channel. To determine the best possible locations to select from, we must first understand how affecting different areas of the quantized DCT Coefficients will affect the image as well as how they are affected by compression.

4.2.3.1 Introducing Distortion

Since the goal of Steganography is to communicate in such a way so as to raise least amount of suspicion, the masking process of our proposed systems needs to select candidate coefficients in such a way as to minimise the visual distortion that may occur by affecting those coefficients.

As mentioned in Chapter 3, the human eye is rather proficient at detecting high frequency changes. For example, our eyes tend to notice sharp changes in contrast much easier than a slight, smooth change in colour, a fact illustrated in Figure 4-14.

We notice that with the image on the right, our eye manages to determine distinct borders between the white and black pixels shown since these represent high frequency change in the image. On the other hand, one notices that it is almost impossible to make out any distinct colour boundaries in the image on the left since each neighbouring pixel is only slightly different from its neighbour. This represents low frequency change in an image.

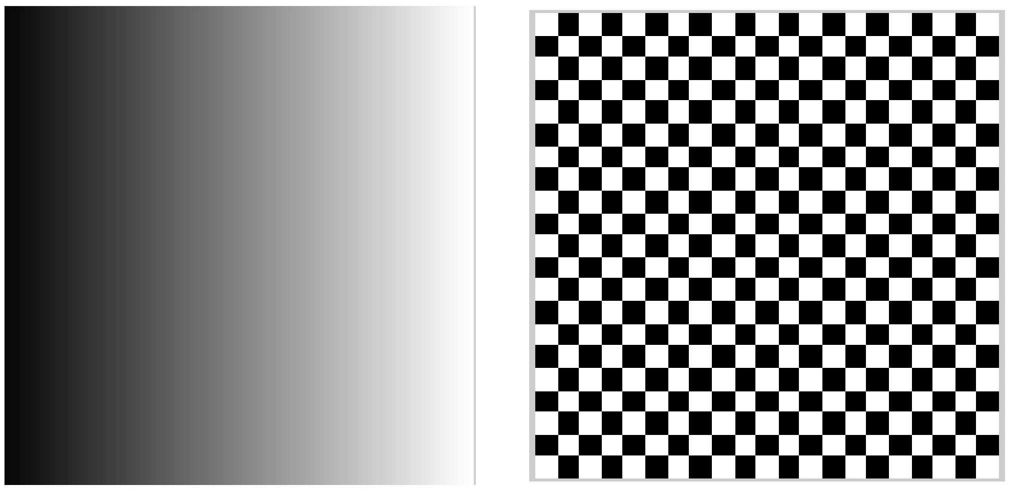


Figure 4-14: Example of a human eye's ability to see high frequency change

This then of course raises the question of why the JPEG compression algorithm quantises most heavily in the high frequency coefficients if this is where humans see changes so well. The simple answer is that in most images, high frequency almost always relates to fine detail in an image. In most cases, this fine detail is not vital in the overall image and can thus be dropped to compress the image. What is important is the lower frequency detail that determines colour saturation as well as region gradients within the image.

Therefore, any image compressed using the JPEG compression scheme would more than likely have most of its high frequency information dropped. It therefore comes as no surprise that embedding in high frequency coefficients is far from advisable. That said hiding in lower frequency components does also have its draw backs.

Embedding in low frequency coefficients at this point seems like a rather attractive proposition. With a high probability of surviving any potential subsequent compression that may occur, it would make the most sense to embed in this region of the frequency spectrum. Unfortunately, there is a trade off one must consider when deciding to embed in low frequency coefficients. Take, as an example, the image shown in Figure 4-15.

The original image is shown on top. The first stage, much like the proposed system is to crop an E-block, in this case, a simple 8 by 8 block will do. This crop is show on the left, below the original image. Once the crop is complete, the 2D DCT is applied to red spectrum and then quantized as detailed in the previous section. At this point, the three lowest frequency components were decremented by one and the reverse process was applied, i.e. expansion was performed, followed by the inverse 2D DCT. The resulting 8 by 8 grid is shown on the right below the original image.

One can clearly see that by affecting the lower frequency components, we have had a rather obvious effect on the image. Anyone looking at the resulting image would notice the slightly unnatural blue hue that has resulted due to the embedding and thus would be very suspicious of the image as well as the communication.

This simple example would suggest that embedding in the lowest frequencies is also a rather risky proposition.

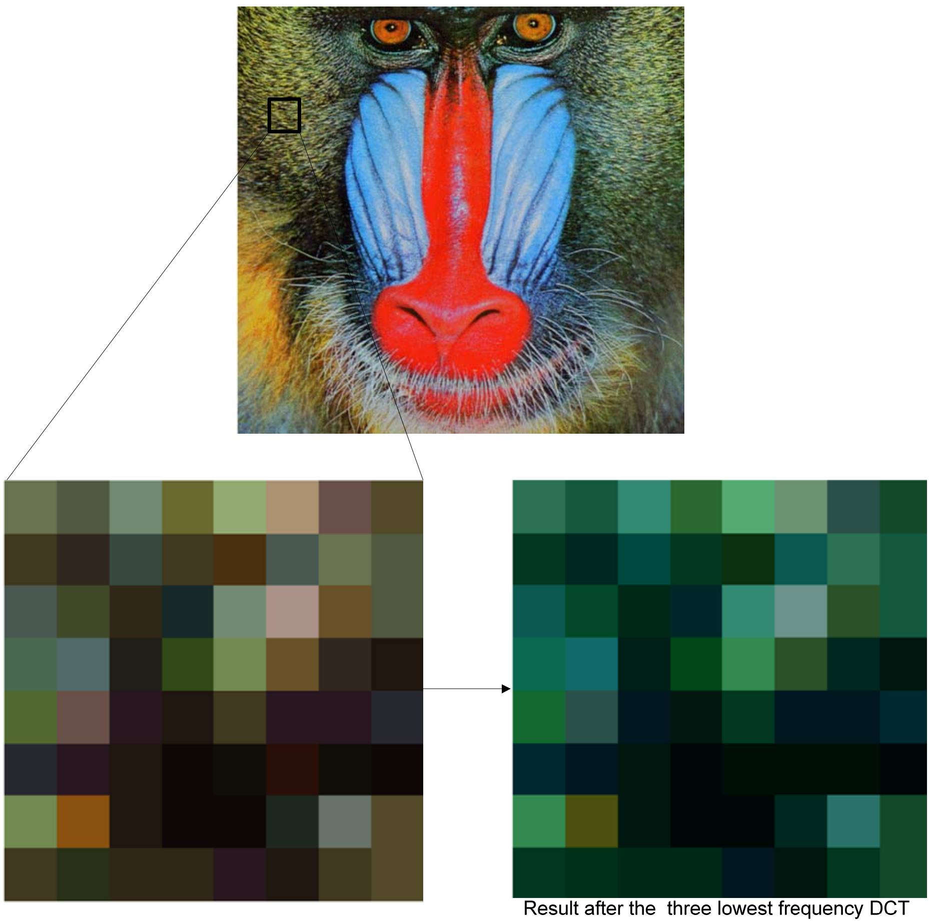

Original 8 by 8 pixel crop from image
Result after the three lowest frequency DCT coefficients (quantized) were decreased by one

Figure 4-15: Example of effects of lowest frequency embedding

4.2.3.2 Compression Effects

As mentioned earlier, in most cases, images are commonly compressed before entering a communication channel. This is normally due to the limitations of the bandwidth available on the channel and thus the need to compress to save some of that bandwidth. Faced with this, one must be careful as to where the stego-message is embedded.

Since the JPEG compression standard is widely used to compress and store images, it is safe to assume that there is a very good chance the stego-image will undergo compression via this system. As mentioned above, the JPEG compression strategy relies on quantizing higher frequency components using a larger quantization step than lower frequency coefficients. This heavy quantization results in some high frequency components being quantized to zero and thus are later dropped due to the entropy coding that occurs later in the process.

To explain this concept further we utilize a simple 8 by 8 grid. Assuming the DCT coefficients have been equalized about zero, we can monitor the effect of changing each individual coefficient as they are changed. Figure 4-16 shows the effects of changing each coefficient, where the 8 by 8 grid of pixels located in position (i,j) represents the changes to the pixel values if the corresponding $C_{i,j}$ DCT coefficient is incremented by one.

We notice that a change to the DC coefficient and other lower frequency coefficients changes large regions of pixel intensities. On the other hand, changes to higher frequency coefficients have the effect of changing neighbouring pixels rather than regions of pixels thus creating a higher frequency pattern.

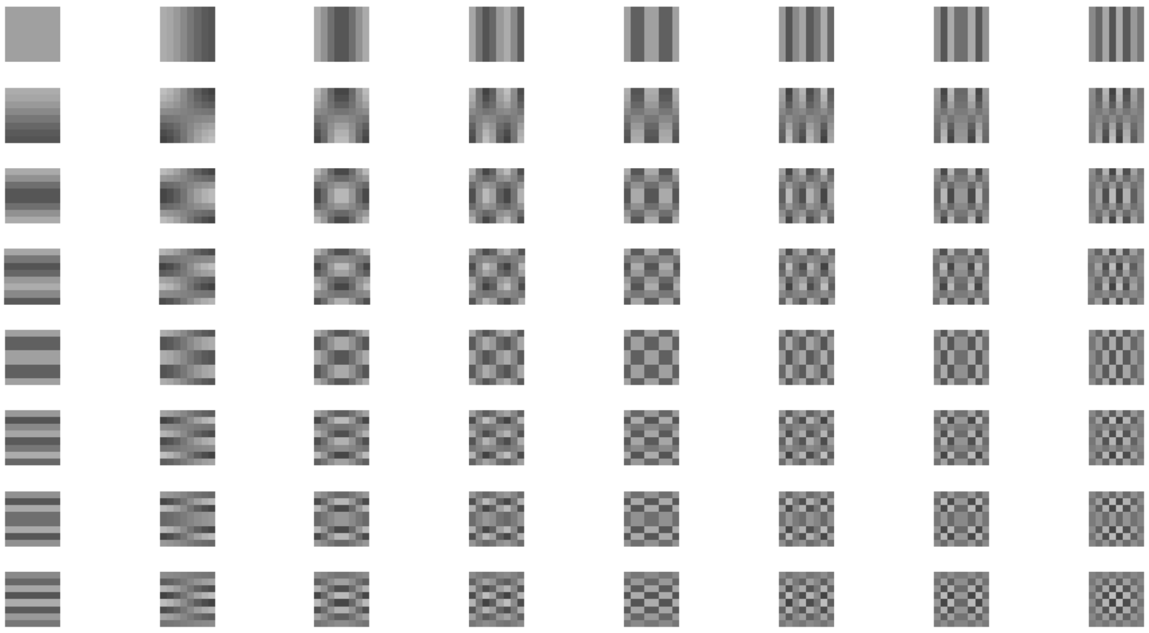


Figure 4-16: Effect of change DCT Coefficient on Pixel value

Now, we take the 64 generated images from Figure 4-16 and apply the standard JPEG compression algorithm. At this point, visually, there is very little to differentiate between the original 8 by 8 grid and the new, compressed grids. That said, from a pixel value point of view, we notice that there is a change to the pixels intensities. Figure 4-17 shows these changes, where a white pixel represents a change in that pixel's intensity value.

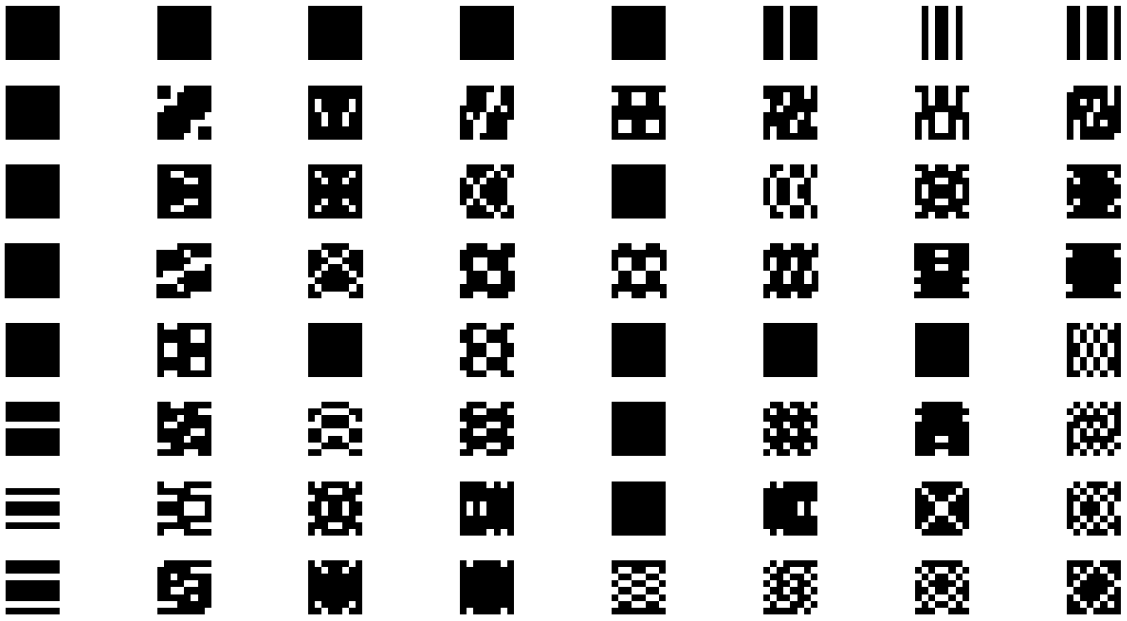


Figure 4-17: Error introduced due to compression (white pixel represent a change in value)

From Figure 4-17 we notice that grids generated by changing lower frequency coefficients manage to survive the compression with little to no change to the pixels, while grids generated from higher frequency coefficients suffer with higher errors due to the compression applied.

This proves that our original statement warning against embedding in high frequency components was well founded since even at an average compression rate, errors can be noted due to the compression.

At this point, we have shown that embedding in the lower frequency coefficients may result in visual distortion which could be easily detectable. We have also shown that embedding in high frequency coefficients may result in large errors in the hidden message due to the likelihood of the high frequency components being compressed out of the image.

The masking algorithm is responsible for selecting the best region for embedding, and given the discussion in the previous two sections, the coefficients with middle frequency components have the best of both worlds. Embedding in those coefficients would result in a good trade off between the potential to be lost due to compression versus the risk of creating visually obvious distortion.

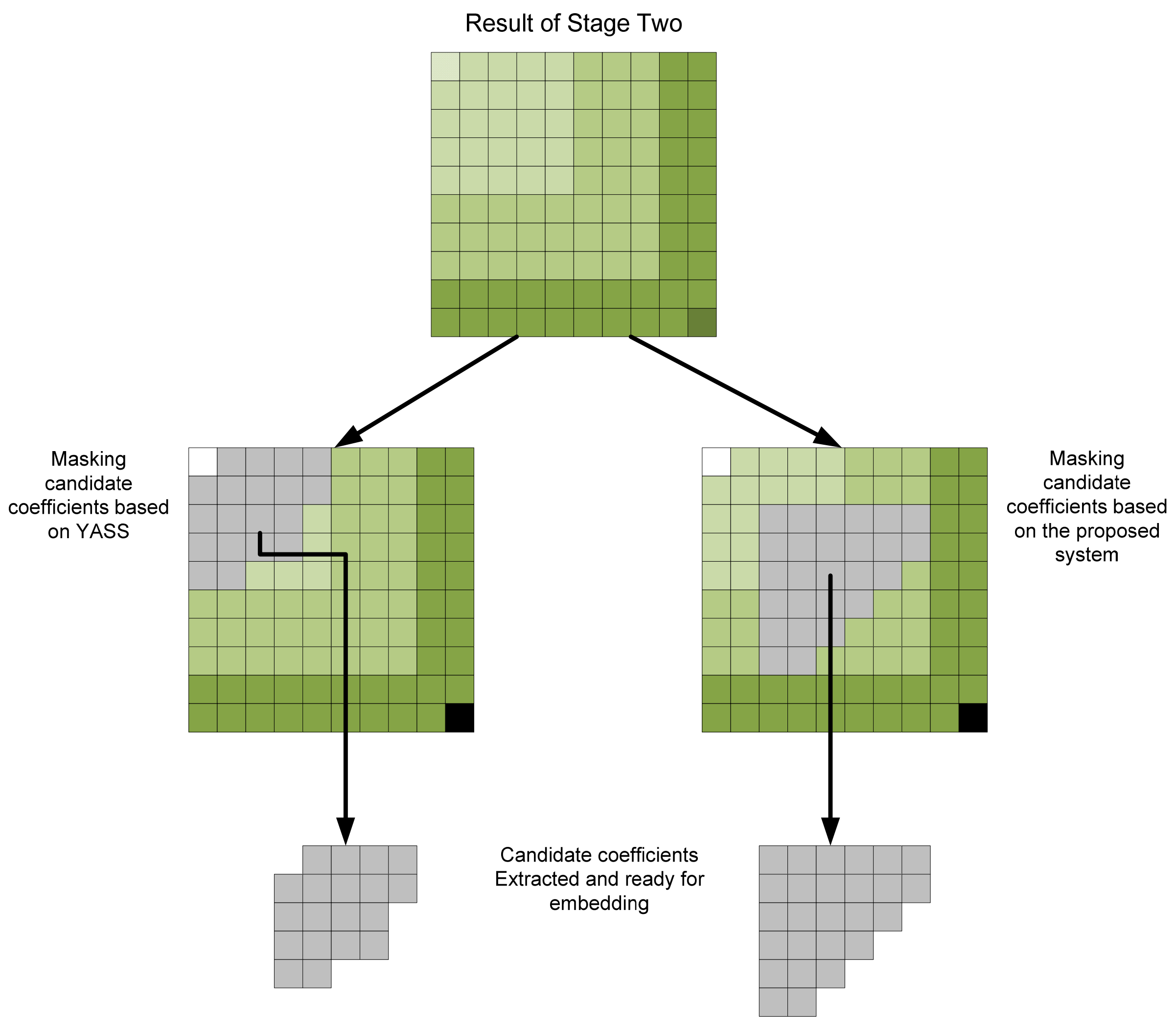

Result of Stage Two
Masking candidate coefficients based on YASS
Masking candidate coefficients based on the proposed system
Candidate coefficients Extracted and ready for embedding

Figure 4-18: Possible masking paths (YASS or Proposed system)

The YASS framework, as shown in Figure 4-18, selects the 19 lowest frequency coefficients and attempts to embed within them. Note that the DC Coefficient is not considered. Selecting the candidate coefficients using the mask suggested by the YASS framework has two main drawbacks.

The first of these drawbacks is the fact that selection is always limited to the first 19 low frequency coefficients. This approach will waste a lot of potential candidate coefficients generated due to the larger E-block size in our proposed system. The second major issue with using the YASS framework approach is we have no way of controlling how aggressively we wish to embed.

We now present the newly formed masking strategy used with our proposed system shown in Figure 4-18.

4.2.3.3 *Masking the right region*

Our masking algorithm aims to address three main issues, namely:

1. Avoid embedding in regions that will result in visual distortion (low frequency coefficients) or regions which are affected heavily by compression.
2. Allow for user controlled aggression, allowing users the option of selecting how aggressive they which to embed within a given image.
3. Allow for the selection of a high number of candidate coefficients to increase embedding capacity.

To address these three issues we introduce an important parameter to the proposed system, namely the *Aggression level* (A_l) . This parameter shall allow the user to not only control how many candidate coefficients are present, but also how many of them to embed in and which region in the DCT grid the system is allowed to select from.

4.2.3.3.1 *Aggression Level*

The aggression level (A_l) is the parameter that controls how many candidate coefficients can selected. This is accomplished by increasing the area in which possible candidate coefficients can be selected from.

As mentioned earlier, medium frequency coefficients have a better chance of surviving compression as compared to higher frequency coefficients, and medium frequency coefficients have less of an effect on visual distortion than lower frequency coefficients and are thus prime candidates for selection.

However, as the aggression level increases, we are faced with the decision of which direction to increase the search area so as to look for more candidate coefficients. If we increase equally in the direction of low and high frequency components, we note that we increase the risk of introducing more errors due to compression as well increase the visual effects due to embedding.

Since an increase in aggression implies a greater risk of detection, we make a choice to increase the search area to maximize the potential increase in the embedding capacity. Embedding in higher frequency components has a rather big drawback. Since these coefficients are affected the most by the expected JPEG compression that will be performed as the image enters the channel, we realise that any increase in the embedding capacity will be negated by the increase in potential errors generated due to compression.

Increasing the search area into the lower frequency coefficients on the other hand has a much more positive affect. Although the likelihood of some visual distortion occurring due to aggressive embedding, this trade-off will be well worth the risk due to the increase of capacity, since lower frequency coefficients are not affected by normal compression.

We therefore propose a masking strategy that increases the masking area with an increase in aggression level such at it always favours lower frequency coefficients. An example of such a proposal is shown in Figure 4-19.

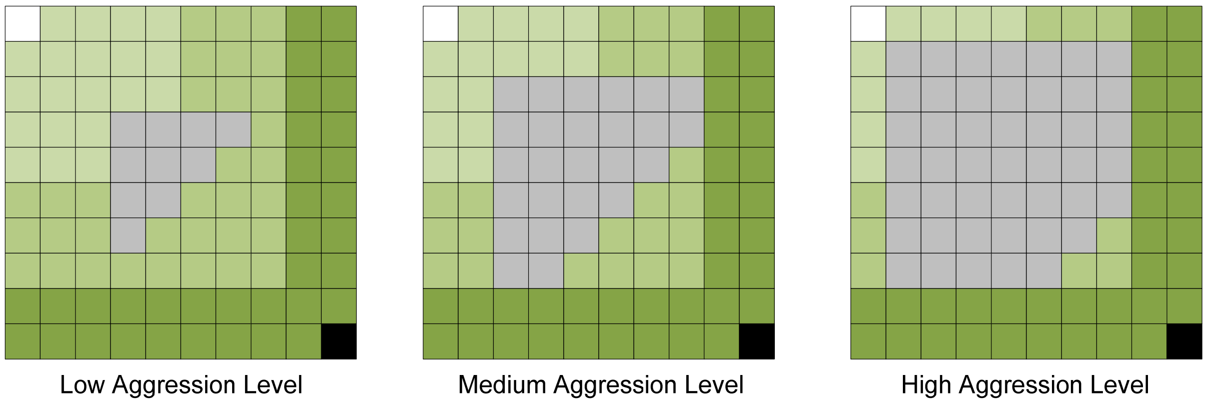


Figure 4-19: Example of aggression level selection regions

We now note that to address the second issue mentioned at the start of this section, namely, allowing for the control of embedding aggression, there is a need to compromise on the first issue, namely, avoiding low frequency and high frequency coefficients. This is an obvious trade off that will depend on the context of use. If the user wishes to maximize their security, clearly a very low aggression level would be selected, resulting in an embedding process that has minimal effect on the cover-image. On the other hand if a user needs to send a lot of data using as few cover-images as possible, a higher aggression level would be much more suitable.

The use of the Aggression level control allows for user controlled embedding capacity. This is something that the YASS framework did not allow for and is new to the proposed system. The concept of embedding in medium frequency coefficients is also new. Previously, the YASS framework only focused on embedding in the lowest frequency coefficients and did not try to optimise for a fair trade off between the risk of introducing visual distortion and risk of message corruption due to compression.

With an understanding of the aggression level and how we use it to determine the region in which to search of candidate coefficients, we now look at the selection process and how the stego-message is embedded into those selected coefficients.

4.2.4 Embedding Data

After the correct region has been determined, the next step is to select candidate coefficients that can be used to embed data. The selection process of these candidate coefficients is largely dependent on the method used in embedding the stego-message. Embedding data into coefficients can be done in many different ways, and depending on which method is utilized, coefficients behave in a certain way. We therefore first present the proposed embedding method and follow the description with our selection criterion.

4.2.4.1 *Hiding the Data*

Before any embedding is performed, the stego-message needs to go through some processing. We note that since we are embedding in a region which trades off visual distortion with compression losses, we need an error handling technique that can provide good recovery, even with high bit error rates. For the purpose of the proposed system, we utilize the Repeat-Accumulative (RA) error correction code to generate a codeword than can be embedded.

Thus the very first task is to divide the stego-message into small stego-blocks. By doing so, we can embed each stego-block into a single B-block rather than across a few B-blocks, meaning that if the error handling fails to recover the data, a smaller piece of the stego-message is lost and not all of the message. Once the stego-message is blocked, each block is coded into its representative codeword and it is this code word that shall be embedded into the candidate coefficients in a particular region.

The YASS framework utilized the QIM embedding method as detailed in Chapter 3. As mentioned, this embedding strategy results in an increase of zero coefficients after the embedding process. This increase in zero coefficients resulted in a statistical attack that allowed attackers to detect embedding with an 8 by 8 grid (Solanki K. et al 2008). We therefore avoid using QIM as the embedding procedure to avoid this disadvantage of the YASS framework.

Another option is to utilize the matrix embedding used in F5, but as shown in Chapter 3, this will also result in an attacker being able to detect the embedding process through statistical attacks. Therefore we suggest a new embedding scheme that will allow for data to be embedded into candidate coefficients without resulting in an increase in generated zero coefficients.

For the purpose of our system we utilize the *Random ± ternary modulation.* This embedding technique will allow us to embed in medium frequency candidate coefficients as well as maintain the average number of zero coefficients in the region. This solves the problem introduced by the QIM used in the YASS framework. Assume x is a DCT coefficient, b is the message bit, and y is the DCT coefficient after embedding b; the embedding strategy for Random ± ternary modulation is:

$$y = \begin{cases} x, & if\ b = LSB(x) \\ (x-1), & if\ \big(b \neq LSB(x)\big)\ and\ (r > 0\,) \\ (x+1), & if\ \big(b \neq LSB(x)\big)\ and\ (r < 0\,) \end{cases}$$

Where r is a pseudo-random variable with a uniform distribution on {-1, +1}. We note that by using this modulation technique, coefficients with an absolute value of one have the same probability of being changed to a zero coefficient as zero coefficients have of changing to a coefficient of absolute value one. This means that on average, the number of zero coefficients before embedding will equal the number of zero coefficients after embedding has occurred.

With the embedding scheme in mind, we now present a set of criteria that a coefficient needs to satisfy for it to be selected for embedding.

4.2.4.2 *Selecting Candidates*

Although the modulation scheme allows for message bits to be saved in any coefficient, we select a set of criteria that will guard against embedding that will result in high visual distortion or result in embedding that can be easily detected.

We note that although the scheme allows for embedding in coefficients that are equal to zero, changes to those coefficients may have a very noticeable effect on the cover-image, especially if the coefficients being changed are of low frequency. We also note that coefficients with an absolute value of one will have a 0.5 probability of being changed to a zero, once again having the potential of distorting the cover-image rather heavily. We therefore place a criterion that all candidate coefficients selected in the masked region must have an absolute value greater than two, i.e.

$$|x| > 2.$$

This serves as the only constraint we place on the embedding process and although the embedding scheme allows for embedding in other coefficients, we restrict embedding to minimize distortion. Figure 4-20 shows the full process of selecting candidate coefficients within the masked region and the embedding of a random message into the candidates.

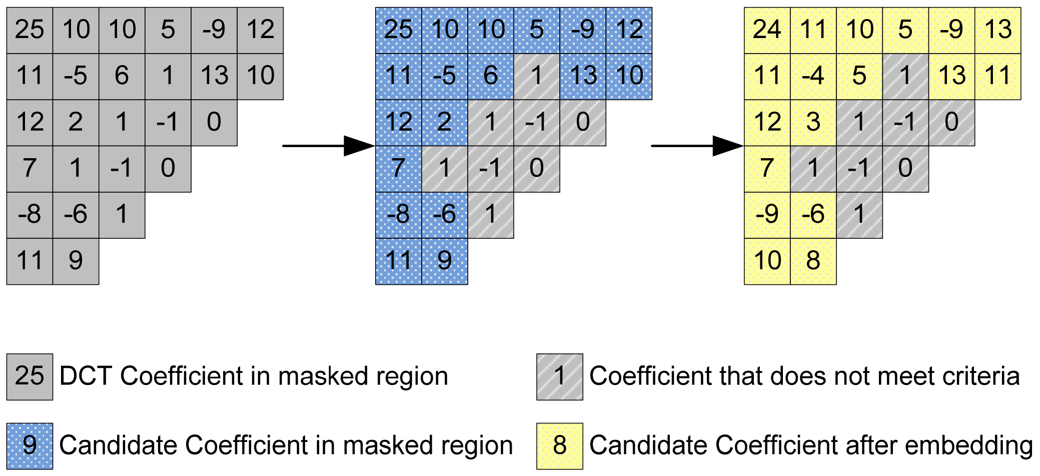

DCT Coefficient in masked region
Coefficient that does not meet criteria
Candidate Coefficient in masked region
Candidate Coefficient after embedding

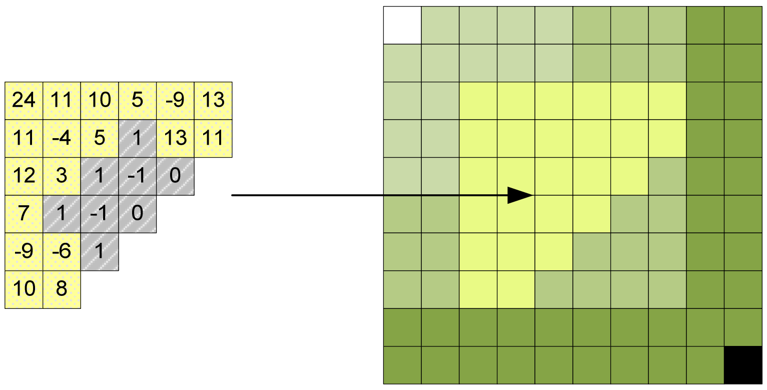
Figure 4-20: Selection and embedding within masked region

Once a block of the stego-message has been embedded into candidate coefficients within the masked region, a reversal process is applied to return the quantized DCT coefficients to the spatial domain and thus to the cover-image. The following section details this reversal.

4.2.5 Generating the Stego-Image

Since our stego-message is embedded within candidate DCT coefficients, the processes detailed in the previous section needs to be reversed so as to once again arrive at an E-block that is in the spatial domain and can be returned to the cover-image to generate a stego-image.

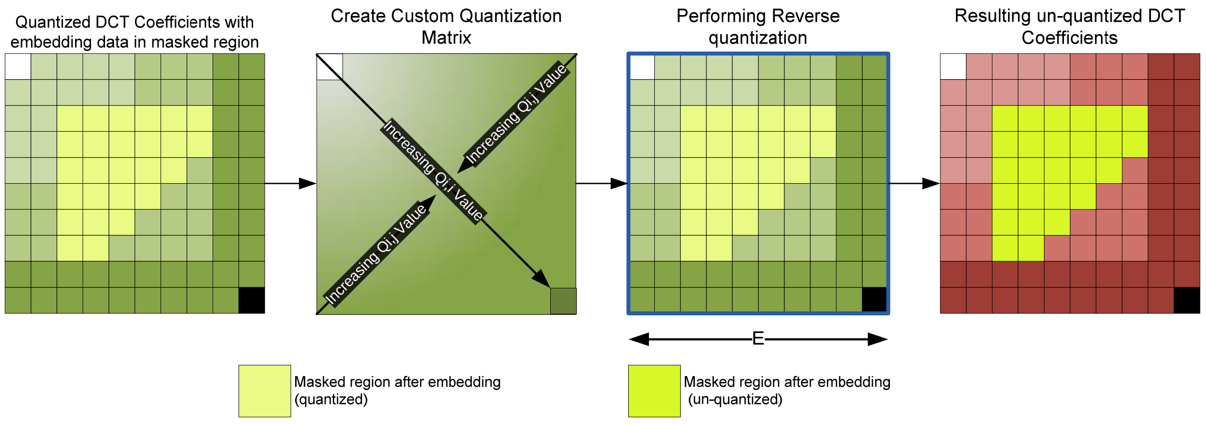
Once a block from the stego-message has been embedded in a masked region, the first stage in the reversal process is to return the masked region, with the effected coefficients, to the E-block. This process simply replaces the entire masked region from the original E-block with that masked region after embedding. Figure 4-21 shows this process.



🟨 Masked region returned after embedding

Figure 4-21: Returning Masked region after embedding

Once the masked region has been returned to the E-block, the next process is to reverse the quantization step that was performed previously. To perform this step, we simple regenerate the quantization matrix for an $E\ by\ E$ E-block and instead of dividing each coefficient by its corresponding quantization value we multiply each coefficient by that quantization value. Figure 4-22 shows this process.



Quantized DCT Coefficients with embedding data in masked region
Create Custom Quantization Matrix
Increasing Qi,j Value
Increasing Qi,j Value
Increasing Qi,j Value
Performing Reverse quantization
Resulting un-quantized DCT Coefficients
E
Masked region after embedding (quantized)
Masked region after embedding (un-quantized)

Figure 4-22: Reversing Quantization

Once we have reversed the quantization stage, the next step in the reversal process is to perform the inverse 2D DCT (4.5) on an $E\ by\ E$ E-block. The purpose of this step is to return the E-block from the frequency domain to the spatial domain. Figure 4-23 shows this process.

$$f(x,y) = \frac{1}{4}\sum_{u=0}^{E-1}\sum_{v=0}^{E-1} C(u)C(v)F(u,v)\cos\left(\frac{(2x+1)u\pi}{16}\right)\cos\left(\frac{(2y+1)v\pi}{16}\right), \qquad (4.5)$$

$$where\ C(z) = \begin{cases} \frac{1}{\sqrt{2}} & if\ z = 0, \\ 1 & otherwise. \end{cases}$$

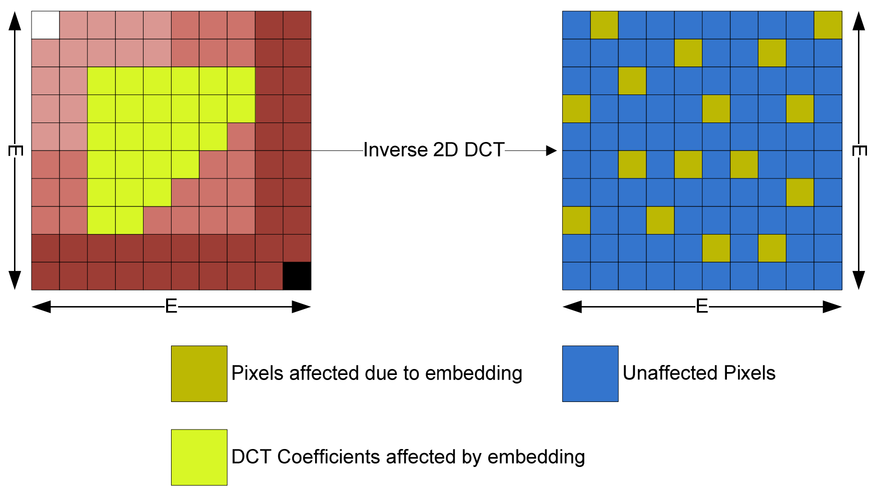

E
E
Inverse 2D DCT
E
E
Pixels affected due to embedding
Unaffected Pixels
DCT Coefficients affected by embedding

Figure 4-23: Performing the inverse 2 Dimensional Discrete Cosine Transform

Once the E-block has been returned to the spatial domain, the last step in the reversal process is to return it to its corresponding location in the cover-image. This process is repeated for every block of stego-message embedded into the cover-image.

Once the whole message has been hidden within the cover-image or all available E-blocks have been used, the resulting image is the stego-image.

4.3 Summary

This chapter presented the development of the proposed system. The proposed system is designed as an extension of the framework presented by the YASS framework in Chapter 3 and builds on the main principles of YASS to not only improve security and potentially increase the embedding capacity, but counteract some of the shortcomings of the YASS framework, currently being used to attack and detect information saved using the framework.

The proposed system is formally defined as follows:

$Let\ C\ be\ a\ set\ of\ all\ possible\ Cover-Images,$

$M\ be\ a\ set\ of\ all\ possible\ Stego-Messages,$

$K\ be\ a\ set\ of\ all\ possible\ Stego-Keys,$

$S\ be\ a\ set\ of\ all\ possible\ Stego-Images.$

$\therefore\ we\ have\ that\ c \in C, m \in M, \kappa \in K\ and\ s \in S.$

We first encode m into a sequence of code words by using:

$$Encode(m) \rightarrow M_{block}, a\ set\ of\ all\ possible\ message\ code\ words.$$

We also perform a blocking algorithm on the cover-image to generate a sequence of E-blocks by using:

$$CoverBlock(c, \kappa) \rightarrow E_{block}, a\ set\ of\ all\ generated\ E-blocks.$$

We now define the embedding process for a given $m_{block} \in M_{block}$into a given $e_{block} \in E_{block}$ follows:

$$DCT_{2D}(e_{block}) \rightarrow DCT_{eblock}$$

The function DCT_{2D}defines a $E\ by\ E$ two dimensional Discrete Cosine Transform, which is performed on the E-block. The result of this function, DCT_{eblock} , represents the frequency domain representation of the pixel intensities.

We then take DCT_{eblock} , and performing quantization using:

$$Q(DCT_{eblock}\,, Q_h) \rightarrow DCT_Q$$

The function Q defines the quantizing process where each DCT coefficient in $DCT_{eblock}\,, DCT_{i,j}$, is divided by the corresponding quantization element $Q_{i,j}$ and where Q_h refers to the quality factor used during the quantization process. Q_h is simply a scalar value that is used to control the quantization step used during quantization. The greater the value of Q_h the smaller the quantization step and vice versa. The result of the quantization function is the $E\ by\ E$ quantized DCT block referred to as DCT_Q.

We now take $DCT_Q$and perform a masked scheme on it to extract a region in which candidate coefficients can be selected:

$$Mask(DCT_Q, A_l) \rightarrow Mask_{region}$$

The masking function, $Mask$, is designed to selected a region of DCT coefficients that have the best trade off between risk of introducing visual distortion and surviving the inevitable JPEG compression that will occur once the stego-image enters a compressed channel. The size and region masked off by the function is controlled by the aggression level, A_l, which is a scale variable from 1 to 10, and represents how aggressive an embedding is required. The larger the value of A_l, the greater the region selected by the mask.

Once the masked region has been determined, the next process in the embedding process is the selection of the candidate coefficients and the actual embedding of the stego-message block. We therefore have the following function:

$$Embed\left(Mask_{region}, m_{block}\right) \rightarrow Mask_{embedded}$$

The $Embed$ function performs two steps. The first of these steps is the selection of all candidate coefficients within the $Mask_{region}$. The next stage is the actual embedding of the stego-message block into those candidate coefficients. The selection process is done via a criterion set to minimize the effect of embedding on the cover-image, so as to generate a stego-image that results in little to no suspicion. This selection criterion is simply: $|DCT_{candidate}| > 1$.

The embedding process utilizes Random ± ternary modulation. The result of the $Embed$ function is $Mask_{embedded}$. This is the original masked region, but with candidate coefficients affected by the stego-message m_{block}.

At this point, $Mask_{embedded}$ needs to be returned to the original cover-image so as to generate the stego-image. This is done through a set of reverse processes. These processes include:

$$UnMask\left(Mask_{embedded}, DCT_Q, A_l\right) \rightarrow DCT'_Q,$$

which returns the masked region back into the quantized $E\ by\ E$ DCT block.

$$ReverseQ\left(DCT'_Q, Q_h\right) \rightarrow DCT'_{eblock},$$

which reverses the quantization which occurred earlier in the embedding process.

$$IDCT_{2D}\left(DCT'_{eblock}\right) \rightarrow e'_{block},$$

which performs the $E\ by\ E$ inverse two dimensional discrete cosine transform, to return the embedded E-block from the frequency domain to the spatial domain.

$$Replace(e'_{block}, \kappa, c) \rightarrow s,$$

which returns the spatial domain E-block to the cover-image to generate the stego-image. This embedding cycle is repeated until either all the E-blocks available in E_{block} are used or all the message blocks in M_{block} are embedded into the cover-image.

We therefore define the general embedding and decoding functions from the proposed system as:

$$E(c, m, \kappa, A_l, Q_h) \rightarrow s,$$

$$D(s, \kappa, A_l, Q_h) \rightarrow m.$$

The proposed system introduced five major improvements over the original YASS framework. The first two major improvements were the introduction of randomisation to the B-block and E-block sizes. These randomisations were shown to increase the average attack time by a factor of one, for the Hybrid system (only B-block size randomization) and by an order of two for the full proposed system. This increase of order in the average attack time suggests a major increase in the security of the proposed system versus the original YASS framework.

The next major improvement is the introduction of the Masking process. Originally the YASS framework would simply embed message blocks into the 19 lowest frequency coefficients. But as shown earlier in the chapter, this may result in obvious visual distortion and the proposed system avoids this by utilizing the different masking function.

Instead of selecting the 19 lowest frequency coefficients, the proposed system aims to embed information in the medium frequency coefficients generated using the 2D DCT. By embedding in these medium frequency coefficients we limit the embedding process's effect on the cover-image, all the while limiting the effects of possible compression errors that maybe introduced when the stego-image enters a compressed channel.

This change in frequency location allowed for the introduction of the fourth major improvement, namely the ability to control the embedding capacity. By utilizing the aggression level, the user can select a preference towards high embedding or high security. The higher the aggression level, the greater a region is selected by the masking function, selecting lower frequency coefficients in favour of higher frequency coefficients, increasing the possibility of more candidate coefficients being present in the masked region.

The final improvement on the original YASS framework is the use of Random ± ternary modulation instead of QIM. This is to reduce the effect of increasing the number of zero coefficients generated during the embedding process. By using this modulation scheme, along with a selection criterion, the proposed system has an embedding procedure that ensures that the number of zero coefficients before and after embedding is the same, solving one of the YASS framework's short comings.

The next chapter looks at the performance of the proposed system. We take a careful look at the improvements to the embedding capacity as well as look at the effects of compression on the performance of the system. The chapter then focuses on the security of the proposed system by analyzing the results of a visual distortion survey run to test if visual distortion is detectable at high embedding rates. We also compare the proposed system's performance to that of the YASS framework against steganalysis systems.

5 Performance and Security Evaluation of Proposed System

Chapter 4 presented the newly proposed system referred to as Multi-blocking (MULTI). MULTI aims to improve upon the YASS framework, detailed in Chapter 3, by introducing two extra layers of randomization as well as a new embedding scheme. This Chapter provides a systematic performance analysis of MULTI and provides comparisons to the YASS framework as well as the HYBRID system, proposed during the development of MULTI.

The performance analysis is divided into five main sections. We first look at the blocking rates achieved when comparing MULTI with the YASS framework. Once the blocking rates of the two systems have been established, the next section looks at the Embedding and Error Rates achieved using the HYBRID and MULTI systems and compares them to a similar YASS system. We then test the transparency of the MUTLI system by utilizing the Power to Signal Noise Ratio (PSNR) test mentioned in Chapter 2. We then present a visual distortion survey run on a randomly selected group of University staff and students to test MULTI's ability to avoid suspicion of an average viewer. We finally conclude the performance analysis by running MULTI against a set of blind steganalysis systems, as used originally to determine the performance of the YASS framework.

5.1 Blocking Comparison

Due to the introduction of the randomization of B-blocks used in our proposed system, it is important to demonstrate that the new MULTI system can achieve, and potentially better, the rate at which the YASS framework can block a cover-image.

This performance test aims to show that MULTI, although allowing for a range of B-block sizes, on average generates the same, if not more, B-blocks that can be used for embedding as the YASS framework. The original YASS framework was implemented and used to generate a set of B-blocks in a 500 by 500 pixel image. This process was repeated for B-block sizes ranging from 9 to 16. Once the YASS blocking was run, the same 500 by 500 pixel image was used to test the blocking rate for MULTI. To match the B-block sizes used in the YASS test, we used eight different ranges for MULTI, namely 9-10 to 9-17.

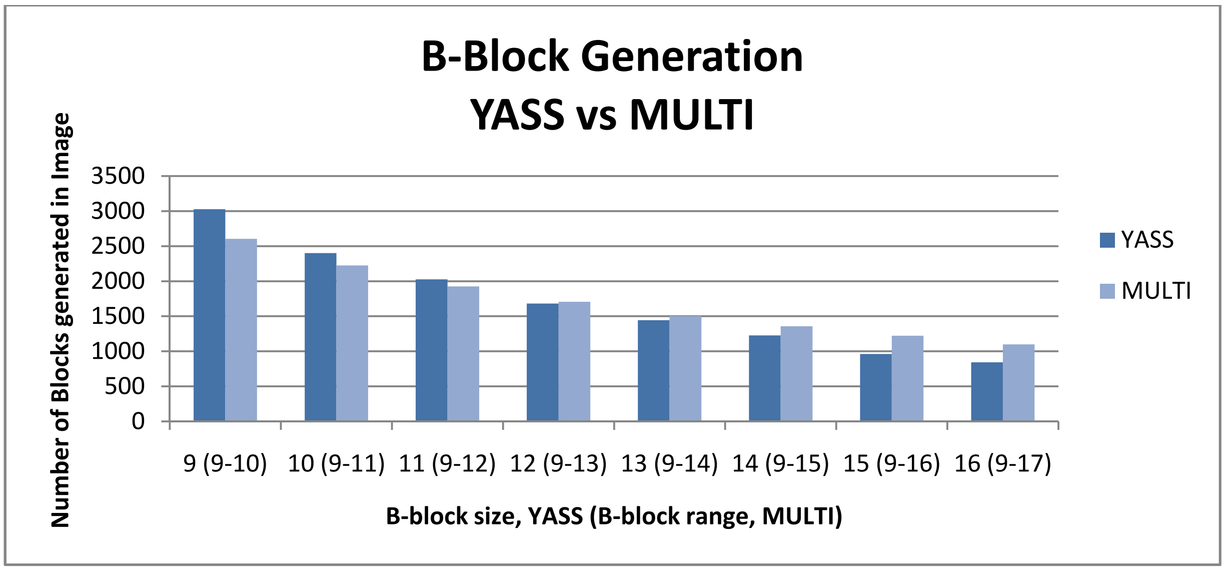


Figure 5-1: Comparison of B-block Generation between YASS and MULTI

Figure 5-1 shows the results of the blocking rate test. We notice that for very small B-block size, namely 9, 10 and 11, the YASS framework results in a slightly higher block count. That said, as the B-block size increases, we can clearly see that MULTI generates more blocks within the same area than YASS does.

With a B-block size of 12 for YASS and a B-block range of 9-13 for MULTI, we notice that both systems share a very similar blocking rate. From this point on, if the same embedding scheme is used for both YASS and MULTI, MULTI would result in a higher embedding capacity, since it generates more blocks that can be used to embed in.

In Chapter 4, it was stated that a PRNG would be used in the generating of the B-block size sequence used in the blocking of the MULTI system. To show that the uniform distribution assumed in the theory is true in implementation, we use the 9-13 B-block range on an image of 2048 by 1536 pixels. This increase in image size allows the generation of more samples that can give a clearer indication of any trends that maybe present in the data collected.

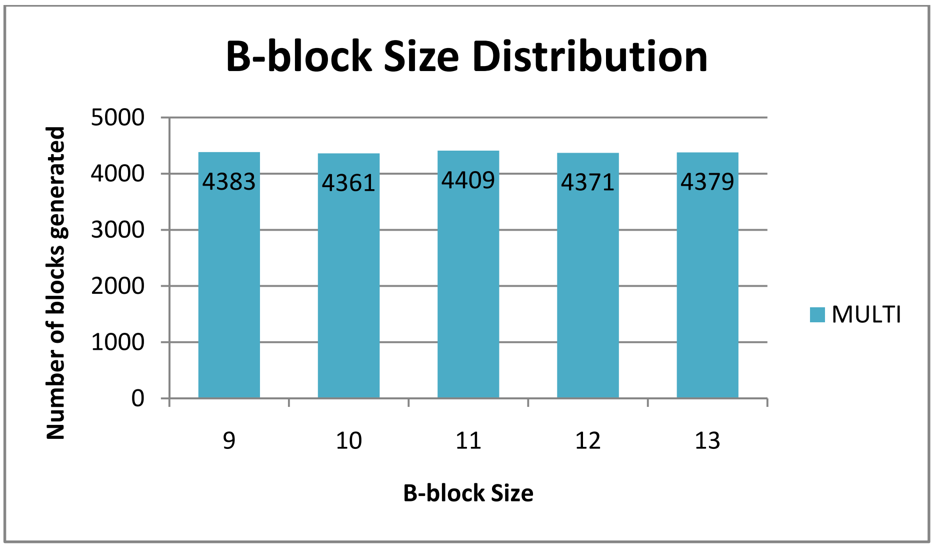


Figure 5-2: B-block size distribution

Figure 5-2 shows the resulting block counts for each of the five different B-block sizes that could be generated within the selected range. We notice that with just less than 22 000 blocks generated in the image, each of the five sizes generates the same block count within 0.13%. This confirms that on average, each size has the same probability of being generated, i.e.

$$p_9 \approx p_{10} \approx p_{11} \approx p_{12} \approx p_{13} \approx \frac{1}{5},$$

Where p_i is the probability of a B-block being generated with a size of i.

Once the B-block sequence has been generated, MULTI then determines the location and size of the E-block that will be placed within each of those B-blocks. The range of sizes of E blocks is dependent upon the size of the parent B block and an understanding of the distribution of E-block sizes will give a clearer indication of the increase of randomization introduced due to E-block size randomization within the MULTI system.

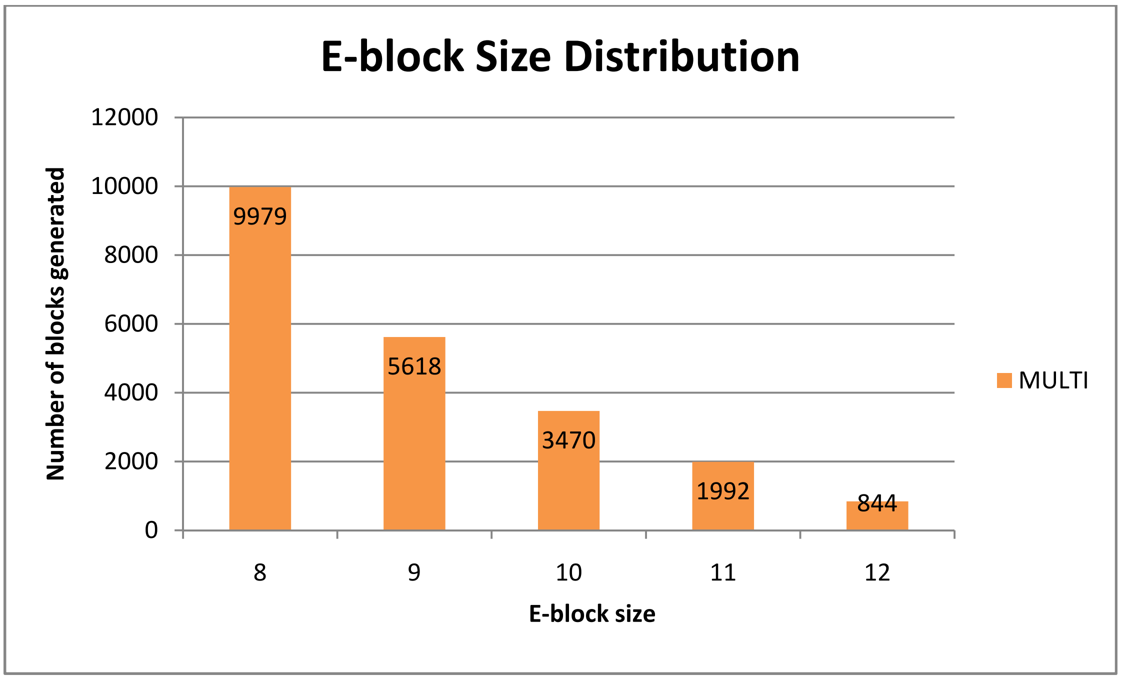


Figure 5-3: E-block size distribution for MULTI systems

Once again, the same 2048 by 1536 pixel image is used to generate a sequence of B-blocks along with their E-blocks. Figure 5-3 shows the distribution of E-blocks throughout the image. When MULTI was designed, the size of E-blocks was restricted to within a range bounded by $8 \leq e \leq B - 1$ where e is the size of the current E-block and B is the size of its parent B-block.

We note that for all the B-blocks of size 9, all their respective E-blocks would be of size 8. Similarly, B-blocks of size 10 can accommodate E-blocks of either size 8 or size 9. The probability of selecting an E-block of size 8 is the same as selecting one of size 9; therefore we expect that half of the B-blocks would be parent to E-blocks of size 8 and the other half parent to E-blocks of size 9. This behaviour can be extended to B-blocks of size 11, 12 and 13.

Based on the theory mentioned in Chapter 4, we expect the E-blocks generated throughout the image to have a certain, set, probability. We note that the probability of generating an E-block of size 8, P_{E8}:

$$P_{E8} = p_9 + p_{10} * p_8{}^{10} + p_{11} * p_8{}^{11} + p_{12} * p_8{}^{12} + p_{13} * p_8{}^{12},$$

$$where\ p_9 = p_{10} \ = p_{11} \ = p_{12} \ = p_{13} \ = \frac{1}{5}$$

$$and$$

$$p_8{}^{i} = probability\ of\ a\ B - block\ size\ i\ generating\ an\ E - block\ of\ size\ 8.$$

$$\therefore\ p_8{}^{10} = \frac{1}{2}, p_8{}^{11} = \frac{1}{3}, p_8{}^{12} = \frac{1}{4}, p_8{}^{13} = \frac{1}{5}.$$

$$\therefore P_{E8} = \frac{1}{5} + \left(\frac{1}{5} * \frac{1}{2}\right) + \left(\frac{1}{5} * \frac{1}{3}\right) + \left(\frac{1}{5} * \frac{1}{4}\right) + \left(\frac{1}{5} * \frac{1}{5}\right) = \frac{137}{300}.$$

Similarly, we have:

$$P_{E9} = \left(\frac{1}{5} * \frac{1}{2}\right) + \left(\frac{1}{5} * \frac{1}{3}\right) + \left(\frac{1}{5} * \frac{1}{4}\right) + \left(\frac{1}{5} * \frac{1}{5}\right) = \frac{77}{300},$$

$$P_{E10} = \left(\frac{1}{5} * \frac{1}{4}\right) + \left(\frac{1}{5} * \frac{1}{5}\right) = \frac{9}{100},$$

$$P_{E11} = \left(\frac{1}{5} * \frac{1}{5}\right) = \frac{1}{25}.$$

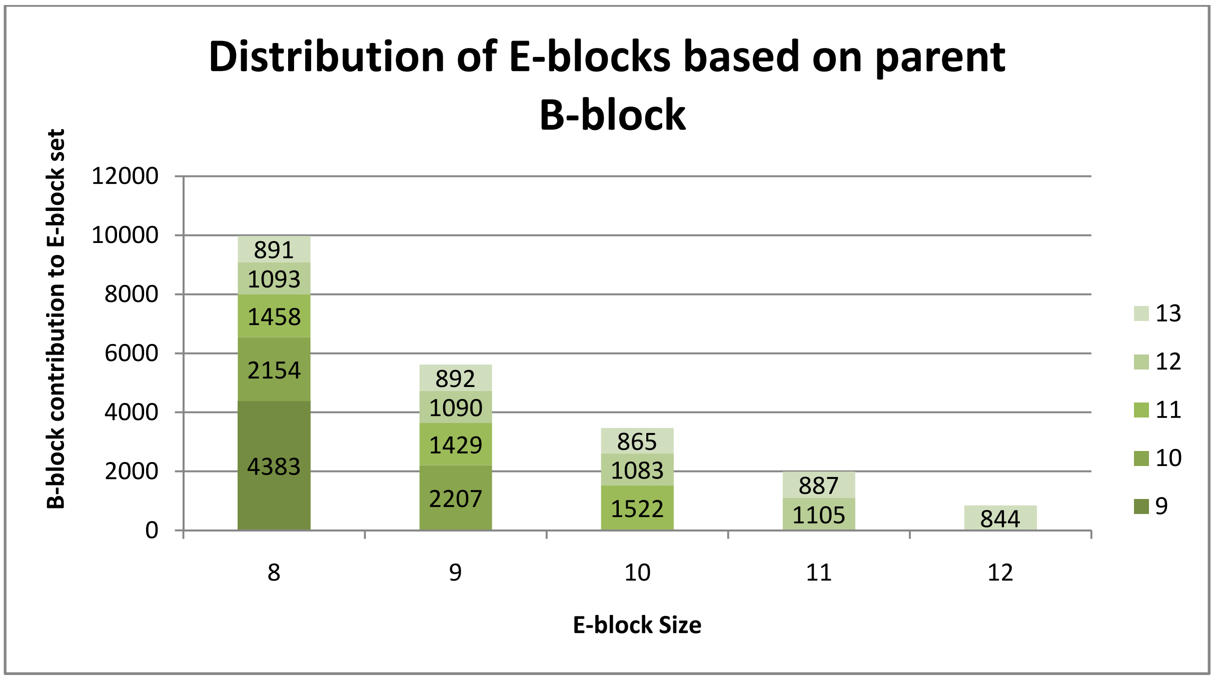


Figure 5-4: Distribution of E-block based on parent B-block

Figure 5-4 shows the distribution of E-blocks based on the contributions of their parent B-blocks. As was predicted, the majority of E-blocks of size 8 are all generated due to B-blocks of size 9. Similarly, we note that E-blocks generated by B-blocks of size 10 are evenly split between sizes of 8 and 9. This trend continues as expected for the remaining E-blocks.

Table 5-1 shows a comparison of the expected, theoretical E-block distribution ratio and the distribution ratio achieved when the MULTI system was implemented and simulated. We can see that although not perfectly matched, the implemented PRNG achieves the goal of randomizing E-block generation in the expected way.

Table 5-1: A comparison of Theoretical distribution and implemented distribution of E-block

E-block size	Theoretical	Simulated
8	0.457	0.456
9	0.257	0.256
10	0.157	0.158
11	0.089	0.091
12	0.040	0.039

Figure 5-5 shows the generation of B-blocks on a sample image. The image used is a 256 by 256 pixel greyscale image and the range used in the generation algorithm is 9-13. The yellow blocks in Figure 5-5 represent the boundaries of each generated B-block. We note that as motivated in Chapter 4, the B-blocks are placed in order of generation and not in any optimized manner. This approach clearly results in unused space throughout the image, an issue YASS does not have. However, even with this unused space, we note that at this range, MULTI still generates around the same number of B-blocks as YASS and therefore shares a very similar embed rate.

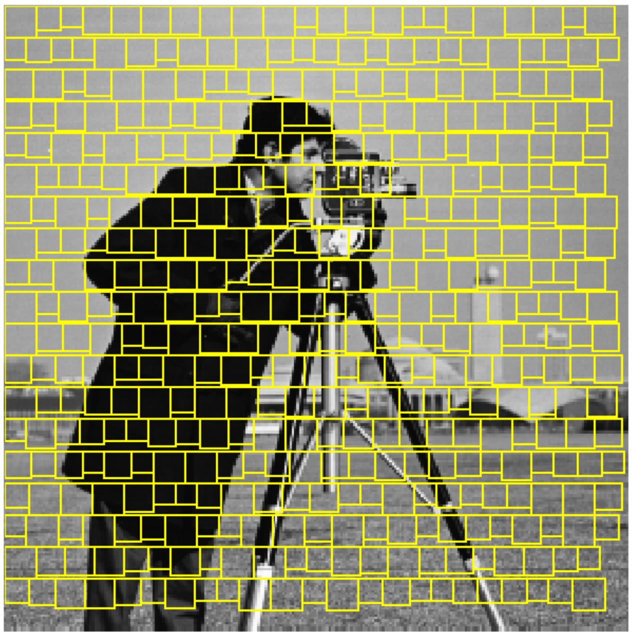


Figure 5-5: Generated B-block

Figure 5-6 shows the resulting E-blocks generated in the process of blocking the sample image. As mentioned in Chapter 4, for each B-block, a location and size is randomly assigned to generate its E-block. This process is repeated for all B-blocks within the image.

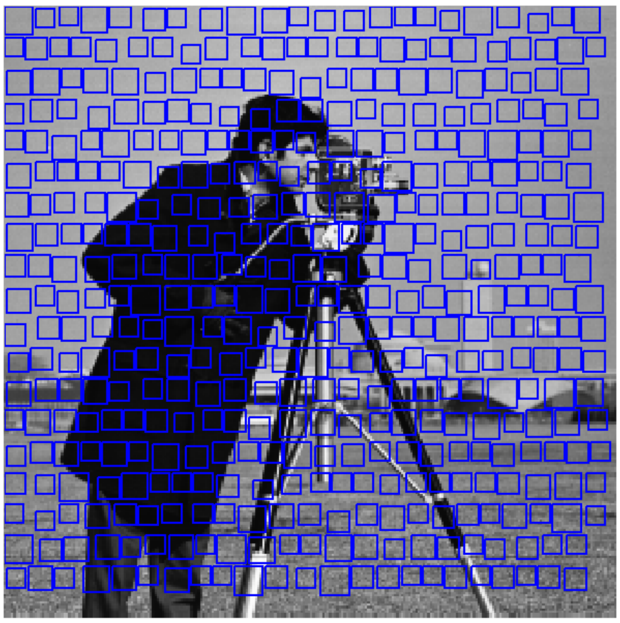


Figure 5-6: Generated E-blocks

5.2 Embed/Error Rates

The previous section presented the blocking scheme used in MULTI and presented a comparison of the blocking rates achieved by both YASS and MULTI. During the analysis, it was shown that after a certain threshold, MULTI would be far more effective at embedding data than the YASS framework. For this section, we aim to test the embed rate as well as the error rate for MULTI and compare it to YASS.

The aim is to investigate whether the introduction of the new embedding scheme, namely medium frequency embedding and masking region control via the aggression level, increases the embedding potential as compared to YASS. Therefore, we must select testing conditions that minimize the effects of other variables to ensure that we are comparing the effect of the embedding scheme on capacity rather than the effect of blocking on embedding.

To accomplish this, we make the restriction that YASS will be blocking with a B-block size of 12 and MULTI will use a range of 9-13. The selection of these two parameters results in each system generating very similar numbers of blocks, and thus the embedding scheme is what determines embedding capacity not block number.

5.2.1 Testing setup

To test for embedding, we perform the embedding procedure of YASS, HYBRID and MULTI on a set of images taken from the *"Zurich Natural Images"* database. This database is a group of 128 uncompressed high resolution photos taken of natural scenes. The term "Natural Images" refers to images that have not been digitally filtered or compressed; in other words, these images represent the original raw image capture by the camera.

The reason for selecting this database is due to the fact that the images included within the database covers a wide range of sample images. These range from simple scenes, with very little high frequency components, to very complex scenes with higher frequency components. The variety in images will allow for a more conclusive representation of natural images that can be used for hiding.

Since the aim of these tests is to determine the embedding rate and error rate for each of the three systems, the tests shall be divided into two stages:

1. Each system will be allowed to perform embedding as defined in Chapter 3 for YASS and Chapter 4 for HYBRID and MULTI. To determine the maximum embedding rate, the total number of candidate coefficients in each E-block generated in the cover image is counted. This number is then averaged to give an indication of each system's embedding rate *per E-block*.

2. For each E-block generated in the cover-image, the worst case message is embedded in to the candidate coefficients. The worst case message represents a message which requires that each candidate coefficient be changed in the embedding process.

   Once the entire cover-image has been blocked and all candidate coefficients changed due to embedding, the resulting coefficients are saved. The stego-image is then compressed using the JPEG Compression Standard. The resulting compressed image is then decoded to extract the candidate coefficients once more.

   The LSB of the newly extracted coefficients are compared to the coefficients saved before compression; if the LSBs are unchanged, then an embedded bit in that coefficient would have survived compression, otherwise it is considered an error and noted. Once again, the error rate is averaged so as to represent the error rate *per E-block*.

Since the stego-images are passed through a compression algorithm, we test each of the systems' performance against a few Quality factors to see how aggressive compression affects overall performance.

5.2.2 YASS Performance

Since the YASS framework uses a hiding quality factor during its embedding process, Q_h, we select a range of Q_h that optimizes the frameworks performance based on (Solanki K. et al 2008). Since in the case of YASS Q_h controls the number of potential candidate coefficients available in an E-block we expect that by keeping Q_h constant throughout the test, the embedding rate per E-block for YASS will remain constant. This is due to the fact that candidate coefficients are selected before compression takes place and thus will allows be present when coefficient selection is made.

Table 5-2: Embedding Rate per E-block for YASS (bits per E-block)

Compression Quality(Q_a)	Embedding Quality (Q_h)			
	100	**80**	**75**	**60**
60	16.79	14.97	12.55	9.91
70	16.79	14.97	12.55	9.91
80	16.79	14.97	12.55	9.91
90	16.79	14.97	12.55	9.91
100	16.79	14.97	12.55	9.91

Table 5-2 presents the embedding rates for the YASS framework in bits per E-block. As expected, we note that the embedding rate per E-block for YASS decreases as the embedding quality factor decreases. This is due to more candidate coefficients being quantized to zero as the quantization steps increase. We also note that the embedding rate of the framework is independent of the compression quality factor, since compression happens after embedding takes place and thus has no effect on the availability of candidate coefficients.

Since YASS attempted to embed in the first 19 low frequency coefficients, excluding the DC coefficient, we expected a peak embedding rate of around 19 bits per E-block. The simulation results suggest a slightly lower peak embedding rate of about 17 bits per E-block.

Table 5-3: Error rate per E-block for YASS (bits per E-block)

Compression Quality(Q_a)	Embedding Quality (Q_h)			
	100	**80**	**75**	**60**
60	10.85	9.03	6.48	4.01
70	10.43	8.61	6.13	3.74
80	9.47	7.76	5.47	3.32
90	8.04	6.57	4.61	2.78
100	6.76	5.52	3.87	2.33

Table 5-3 shows the error rate per E-block for YASS in bits per E-block. We notice that as the compression quality is lowered, the expected increase in error rate is observed. This is due to the fact that as the compression quality factor decreases, the image undergoes harsher compression. This results in errors in even low frequency coefficients which in turn corrupts the embedded message.

From the results shown in

Table 5-3, we can make a rather important observation. YASS is noticeably more efficient when $Q_a > Q_h$, that is to say, when the compression quality factor is higher than the quality factor used in the embedding procedure. This result was stated in (Solanki K. et al 2008) and reproduced during this test.

Comparing the error rate results with those obtained previously for the embedding rate, we find two extremes. Figure 5-7 and Figure 5-8 represent those extremes, namely the best case embedding, which results in the best possible overall embedding rate and the worst case embedding, which results in the lowest possible embedding rate.

Figure 5-7 shows the YASS embedding rate versus error rate with a compression quality factor of 100. This represents the best possible compression that the stego-image can undergo, short of not being compressed at all. We note that the best possible embedding rate will achieved when $Q_a = Q_h = 100$.

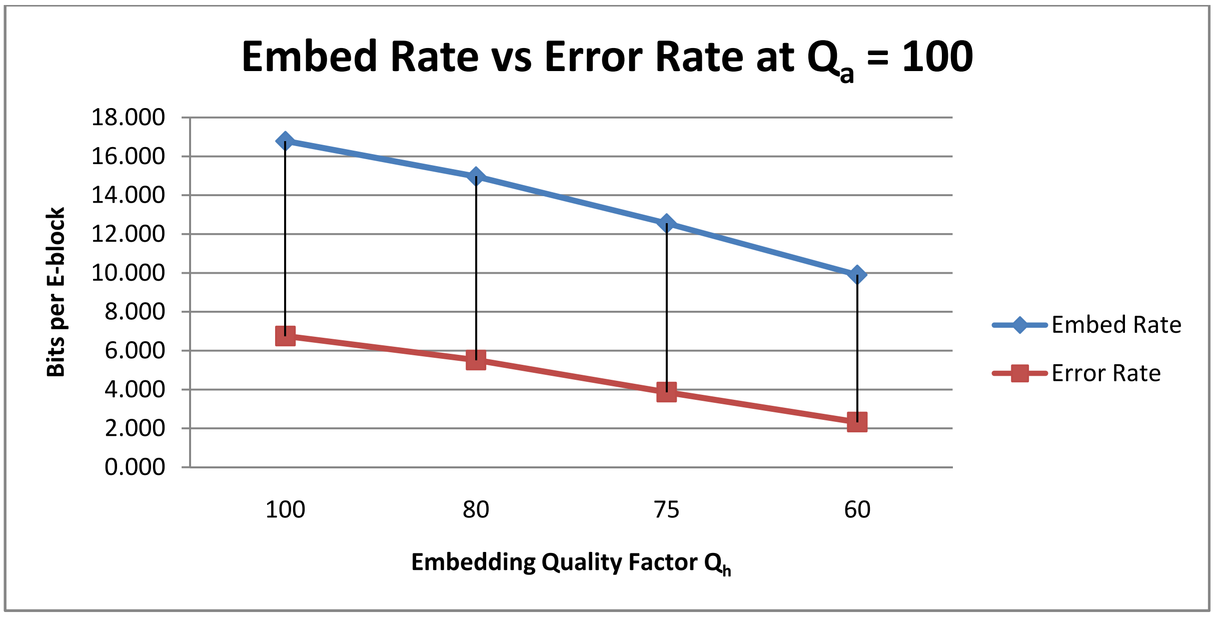


Figure 5-7: Embedding Rate versus Error Rate for YASS at $Q_a = 100$

Figure 5-8 on the other hand shows the worst possible configuration for the framework. With a compression quality factor of 60, we find that peak error rate increases from 6.75 to 10.84, or an increase of about 60%. Therefore the embedding efficiency of the YASS framework is greatly reduced.

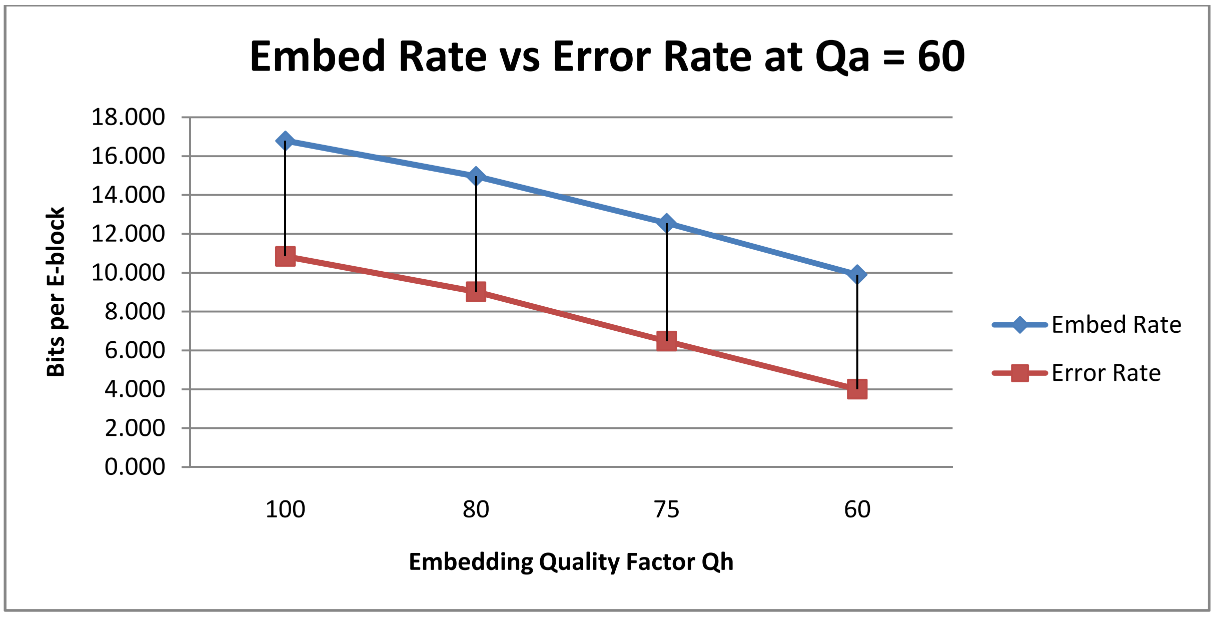


Figure 5-8: Embedding Rate versus Error Rate for YASS at $Q_a = 60$

The embedding capacity of a steganographic system is defined as:

$$Embedding\ Capacity = \frac{Maximum\ Embedded\ message\ size}{(Stegoimage\ size - Coverimage\ size)}$$

If the Embedding Capacity results in a ratio of one or greater, i.e. $Embedding\ Capacity \geq 1$, this implies that the information embedded into the cover-image is greater than the net change in cover-image size.

Taking the best case scenario for YASS, $Q_a = Q_h = 100$, we calculate the highest embedding capacity for the framework using a set of 500 by 500 pixel images. We therefore have:

$$Maximum\ Embedded\ message\ size = 22\,459\ bits = 2807.375\ bytes,$$

$$Stegoimage\ size = 224\,218\ bytes,$$

$$Coverimage\ size = 211\,451\ bytes,$$

$$\therefore Embedding\ capacity = \frac{2807.375\ bytes}{(224\,218\ bytes - 211\,451\ bytes)},$$

$$Embedding\ capacity = \frac{2807.375}{12\,767} = 0.22.$$

The embedding capacity of YASS would suggest that the system increases the size of the cover-image by about 5 times the size of the message you embed. For low embed rates this isn't a huge problem but as the message size increases, the increase in file size may result in suspicion.

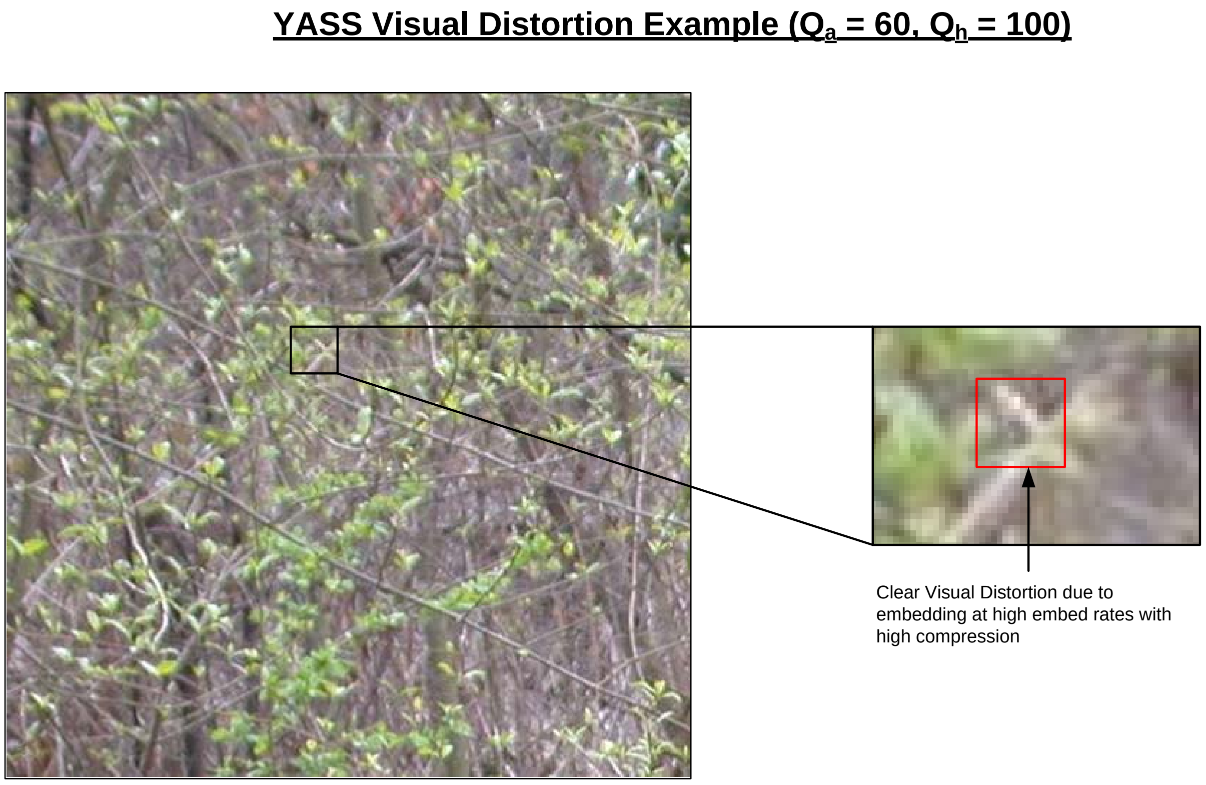

YASS Visual Distortion Example (Q_a = 60, Q_h = 100)
Clear Visual Distortion due to embedding at high embed rates with high compression

Figure 5-9: Example of Visual Distortion due to embedding with YASS

Figure 5-9 shows an example of visual distortion that can occur when embedding data using YASS. In this case a 500 by 500 pixel image is used as cover for very aggressive embedding. An embedding quality factor of $Q_h = 100$ was used, but to demonstrate the presence of errors, a compression quality factor of $Q_a = 60$ was used. Due to this combination, clear visual distortion occurs, although lost in the original image, zooming into a region shows the introduction of high frequency noise, something not expected in an image compressed with such a low quality factor.

We now present the performance analysis of the HYBRID system.

5.2.3 HYBRID Performance

The HYBRID system was created as a stepping stone between the original YASS framework and the final MULTI system proposed. The HYBRID system uses the same embedding scheme as MULTI but instead of allowing the randomization of E-blocks, it retains the original E-block size of 8. This allows us to perform a more direct comparison between the embedding scheme used in YASS and that used in HYBRID/MULTI.

Similar to the behaviour observed for YASS, we expect that the HYBRID system's embedding rate is independent of the compression quality factor used. Once more this is due to the fact that HYBRID embeds data based on an aggression level control, A_l, which takes place before the stego-image enters the compression channel and therefore is independent from the compression quality factor of the channel.

Table 5-4: Embedding Rate per E-block for HYBRID (bits per E-block)

Compression Quality (Q_a)	**Aggression level** (A_l)							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
60	3.37	9.92	19.41	31.67	40.63	46.41	48.84	48.84
70	3.37	9.92	19.41	31.67	40.63	46.41	48.84	48.84
80	3.37	9.92	19.41	31.67	40.63	46.41	48.84	48.84
90	3.37	9.92	19.41	31.67	40.63	46.41	48.84	48.84
100	3.37	9.92	19.41	31.67	40.63	46.41	48.84	48.84

Table 5-4 presents the embedding rate in bits per E-block for the HYBRID system. We note that as mentioned, the embedding rate is independent of the compression quality factor used, but is very much dependent on the level of aggression chosen.

As the aggression level of the HYBRID system is increased, we notice a steady increase in the bits per E-block. The peak embedding rate of about 49 bits per E-block represents embedding in just over 75% of all coefficients available within the 8 by 8 E-block. This is due to the change in embedding scheme, utilizing the masking system that allows for progressively more coefficients being selected as candidates as the aggression level increases.

On the other hand, we note that our new embedding scheme also allows for a very low embedding rate to be obtained, about 3 bits per E-block. This gives the user far more control over how much information they wish to embed into the cover-image, something the YASS framework could not provide.

Since the embedding scheme used in HYBRID embeds in the medium frequency coefficients rather than the low frequency coefficients used in YASS, we expect that the error rate for HYBRID system to be higher that YASS, especially when the stego-image undergoes high compression.

Table 5-5: Error rate per E-block for HYBRID (bits per E-block)

Compression Quality (Q_a)	**Aggression level** (A_l)							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
60	1.97	5.89	11.70	19.38	25.09	28.84	30.71	30.71
70	1.97	5.88	11.70	19.38	25.09	28.85	30.71	30.71
80	1.97	5.88	11.69	19.37	25.07	28.83	30.69	30.69
90	1.94	5.81	11.54	19.12	24.75	28.45	30.28	30.28
100	1.84	5.50	10.94	18.13	23.49	27.02	28.78	28.78

Table 5-5 presents the error rate of the HYBRID system in bits per E-block. As expected, the error rate per E-block increases as the aggression level increases. The peak error occurs when the highest aggression level is used at the lowest compression quality, an error rate of 31 bits per E-block.

As mentioned earlier, the error rate is not only affected by the aggression level but is also affected by the compression quality factor used in compressing the stego-image as it enters the channel. This can be seen by focusing on one aggression level. For example, looking at the error rates at different compression quality factors with an aggression level of 8, we notice that the error increases by 7% due to the decreasing in compression quality factor.

Comparing HYBRID's embedding and error performance, we notice with $Q_a = 100$, HYBRID achieves maximum performance. Figure 5-10 shows the embedding rate versus error rate for the HYRBID scheme at $Q_a = 100$. We note that as the aggression level increases from 1 to8 the embedding rate increases at a faster rate than the error rate. It is within this region where HYBRID becomes more efficient. With an aggression of 7 or 8, we notice the greatest difference between the embedding rate and error rate, a difference of about 20 bits. This represents the most efficient embedding strategy for the HYBRID scheme.

Figure 5-11 shows the opposite of Figure 5-10. This graph represents the worst embedding strategy for the HYBRID scheme. By using a compression quality factor of $Q_a = 60$, the HYBRID scheme finds its maximum error rate jump from 28.8 to 30.7. This 7% increase in error rate is noticed throughout all aggression levels at this compression quality. The increase in error is due to the harsher compression occurring at such low compression qualities and since most data is saved within medium frequency coefficients, the higher error rates were expected.

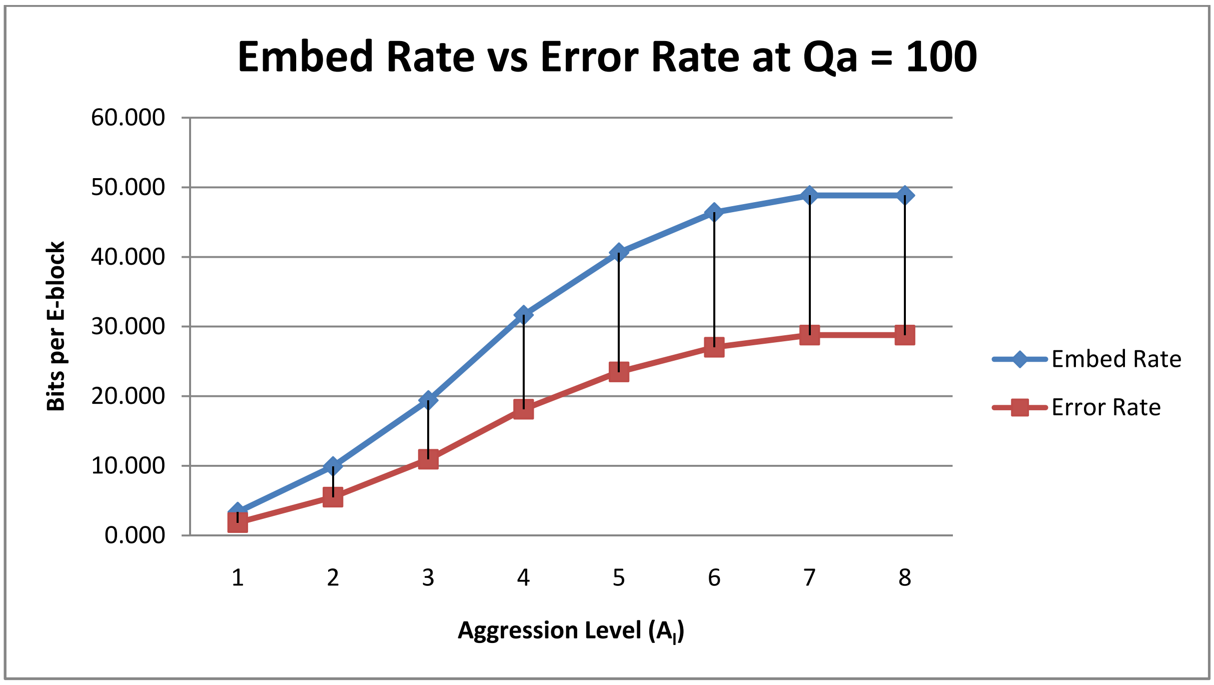


Figure 5-10: Embedding Rate versus Error Rate for HYBRID at $Q_a = 100$

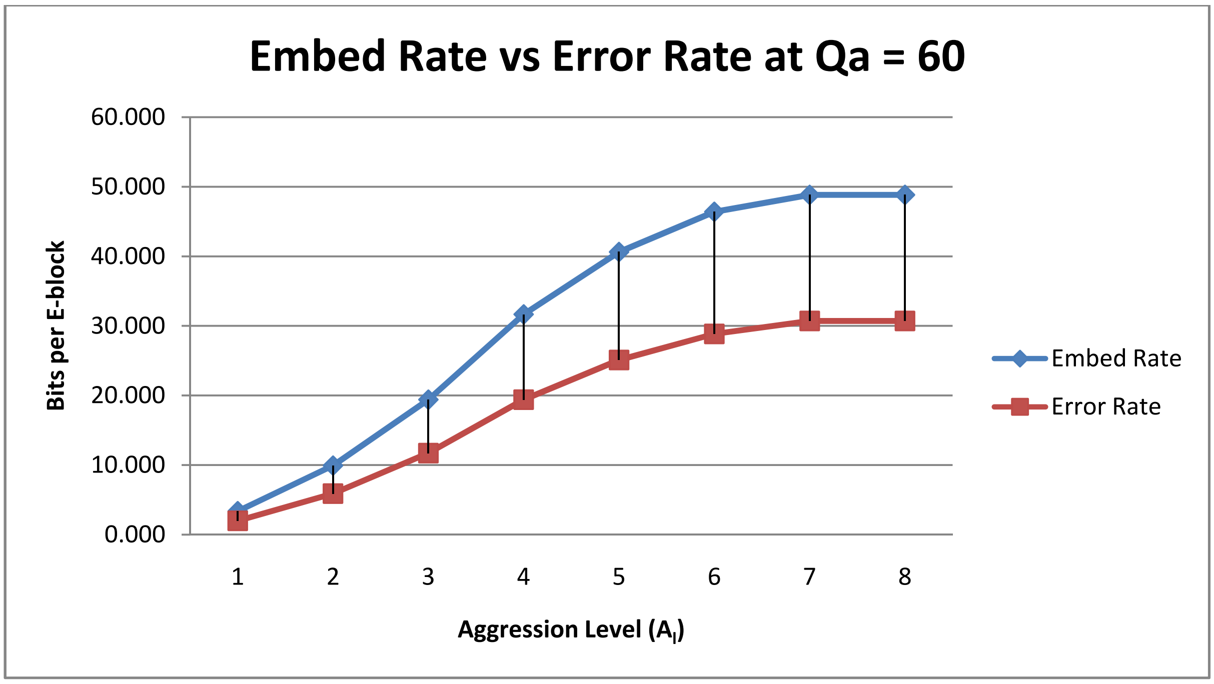


Figure 5-11: Embedding Rate versus Error Rate for HYBRID at $Q_a = 60$

We now determine the embedding capacity for the HYBRID scheme by using a set of 500 by 500 pixel images and selecting the best possible embedding strategy for HYBRID, i.e. $Q_a = 100 \; and \; A_l = 8$,

$$Maximum\ Embedded\ message\ size = 117\,678\ bits = 14\,709.75\ bytes,$$

$$Stegoimage\ size = \ 220\,447\ bytes,$$

$$Coverimage\ size = \ 211\,451\ bytes,$$

$$\therefore Embedding\ capacity = \frac{14\,709.75\ bytes}{(220\,447\ bytes - \ 211\,451\ bytes)},$$

$$Embedding\ capacity = \frac{14\,709.75}{8996} = 1.64.$$

We notice that the HYBRID scheme manages to achieve an embedding capacity of greater than one. What this represents is that the HYBRID scheme can embed 1.6 bits of data for every bit increase in the cover-image. Having such a high embedding capacity allows for larger message embedding without the risk of raising too much suspicion due to file size increase. This is a very desirable property of the HYBRID scheme.

Finally, Figure 5-12 shows a sample image in which a worst case message was embedded using the HYBRID scheme. For this embedding, we used the worst possible strategy to create the most visual distortion, namely $Q_a = 60 \; and \; A_l = 8$. Unlike the YASS frame, it is virtually impossible to visually detect artefacts or imperfections introduced by the embedding.

This represents the motivation behind embedding in the medium frequency components of the image. Since these components have little effect on the image, embedding data in them has minimal effect on the image.

Figure 5-12: Example of embedding with HYBRID

5.2.4 MULTI Performance

The MULTI system represents the full proposal of this thesis. It introduces two extra layers of randomization to the YASS framework as well as a new embedding scheme to allow for higher embedding rates without risking much visual distortion. The HYBRID scheme discussed in the previous section was developed during the design of the MULTI system, focusing only in the randomization of the bigger B-blocks and the introduction of the new embedding scheme.

MULTI now introduces the second layer of randomization, namely the randomization of the E-blocks. This randomization is expected to improve upon the embedding rates and embedding capacity shown by the HYBRID scheme as well as aid in increasing security against attacks, discussed later.

Since the MULTI system allows the user to set the range of B-block sizes as well as the aggression level used during the embedding process, we select a B-block range used in the HYBRID test, namely a range of 9-13. This allows the MULTI system to generate the same number of B-blocks on average as YASS and thus allow for a better comparison of embedding scheme performance as the aggression level of the system is changed. Once again we perform our performance simulation of the full set of images available in the *Zurich Natural Image Database.*

Table 5-6: Embedding Rate per E-block for MULTI (bits per E-block)

Compression Quality (Q_a)	Aggression level (A_l)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
60	3.68	10.30	19.93	32.41	44.19	53.31	59.13	62.11	63.51	64.05
70	3.68	10.30	19.93	32.41	44.19	53.31	59.13	62.11	63.51	64.05
80	3.68	10.30	19.93	32.41	44.19	53.31	59.13	62.11	63.51	64.05
90	3.68	10.30	19.93	32.41	44.19	53.31	59.13	62.11	63.51	64.05
100	3.68	10.30	19.93	32.41	44.19	53.31	59.13	62.11	63.51	64.05

Table 5-6 presents the embedding rates per E-block for the MULTI system. These embedding rates were determined by averaging the number of bits available for use in each generated E-block across all images in the database used. Much like the HYBRID scheme, we expect that the embedding rate is independent of the compression quality used in the channel but rather dependent on the aggression level selected. This can be seen in Table 5-6. We notice a peak embedding rate of 64 bits per E-block.

This peak embed rate is impossible to achieve in both the YASS framework and the HYBRID scheme since their E-blocks are restricted to 8 by 8 pixels. Therefore both systems have at most 64 coefficients, one of which is the DC coefficient. Therefore the theoretical maximum for each of those schemes is 63 bit per E-block. Since MULTI allows for a range of E-block sizes, we note that the average E-block size is now greater than 8 by 8 and thus allows for more candidate coefficients.

This presents the first major advantage of introducing the randomization of E-block size. We are now able to embed more data into an E-block on average than there are coefficients in the other systems' E-blocks. We also note that the aggression level once again provides the user with much greater control over how much information they wish to embed into the cover-image. Using an aggression level of 1, the MULTI system allows users to embed up to 3 bits per E-block.

This massive increase in embedding rate comes at a price. Table 5-7 presents the error rate per E-block for the MULTI system. We notice that much like the HYBRID scheme, the error rate is dependent on both the aggression level and the compression quality factor. As the aggression level is increased and higher frequency coefficients are selected as candidates for hiding, the higher the chance of them being changed due to compression and thus the higher the likely hood of error.

Table 5-7: Error Rate per E-block for MULTI (bits per E-block)

Compression Quality (Q_a)	Aggression level (A_l)									
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
60	2.15	6.08	11.95	19.69	27.12	32.95	36.85	38.84	39.78	40.14
70	2.15	6.08	11.94	19.69	27.12	32.96	36.85	38.83	39.77	40.13
80	2.15	6.08	11.94	19.69	27.12	32.95	36.83	38.81	39.75	40.11
90	2.12	6.01	11.78	19.43	26.75	32.50	36.32	38.27	39.20	39.55
100	2.01	5.68	11.16	18.41	25.36	30.82	34.46	36.32	37.20	37.54

Similarly, as the compression quality factor is decreased, the stego-image is compressed more harshly with progressively lower frequency coefficients being dropped during the compression. This in turn results in an increase in error. We note a peak error rate of 40 bits per E-block when $Q_a = 60 \; and \; A_l = 10$.

When comparing the increase of the error rate due to compression quality, we notice that for an aggression level of 10, the error rate changes from 37.5 to 40.1. This represents an increase of 7 %. We notice that this increase of 7% is seen for each aggression level as the compression quality is decreased.

Comparing the embedding rates and error rates of the MULTI systems, we can generate the best case embedding and worst case embedding strategies. Figure 5-13 shows the performance of MULTI at a compression quality factor of $Q_a = 100$. We note that as the aggression level increase from 1 to 8, the embedding rate increases at a faster rate than the error rate.

At an aggression level of 9 and 10, although achieving the best performance for the system, we notice that the rate of growth is petering out. However, at an aggression level of 10, we do have the greatest difference between the embedding rate and error rate, a difference of about 26 bits.

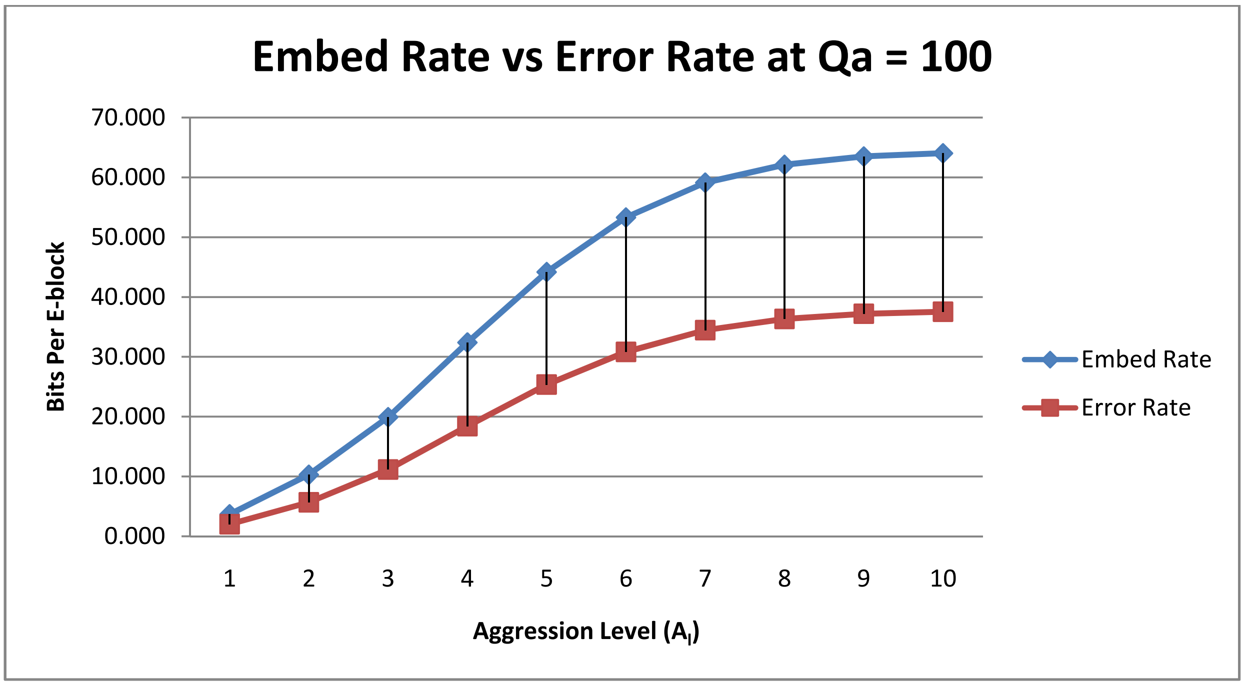


Figure 5-13: Embedding Rate versus Error Rate for MULTI at $Q_a = 100$

Figure 5-14 on the other hand represents the worst case embedding strategy. Selecting a compression quality factor of $Q_a = 60$, we find that the error rate increases. Once again, this increase in error is due to the harsher compression that occurs when the stego-image enters the compression channel. At such low compression quality factors, a lot of medium frequency components are removed, and most if not all high frequency components.

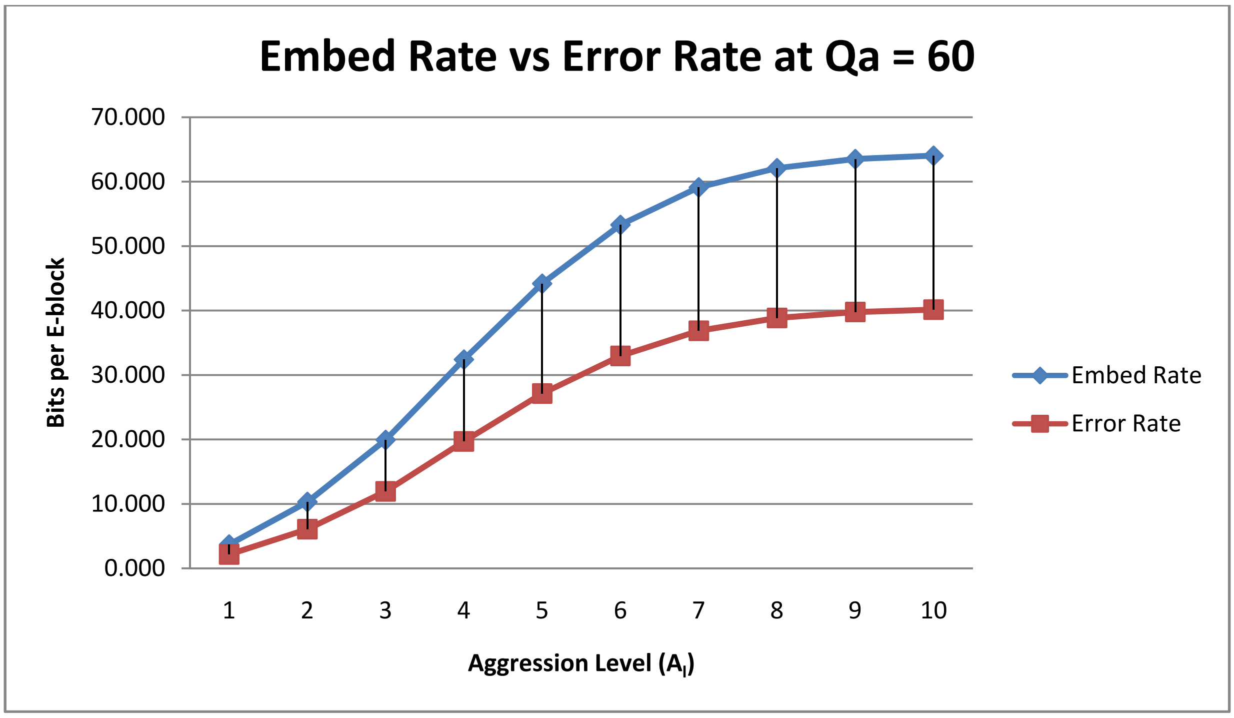


Figure 5-14: Embedding Rate versus Error Rate for MULTI at $Q_a = 60$

We now calculate the embedding capacity of the MULTI system by once again using a set of 500 by 500 pixel images and embed using the most optimum embedding strategy, $Q_a = 100 \; and \; A_l = 10$,

$$Maximum\ Embedded\ message\ size = 117\,973 bits = 14\,747\ bytes,$$

$$Stegoimage\ size = \ 220\,342\ bytes,$$

$$Coverimage\ size = \ 211\,451\ bytes,$$

$$\therefore Embedding\ capacity = \frac{14\,747\ bytes}{(220\,342\ bytes - \ 211\,451\ bytes)},$$

$$Embedding\ capacity = \frac{14\,747}{8891} = 1.66.$$

From the above result we notice that the MULTI system manages to achieve an embedding capacity of greater than one. Much like HYBRID, this represents the system's ability to embed data at a higher rate than it increases the file size of the cover-image. For every 1.66 bits of data embedded using the MULTI system, the cover-image only increases by one bit. This allows for higher capacity embedding without risking suspicion due to file size increase. Much like the HYBRID scheme, this is a very desirable feature of the MULTI system.

Finally, Figure 5-15 presents a 500 by 500 pixel image that demonstrates the embedding of a worst case message using the MULTI system. To maximize visual distortion, we utilized the worst case embedding strategy, i.e. $Q_a = 60 \; and \; A_l = 10$. Once again, this example shows the benefits of the new embedding scheme.

By embedding data in the medium frequency coefficients, we note the effect of the embedding is virtually impossible to see.

Figure 5-15: Example of embedding with MULTI

5.2.5 Comparing the Systems

The previous three sections presented the performance of each system separately. We now present an analysis of those results with relation to the other systems. The main focus of this section is to determine if the newly proposed system, MULTI, achieves any improvements over the original YASS framework.

Before we compare YASS directly to MULTI, we first compare YASS to the HYBRID scheme.

5.2.5.1 YASS vs. HYBRID

As mentioned before, the HYBRID scheme is a stepping stone in the path of developing the final MULTI system. It utilizes the randomization of B-blocks used in the MULTI system while keeping the 8 by 8 E-block size used by the YASS framework. It also incorporates the new embedding scheme designed for the MULTI system.

As mentioned at the start of the performance analysis, we set the B-block size for YASS equal to B = 12 and set the range for HYBRID to 9-13. This meant that both systems generate about the same number of B-blocks on average per image and since the E-block size is the same for both YASS and HYBRID, these two systems have the same number of coefficients to work with per E-block and thus the embedding rates are affected by the scheme used for embedding rather than the number of coefficients available. Therefore this comparison will give a good indication of the newly designed embedding schemes performance as compared to the scheme used in the YASS framework.

5.2.5.1.1 Embedding Capacity

When comparing the two embedding capacities calculated previously, one gets a sense of just how much of an improvement the HYBRID system is over YASS. YASS achieved an embedding capacity of 0.22, while HYBRID, on the same set of images, achieved an embedding capacity of 1.64.

This represents an increase of 7.4 times from YASS to HYBRID. Out of context this seems like a huge improvement. Although the increase is substantial, the important factor to note is that HYBRID achieves an embedding capacity of greater than one. Unlike YASS which increases the file size by five bits per bit embedded, embedding using HYBRID does not rapidly increase the stego-image's file size.

Although this improvement in embedding capacity may demonstrate HYBRID's ability to embed efficiently into a cover-image, one must realize that there is a definite trade-off for achieving such a high embedding capacity. This trade-off is clearly the introduction of a lot more error.

5.2.5.1.2 Error Rate

When designing the embedding scheme used in HYBRID and MULTI, it was noted that by embedding in medium frequency coefficients, a trade-off was being made between risking losses due to compression and introducing visual distortion. It was therefore an expected result when the error rate of HYBRID was higher than the error rate of YASS.

Comparing the maximum error rate of each system, we notice that YASS achieved a peak error rate of 10.85 bits per E-block while HYBRID achieved a peak error rate of 30.71 bits per E-block. This represents an increase of 2.83 times from YASS to HYBRID. This is most certainly a huge increase in error rate, but this was expected when you consider the region in which HYBRID embeds data is heavily compressed at lower compression quality factors.

In order to make a fair comparison, it is not enough to look to the error rate, but we must also consider the gain in the embedding rate

5.2.5.1.3 Embedding Rate

The point of designing a new embedding scheme for HYBRID and MULTI was to try and achieve a higher embedding capacity for the systems. Overall, the embedding capacity is dependent mostly on how many bits of data the scheme can embed per E-block. Therefore the more bit we can embed in a single E-block the better.

YASS managed to achieve a peak embedding rate of 16.8 bits per E-block, just shy of its theoretical maximum of 19 bits per E-block. On the other hand, HYBRID managed an embedding rate of 48.84 bits per E-block. This represented an increase of 2.91 times from YASS to HYBRID. An interesting observation to make at this point is that although the error rate increased by 2.83 times, the embedding rate increased more. Therefore, we have a net effect of being able to embed more overall by using HYBRID over YASS.

A final comparison can be made between the two systems by matching their error rates.

5.2.5.1.4 Matching Error Rates

Since HYBRID allows the user to control how aggressively they wish to embed into a cover-image, comparing HYBRID's performance with YASS directly is rather tricky since HYBRID generates a range of error and embedding rate pairs.

Therefore the easiest way to determine which system performs better would be to select an error rate from one system and compare both systems embedding rates at that error rate. This is a direct comparison which can then be used to determine which system has a better overall embedding rate.

Since YASS has the lower error rates, we select the worst case error rate for YASS as our baseline. Therefore we have:

$$Error_{YASS} = 10.85\ bits\ per\ E - block,$$

At this error rate we note that the embedding rate for YASS is:

$$Embed_{YASS} = 16.79\ bits\ per\ E - block,$$

We can therefore calculate the net difference between the embedding rate and error rate:

$$Net\ difference\ for\ YASS = 16.79 - 10.85 = 5.94\ bits\ per\ E - block.$$

This represents the number of bits YASS generates over the error rate. In other words, this represents the number of useful bits that the system provides a user. In other words, a user of the system has a throughput of 5.94 bits per E-block.

We now select a worst case error rate for HYBRID that matches that of YASS as closely as possible.

$$Error_{YASS} \approx Error_{HYBRID} = 11.7\ bits\ per\ E - block,$$

At this error rate results in an embedding rate of:

$$Embed_{HYBRID} = 19.41\ bits\ per\ E - block,$$

We now find the net difference for the HYBRID system:

$$Net\ difference\ or\ HYBRID = 19.41 - 11.7 = 7.71\ bits\ per\ E - block.$$

With the two net differences, we can now compare the two systems. The comparison shows that HYBRID has a net improvement of:

$$7.71 - 5.94 = 1.77\ bits\ per\ E - block.$$

Therefore, given a similar error rate to YASS, HYRBID generates a throughput of 1.8 bits more per E-block that YASS. This shows the HYBRID system has a higher embedding capacity than YASS at similar error rates.

We now perform the same analysis for YASS and MULTI.

5.2.5.2 YASS vs. MULTI

As mentioned at the beginning of the section, the comparison between YASS and MULTI is the main focus of the section. Although we showed that the HYBRID scheme achieves a higher embedding capacity than YASS, it does not represent the full system. MULTI not only incorporates all of HYBRID's structure but also includes an added layer of randomization, introduced by making the E-block sizes random.

This randomization results in just over half the E-blocks generated in the image having a size greater than 8 by 8. It is therefore expected that the embedding rate for the MULTI system would increase since there are more coefficients available within the E-block for selection. We therefore start by investigating the two systems' embedding capacity.

Once again, all the results generated were done so by performing a simulation on the full set of 128 images available to us in the image database.

5.2.5.2.1 Embedding Capacity

As mentioned earlier, YASS managed to achieve an embedding capacity of 0.2199. This represented an increase of 5 bits for every bit of data embedded into the cover-image. On the other hand, MULTI managed to achieve an embedding capacity of 1.6586. This represents an increase of 7.5425 times from YASS to MULTI.

Comparing this to the increase achieved with HYBRID, we notice that MULTI manages to improve slightly on HYBRID's performance. That said, the same trade-off is present for the MULTI system as was present for the HYBRID scheme, i.e. an introduction of a higher error rate.

5.2.5.2.2 Error Rate

Since MULTI utilizes the same embedding scheme as HYBRID, it is expected that it also suffers for the increase of error rate noted by the HYBRID system. MULTI also has to contend with potentially higher error rates due to a bigger E-block size. More medium frequency coefficients results in more data embedding in a region which is affected heavily by compression.

Due to this we note that MULTI achieves a peak average error rate of 40.14 bits per E-block, while YASS achieves a peak average error rate of 10.85 bits per E-block. This represents an increase of 3.7 times from YASS to MULTI, which is almost a 30% increase of the YASS/HYBRID comparison. Another interesting point to note is that the peak error rate of MULTI is 33% larger than the peak error rate of HYBRID.

Once again, it is not enough to take into consideration the error rate but it is necessarily to also consider the increase in the embedding rate.

5.2.5.2.3 Embedding Rate

Since the E-blocks generated with MULTI have the potential to be greater than the 8 by 8 E-blocks used by YASS and HYBRID, we expect a much higher embedding rate for MULTI than the two other systems. With the range of 9-13 used for this simulation, we note that on average; the E-block size in MULTI is 9 by 9, which generates just over 26% more coefficients than the 8 by 8 E-blocks used in YASS and HYBRID.

This increase of the number of coefficients that can be used for hiding explains the reason that MULTI achieves a peak embedding rate of 64.05bits per E-block as compared to YASS's 16.8 bits per E-block. This represents an increase of 3.8 times from YASS to MULTI. This increase is compares to YASS almost in the same fashion as HYBRID's did. Although there is a huge increase in the error rate, we find that the embedding rate has increased more. This would suggest that MULTI is more efficient at embedding than YASS and thus achieves a higher embedding capacity.

We perform the same error matching comparison to complete the analysis of the two systems.

5.2.5.2.4 Matching Error Rate

Using the same strategy used in the YASS vs. HYBRID comparison, we select YASS's peak error rate as the baseline to perform our comparison:

$$Error_{YASS} = 10.85\ bits\ per\ E - block,$$

At this error rate we note that the embedding rate for YASS is:

$$Embed_{YASS} = 16.79\ bits\ per\ E - block,$$

We can therefore calculate the net difference between the embedding rate and error rate:

$$Net\ difference\ for\ YASS = 16.79 - 10.85 = 5.94\ bits\ per\ E - block.$$

We now select a worst case error rate for HYBRID that matches that of YASS as closely as possible.

$$Error_{YASS} \approx Error_{MULTI} = 11.95\ bits\ per\ E - block,$$

At this error rate results in an embedding rate of:

$$Embed_{MULTI} = 19.926\ bits\ per\ E - block,$$

We now find the net difference for the HYBRID system:

$$Net\ difference\ or\ HYBRID = 19.926 - 11.95 = 7.976\ bits\ per\ E - block.$$

We therefore note that MULTI has almost 8 bits more per E-block to match YASS.

With the two net differences, we can now compare the two to determine which show that MULTI has a net improvement of:

$$7.976 - 5.94 = 2.036\ bits\ per\ E - block.$$

Therefore, given a similar error rate to YASS, MULTI generates 2 bits more per E-block that YASS. This shows the MULTI system has a higher throughput of useful bits than YASS. When you consider the system as a whole, we note that MULTI's design allows for the generation of more B-blocks than YASS and thus more E-blocks to allow for higher embedding. And when the embedding scheme is considered, the newly developed MULTI system out performs YASS.

5.3 Peak Signal to Noise Ratio

As mentioned in Chapter 2, a possible way of detecting visual distortion is through the use of the Peak Signal to Noise Ratio (PSNR). This is a ratio of the maximum possible power of a signal and the power of the noise that can be injected into the signal due to the channel. In the case of images, PSNR is normally used as a way of representing noise injected due to compression. In the case of steganography, it is used as a way of testing for any visual distortion that may have occurred due to the embedding process.

The PSNR for a monochrome (grey scale) image is defined as:

$$PSNR = 10.\log_{10}\left(\frac{{MAX_I}^2}{MSE}\right)$$

$$where\ MAX_I = maximum\ possible\ pixel\ value = 255,$$

$$MSE = \frac{1}{M * N}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j) - K(i,j)]^2.$$

$where\ we\ define\ I\ and\ K\ as\ two\ monochrome\ images.$

For colour images, the definition of PSNR is the same, with the exception of the MSE now being the overall sum of the squared differences divided by the size of the images times 3 i.e.

$$MSE = \frac{1}{M*N*3}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}\{[I_R(i,j) - K_R(i,j)]^2 + [I_G(i,j) - K_G(i,j)]^2 + [I_B(i,j) - K_B(i,j)]^2\}.$$

To give an idea of the expected PSNR values, Table 5-8 presents a list of PSNR values for different compression quality factors. These values were generated by averaging the PSNR for 128, 500 by 500 pixel colour images compressed at each of those quality factors. We notice that for acceptable image qualities, PSNR is normally between 30dB to 50dB, the larger the PSNR the better the quality for the image.

Table 5-8: PSNR values for different Compression Quality Factors

	Compression Quality Factor								
	60	**65**	**70**	**75**	**80**	**85**	**90**	**95**	**100**
PSNR (dB)	36.43	36.76	37.06	38.16	40.29	42.33	42.95	46.25	49.02

Although PSNR is good for representing compression quality, as a tool for detecting visual distortion it is far from reliable. A simple example shall be used to demonstrate why this is the case.

Taking the sample image shown in Figure 5-16, we compress it using a range of quality factors, ranging from 60 to 100. Once these compressed images have been saved, we calculate the PSNR and present them in Table 5-9.

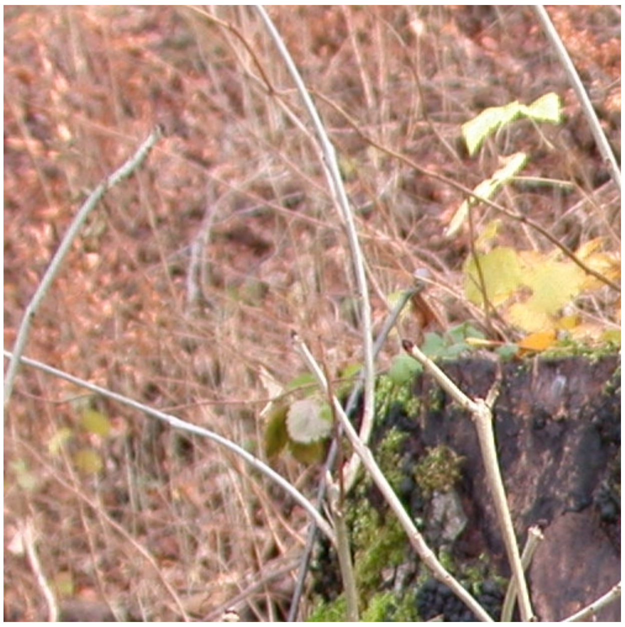


Figure 5-16: Sample Image

Table 5-9: PSNR results for sample image at different compression quality factors

	Compression Quality Factor								
	60	**65**	**70**	**75**	**80**	**85**	**90**	**95**	**100**
PSNR (dB)	37.24	37.69	38.09	38.65	40.70	42.71	43.76	46.28	49.31

We now take the original image and edit the colour range of a group of pixels in the upper left corner, generating the image shown Figure 5-17. Calculating the PSNR of this generated image, we find it to be PSNR = 42.82dB.

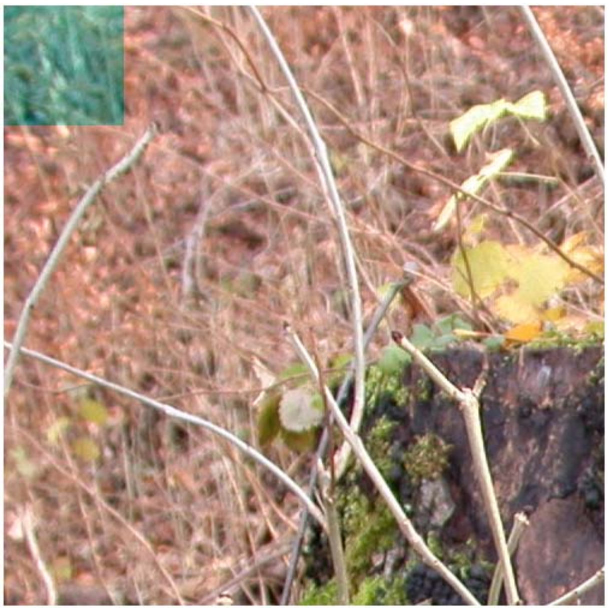


Figure 5-17: Edited Sample Image

Looking at Table 5-9 we see that the PSNR value obtained by the generated image is very close to the value obtained with a compression quality factor of 85. Figure 5-18 shows the two images side by side. It is clear that the PSNR value in this case misrepresents the distortion in the images since the generated image on the left has far clearer distortion than the compressed image of the right, which appears almost identical to the original sample image.

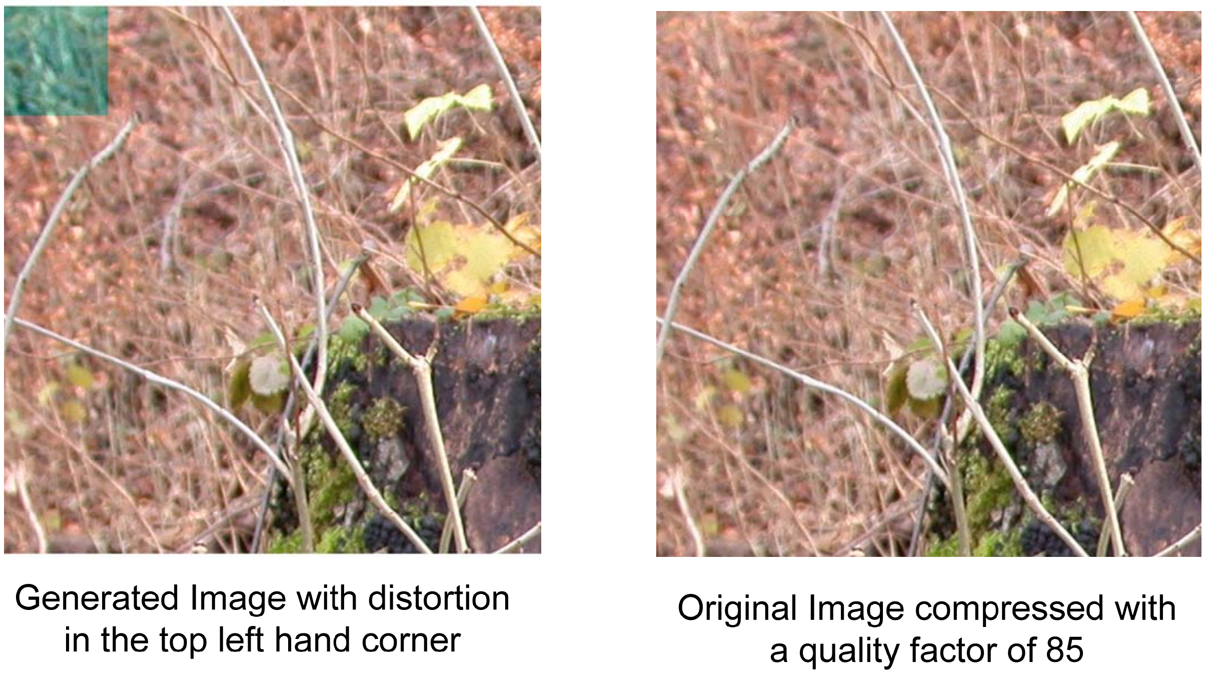

Generated Image with distortion in the top left hand corner
Original Image compressed with a quality factor of 85

Figure 5-18: Two Images with the same PSNR with relation to the original Sample Image

Due to this discrepancy, the PSNR performance for YASS, HYBRID and MULTI shall not be presented.

5.4 Visual Distortion Survey

Since the PSNR did not provide a reliable mechanism to test for visual distortion in images, we needed to design a different approach to testing the visual distortion that may be generated by the MULTI system as it embeds information into images. Instead of relying on a mathematical function or method to perform this analysis we designed a simple survey to test MULTI's performance against human analysts.

The purpose of this survey is to test MULTI's ability to avoid detection by a human observer. In most cases, if a stego-image can pass an attacker without visually raising suspicion, the attacker will pay no attention to it and the steganographic system's goal would have been achieved.

As shown previously in Figure 5-9, YASS produces clear visual signs of embedding when used to embed aggressively. HYBRID and MULTI on the other hand showed very little in the way of obvious distortion. This survey aims to present a series of images and ask of the participants to determine if the presented images show any signs of distortion.

To maintain independence from the sample group, we do not present them with any information regarding how visual distortion may manifest itself in a digital image. By doing so, each member of the sample group uses their own predefined knowledge set to determine the state of the image and gives us a better indication of the system's performance against a general group of people.

For convenience, the sample group is made up of people found on any normal university campus. To keep the sample as diverse and representative as possible, people were asked from a variety of faculties, social backgrounds and age groups. The sample group includes everyone from engineering master and PhD students to professors, final year art students, some of the cleaning and security staff on campus as well as general undergraduate students from around the university.

Each member of the test group was presented with a set of ten images randomly selected from the Zurich Natural Image Database. Of these ten images, five were randomly selected and a message embedded into them using the MULTI system. At that point the test subject is asked to determine which images had signs of visual distortion and which showed no evidence of tampering. This was accomplished by the test subject awarding a value of one ("1") to an image they felt contained distortion of some kind and a zero to images they felt were untouched. A mark was awarded to every correct classification of the images.

Table 5-10 shows a simple example of how one of the samples was obtained. First, of the ten selected images, five were randomly selected and MULTI used to embed a message. The test subject was then asked to classify each image by assigning a 0 to an image they felt was not tampered with and a 1 to an image they felt was tampered or distorted. Once the classification was complete, the classifications were marked by awarding a point for correctly classifying an image and no points for incorrectly classifying an image. A total out of ten was then given to each test subject. It is important to note that the test subjects were never told how many images in the set of ten were distorted.

Table 5-10: Example Result from Survey (A.C = Actual Classification, TS.C = Test Subject Classification, M = Modified Image, N = Normal (unmodified) Image)

	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10
A.C	N	M	M	N	M	M	N	M	N	M
TS.C	1	0	0	0	1	1	0	0	1	1
Marks	0	1	0	1	1	1	1	0	0	1
Total for test										**6**

We note that by awarding a mark for correctly classifying both the untouched and distorted images we remove the possibility of a test subject doing well by simply guessing that each image is distorted. For example, if marks were only awarded for correctly classifying distorted images, a test subject guessing that all ten images are distorted would achieve a mark of 5 out of 5, but in reality, the test subject simply picked one of the two answers and stuck with it throughout the test. By awarding marks for classifying both types of images, this particular test subject's score would be 5 out of 10, indicating that he was most likely guessing his way through the test.

When selecting the message to be used in the embedding procedure, we used the worst case message; the case which represented a message that would require that each candidate coefficient be affected in the embedding procedure. This maximized the effect of the embedding to really give an indication of MULTI's performance. To achieve this, an aggression level of 8 was selected. This resulted in a very high embedding capacity while not embedding too aggressively to the point where it was obvious that embedding was occurring.

To determine a suitable sample size for the survey, we approached Ms. Tonya Esterhuizen, a biostatistician at the University of Kwazulu Natal. Based on her expertise, it was determined that a sample of 80 or more would allow for a fair representation of a general trend.

Appendix A contains the detailed results obtained from all 90 test subjects tested during the survey. Looking through the results we notice that the highest mark awarded to any subject was 8 out of 10, indicating a very high classification rate. This rate was achieved only four times throughout the survey, which represents less that 5% of the sample group. On the other hand, we find that a member of the sample grouped failed to classify a single image correctly.

Figure 5-19 presents the histogram obtained from the Survey. We note that the mean mark achieved by the 90 sample group was 5.1.

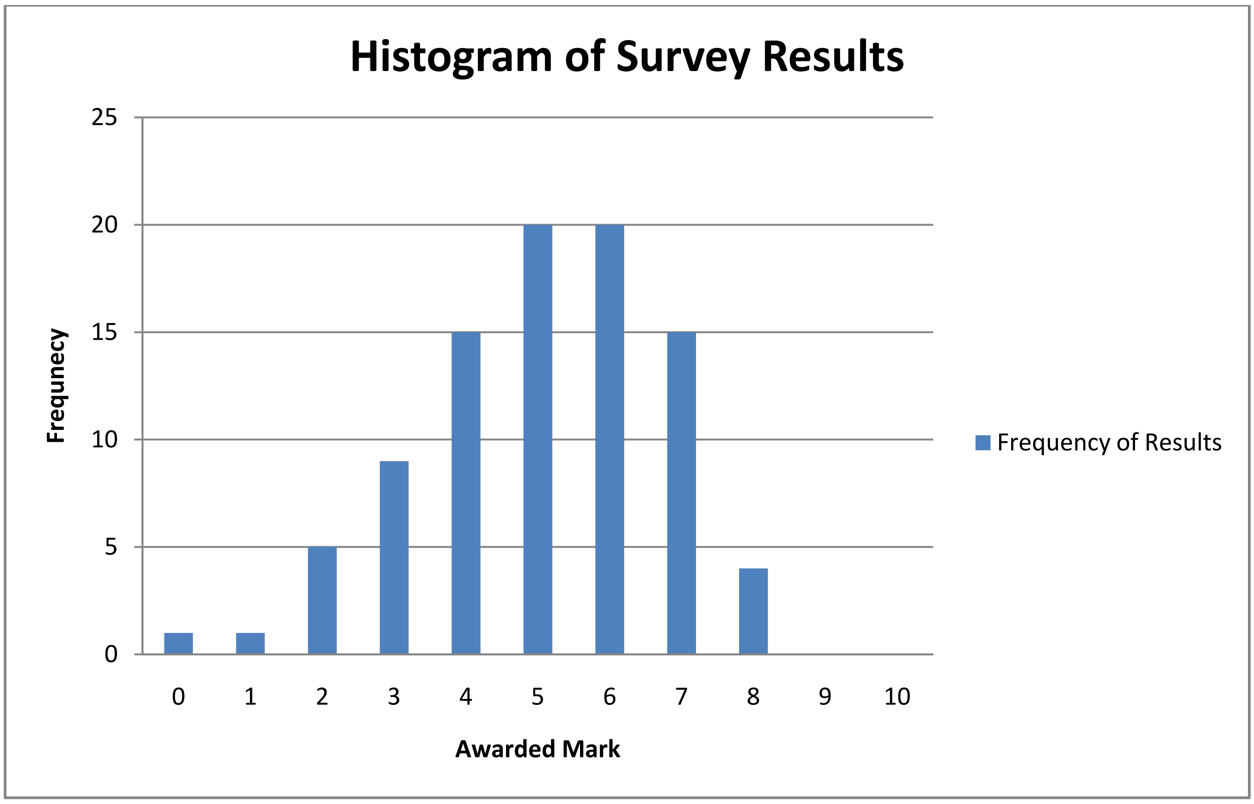


Figure 5-19: Histogram of Results from Visual Distortion Survey

A person presented with an image and asked to determine if the image contains any distortion has a 1 in 2 chance of guessing the answer correctly. When considering this, we get an average, of just over 5 out of 10. This suggests that on average the sample group had a guessing chance of detecting distortion in images.

This result means that the MULTI system does not introduce enough visual distortion to aid a normal human observer in determining whether information was embedded in a presented image.

5.5 Blind Steganalysis

Now that we have investigated the ability for a human observer to detect the presence of distortion in an image, we now turn our focus on MULTI's performance against blind steganalysis systems. As discussed in Chapter 3, these systems are designed to test and analyze higher order image features in the hopes of detecting any irregularities that may give away the presence of hidden information.

These Blind steganalysis systems are designed to distinguish between two different classes of images, namely images that do not contain steganographic content and those that do contain such material. This mirrors the question asked to the participants of the survey performed in the previous section.

What interests us is referred to as the probability of detection, defined as P_d. This probability refers to the likelihood of the steganalysis system detecting an image containing steganographic material. We consider the following scenario:

Let X_0 represent the event of an image containing no stego material and X_1an event of an image containing stego material. On the other hand, we denote Y_0 as the event of detecting an image that contains no stego material and similarly Y_1 the event of detecting an image that contains stego material. We therefore formally define P_d as:

$$P_d = 1 - P_{error},$$

$$P_{error} = P(X_0)P(Y_1|X_0) + P(X_1)P(Y_0|X_1),$$

$$P_{error} = \frac{1}{2}P_{FP} + \frac{1}{2}P_{FN}.$$

$$where P(X_0) = P(X_1) = \frac{1}{2}, P(Y_1|X_0) = P_{FP} \; and \; P(Y_0|X_1) = P_{FN}.$$

We note that for a dataset of images where we have the same number of stego-images as well as plan cover-images, the assumption $P(X_0) = P(X_1) = \frac{1}{2}$ always holds.

We define P_{FP} as the False Positive rate. This represents the probability of the detector misinterpreting a plain cover-image as an image containing stego material. We also define P_{FN}as the False Negative rate. This represents the detector missing the occurrence of a stego-image and interprets it as a plain cover-image.

Since a uniform detector can classify images as either always stego or always cover, for a dataset containing an equal number of both classes, it will achieve a $P_d = \frac{1}{2}$. Therefore we consider our steganographic system as secure if a blind steganalysis detector achieves a $P_d \approx \frac{1}{2}$, with a $P_d \leq 0.6$ still considered secure. Clearly as the detectability of the system increases, P_d tends towards a maximum of 1.

Before we test the detectability of the MULTI system, we require a baseline of performance for the YASS framework. As detailed in (Solanki K. et al 2007), the original YASS was tested against the following six steganalysis systems:

1. **Farid**: 72-dimensional feature vector based on moments in the wavelet domain (Farid H. 2006).
2. **PF-23**: Pevny and Fridrich's 23-dimensional DCT feature vector (Pevny T. et al 2006).
3. **PF-274**: Pevny and Fridrich's 274-dimensional feature vector that merges Markov and DCT features (Pevny T. et al 2007).
4. **DCT hist.**: Histogram of DCT coefficients from a low-frequency band (Solanki K. et al 2006).
5. **Xuan-39**: Spatial domain steganalysis proposed (Xuan G. et al 2005) (a 39-dimensional feature vector).

6. **Chen-324**: JPEG steganalysis based on statistical moments of wavelet characteristic functions proposed (Chen C. et al 2006).

Before performing the steganalysis on the YASS framework, we are required to select values for the two variables under the user's control. We firstly need to determine a value for the B-block size. In (Solanki K. et al 2008), a value of B=9 and B=14 was selected to allow testing in two different situations. When B=9, it represents the highest possible density of hidden information since generated E-blocks are at most two pixels apart. On the other hand, for B=14, it represents a rather low embedding density since E-block have the potential to be separated by larger distances.

The second parameter for selection is the quality factor used during the hiding procedure. Q_h can take up any value one requires, but as mentioned previously and in (Solanki K. et al 2007), the YASS framework operates at its peak when $Q_h = Q_a$, where once again Q_a represents the compression quality factor.

Since $Q_a = 75$ is an extremely popular compression quality factor (Solanki K. et al 2007), compressing an image with this factor would raise the least amount of suspicion and thus is selected for the purposes of our test. We are therefore faced with selecting a $Q_h \leq 75$. To attempt to match the results published previously, we use the same two Q_h values as in (Solanki K. et al 2007) namely $Q_h = 50$ and $Q_h = 75$.

Table 5-11 present the results from the blind steganalysis detectors presented to two significant figures after the decimal point. As stated before, a detection rate of greater than 0.6 is considered as successfully detecting a steganographic system. Looking through the results, we notice that for the original YASS framework, under certain conditions, some of the steganalysis schemes were able to achieve a detection rate of 0.6 or greater (highlighted in Table 5-11).

We also make note that of all the steganalysis schemes used, the **PF-274** and **Chen-324** schemes achieved the greatest success in detection therefore we select only these two schemes to provide a comparison test for MULTI.

Similar to the YASS framework, MULTI also allows the user to set the range of the B-block values to be used during the blocking phase. Since MULTI uses a range of values, we select a range of 9-10 to allow for a similar comparison to that of $B = 9$ for YASS. We also select a range of 9-15 to match the lower density produced by a B-block size of $B = 14$.

We also need to select a suitable aggression level to be used during the test. Since the same compression quality factors will be used, namely $Q_a = 50$ and $Q_a = 75$, we require an aggression level that will embed at a reasonable rate without risking too many errors due to the compression occurring. We therefore select an $A_l = 5$ for the test.

Table 5-11: Steganalysis Results for YASS

Q_h (used for hiding)	Q_a (used for steganalysis)	Steganalysis method	Detection rate: B=9	Detection rate: B=14
50	50	Farid	0.51	0.51
50	75	Farid	0.53	0.51
75	75	Farid	0.52	0.50
50	50	PF-23	0.55	0.54
50	75	PF-23	0.58	**0.6**
75	75	PF-23	0.53	0.52
50	50	PF-274	0.59	0.54
50	75	PF-274	**0.77**	**0.65**
75	75	PF-274	**0.59**	0.54
50	50	DCT-hist	0.54	0.53
50	75	DCT-hist	**0.64**	0.53
75	75	DCT-hist	0.54	0.53
50	50	Xuan-39	0.54	0.51
50	75	Xuan-39	0.63	0.52
75	75	Xuan-39	0.52	0.52
50	50	Chen-324	0.57	0.54
50	75	Chen-324	**0.75**	0.53
75	75	Chen-324	0.54	0.53

Table 5-12 presents the blind steganalysis results for MULTI. We notice that the two schemes tested failed to achieve a detection rate of higher than 0.55. This would suggest that MULTI can successfully resist blind steganalysis techniques that could previously detect YASS.

Table 5-12: Steganalysis Results for MULTI (A_l = 5)

Aggression level	Q_a (used for steganalysis)	Steganalysis method	Detection accuracy: B=9-10	Detection accuracy: B=9-14
5	50	PF-274	0.52	0.53
5	75	PF-274	0.55	0.5
5	50	Chen-324	0.51	0.51
5	75	Chen-324	0.53	0.52

Considering the results in Table 5-12 were obtained with an aggression level 5, we find that there is still a lot of potential embedding capacity that the system is not taking advantage of. We therefore increased the aggression level to 8 while still maintaining the same compression quality factors.

Table 5-13: Steganalysis Results for MULTI ($A_l = 8$)

Aggression level	Q_a (used for steganalysis)	Steganalysis method	Detection accuracy: B=9-10	Detection accuracy: B=9-14
8	50	PF-274	0.55	0.57
8	75	PF-274	**0.6**	0.54
8	50	Chen-324	0.57	0.58
8	75	Chen-324	0.57	**0.59**

Having increased the aggression level, we present the result in Table 5-13. We find that **PF-274** and **Chen-324** manage to detect stego content when the image is compressed with a quality factor of 75. At an aggression level of 5 with a compression quality factor of 75, MULTI embeds an average of about 46 bits per E-block. On the other hand, at an aggression level of 8, MULTI manages to embed an average of just over 64 bits per E-block. This massive increase in embedding capacity maybe the reason behind the steganalysis schemes success in detecting the stego content.

That said, we notice that even at much higher embedding capacities to that achievable by the YASS framework, MULTI still presents a secure steganographic system.

5.6 Summary

This chapter presented a performance analysis for the proposed MULTI system. The focus was to demonstrate the improvements the new system achieved over the original YASS framework, the baseline for the proposals design.

The first of these improvements made by the MULTI system was the introduction of a new layer of randomization to the B-block size used in the blocking scheme. The chapter presented a comparison between the original blocking scheme used by the YASS framework and the scheme designed for MULTI. It was shown that YASS achieves a higher blocking rate for B-block sizes smaller than B=11, while MULTI successfully manages to achieve a higher blocking rate than YASS for B-blocks of size 12 and higher. According to Solanki K. et al (2007), YASS achieves its optimum performance when B=14 and it is shown that MULTI can achieve a better blocking rate at that range.

The second improvement made to the MULTI system was the introduction of a second layer of randomization, namely the randomization of the E-block size. By allowing the E-blocks to vary in size, we showed that it generated distribution of E-blocks results in an average E-block size of just over 9 for a B-block range of 9-13. This represented a 26% increase of candidate coefficients over the original YASS E-block size, which aids greatly in improving the overall embedding capacity of the new MULTI system.

Once the blocking rate was tested and the PRNG performance shown to match the expected theoretical distribution, we showed the embedding and error rate performance of the new MULTI system as compared to the original YASS framework.

It was shown that MULTI achieves an embedding capacity increase of 7.54 times over that achieved by YASS, 1.66 for MULTI compared to 0.22 for MULTI. This result indicates MULTI's ability to embed data into cover without grossly increasing the resulting stego-image's size. Although this is a very desirable feature of MULTI, by achieves this increase in capacity it also increases the error rate the system needs to handle. With a peak error rate of just over 40 bits per E-block, this represented a 3.7 times increase over YASS's 10 bits per E-block in error. This huge increase in error rate was matched by the embedding rate achieved by the MULTI system, managing to embed just over 64 bits per E-block on average, a increase of 3.8 times over YASS's 16 bits per E-block.

To compare the two systems fairly, we matched the error rates of both systems, using YASS's peak error rate of 10 bits per E-block as the baseline for the comparison. It was shown that by matching YASS's error rate, MULTI could embed 2 bits more per E-block than YASS, showing the MULTI has indeed increased on the embedding capacity of YASS.

The chapter then focused on presenting a security analysis to compare YASS and MULTI's ability to avoid suspicion. The first method used was the Peak Signal to Noise Ratio, or PSNR. Although this method can be used as a method of determining image degradation, for example as a means of judging compression quality, it was shown that as a means of determine visual distortion, it presented a discrepancy that left the results generated by this method in question.

Since the use of the PSNR was far from ideal, a simple survey was performed with the aim of testing MULTI's ability to embed stego content without introducing obvious visual distortion. A group of 90 people found on a normal university campus were asked to examine a set of 10 images. Of these ten images, half of them were randomly selected and used as cover-images for an aggressive embedding pass by MULTI. The group were then asked to classify each image as with a distorted image or an image containing no distortion. A mark was awarded for every image they classified correctly.

Once the survey was complete, the average mark attended by the sample group was 5 out of ten. This suggests that on average, a person has a 50 % chance or guessing chance of classifying an image correctly, meaning that MULTI, at least with regards to human interpretation, is a secure system.

The final security analysis involved testing MULTI's performance against blind steganalysis. We first performed the analysis on YASS to achieve a baseline that allows us to compare MULTI's performance. We successfully managed to replicate YASS's original performance as stated in (Solanki K. et al 2007). We then selected two of the steganalysis techniques, namely the **PF-274** and **Chen-324** schemes since they achieved the highest detection rates against YASS.

Performing the same analysis on MULTI using comparable aggression levels, we find the steganalysis schemes failed in detecting MULTI. Where the **PF-274** managed to achieve a maximum detection rate of 0.77 for YASS, it managed a detection rate of only 0.55 for MULTI under the same aggression levels. Similarly, where **Chn-324** managed a peak detection rate of 0.75 for YASS, it managed a detection rate of only 0.53 for MULTI.

We then increased the aggression level of MULTI even higher to determine a level which would result in one of the two systems successfully detecting MULTI. At an aggression level of 8, we found that **PF-274** managed a detection rate of 0.6, representing successfully detecting MULTI while **Chen-324** managed a detection rate of 0.59, close enough to represent some level of reliable detection.

An interesting comparison to perform at this point is how both YASS and MULTI compare to each other at the same detectability level. We note that **PF-274** manages to detect YASS with a rate 0.59

with $B = 9$, $Q_h = 75$ and $Q_a = 75$, and manages a similar rate of 0.6 for MULTI with a B-block range of 9-13, an aggression level of 8 and a $Q_a = 75$.

With the two systems embedding with their parameters set to those mentioned above we find that YASS achieves an error rate of 6 bits per E-block and an embedding rate of 12 bits per E-block, resulting in a throughput of 6 useful bits per E-block. On the other hand, the MULTI system achieves an error rate of 38.8 bits per E-block and an embedding rate of 62 bits per E-block. This results in a throughput of 23 useful bits per E-block. Therefore at a similar level of detectability, we find that MULTI clearly out performs YASS.

Considering these results, we find that MULTI successfully managed to achieve its goal of increase upon the embedding capacity of YASS. The introduction of random E-block sizes, aid greatly in this as well as the design of the new embedding scheme which allowed for more aggressive embedding in medium frequency coefficients.

MULTI also managed to improve upon YASS's security by avoid human detection as well as blind steganalysis at a comparable aggression level with YASS. We also find that at an increased capacity of over 7.54 times that of YASS, MULTI is only just detectable by currently blind steganalysis schemes.

6 Conclusion

The science of Steganography is still relatively young in a digital context. Currently there is a lot of research being done on the study of stego-systems and with the popularity of the internet; most of this research is focused on hiding information in cover-objects that exist within the internet. Everything from video to text based steganography is being researched as a possible carrier of covert information.

We noted that with the massive popularity of social networks, the way the internet is being used has changed. Before the internet represented a tool for distributing knowledge and sharing of ideas, but with the advent of sites such as MySpace, Facebook and Youtube, we are finding that more and more users of the internet spend time sharing personal data such as images, videos and blogs. This provides a huge set of possible objects for use as cover for covert communication.

We presented statistics that showed that Facebook alone accounts for over 30 billion images being uploaded a year, which represents about 950 images being uploaded every second. This fact provides any designer of a steganographic system with two very important properties.

The first is an innocent cover-object. Due to the popularity of sharing images online, it has become almost an expectation to view images on any website visited while on the internet and this provides a steganographer with the perfect cover-object. The second property is a culture of sharing. Due to the open nature of social networks, it is expected that people will view profiles of strangers. This property of social networks allows for the sharing of information between two parties without either having to establish a previous relationship. Combining these two properties motivate the need to research image based steganography and more importantly, JPEG based steganography.

The first part of the dissertation investigated the history of steganography and presented an overview of the evolution of steganography from physical based to digital based steganography. We then introduced the current science of the subject since no formal theoretical base has yet been developed for the science due to relative infancy. We define the notion of a ε-secure steganographic system and detailed the properties of any steganographic system, regardless of cover-object. We then introduced a formal method for classifying steganographic objects based on the process used in the embedding of the hidden message.

The second part of the dissertation surveyed the development of image based steganographic systems with a focus of systems that embedded in JPEG compressed images. Since JPEG is one of the most popular compression algorithms currently used on images, most images stored on websites and on the internet are compressed with this standard. We investigated the properties of JPEG based steganographic systems and highlighted the motivation behind their development. We then introduce possible weaknesses for each of those systems as well as present methods that are used in detecting them.

The third part of the dissertation introduced the design of a novel JPEG based steganographic system built on the YASS framework. This MULTI system introduces two added layers of randomization to the embedding process of the YASS framework to increase security. The introduction of randomization to the B-block generation phase increased the order of complexity of the detection algorithm by a factor of two, while the introduction of randomization to the E-block layer allows for an increase in security as well as allowed for the potential for increasing embedding capacity.

The proposed system also introduced a novel embedding method, different to that used by the YASS framework. This embedding scheme firstly introduced a masking step to the embedding process that allows the user of the stego-system to determine the size of the embedding region. This masking region is controlled by the aggression level. The higher the aggression, the greater the region the embedding scheme searches for candidate coefficients that can be used for embedding.

Another important concept introduced by the embedding scheme is the modulation used to embed the coded stego-message into the coefficient data. Instead of using QIM, the proposed system uses the simpler Random ± ternary modulation. It is expected that using this scheme to embed into candidate coefficients will avoid QIM's problem of increasing the number of zero coefficients as noted in Chapter 4.

The final and arguably the most important addition made by the new embedding scheme is the introduction of embedding in medium frequency coefficients. The YASS framework previously embedded in the 19 lowest frequency coefficients, DC coefficient not included, which resulted in clear visual distortion at higher embedding rates as demonstrated in Chapter 4. But using this novel technique of hiding in medium frequency coefficients, we traded off visual distortion with the risk of higher error rates. This technique of hiding in medium frequency coefficient is new to JPEG steganography and it was shown that although the increase of error is high, at matched error rates, the new proposed system does outperform YASS.

6.1 Summary of Contributions

The following lists the contributions of the dissertation:

- A complete overview of the development of the science of steganography and the collection and formalization of a theoretical basis for the science. We also presented a formal classification of different steganographic techniques.

- A detailed survey of currently available image based steganographic systems with a focus of JPEG based systems. A detailed discussion of the motivation of each system was also included as well as an overview of weakness inherit within each system.

- The development of a new JPEG based steganographic system that built upon the YASS framework but introduced measures to counter its original weakness and improve upon both security and embedding capacity.

- A detail analysis of the effects of compression on embedding in DCT coefficients and the development of a novel embedding scheme that utilizes medium frequency coefficients which lowers the risk of visual distortion at higher embedding rates.

- The introduction of user controlled embedding rates through the use of an aggression level that allows for on the fly embedding rate changes.

6.2 Direction of Future Work

Future work related to JPEG based steganography and Image based steganography includes the following:

- An investigation into the effects of the quantization matrix used in the embedding phase with the aim of determining its effects on the systems overall error rates.
- The performance of the MULTI and HYBRID systems against other types of image compression algorithms. Compression algorithms just as JPEG2000 and Portable Network Graphics (PNG).
- An investigation of the potential to perform repeat compression and embedding cycles. As mentioned in (Solanki K. et al 2008), by repeatedly compression an image with same quality factor, the image's raw pixel data as well as DCT coefficients as a whole tend to a constant value. This tendency to approach a constant state, may aid the embedding rate by allow for the reduction of error rates during the embedding process. This occurs due to the stego-image tending towards a constant state and thus would not be affected by any further compression.
- An investigation at the potential to adapt the HYBRID and MULTI system to take advantage of the wavelet based JPEG2000.
- A look at video based embedding would be an interesting extension to image based hiding.

7 References

Anderson R.J.: *ed., Information hiding: first international workshop,* vol. 1174 of L*ecture Notes in Computer Science,* Isaac Newton Institute, Cambridge, England, May, Springer-Verlag, Berlin, Germany, ISBN 3-540-61996-8. (1996)

Cahin C.: "An information-theoretic model for steganography", Information and Computation, Vol. 192(1). Pp.41-56. (2004).

Cancelli G., Barni M., Menegaz G.: "Mpsteg: hiding a message in the matching pursuit domain," Proc. SPIE, Security, Steganography, and Watermarking of Multimedia Contents VIII, California USA, vol. 6072. (2006)

Chapman M., and Davida G.: "Hiding he Hidden: A Software System for Concealing Ciphertext as Innocuous Text," in the *proceedings of the International Conference on Information and Communications Security,* Vol. 1334 of Lecture Notes in Computer Science, Springer, pp.335-345, Beijing, P. R. China, November. (1997)

Chen C., Shi Y.Q., Chen W., Xuan G.: Statistical moments based universal steganalysis using JPEG-2D array and 2-D characteristic function. In: Proc. ICIP, Atlanta, GA, USA pp105–108 (2006)

Dabeer O., Sullivan K., Madhow U., Chandrasekaran S.: "Detection of hiding in the least significant bit", IEEE Trans. On Signal Processing, vol.52, pp. 3046-3058. (2004)

Desoky A.: Nostega: A Novel Noiseless Steganography Paradigm, Journal of Digital Forensic Practice, Volume 2, Issue 3 July, pages 132 – 139. (2008)

Farid H.: http://www.cs.dartmouth.edu/farid/research/steg.m. (Code for generating wavelet-based feature vectors for steganalysis.) (2006)

Fridrich J., Goljan M., Hogea D.: "Steganalysis of JPEG Images: Breaking F5 Algorithm," In: F.A.P. Petitcolas, Editor, *Information-Hiding*, *Lecture Notes in Computer Science* **vol. 2578**, Springer-Verlag, New York, pp. 310–323. (2002)

Fridrich J., Goljan M., Hogea D.: Attacking the OutGuess. Proc. ACM: Special Session on Multimedia Security and Watermarking, Juan-les-Paris, France. (2002)

Fu D., Shi Y.Q., Zou, D., Xuan G.: "JPEG steganalysis using empirical transition matrix in block DCT domain", *International Workshop on Multimedia Signal Processing*, Victoria, BC, Canada. (2006)

JPEG Standard: ISO/IEC IS 10918-1 | ITU-T Recommendation T.81. (1986)

Kahn D.: *The Codebreakers: The Story of Secret Writting.* Revised Ed. Scribner, December. (1996)

Katzenbeisser S. and Petitcolas F. A.P. (Ed.): Information Hiding techniques for Steganography and digital watermarking, Artech House (2000).

Kerckhoffs A.: "La cryptographie militaire", *Journal des sciences militaires*, vol. IX, pp. 5–83, Jan., pp. 161–191, Feb. (1883)

Kharrazi M., Sencar H.T., Memon N.: "Cover selection for steganographic embedding," Proc. ICIP, pp. 117-120. (2006)

Konrad M., Stogner H., and Uhl A.: "Custom Design of JPEG Quantisation Tables for Compressing Iris Polar Images to Improve Recognition Accuracy," *ICB 2009*, LNCS 5558, pp. 1091–1101. (2009)

Lenti J.: Steganographic Methods. Periodica Polytechnica, 44 (3-4), pages 249–258, June. (2000)

Lin E.T. and Delp E.J.: Review of Data Hiding in Digital Images. Video and Image Processing Laboratory (VIPER) School of Electrical and Computer Engineering Purdue University West Lafayette, Indiana. (1999)

Lyu S., Farid H.: "Detecting Hidden Messages Using Higher-Order Statistics and Support Vector Machines," *Proc. 5th Int'l Workshop on Information Hiding,* Springer- Verlag. (2002)

Marvel L.M., Boncelet C.G.and Retter C.T.: (1999). Spread-Spectrum Image Steganography. Set to appear in the IEEE Transaction on Image Processing. (1999)

Pevny T., Fridrich J.: Merging Markov and DCT features for multi-class JPEG steganalysis. In: Proc. of SPIE, San Jose, CA (2007)

Pevny T., Fridrich J.: Multi-class blind steganalysis for JPEG images. In: Proc. of SPIE, San Jose, CA (2006)

Pfitzmann B.: "Information hiding terminology." In Anderson R.J., pp.347-350, ISBN 3-540-61996-8, results of an informal plenary meeting and additional proposal.

Provos N. and Honey P.: "Hide and Seek: An Introduction to Steganography", Published by the IEEE Computer Society, 1540-7993/03. (2003)

Provos N.: "Defending Against Statistical Steganalysis," *Proc.10th Usenix Security Symp.,* Usenix Assoc., pp 323-335. (2001)

Sarkar A., Solanki K., and Manjunath B.S.: Further study on YASS: Steganography based on randomized embedding to resist blind steganalysis. In E. J. Delp and P. W. Wong, editors, Proceedings SPIE, Electronic Imaging, Security, Forensics, Steganography, and Watermarking of Multimedia Contents X, volume 6819, pages 16–31, San Jose, CA, January 27–31. (2008)

Shi Y.Q., Chen C., Chen W.: "A Markov process based approach to effective attacking JPEG steganography". *Leilich, H.-O. (ed.) GI-NTG Fachtagung Struktur und Betrieb von Rechensystemen.* LNCS, Springer, Heidelberg. (1974)

Simmons G.J.: The Prisoners' Problem and the subliminal channel, in Proceedings of Crypto'83, Plenum Press, New York, pp. 51-67. (1984)

Smith J.R., Comiskey B.O.: "Modulation and Information Hiding in Images" in First International Workshop on Information Hiding, LNCS 1174, pp. 207–226, Springer. (1996)

Solanki K., Jacobsen N., Madhow U., Manjunath B.S., Chandrasekaran S.: Robust image-adaptive data hiding based on erasure and error correction. IEEE Trans. on Image Processing 13(12) pp1627 – 1639 (2004)

Solanki K., Sarkar A., Manjunath B.: Further Study on YASS: Steganography Based on Randomized Embedding to Resist Blind Steganalysis, Proc. SPIE - Security, Steganography, and Watermarking of Multimedia Contents (X), San Jose, California, Jan. (2008).

Solanki K., Sarkar A., Manjunath B.: YASS: Yet Another Steganographic scheme that resists Blind Steganalysis, In: Information Hiding, 9th International workshop, IH2007, Saint Malo, France, Springer, pp.16-31. (2007)

Solanki K., Sullivan K., Madhow U., Manjunath B.S., Chandrasekaran S.: Provably secure steganography: Achieving zero K-L divergence using statistical restoration. In: Proc. ICIP pp125–128 (2006)

Sullivan K., Madhow U., Chandrasekaran S., Manjunath B.: “Steganalysis for Markov cover data with applications to images.” *IEEE Transactions on Information Forensics and Security 1*, 275–287. (2006)

Topkara M., Topkara U., Atallah M.J.: “Information hiding through errors: A confusing approach” *Proceedings of the SPIE International Conference on Security, Steganography, and Watermarking of Multimedia Contents,* January. (2007)

The Radicati Group, Inc. Releases “Email Statistics Report, 2009-2013” published May, 2009.

van Lint J.H.: *Introduction to coding theory*, 2nd ed. Springer-Verlag. (1992)

van Tilborg H.C.A (Ed.): Encyclopedia of cryptography and security, pp.159. Springer (2005).

Wayner P.: “Mimic functions” in Cryptologia, Vol. XVI(3), pp. 285–299. (1992)

Westfeld A.: “F5 – A Steganographic Algorithm: High Capacity Despite Better Steganalysis,” *Proc. 4th Int’l Workshop Information Hiding,* Springer-Verlag, pp. 289-302. (2001)

Xuan G., Shi Y.Q., Gao J., Zou D., Yang C., Yang C., Zhang Z., Chai P., Chen C., Chen W.: Steganalysis based on multiple features formed by statistical moments of wavelet characteristic functions. In: Lecture notes in computer science: 7th International Workshop on Information Hiding. (2005)

Zeng W., Yu H., Lin C.: “Multimedia Security Technologies for Digital Rights Management”. Academic Press. Elsevier. (2006).

8 Bibliography

Avcibas I., Sankur B., Memon N.: "Image steganalysis with binary similarity measures," Proc. ICIP, pp. 645-648. (2002)

Cancelli G., Barni M., Menegaz G.: "Mpsteg: hiding a message in the matching pursuit domain," Proc. SPIE, Security, Steganography, and Watermarking of Multimedia Contents VIII, California USA, vol. 6072. (2006)

Fridrich J., Goljan M.: "Digital image steganography using stochastic modulation," Proceedings of SPIE Electronic Imaging, Security, Steganography, and Watermarking of Multimedia Contents V 5020, pp. 191–202. (2003)

Gruhl D., Bender W., and Lu A.: Echo Hiding. In: Information Hiding: First International Workshop, pages 295-315. (1996)

Hetzl S., Mutzel P.: A graph theoretic approach to steganography. In: 9th IFIP TC-6 TC-11 International Conference, Communication and Multimedia Security, Salzburg, Austria, vol. 3677, pp. 119-128. (2005)

Jagpal G.: Steganography in Digital Images Thesis, Cambridge University Computer Laboratory, May. (1995)

Kharrazi M., Sencar H.T., Memon N.: "Cover selection for steganographic embedding," Proc. ICIP, pp. 117-120. (2006)

Lee I.S., *Tsai, W.H*, "Data Hiding in Binary Images with Distortion-Minimizing Capabilities by Optimal Block Pattern Coding and Dynamic Programming Techniques," *IEICE(E90-D)*, No. 8, pp. 1142-1150. (2007)

Marvel M., Boncelet C., Retter, C.: "Spread spectrum image steganography," *IEEE Transactions on Image Processing* 8, pp. 1075–1083. (1999)

Matsui K. and Tanaka K.: Video-steganography. In: IMA Intellectual Property Project Proceedings, volume 1, pages 187-206. (1994)

Neubauer C., Herre J., and Brandenburg k.: Continuous Steganographic Data Transmission Using Uncompressed Audio. In: Information Hiding: Second International Workshop, pages 208-217, (1998)

Petitcolas F.A.P., Anderson R.J and Kuhn M.G.: "Information Hiding – A Survey" in *Proceedings of the IEEE*, special issue on protection of multimedia content, 87(7):1062-1078, July. (1999)

Sallee P.: Model-based steganography. In: Kalker, T., Cox, I (eds) IWDW 2003, LNCS, vol. 2939, pp. 154-167. Springer. (2004)

Solanki K., Sullivan K., Madhow U., Manjunath B.S., Chandrasekaran S.: "Probably secure steganography: Achieving zero K-L divergence using statistical restorartion," Proc. ICIP, Atlanta, USA, pp. 125-128. (2006)

Solanki K., Sullivan K., Madhow U., Manjunath B.S., Chandrasekaran S.: "Statistical restoration for robust and secure steganography." Proc. ICIP, Genova, Italy, pp. 1118-1121. (2005)

Wang Y., Moulin P.: "Optimized features extraction for learning-based image steganalysis," IEEE Trans. In Information Forensics and Security, vol.2, no.1, pp. 31-45. (2007)

Wu D.C., *Tsai, W.H.:* “A steganographic method for images by pixel-value differencing,” *PRL(24)*, No. 9-10, June, pp. 1613-1626. (2003)

Wu D.C., *Tsai, W.H.:* “Spatial-domain image hiding in using image differencing,” *VISP(147)*, No. 1, February, pp. 29. (2000)

Xydeag C.S., *Kostic B.*, *Steele R.*, “Embedding Data into Pictures by Modulo Masking,” *Commun(32)*, pp. 56-69. (1984)

9 Appendix A

Table 9-1: Full Survey Results (1 represents correct classification, 0 represents incorrect classification)

Sample #	Image 1	Image 2	Image 3	Image 4	Image 5	Image 6	Image 7	Image 8	Image 9	Image 10	Total
1	0	0	0	1	1	0	1	0	1	0	**4**
2	1	1	1	0	0	0	1	0	0	1	**5**
3	1	1	0	1	1	0	0	0	1	0	**5**
4	0	0	1	1	0	1	1	0	1	0	**5**
5	0	1	0	0	0	1	1	1	0	0	**4**
6	1	0	1	1	1	0	1	1	0	1	**7**
7	0	0	0	0	1	1	1	1	0	1	**5**
8	1	1	1	1	1	1	0	1	1	0	**8**
9	0	0	1	1	1	1	0	0	1	0	**5**
10	0	1	1	0	1	0	1	0	1	1	**6**
11	1	0	0	0	1	0	0	0	1	1	**4**
12	1	0	1	0	1	1	1	0	1	0	**6**
13	1	0	0	1	0	0	1	0	1	1	**5**
14	1	1	0	0	0	1	0	1	1	1	**6**
15	1	0	0	0	1	1	1	0	0	1	**5**
16	0	1	1	0	1	0	0	1	1	0	**5**
17	1	0	1	0	1	1	1	1	1	1	**8**
18	1	0	1	0	0	0	1	1	1	1	**6**
19	0	0	1	1	0	1	0	0	0	0	**3**
20	0	0	1	1	1	1	0	0	1	0	**5**
21	0	1	1	1	0	0	0	1	1	0	**5**
22	1	1	1	0	0	1	1	1	0	0	**6**
23	1	1	0	0	1	1	1	0	1	1	**7**
24	0	1	1	0	1	0	1	0	0	1	**5**
25	1	1	1	0	0	1	0	0	0	1	**5**
26	1	1	0	1	0	0	1	1	1	0	**6**
27	0	0	1	0	0	0	0	0	1	1	**3**
28	0	1	0	0	1	1	1	1	1	1	**7**
29	1	1	1	0	1	1	0	1	0	0	**6**

30	0	1	0	0	1	1	0	0	1	0	**4**
31	0	0	0	1	1	1	1	0	1	1	**6**
32	0	1	1	0	1	0	1	1	1	0	**6**
33	0	1	1	1	1	1	1	0	1	0	**7**
34	0	0	0	1	0	0	1	0	0	0	**2**
35	1	1	0	1	0	0	1	0	1	1	**6**
36	1	1	1	0	0	1	0	1	1	1	**7**
37	0	1	0	0	0	1	0	0	1	0	**3**
38	0	0	0	1	0	1	0	0	1	1	**4**
39	0	0	0	0	1	1	1	0	0	1	**4**
40	1	0	1	1	1	1	1	1	1	0	**8**
41	1	0	0	0	1	1	0	0	1	1	**5**
42	0	1	1	0	0	0	0	0	1	0	**3**
43	1	1	1	0	0	0	0	0	1	0	**4**
44	1	1	0	1	0	1	1	1	0	0	**6**
45	0	0	1	1	0	1	0	1	0	1	**5**
46	1	0	1	0	0	0	0	0	0	0	**2**
47	0	1	1	1	0	0	1	1	0	1	**6**
48	0	1	0	0	0	0	0	0	0	1	**2**
49	1	0	0	0	0	1	1	0	0	1	**4**
50	1	0	1	0	0	1	1	1	1	1	**7**
51	0	1	0	1	1	1	0	1	1	0	**6**
52	1	1	0	1	0	1	0	1	0	0	**5**
53	1	0	1	1	0	0	1	1	0	0	**5**
54	1	0	1	0	0	0	1	0	0	0	**3**
55	0	1	0	0	0	1	0	0	1	0	**3**
56	0	1	1	1	0	0	1	1	1	1	**7**
57	0	0	0	0	0	1	0	1	1	1	**4**
58	1	1	1	1	0	1	1	0	1	1	**8**
59	0	0	1	1	0	0	0	0	0	0	**2**
60	1	0	1	0	1	0	0	0	0	0	**3**
61	0	0	0	0	0	1	0	0	0	0	**1**
62	0	0	1	0	1	0	1	1	1	0	**5**
63	1	1	0	0	0	0	1	1	0	0	**4**

64	1	0	0	1	1	1	1	0	1	1	**7**
65	1	0	1	1	0	0	0	0	1	0	**4**
66	1	0	1	0	1	1	1	0	0	0	**5**
67	0	0	0	0	1	0	0	1	1	1	**4**
68	1	1	0	1	0	1	1	0	1	1	**7**
69	0	0	1	0	1	0	0	1	0	0	**3**
70	1	1	1	1	0	1	0	0	1	0	**6**
71	1	0	0	0	1	0	1	1	1	1	**6**
72	1	1	0	0	1	1	1	1	1	0	**7**
73	1	1	1	1	1	1	0	0	0	0	**6**
74	1	1	1	1	0	0	1	0	1	0	**6**
75	1	1	0	0	0	1	1	1	1	1	**7**
76	1	0	0	1	1	0	1	1	1	1	**7**
77	1	1	0	0	1	1	1	1	0	1	**7**
78	0	1	1	0	1	0	1	1	1	0	**6**
79	1	0	0	1	1	1	1	1	0	0	**6**
80	0	1	0	0	0	0	0	0	0	1	**2**
81	1	0	1	0	0	0	1	0	1	0	**4**
82	0	1	0	1	0	0	1	1	1	0	**5**
83	0	1	1	0	1	1	0	1	0	0	**5**
84	1	1	1	0	1	1	0	1	0	1	**7**
85	1	1	1	0	1	0	1	0	1	1	**7**
86	0	1	1	0	1	0	1	0	1	1	**6**
87	0	1	0	0	1	1	0	0	0	0	**3**
88	0	1	1	1	0	0	0	1	0	0	**4**
89	0	0	0	0	0	0	0	0	0	0	**0**
90	0	0	1	0	1	1	0	0	1	0	**4**
	47	**48**	**49**	**36**	**44**	**48**	**49**	**40**	**54**	**40**	