University of Glasgow | School of Computing Science

Steganography in Lossy and Lossless Images

Aidan Rutherford

School of Computing Science
Sir Alwyn Williams Building
University of Glasgow
G12 8QQ

Level 4 Project — March 28, 2014

Abstract

Chartering the progress of an investigation into the Bit-plane Complexity Segmentation steganography algorithm, this report examines the suitability of using BPCS in JPEG 2000 images. Subjecting the algorithm to statistical and visual analyse, the report suggests the original algorithm needs considerable revision to be viable for lossy JPEG 2000. Instead, it details an algorithm which takes advantage of the characteristics of JPEG 2000, combined with the results of the investigation which suggest using a maximum embedding bit-plane, as opposed to the standard black-and-white border measure in traditional BPCS.

Acknowledgements

I would like to take this opportunity to thank my supervisor, Dr Ron Poet, for his time and assistance.

Education Use Consent

Contents

List of Figures

Chapter 1

Introduction

Steganography, the art and science of hidden communication, has been practiced for thousands of years, and practitioners have adapted their methods to take advantage of the latest and most popular techniques throughout its history. The advent and boom of digital communication is no exception, facilitating the emergence of steganography as an academic discipline in its own right. Growing in tandem with steganography is its antithesis, steganalysis, which aims to detect the presence of hidden messages within otherwise innocuous cover media. Together, these fields create a vibrant and evolving academic community focussed on creating a perfectly secure steganographic technique.

Outwith academia, various industries have also taken up the practice, most prevalent of which are copyright holders within the creative industries, who aim to identify copyright infringers based on buyer details embedded within their content. Many of the techniques used within industry have not been publicly released, relying in part on security through obscurity rather than the methods themselves. By relying on obscurity, they violate Shannon's Maxim of "The enemy knows the system"[1] and cannot be deemed secure.

The most widely studied subset of steganography uses digital images as the cover medium, with forays into other media being a relatively new addition to the state of the art. One of the major problems facing all steganographic techniques, but particularly those based on digital images, is the inability to withstand alterations to the cover media without sacrificing the integrity of the secret. Most techniques strive to create a balance between maintaining the undetectability of the technique and its robustness. For this reason, lossy image compression presents an interesting challenge for steganography; lossy compression irrevocably removes data from the original, resulting in a reduced data set that is suitable for embedding with steganography. Steganography within lossy media is also essential for the continued preservation of secret communication, especially if the technique is used as a covert means of communication on the Internet because, as highlighted by Eggers, Bäuml, and Girod [13, 27], "uncompressed image data looks to [an observer] as suspicious as encrypted data."

Recent studies have begun to investigate this idea, often adapting existing techniques and generating new ones for use during lossy compression. One such algorithm is BPCS [32] which has been converted from its original usage in lossless BMP image files to a variety of media, including both EZW [58] and JPEG2000 [45], by changing the embedding domain from the spatial to the transform. However neither of the above techniques have researched the impact of changing the domain in which the algorithm is performed on the detectability of the hidden information. Considering the current state of the art for steganalysis, and the importance of maintaining secrecy, investigating the detectability of these modified algorithms is an important research question for assessing their practicality.

Drawing on the previous work on BPCS in lossy JPEG 2000 covers conducted by Noda et al. [45], in conjunction with Kawaguchi and Eason's [32] original algorithm, this investigation aimed to examine the suitability and security of BPCS within JPEG 2000, with particular emphasis on the detectability of embedded data. Noda's

[1]Widely accepted as Shannon's theory, though unattributed in literature

algorithm was implemented for JPEG 2000 and its performance was compared to the both the body of BPCS literature and an implementation of the original algorithm for BMP covers.

Subjecting the stego-objects produced by the system to visual and statistical analysis highlighted significant anomalies, prompting a closer examination of the intricacies of the algorithm. The results of these investigations are summarised in this report, detailing the irregularities discovered and potential explanations for their occurrence. Finally it expounds a modified steganography technique which takes the results into consideration and is customised to take account of the characteristics of JPEG 2000.

Chapter 2

Background & Related Literature

2.1 The History of Steganography

Steganography is the art and science of hiding information in plain sight. Over the course of millennia it has evolved into a rigorous academic discipline with a rapidly growing following, but it has its roots in antiquity. The term itself, descended from the Greek στεγανός, meaning "to cover", and γράφω, "I write", was not coined until the late 15th century when it was used by Johannes Trithemius in his *Steganographia* [60], which in itself was an example of steganography and one which has only recently been fully recovered [50].

While as yet unnamed, the earliest literary record of steganography appears in the work of the first antique historiographer, Herodotus, which can be dated to mid 5th Century B.C.E. [52]. Herodotus uses three examples of hidden messages [23, 1.123, 5.35, 7.241], which were each instrumental in military victory. His most famous example features a slave who's shaven head has been tattooed with the word "revolt" and, after his hair has grown back, he is sent to the Ionians to insight an uprising against the Persians. The examples cited in Herodotus are very distant cousins to the techniques used today, but the work of Aeneas the Tactician [2, 31.2], one of Herodotus' contemporaries, resembles methods which were being used up to the end of World War II; Aeneas Tacticus describes a form of acrostics, or literary steganography, in which the spacing and size of the letters in a cover text hides the secret message [2].

Other forms of steganography have been used throughout history and many different techniques were utilised by espionage agents during both World Wars in order to facilitate the communication of critical information. One such method, which was popular during both wars, concealed the information by scaling it down on to a small disk, called a microdot, which could then be concealed as a full stop in a letter [38]. Another example, made famous through the media, was the case of Jeremiah Denton who in 1966, while held as a prisoner of war by the North Vietnamese, blinked the word "torture" in morse code while being interviewed for a Japanese television broadcast [14].

Modern techniques stray away from such physical manifestations, instead focussing on means of communicating information within digital media. The advent of steganography as an academic discipline occurred in conjunction with the rise of digital communication and the increased pressure to be able to communicate secretly along these new channels. In recent years, steganography has received another boom following in the wake of the battle being waged between content providers and copyright infringers. Publishers and broadcasting agencies are constantly seeking new ways of 'fingerprinting' their property, to help identify the source of leaked material [18]. Outwith the publishing industry, steganography has been used by medical and military groups for transmitting sensitive data.

More controversial uses of steganography have also been reported, with criminal and terror organisations allegedly using it as a covert means of communication. This was first brought into the public eye in 2001 when it was reported by Jack Kelley that al-Queda were hiding messages in images for distribution online [34][1]. It was seen as a credible threat to national security and, in 2006, the American Government produced a report which highlighted the problem posed by the criminal use of steganography [25]. Contrarily, studies have been conducted which downplay the threat posed by online steganography, highlighting a dearth of evidence [49]. Lately, steganography has been used to conduct malware attacks, with one of the most recent attacks hiding a trojan horse within a PNG image file [20].

2.2 Towards a definition of Steganography

Despite spanning millennia and utilising vastly different communication media, all the examples described above obey the key steganographic principle of secrecy. The definition of secrecy is an evolving entity, which needs to match the technology and requirements of both the period and environment; secrecy is an absolute and once it is broken, it cannot be restored. All too often, secrecy has been neglected during the past 20 years, while research focussed on furthering other metrics such as robustness and capacity.

In the classical period, steganography was based on security through obscurity; if the Persians knew messages were tattooed on slaves' heads it would be a simple step to shave them and reveal the message. In a similar vein, acrostics only uphold their secrecy while the rules for encoding the message are unknown. Following from Kerkhoff's principle [37], modern steganography assumes that the method of hiding the information is known and therefore the hidden message must be imperceptible from normal communication. Most of the early algorithms and naive steganography techniques obeyed this principle, but it quickly became apparent that imperceptibility did not provide enough security to withstand computational analysis and therefore techniques adopted undetectability as the ultimate metric.

Undetectability is a narrow definition of secrecy which requires hidden messages to be capable of avoiding detection by both the Human Vision System (HVS) and computational analysis, as opposed to imperceptibility which only requires passing the HVS. If a system does not embed undetectably, it cannot be classified as a successful stenographic algorithm. Algorithms can be deemed to produce undetectable output only for a given state of the art, therefore, due to the constantly evolving nature of the research, a fail safe is often employed to protect the secrecy of the information. Currently, systems either make use of a stego-key, which randomly distributes the data within the image, or encrypt the data prior to embedding, thus providing another layer of security, should the secrecy of the embedded data be compromised.

Every steganographic system utilises a common set of components, which are expanded or adapted depending on the algorithm. Figure 2.1 describes a generic system based on Simmons's [57] Prisoners' Problem, which both embeds and extracts the secret data. The components of Figure 2.1 are defined as follows:

Payload: The secret information to be communicated.

Cover: The medium in which the message will be embedded.

Stego-key: The stego-key used to encrypt the secret.

Stego-object: The cover object containing the embedded and encrypted secret.

These definitions will be used throughout this report to specify the components of all steganaographic systems.

The stego-key is an optional component for a generic steganography system, though one which has gained traction in practical implementations because of the additional security it provides [16]. Those which do not use it will be referred to as 'pure' steganography systems throughout this project, as they rely solely on the principle of undetectability to establish security. Therefore the overarching definition can be described as:

[1]Kelley has since been removed from his post at USA Today, following accusations of false-reporting.

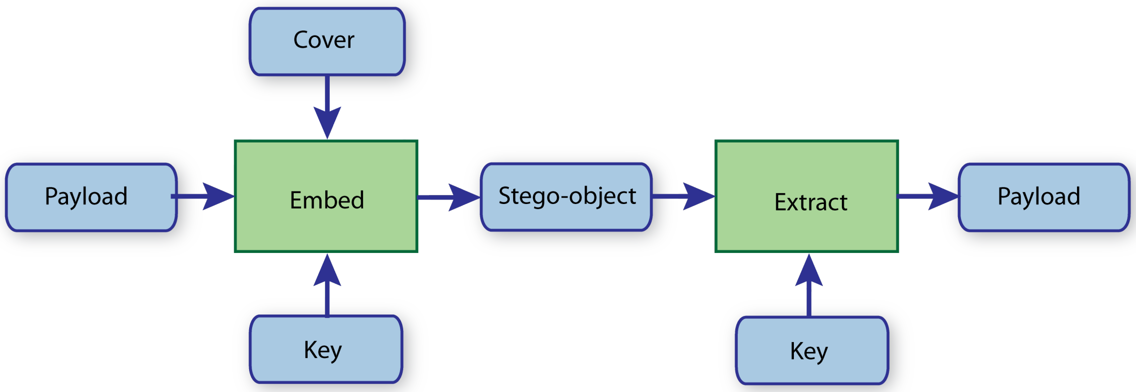


Figure 2.1: Generic stegonography process

Steganography 1. *Any method which embeds secret information into a cover medium in such a way that it is undetectable by both the Human Vision System and computational analysis, though which can be recovered by those who are aware of the secret's existence and the method used to embed.*

2.3 Modern Steganography

The current state of the art is expansive and includes a disparate variety of embedding techniques and cover media. Predominantly, techniques focus on subtracting image data and replacing it with an equivalent number of bits from the payload, though implementations using this method have been proven to be detectable by statistical attacks, prompting the emergence of additive techniques such as matrix encoding. This section focuses on the research associated with steganography in digital images, while also providing a brief overview of other media where pertinent.

2.3.1 Least Significant Bit Steganography

Least Significant Bit (LSB) steganography was one of the earliest techniques proposed for embedding within digital images. The algorithm cycles through the LSBs of an image's pixels, replacing each of them with a bit of the payload data, resulting in a maximum embedding capacity of 12.5% of the original image size. The technique has proved to be easily detectable [35] and is now considered obsolete within the current state of the art.

Derivatives of LSB have been proposed with varying degrees of success. Most of the early techniques are now disregarded as naive, though those which employ a metric to determine which bits to replace often fair better. Such is the case of Bender et al.'s [4] technique, which employs statistical analysis to determine both which pixels to consider and how many bits to replace.

2.3.2 Matrix Encoding

Cradall [10] introduced matrix encoding as a more efficient means of embedding payload data in steganographic techniques. The first steganography technique to make use of Cradall's theory was F5, which was proposed by Westfeld [63] in 2001. The algorithm aims to distribute payload data evenly through the cover, in order to avoid the clumping of secret data at the beginning of the stego-object that can occur with other techniques and has been proven to be vulnerable to the χ^2 statistical attack [64]. F5 embeds by incrementing the coefficients resulting from the Discrete Cosine Transform in JPEG encoding, relying on the coefficients tolerance to small changes to prevent perceivable image alterations.

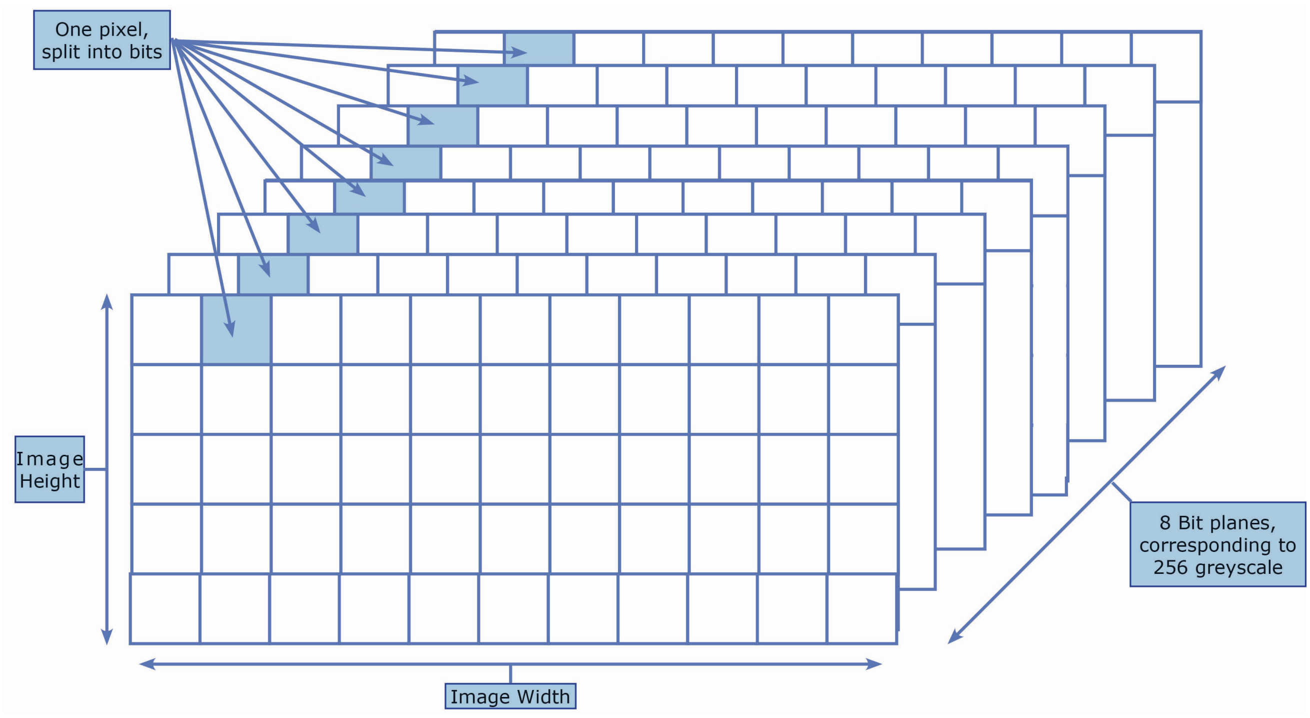

One pixel, split into bits
Image Height
8 Bit planes, corresponding to 256 greyscale
Image Width

Figure 2.2: Bit-plane structure based on pixels

2.3.3 Bit-plane Complexity Segmentation Steganography

Algorithm 1: Basic BPCS algorithm

```
Convert image from Pure Binary Code to Canonical Gray Code;
foreach bit plane in image do
    foreach block in bit plane do
        if NoiseCalculation(block) >= $\alpha$ then
            if NoiseCalculation(payload) < $\alpha$ then
                Conjugate(payload);
            replace block with payload;
Convert image from CGC to PBC;
```

Bit-plane Complexity Segmentation (BPCS) is one of the most actively researched steganography algorithms; first proposed by Kawaguchi and Eason [32] in 1999, it has since been the the subject of 36 papers published by the IEEE[2]. Kawaguchi's algorithm relies on a bit-plane structure and exploits the inability of the HVS to extract meaningful information from sufficiently noise-like patterns, and therefore is unable to discriminate between such regions in an image [17]. The original algorithm (Algorithm 1) decomposes the cover image into bit-planes of increasing significance (Figure 2.2), corresponding to the image's width and height. The planes are further segmented into 8x8 blocks, for each of which their likeness to noise is calculated using the measure established by Niimi, Noda, and Kawaguchi [42]. Niimi uses the dark/light transitions within the block to determine the information potential of the image, such that:

$$c = \frac{\text{The number of B} \rightarrow \text{W changes} + \text{the number of W} \rightarrow \text{B changes in the image}}{\text{The max. number of possible changes in the image}} \tag{2.1}$$

[2]Correct as of 5/4/2014

Resulting in c in the range $[0..1]$. As c tends to 1, the block becomes more noise-like and the ability of the HVS to extract information is reduced; following experimentation, Niimi established 0.5 as the threshold beyond which no information is discernible. Considering the low frequency of true noise-like regions in most images, a threshold below 0.5, α, is determined at which the HVS is unlikely to detect information. Therefore, if c is greater than α, the block can be switched with an equivalent sized section of the payload data so long as the payload is also noise-like. If the payload is less than α, then the segment to be embedded is conjugated with a checkerboard pattern, such that:

$$Seg_{x,y} = Seg_{x,y} \oplus WC_{x,y} \tag{2.2}$$

Where WC is an equally sized checkerboard pattern, beginning with a white bit in the upper left corner. Conjugation results in a symmetric value for c around 0.5, making the payload block suitable for embedding.

This measure often determines borders between regions of colour to be more noise-like than they are due to hamming cliffs, and embedding within such segments results in visual defects. In an effort to mitigate this problem, Kawaguchi suggested converting the image to Canonical Gray Code (CGC) prior to embedding. As a cyclic code, CGC removed the hamming cliffs and not only improved the imperceptibility of the message but also increased the payload capacity of the cover.

The algorithm has been implemented for the spectrum of lossless cover images: Niimi, Noda, and Kawaguchi [43] implemented it for 8-bit greyscale images; Ouellette et al. [47] used 8-bit indexed colour images; and 24-bit colour images were used in Kawaguchi and Eason [31]. In Kawaguchi's original paper, he suggests that "BPCS Steganography is not robust to even small changes in the image" [31, 9], a theory which has been thoroughly disproved in the last decade. By shifting the domain in which embedding takes place, BPCS has been successfully implemented in lossy image formats, including EZW [58]; Spaulding et al. [58] utilises the bit-plane structure inherent in the wavelet coefficients to embed payload data in the transform domain, rather than the spatial. By moving the embedding domain, they are able to overcome the inherent loss of data associated with lossy compression, by exploiting the resilience of the coefficients to small changes.

2.3.4 New Techniques in Steganography

While BPCS has been a research staple for 15 years, other techniques are also beginning to come to the fore, especially those which take advantage of the methods discovered for use in chaotic steganography. Many of these techniques have been designed to embed within lossy images, such as the algorithm proposed by Ghebleh and Kanso [19] which aims to provide a robust steganography technique for use in lossy JPEG 2000 images.

Despite these advances, BPCS is still a very relevant technique due to the lack of secret key required to decode the message and its applicability to lossy media. In addition, it has served as a point of departure for many of the algorithms developed in the past 10 years, meaning it is integral to the study of modern steganography.

2.3.5 Other Media

While most research has been conducted into embedding within digital images, other cover media are growing in popularity, particularly audio and video. Video is often viewed as a simple extension of image steganography; many of the techniques utilise the algorithms designed for images and apply them on frame-by-frame basis. More recent research into video steganography has introduced more cohesive techniques which take advantage of the format and don't rely solely on image techniques [11].

In tandem with the developments in video steganography are those which embed data within audio covers. As with most forms of digital steganography, the LSB has been used as a basic starting point for development [3], though more recent approaches have seen techniques similar to BPCS embedding data within the quantised

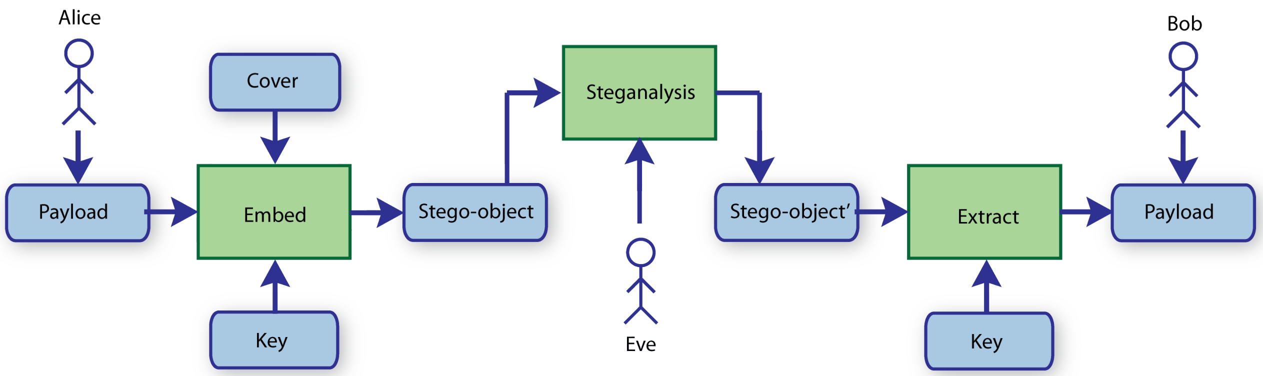


Figure 2.3: Full steganography process, including steganalysis

frequency domain. One of the major challenges facing audio steganography is the sensitivity of the human ear, which can often detect the slight changes to the track introduced by embedding data. Therefore many current techniques exploit the wavelet domain, which is resilient to small changes, in order to minimise these defects [54].

2.4 Steganalysis

Steganalysis is the skill of discovering hidden data, or the presence of hidden data. The techniques employed in steganalysis vary widely in sophistication but they can all be broadly classified as statistical, visual, or structural.

The typical scenario used in data hiding techniques involves prisoners Alice and Bob passing coded messages to each other without Eve, the eavesdropping Warden, understanding the message, as proposed by Simmons [57]. When developing a secure data hiding algorithm, it is assumed that Eve has control over the transmission channel and is aware of the methods by which Alice and Bob may communicate, as dictated by Kerkhoff's Theorem [37]; for academic purposes it is generally assumed that Eve is omniscient, if not omnipotent. But strong steganalysis techniques often rely on very little or no knowledge of the embedding algorithm. Such methods rely on statistical properties of the cover set to identify anomalies and are called blind steganalysis, as opposed to their targeted cousins.

Whether a technique uses blind or targeted steganalysis, Eve can take two approaches with her detection: active or passive. A passive warden does not interfere with the transmission but can test it for observable anomalies. On the other hand, an active warden can attack the transmission either with the purpose of destroying the message (an active attack) or creating a false message (a malicious attack) [38]. With this in mind, the process of steganography outlined in Figure 2.1 needs to be refined. The full flow of steganography is now elucidated in Figure 2.3.

Steganalysis is a probabilistic skill which relies on evaluating the probability that a given image contains hidden information against the probability that it does not. As the degree of variance between the sets of cover and stego-images narrows, the difficulty in establishing an accurate conclusion dwindles. This is clearly elucidated in the example of LSB steganography where, assuming the payload is sufficiently random, it is almost impossible to gauge the likelihood that an LSB is flipped or not [36]. Using information theory, this can be described in terms of the KL divergence between the distributions of cover and stego-images [15, 9], such that:

$$D_{KL}(P_c||P_s) = \sum_{x \in C} P_c(x) \log_2 \frac{P_c(x)}{P_s(x)} \tag{2.3}$$

where P_c is the probability distribution of cover images and P_s is the probability distribution of stego-images. Using this equation, if $D_{KL}(P_c||P_s) = 0$, then the stego-system can be deemed perfectly secure. This is a theoretical measure only, due to the complications inherent in gathering the probability distributions for any real cover set and the changes different payload data can introduce. But in general terms, a stego-system aims for P_s to be as close as possible to P_c, while steganalysis aims to exploit and identify the attributes of $\mathcal{S}$ which differentiate it from $\mathcal{C}$.

Steganalysis techniques depend not just on the embedding technique but also the cover medium used by the system. Broadly speaking, payloads embedded within natural images prove a harder challenge to detect than those embedded within computer generated images. This is primarily due to the increased variance in colour, contrast, technique etc, as well as the noise introduced by digital cameras, which make it difficult to detect anomalies which differentiate a stego-image apart from the set of covers. BPCS embeds most efficiently within natural images, due to the uniformity of computer generated graphics resulting in noticeable defects, and therefore any steganalysis technique used against it must take this variance into account.

2.4.1 Availability of the Cover

Embedding within images which are freely available in the public domain is very risky; if the exact cover is available, a simple subtraction with the stego-object will reveal the embedded data in its entirety. The highest level of security for any steganography technique can only be guaranteed when the the cover itself is kept secret, or destroyed after embedding is complete. On the other hand, many images exist in a multitude of formats and sizes, resulting in a potentially arduous search for the true cover. Even the set of images which is viewed as the 'standard test suite' for image processing exists in multiple forms ([56], c.f. [61])! This ties in with the idea elucidated in Equation 2.3, which focuses on creating an embedding technique which manufactures stego-objects which are, statistically, as close as possible to the set of possible covers. Therefore if a stego-object can masquerade as a different format, or a different compression of the original, it is likely to still pass steganalysis. This makes the task of steganlaysis much harder as the set of possible covers does not just include the medium which manifests itself in the stego-image, but also all similar formats of the image.

Even if the cover comes from the public domain, it is still advantageous for steganalysis to detect the presence of embedded data before extraction by this mechanism; comparing every image passing through a sensor to the set of similar publicly available images has the potential to be computationally non-trivial.

2.4.2 Visual Steganalysis

The simplest form of steganlayis relies on detecting visual anomalies within stego-images. Many visual steganalysis techniques rely on imperfections within the embedding algorithm; producing a functionally identical stego-object is no longer a challenge, but isolated components can often reveal defects. Thus decomposing an image into its elements often produces far better avenues for visual analysis than treating the image as a whole. One of the most enduring visual attacks looks at the LSBs of a stego-image to investigate areas of artificial randomness. Prior to embedding, the LSBs of natural images are far from random, with many factors influencing their periodicity including colour gradations and saturation. After embedding, dependent on the entropy of the payload, these regions will be much closer to random and therefore vulnerable to analysis [62]. While simple, these techniques can prove very effective, as demonstrated by Niimi, Noda, and Segee [44], who describe a visual attack on BPCS that detects defects within the least significant bit-plane.

Detecting periodicity is one of the strongest aspects of visual analysis due to the HVS's capacity to detect repeating patterns [39]. These patterns can often be used as a signature to identify a particular algorithm or form of payload, though only if a set of stego-objects is available for testing.

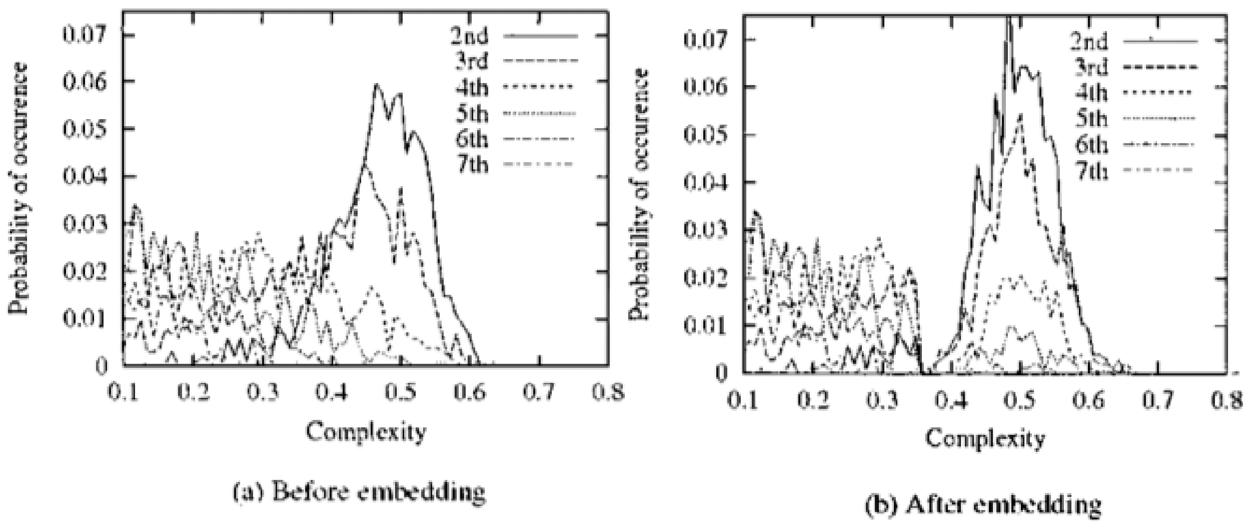


Figure 2.4: Comparison of histograms under BPCS embedding [41, 735]

2.4.3 Structural Steganalysis

Structural steganalysis detects changes to the file format of the stego-object. By comparing the structure against the standard in which it is defined, or by detecting anomalies such as irregular colour mapping, a compromised structure can be detected and the presence of embedded data deduced. Image formats which use colour palettes are particularly vulnerable to structural analysis; in order to mitigate the defects introduced by pixels corresponding to the wrong palette entry, modifications to the palette structure are introduced which can then be detected by structural analysis [62].

2.4.4 Statistical Steganalysis

Analysing the statistical attributes of a stego-object, in comparison with the set of possible covers, frequently reveals the presence of hidden data. Such techniques usually detect the increased randomness introduced by embedding an encrypted or compressed payload. These techniques often target a particular embedding medium for which a statistical baseline has been generated from the set of possible covers. Alternatively a known-payload attack can be employed which obeys the Central Limit Theorem, such that the more covers the warden possesses, the fewer changes can be introduced by the embedding technique, as the warden will be able to create a more refined baseline.

The χ^2 test examines how close to random the distribution of bits are within an image by summing the occurrences of a particular set of events, such as the LSB being 1 or 0. Its use in steganography was proposed by Westfeld and Pfitzmann [64] and has been used to analyse the parity of the LSB in order to deduce the presence of embedded material.

Another statistical test employed in steganalysis aims to detect anomalies using histograms charting various attributes of the stego and cover images. This technique has been successfully applied to a wide range of steganographic techniques, and has also been developed as a blind attack, though with less success [48]. Through histogram analysis both the kurtosis and variation of bytes within the image can be gauged, though kurtosis requires a knowledge of underlying algorithm due to the lack of clean break point [7]. Niimi et al. [41] proposed an attack on the BPCS algorithm which exploits one of the fundamental aspects of the algorithm: conjugation. Due to conjugation's symmetric property, the histogram of the stego-image betrays the presence of hidden data by manifesting a valley close to the threshold, as shown in Figure 2.4

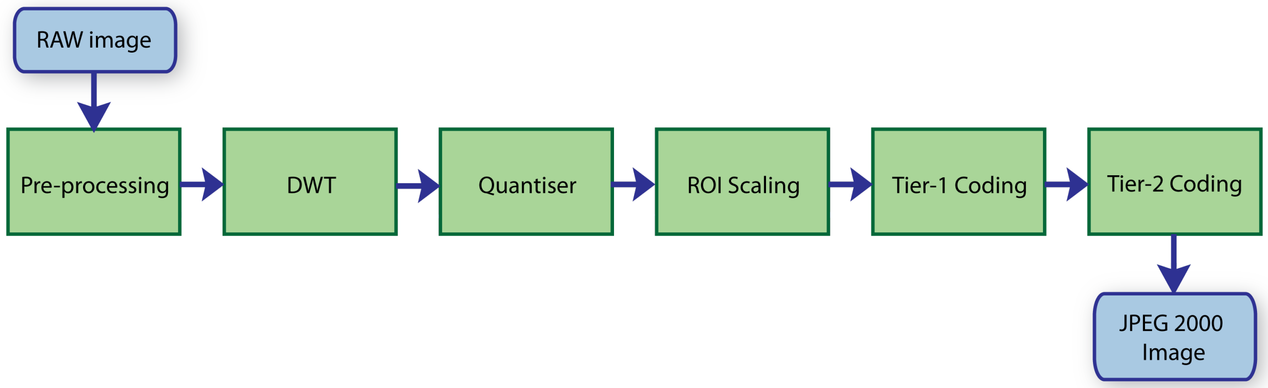


Figure 2.5: Workflow of JPEG2000

2.5 Image Compression

Compressing images is a common practice used to facilitate more efficient storage and communication of data. Image compression is either lossless, in which case the original image can be restored in its entirety, or lossy, where part of the original is irrecoverably lost.

Lossy image compression is the norm for any online data transfer, making any uncompressed image data suspicious [13]. With the fundamental principle of steganography requiring a high degree of secrecy, anything which deviates from the norms of the environment in which it is likely to be used is disadvantageous. Therefore utilising lossy images as cover media is a fundamental aspect of preserving the secrecy of any stenographic communication which uses the Internet as the communication channel.

2.5.1 JPEG 2000

JPEG 2000 is an image standard proposed by the Joint Photographic Experts Group [26], aimed at supplanting the original 1992 JPEG standard [29] by offering extensive customisability and an increase in the compression rate of approximately 20% [53]. The standard offers both lossy and lossless compression by benefit of the Discrete Wavelet Transform (DWT), which is at the heart of the standard and replaces the Discrete Cosine Transform (DCT) used by its predecessor. Uptake for the new standard has been slow, which can largely be attributed to the increased complexity of the encoding procedure and the lack of backwards compatibility with JPEG [6]. Despite not being popular within the wider market, it has received considerable attention within the Data Hiding community due to the bit-plane structure which can be extrapolated from the wavelet coefficients, making it an ideal candidate for BPCS embedding.

Generating a JPEG 2000 bitstream follows a distinct set of steps, outlined in Figure 2.5. Pre-processing is the initial component which takes an uncompressed image as input and prepares it for encoding by decomposing it into a series of tiles, or equally sized segments, which are encoded separately. The size of the tiles is arbitrary, ranging from the default of one tile spanning the whole image, to a single pixel! Each tile is passed to the DWT module where the Daubechies(9,7) or Daubechies(5,3) analysis filter bank is used to further decompose the tile into a series of sub bands, corresponding to a predetermined number of resolution levels, which can be visualised using Figure 2.6.

The resulting sub bands can be used during decoding to reconstruct the image at various resolution levels simply by decoding only those sub bands corresponding to the desired resolution and the previous one. Using figure 2.6 as reference, to reconstruct an image at resolution level r from an n-step decomposition, the sub bands $(n-r+1)$HH, $(n-r+1)$HL and $(n-r+1)$LH are combined with the image at resolution level $r-1$.

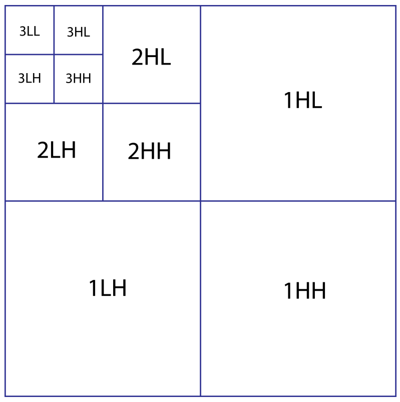


Figure 2.6: A 3 step DWT sub band structure for JPEG 2000

Following DWT, all coefficients are quantised, resulting in a set of quantised wavelet coefficients. It is at this stage that BPCS would ideally take place in the encoding process, as the wavelet coefficients and an adequate bit-plane structure are readily available. However, as Noda et al. [45] explain, it is impossible to perform BPCS at this stage without risking compromising the data further down the chain.

Region of Interest (ROI) scaling is an optional stage of the process which can be bypassed without jeopardising the resulting bitstream. The process takes predefined regions and shifts the associated wavelets into more significant bit-planes, creating more detailed regions in the final image.

Tier-1 and Tier-2 coding refer to arithmetic coding and packetisation respectively, which collectively partition the sub bands into code blocks before organising them into the JPEG 2000 bitstream. Importantly, this section of the encoder performs post-compression rate-distortion optimisation which aims to bring the encoded image's bitrate inline with a given target bitrate, while simultaneously introducing a minimum of distortion to the image [59]. In so doing, it determines the number of bit-planes to encode for a given bitrate, resulting in truncation of low-significance planes. It is for this reason that embedding cannot take place immediately after quantisation.

2.5.2 JPEG 2000 and BPCS

BPCS cannot be employed in the usual spatial domain within JPEG 2000 covers, therefore the embedding process needs to be shifted into the coefficients resulting from the DWT. The quantised wavelet coefficients are an ideal embedding platform for BPCS because they mirror the properties of the spatial domain and have been quantised into a bit-plane structure. Ultimately, wavelets correspond spatially to pixels in the resulting image, though this relationship is scaled with a 16x16 block of pixels in the original image translating to an 8x8 set of coefficients in the finest sub band. This means that the likeness to noise of a given bit-plane is likely to be proportional to the corresponding bit-plane in the original, allowing for the same measures used in the original BPCS algorithm to be used for JPEG 2000 embedding. Importantly, this also means alterations to the noise-like segments in the coefficients won't cause disruptions in informative regions of the image, preserving the ability of the algorithm to confound the HVS into seeing an innocuous image. Similarly, BPCS exploits the robustness of the coefficients, to make small changes which do not affect the image disproportionately. This idea draws on the pre-existing work on speech recognition which utilises this property as a measure of similarity [22].

Chapter 3

Design & Implementation

In order to investigate the suitability of BPCS for JPEG 2000 cover images, two separate systems were created. The first used Kawaguchi and Eason's [32] original embedding algorithm, with few amendments, for lossless BMP covers, while the second implemented Noda et al.'s [45] modified algorithm for use in JPEG 2000 images. The lossless BMP implementation was required in order to ensure the system functioned comparably with Kawaguhi's findings and also as a point of comparison for the performance of the JPEG 2000 system. To that end, the implementations of the algorithms themselves were based on a common design, with differences stemming from the change in embedding domain and integration with the image libraries.

The design for both systems focussed on efficiency and testability, resulting in a pair of highly paramatised systems bundled within batch wrappers to facilitate iterative testing. As a result, the usability of the systems was given a low priority, and therefore the implementations of the algorithms are not suitable for wider distribution and should be used for research purposes only.

3.1 Common Design Decisions

In order to facilitate accurate comparisons between the BMP-based system and the JPEG 2000 implementation, the design of the systems were kept as analogous as possible. This section details elements which were common to both systems.

3.1.1 Language Choice

Java is the language of choice for most of the extant steganography literature, in particular it was used as the basis for Noda et al.'s [45] implementation of BPCS in JPEG 2000. Therefore, in order to enable comparisons not just with both algorithms but within the wider context of steganography, Java was chosen as the implementation language.

3.1.2 Bit-planes & Segments

Bit-planes were implemented as two-dimensional arrays of boolean values, in order to keep the storage footprint of the programmes small and to ease conceptual understanding of the mechanics of the algorithm. Segments, the regular sized embedding regions, were treated as 8x8 bit-planes, corresponding to spatial areas within the parent plane. The number of segments within a plane was defined as:

$$n = (width - width \bmod 8)(height - height \bmod 8) \tag{3.1}$$

Resulting in bits to the far right and bottom of the plane being ignored if the dimensions of the plane were not multiples of 8. This method was chosen so as to ensure all blocks were uniform, allowing for better estimation of the expected payload for any given image.

Regardless of cover medium, the bit-planes are accessed in ascending order of significance, ensuring that those bits which are most likely to introduce a change in the image are the least likely to be altered. For JPEG 2000, significance ascends as if through a little-endian 32-bit integer, while in BMP significance is based within each RGB channel. Therefore the n^{th} significant plane is accessed as:

$$n = 3(i \bmod 8) + \lfloor i/8 \rfloor \tag{3.2}$$

where i is the sequential index of each bit in the pixel and 8 bits are used per colour channel.

3.1.3 Likeness to Noise

A measurement of how much significant information interpretable by the HVS contained within each segment of the cover is fundamental to the BPCS algorithm. By treating each segment as a binary image, Niimi, Noda, and Kawaguchi [43] proposed using the number of changes in the black and white border to calculate the segment's likeness to noise. Using Algorithm 2, the measurement estimates the likelihood of the segment containing meaningful information by summing the light/dark transitions in both the rows and columns. 8x8 patterns which can be interpreted as shapes by the HVS measure at less than 0.5 on this scale, while all those above are guaranteed to be indistinguishable from random noise.

Algorithm 2: Block Black and White Border Calculation

```
previous ← the first bit in the block ;
foreach row in the block do
    foreach bit in the row do
        if bit equals previous then
            increment rowTotal;
            previous ← bit;
foreach column in the block do
    foreach bit in the column do
        if bit equals previous then
            increment columnTotal;
            previous ← bit;
maxNumberOfChanges ← (2× width of block × (width of block - 1)) ;
return $\frac{(\text{rowTotal}+\text{columnTotal})}{\text{maxNumberOfChanges}}$
```

3.1.4 Conjugation

Conjugation is a reversible operation which alters a segments likeness to noise; it is used on all payload data which is below the noise threshold used for embedding, which prevents data mistakable as shape information being embedded into the image. The implementation of conjugation used in both projects was defined by Kawaguchi, Endo, and Matsunaga [33] and is summarised in Equation 2.2. This method of conjugation (Figure 3.1) replaces the background of the original block with the WC pattern and the foreground with an alternate

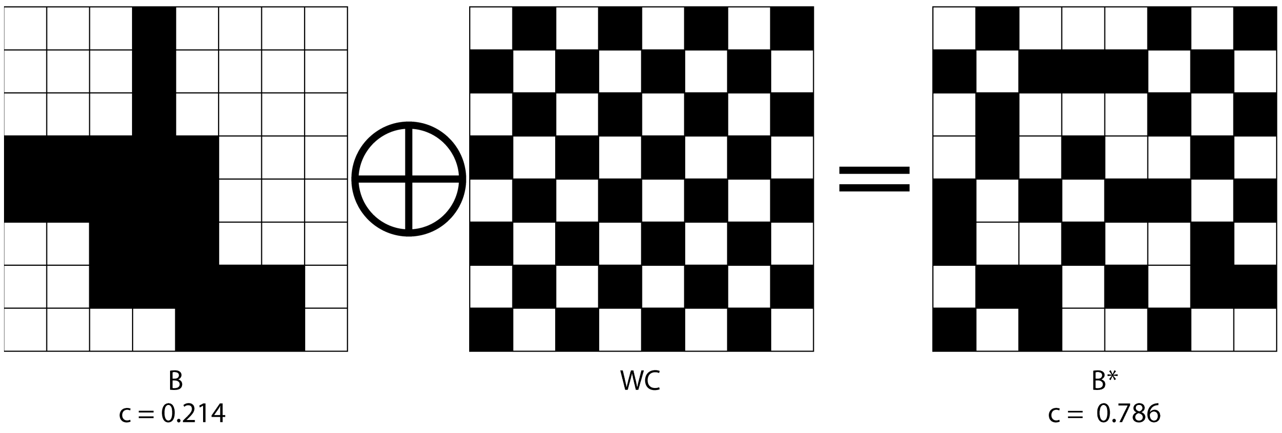


Figure 3.1: The conjugation process

checkerboard pattern, beginning with a black section in the upper left corner. The entire process is reversible, allowing for the original block to be extracted by repeating the process during extraction.

Kawaguchi's original paper suggests storing conjugation metadata as a map in the LSB-plane, though Niimi et al. [41] prove that conjugation maps are easily detectable with complexity histogram analysis. Therefore this implementation reserves the last bit of each embedding block to store a conjugation flag; the bit marked `true` if conjugation was used on the block. This reduces the payload capacity by one bit per segment, introducing an overhead of 1.56% considering the 8x8 segment size.

3.1.5 Payloads & Metadata

Payloads are embedded sequentially in blocks of 63 bits, with each block being treated as an 8x8 bit-plane. If the file size isn't divisible by 63, the remaining bits are stored in an extra segment which is left partially empty.

With BPCS, knowing the payload's type and size are essential for successful recovery. With this in mind, the first two complex segments in the cover were reserved for the payload's metadata; the first block holding the file size as a 32-bit integer, and the second holding the file extension.

A 32-bit 2s-compliment integer provides a maximum payload size of approximately 2GB. With most steganography algorithms providing a theoretic payload capacity of less than 50%, a 2GB maximum payload size would require a cover image which is greater than 4GB. Taking into account the suspicious nature of uncompressed data [13] and the average online image being less than 30KB[1], this was seen as a justifiable maximum.

Storing the extension within a single segment allows for a maximum of 7 characters, as each 8x8 segment can contain 63 bits or data plus a conjugation bit. Considering the majority of common file extensions are 4 characters of less, the 7 character limit imposed by block size is adequate for most payload media. While chaining the blocks to allow for larger extensions is a possibility in the future and would be necessary for a user-ready implementation, 7 characters was deemed adequate for the project.

3.1.6 Gray Code

Canonical Gray Code (CGC) is a single-distance code, proposed by Frank Gray in 1953 [21]. Kawaguchi converts the cover image to CGC prior to embedding in order to both increase the payload capacity and reduce the presence

[1]According to the HTTP Archive [24] as of 1st March 2014

of visual defects within the stego-image. The algorithms (3 and 4) used within this implementation are based on those created by Doran [12].

Following extensive trial and error, it was discovered that if CGC is used, conversion must take place on the whole image, not within the bit-planes nor the segments. Doing otherwise results in colour imbalances and large artefacts in the final stego-image.

Algorithm 3: Conversion from PBC to CGC

```
input  : An integer in PBC form
output : An integer in CGC form
return pbc $\oplus$ RightShift(pbc)
```

Algorithm 4: Conversion from CGC to PBC

```
input  : An integer in CGC form
output : An integer in PBC form
pbc $\leftarrow$ 0 ;
while cgc does not equal 0 do
    pbc $\leftarrow$ pbc $\oplus$ cgc;
    cgc $\leftarrow$ RightShift(cgc);
return pbc;
```

3.1.7 Embedding and Extraction

While the implementations of the algorithm vary greatly between the systems, both combine embedding and extraction into a single programme. This decision was made in order to facilitate batch testing of the system, by introducing the ability to link embedding and extraction together at various stages of the process.

3.2 BPCS within BMP Images

Both the embedding and extraction of data within BMP images rigorously follows the method set out by Kawaguchi, implementing Algorithm 1 for embedding and its inverse for extraction. The program was highly paramatised to allow the algorithm to be customised with each run. In addition a wrapper was created to handle batch tests and iterative testing of multiple parameters in sequence.

3.3 BPCS within JPEG 2000

3.3.1 JJ2000: JPEG 2000 Codec

In order to avoid duplication of existing work, it was deemed prudent to use a 3rd party JPEG 2000 codec within the programme. JJ2000 [28] was chosen as the most appropriate implementation because, of the three[2] publicly available codecs, it is the only one which is open source and suitable for Java development.

JJ2000 implements part 1 of the JPEG 2000 standard in its entirety, while also including elements from part 2, such as enhanced region of interest encoding and variable code-block partitions [27]. The extensions to the original specification were disabled in this implementation in order to conform solely to Part 1 of JPEG 2000.

[2]along with OpenJPEG [8] and Kakuda [30].

This allowed the project to focus on embedding within images obeying the core of the system without requiring the added refinement mandated by the high degree customisability introduced with each subsequent addition to the specification.

3.3.2 Integration with JJ2000

JJ2000 fell out of active development four years ago [3], after it was dropped from the Java Advanced Imaging Library, and documentation of the project is sparse. This led to a significant proportion of the development time being spent analysing the source in order to be able to better adapt it for BPCS embedding.

The codec was designed based on a Chain of Responsibility pattern, where each component is only aware of the block of data it is currently modifying and fetches subsequent blocks from the next step in the chain. Each stage of the JPEG 2000 encoding process, described in Figure 2.5 above, is implemented as a link in this chain. Apart from the beginning and end steps, the only other link in the chain to have an awareness of the full image was the wavelet transform component, which converted the spatial representation of the image to a wavelet representation during encoding, and vice-a-versa during decoding. This ensured data duplication was kept to a minimum, while maintaining an efficient implementation of the standard.

This pattern posed a number of challenges for combining the decoding and encoding modules, as required in order to enable embedding within the correct bit-plane structure. Two approaches were possible: an overhaul of the system could be performed in order to unite the inheritance threads of the data structures which, while functionally similar were kept distinct for both encoding and decoding; alternatively an adaptor could be created which would transfer and convert the data between the various structures. While an overhaul would have been preferable, and allowed for a more efficient implementation of the BPCS algorithm by benefit of being able to link both the encoding and decoding chains of responsibility, an adaptor was chosen due to the need to be able to guarantee blocks were encoded sequentially, which was impossible with the current codec.

Using the adaptor results in a programme which will take a JPEG 2000 file as input and output exactly the same data, despite having recalculated the image's ROIs, as well as both tiers of coding. The adaptor is a two stage process; the first section gathers the image data from the decoder while the second liaises with the encoding chain. The BPCS algorithm works in conjunction with both stages of the algorithm, though it is unnecessary for the program to execute.

The first stage of the adaptor gathers all the image data from the decoder, after ROI Descaling, and stores the data as a series of bit-planes associated with each block. The data is organised into a five-dimensional array, according to the tile, colour component, resolution level, sub band and block respectively. The array allows for data to be gathered and returned in a consistent order, ensuring sequential retrieval of embedded data is possible. Without this guarantee, embedded data can lose its coherence and the output file can become garbled.

The adaptor also manages the relevant metadata for both the decoder and encoder. This includes creating the analysis sub band tree, which while differing from its synthesis parallel only in the type of filter used for generating the wavelets, is treated as different classes by JJ2000.

The second stage of the adaptor manages the interaction between the encoding and decoding chains by keeping a record of returned blocks and providing an interface for the ROI encoder, which replaces the Quantiser used in the original implementation. Blocks are returned in sequential order, based first on the colour component, before stepping through each resolution level and sub band and finally returning blocks in spatial order. This structure remains constant between the original cover and subsequent image, which ensures the payload can be extracted accurately.

[3]The last commit to the repository [28] was made on 18 November 2009

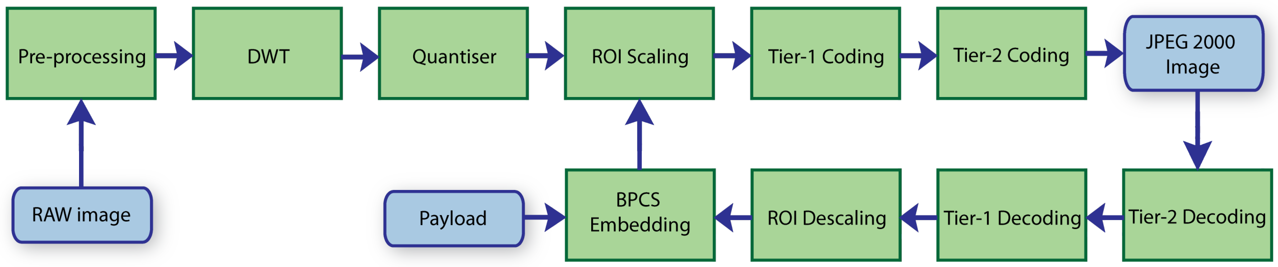


Figure 3.2: BPCS embedding within JPEG 2000

3.3.3 Embedding and Extraction

The BPCS module for JPEG 2000 drew on Noda et al.'s [45] research, modifying Kawaguchi's original algorithm (Algorithm 1) for use in JPEG 2000's wavelet coefficients. The bit-plane structure of wavelet coefficients means major adaption of the algorithm was unnecessary, with the most arduous alteration being the actual shift in domain. Shifting embedding to the transform domain, means embedding data during the encoding process which, as a lossy compression algorithm, is likely to compromise the integrity of the payload.

Ideally, BPCS would take place immediately following the wavelet transform but without bypassing the post-compression rate-distortion optimisation, it is impossible to know the lowest significant bit-plane of each sub band until after encoding is complete. This process is essential for the effective compression of the image and bypassing it would go against the premise of the project, which requires embedding within a lossy medium. Noda et al. [45] suggest combining both the embedding and decoding processes to better facilitate embedding, as shown in Figure 3.2. Utilising this process provides access to the best bit-plane structure while also maintaining the integrity of the JPEG 2000 standard.

Using Noda's scheme, the bit-plane structure of each sub band could be accessed and used for embedding after ROI Decoding, before passing the data back to the encoder for repackaging. Even with Noda's system, care had to be taken in order to ensure the correct bit-planes were used within each block. JJ2000 associates the lowest significant bit plane with the sub band containing the blocks, which could be extracted and used as the basis for the algorithm. Combining both of these ideas resulted in Algorithm 5, which will always embed within in the lowest bit-plane possible, before ascending the planes in order of significance.

Extraction followed similar lines to the embedding process. The decoder was once again utilised until after ROI Descaling, at which point the data was passed into the adaptor which planed the data in preparation for extraction. Finally the extractor accessed the planes in the same order as they were encoded and extracted the data in sequential order, beginning with the payload meta data, which determines how many blocks to scan for data.

3.3.4 Tiling

JPEG 2000 offers the ability to divide an image into a set of equally sized tiles, each of which can be decoded independently of the whole image. This incurs a potential problem for steganography, whereby the embedded payload could be scattered across tiles resulting in complicated extraction and the need to embed a set of payload metadata information segments within each tile. Within this implementation, where an image is decomposed into multiple tiles, the payload is always embedded within the first tile and not spread through the image. While this results in a potential loss of payload capacity, the guarantee of retrieval was deemed more important.

Algorithm 5: BPCS within JPEG 2000 algorithm

```
Convert image from PBC to CGC;
foreach bit-plane in image do
    foreach component in image do
        foreach resolution level in component do
            foreach sub band in resolution level do
                get least significant bit-plane for sub band ;
                if current bit-plane < least significant bit-plane then
                    continue ;
                foreach block in sub band do
                    foreach 8x8 segment in block do
                        if NoiseCalculation(segment) > $\alpha$ then
                            if NoiseCalculation(payload segment) < $\alpha$ then
                                Conjugate(payload) ;
                            replace segment with payload segment ;
```

Chapter 4

Evaluation

4.1 Performance

The performance of both implementations was measured by investigating their payload capacity, ability to retrieve embedded data, and the system's ability to withstand steganalysis. These are standard metrics for evaluating steganographic techniques, and provide a point of comparison between the systems and the previous literature.

In order to provide a fair comparison between the implementations, both were tested against the same set of lossless images, converted into both BMP and lossy JPEG 2000 (Appendix B).

4.1.1 Noise Threshold

Niimi, Noda, and Kawaguchi [42] established the best threshold for BPCS within lossless images as 0.3 according to the black and white border measure, following research into informative and noise-like regions of images. This figure has remained constant throughout the literature and it was expected that the same would hold true when embedding within the transform domain, due to its correlation with the spatial domain.

In order to discover the best threshold for JPEG 2000 covers, an experiment was conducted in the same vein as that of Niimi. Given a threshold, all segments with greater likeness to noise were replaced with randomly generated bit patterns, which were at least as noise-like as the segment they were replacing. The experiment was repeated with thresholds increasing in steps of 0.05 on each of the images within the test suite. The resulting set of images was subject to HVS analysis, beginning with the highest threshold and working down until alterations could be detected. Using this approach, a small range of possible thresholds at which the image would pass as innocuous under the HVS were reached (Table 4.1), all of which were significantly higher than that set out by Niimi.

One of the properties of the black and white border measure states that informative image data is confined to complexities below 0.5, and any data above that limit is guaranteed to be noisy [42]. JPEG 2000 offers increased compression efficiency over its predecessor, which suggests that wavelets are calculated in such a way that non-informative information is kept to a minimum; the JPEG 2000 encoding makes use of the 'concept of irrelevance' which removes details of the image which cannot be perceived by the HVS [1]. Taking this into consideration helps to explain the increased threshold indicated by the results; with less irrelevant data, there is a greater chance of impacting the image when embedding within the informative spectrum. Further, this suggests that embedding should be confined purely to noise-like regions, measuring above complexity 0.5, which is congruent with the results in Table 4.1. In order to ensure stego-images will always pass the HVS a 'safe' complexity threshold needed to be defined as the highest limit provided by the results. Therefore 0.6 became the universal embedding limit for this system, a marked increase to the 0.3 used for BMP images.

File Name	Threshold	Cap. at Threshold (KB)	Cap. (%)
airplane.jp2	0.6	3.58	2.36
baboon.jp2	0.5	12.34	3.07
barbara.jp2	0.6	6.41	2.16
goldhill.jp2	0.6	7.90	2.33
lenna.jp2	0.55	5.65	2.57
mandrill.jp2	0.55	17.95	3.45
pepper.jp2	0.6	4.64	1.72
Average:	0.57		2.52

Table 4.1: Complexity thresholds at which stego-images withstand inspection with HVS

4.1.2 Payload Capacity

The payload capacity is dictated by the informative threshold, so for sake of comparison both implementations were initially tested at the established lossless threshold of 0.3, providing an average capacity of 58.1% for BMP covers (Table 4.2) and 19.2% for JPEG 2000 (Table 4.3). As suggested by Kawaguchi and Eason [32], these results were obtained by converting the images to Canonical Gray Code prior to embedding, without which capacities were, on average, 14.1% lower for BMP covers (Table 4.2) and 5.4% lower for JPEG 2000 covers (Table 4.3), which is consistent with the results in the literature.

Subsequently, the JPEG 2000 implementation was tested with the new safe threshold established above, which dropped the payload capacity to just 2.52% of the image size (Table 4.1). While the lower capacity for JPEG 2000 versus BMP is to be expected, considering the limited bit planes in which to embed and the scaled data set, the sharp decline with the increased threshold warranted further investigation.

Using the capacities established by a threshold of 0.6 as a guide, payloads were generated for each cover and subsequently embedded at both thresholds 0.6 and 0.3. Surprisingly, neither set of the resulting stego-images exhibited visual anomalies (Figure 4.1), suggesting that the defects were associated with the size of the payload rather than the threshold; the threshold only inadvertently preventing distortion by limiting the capacity.

In an effort to pin down the reason for the defects, the next step was to analyse how embedding behaved within each plane. A cumulative test was performed such that an increasing maximum bit-plane was specified for embedding on each iteration of the test. The experiment was repeated for each of the 30 planes within the images of the suite and at each threshold in the range [0..0.7], stepped in increments of 0.05. The results indicated that defects occur more frequently as the maximum bit-plane increases, which is consistent with the upper planes' increased significance. More importantly, for each threshold tested, when embedding was performed at plane 22 and above defects where more abundant, while embedding below 22 did not result in perceivable changes in the final image (Figure 4.2).

As the boundary case, stego-images with data embedded upto and including plane 22 were subject to wider visual analysis in order to ascertain a realistic threshold for the system (Appendix C). Respondents were presented with a series of images, containing both stego and cover images, and asked to indicate whether they believed the images to have been digitally doctored in some way. None of the images in the suite garnered a perfect sweep of false-negatives from the participants, though this may have been caused by bias induced by requesting users to actively seek anomalies in the images. However, a significant proportion (47.3%) of responses were false-negatives, with some images passing inspection by the HVS over 85% of the time. This suggested that while not a suitable threshold for maintaining a constant level of secrecy, if a system requires increased payload capacity instead of guaranteed secrecy then plane 22 may be suitable.

With plane 22 emerging as the boundary case in each of the images, and at each of the tested thresholds, the

File Name	**Size (KB)**	**Cap. (KB)**	**Cap. wi. Gray (KB)**	**Cap. (%)**	**Cap. wi. Gray (%)**	**% increase**
6-4-balloon.bmp	75	28.02	31.98	37.4	42.6	12.4
airplane.bmp	786	346.33	431.82	44.1	54.9	19.8
baboon.bmp	720	454.40	539.94	63.1	75	15.8
barbara.bmp	1215	707.23	839.37	58.2	69.1	15.7
BoatsColor.bmp	1329	611.13	749.30	46	56.4	18.4
goldhill.bmp	1215	662.63	794.68	54.5	65.4	16.6
lenna.bmp	786	299.66	297.90	38.1	37.9	-0.6
pepper.bmp	786	422.92	496.57	53.8	63.2	14.8
			Average:	49.4	58.1	14.1

Table 4.2: BMP embedding performance

File Name	**Size (KB)**	**Cap. (KB)**	**Cap. wi. Gray (KB)**	**Cap. (%)**	**Cap. wi. Gray (%)**	**% increase**
airplane.jp2	152	34.2	31.6	22.5	20.8	7.7
baboon.jp2	402	66.6	64.0	16.6	15.9	3.9
barbara.jp2	297	63.4	60.1	21.4	20.2	5.2
goldhill.jp2	339	72.4	68.1	21.3	20.1	5.8
lenna.jp2	220	41.4	38.8	18.8	17.6	6.3
mandrill.jp2	520	91.9	89.1	17.7	17.1	3.0
pepper.jp2	270	44.4	41.7	16.5	15.5	6.1
			Average:	18.2	19.2	5.4

Table 4.3: JPEG 2000 embedding performance

File Name	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.3**	**0.35**	**0.4**	**0.45**	**0. 5**	**0.55**	**0.6**
airplane.jp2	32.05	30.37	28.33	26.35	25.00	22.83	20.59	18.01	14.56	11.85	6.66	2.23
baboon.jp2	11.79	11.78	11.75	11.68	11.64	11.56	11.46	11.27	10.65	9.55	6.01	2.06
barbara.jp2	27.34	26.22	24.92	23.65	22.69	21.11	19.39	17.22	14.28	11.57	6.37	1.91
goldhill.jp2	24.63	24.10	23.21	22.38	21.74	20.69	19.48	17.82	15.37	12.75	7.28	2.20
lenna.jp2	25.42	24.49	23.22	21.92	21.09	19.65	18.05	19.65	13.25	10.60	5.52	1.65
mandrill.jp2	11.29	11.29	11.27	11.25	11.22	11.17	11.10	10.97	10.53	9.61	6.34	2.17
pepper.jp2	21.24	20.80	20.16	19.40	18.80	17.68	16.40	14.71	12.39	10.10	5.50	1.70
Average:	21.97	21.29	20.41	19.52	18.88	17.81	16.64	15.66	13.00	10.86	6.24	1.99

Table 4.4: Payload capacities (as %) with maximum bit-plane 21 and varying capacity

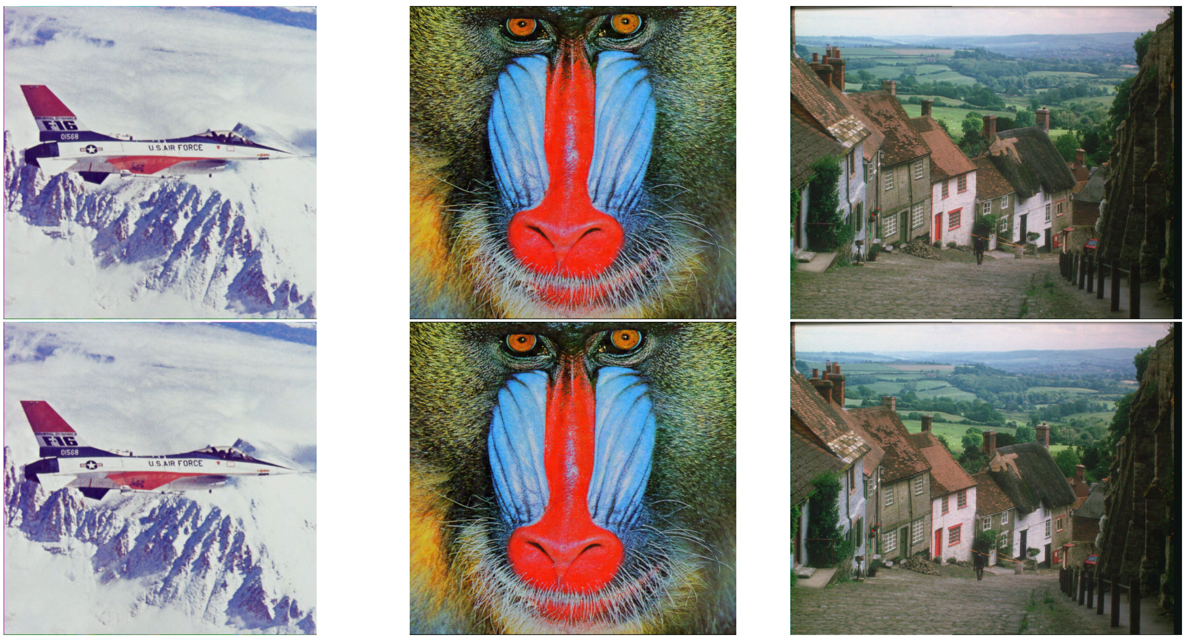


Figure 4.1: JP2 images embedded with capacity from Table 4.1, with a 0.3 threshold (top) and a 0.6 threshold (bottom)

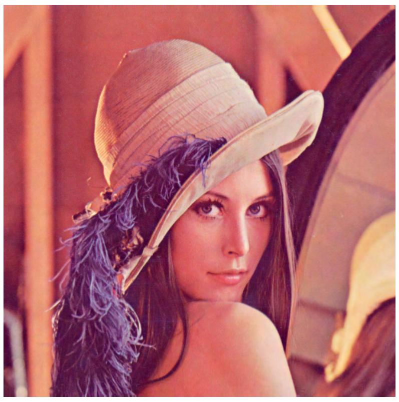
consistency of the results suggest that complexity measures aren't necessary within low significance bit-planes. In addition, using the threshold negatively impacts the maximum payload capacity (Table 4.4). The lack of differentiation between the images in Figure 4.3 highlights the redundancy of distinguishing the information capacity of images below plane 22, at least with regard to the HVS.

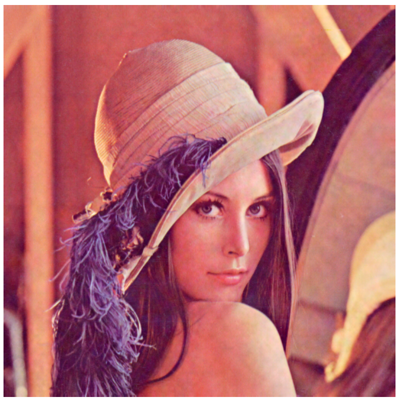
Furthermore, it was noted that the bulk of embedding capacity was contained within planes below 22, with an average increase of just 2% within planes 22-32. This is congruent with the findings in the literature [42, 43, 32] and exemplify the property of the black and white border measure which places informative regions within bit-planes of high significance. Therefore, while a progressive algorithm which utilised the noise-like measure in planes 22 and above is possible, the possibility of introducing perceivable defects within these planes has a significant chance of outweighing the slight capacity increase which could be gained. As such imposing plane 21 as an absolute embedding platform was considered the best conclusion to these results, with the possibility of increasing to 22 if the emphasis of the application is on capacity instead of secrecy.

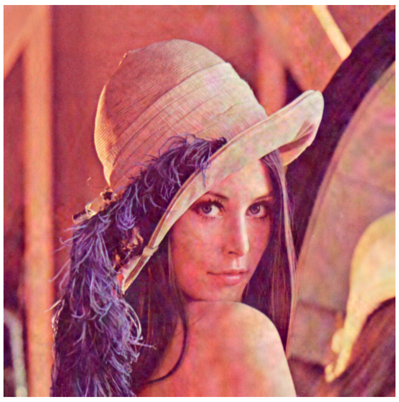
Using the 0.3 threshold defined by Kawaguchi for the BMP implementation, provided payload capacities which were consistent with the literature, but examining the stego-images produced at these limits, highlighted distinctive visual anomalies. These defects were reduced with the use of CGC by preventing colour changes being mistaken for embedding regions, but distortion was still visible. It emerged, that while some images could be embedded with payloads measuring upto 50% of the cover's size, others would begin producing defects with payloads of 40% of the image size. Therefore, a the practical capacity of the system was capped at 40% which is significantly less that proposed by Kawaguchi, though the difference could be a result of the different set of images used within both studies.

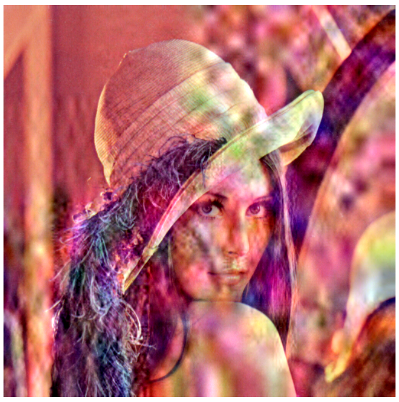
4.1.3 File size

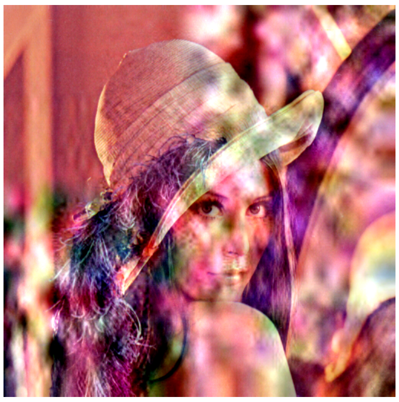
One of the simplest means of detecting the presence of embedded data is comparing the file size of the stego-object to that of similar images, or the original if available. This is an issue which plagues the reversible data hiding community [46] but has proven not to be an issue with either of the implementations for this project. The BMP system produces stego-objects which don't vary in size with the original by even a single byte, which is to be expected due to the direct correspondence between the size of the payload and the number of bits removed


(a) Up to plane 21


(b) Up to plane 22


(c) Up to plane 23


(d) Up to plane 26


(e) Up to plane 29

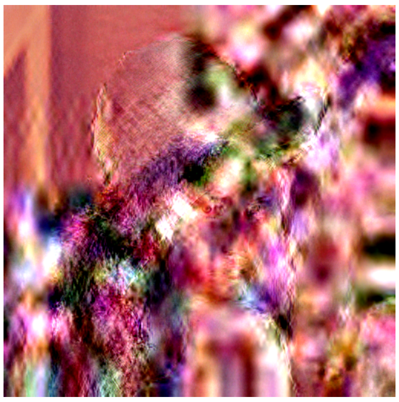
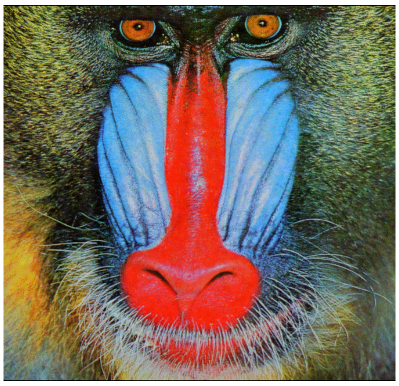
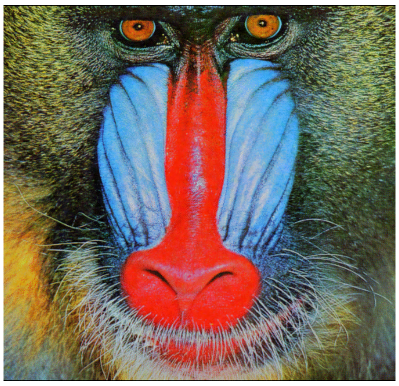

(f) Up to plane 31

Figure 4.2: Embedding in increasing number of planes, using threshold, α=0.3 and the maximum embedding capacity


(a) α = 0.05


(b) α = 0.3

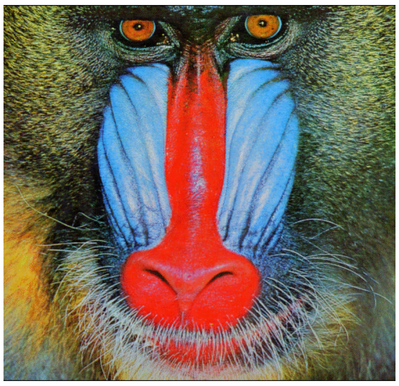

(c) α = 0.6

Figure 4.3: Embedding with a maximum bit-plane of 22, a payload of 20KB and varying threshold, α

from the cover. The JPEG 2000 implementation is not as reliable, with fluctuations +/- 4 bytes introduced into the stego-objects. Due to the degree of customisability within the JPEG 2000 standard and the variations introduced by different compression settings, these fluctuations are not outwith the norm for similar files. As a result, using this metric it is impossible to detect the presence of data embedded by the JPEG 2000 system without the exact cover as the point of comparison.

4.1.4 Extraction

With standard BPCS, both the size and the type of the payload, along with the threshold used for embedding, are required in order to successfully extract the data. The modified algorithm used in JPEG 200 now also required the maximum bit-plane used for embedding. The size and the type are stored in the first two complex blocks of the image, which removes the need for the user to know the specifications of the payload, though it introduces a single point of failure to the system; any corruption to the size segment will result in unsuccessful extraction. On the other hand if a similar corruption occurs in the type segment, extraction can still be completed successfully with the output file defaulting to text data. This means the only requirements for extraction are the threshold and maximum bit-plane.

The problem of corruption compromising the payload isn't a new issue within steganography, and in some cases it is viewed as a beneficial feature of the system. Fragility within the system can be used to wipe the data from the stego-image, resulting in an innocuous cover which is similar to the original. One of the simplest methods of exploiting the system's fragility is to re-compress the image, which will cause disruption within in the lower planes where the payload is stored. On the other hand, the system's lack of robustness to any changes makes malicious attacks very easy to execute. If the goal of the warden is to compromise the payload, in the same way that WWII did by substituting words in letters with synonyms [62], then it is a trivial process to alter the cover slightly and prevent extraction of the secret.

If the cover is not compromised in any way, then extraction can be successfully completed for all payloads which are less than the capacity of the image for any given threshold, minus 16 bytes which are used for storing the payload's metadata.

4.1.5 Performance Caveats

One of the aims for the project was to be able to embed data within pre-encoded JPEG 2000 covers. In its current form, this implementation is only able to perform embedding within images which have been encoded with the JJ2000 package. This is due to the reliance of the BPCS module on chaining the encoding and decoding modules, and utilising information from each simultaneously. JPEG 2000 images which have been generated with other systems will still pass embedding and data can still be extracted from them, but the resulting stego-images have a high likelihood of displaying significant distortion. Both the saturation and colour of the image are markedly different from the original cover, blatantly revealing the existence of embedded information (Figure 4.4). Therefore, despite embedding being successful, the result doesn't uphold the stenographic principle and therefore cannot be deemed a success with external covers.

4.2 Statistical Analysis

The results of visually analysing the JPEG 2000 implementation implied the need to distinguish between noise-like and informative segments, which is key to the success of the original BPCS algorithm, is unnecessary below plane 22 within JPEG 2000 covers. Such analysis is the most primitive barrier a steganography technique needs to pass in order to be deemed secure; it is rudimentary and has proven to be trivially deceivable.

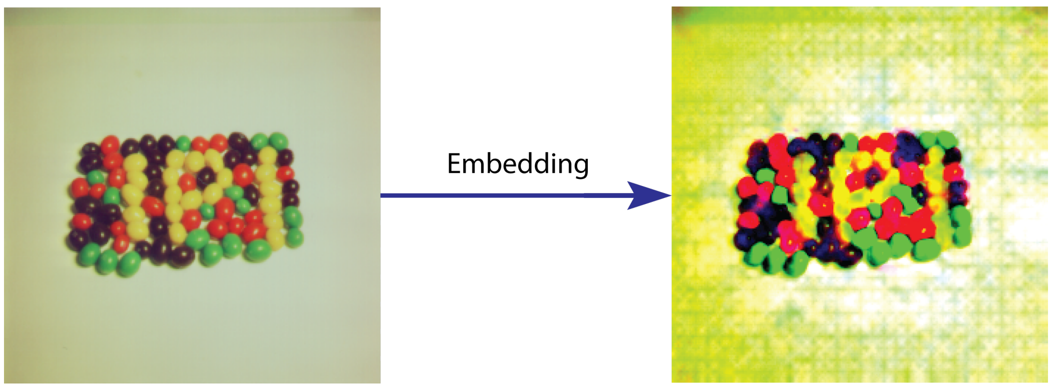
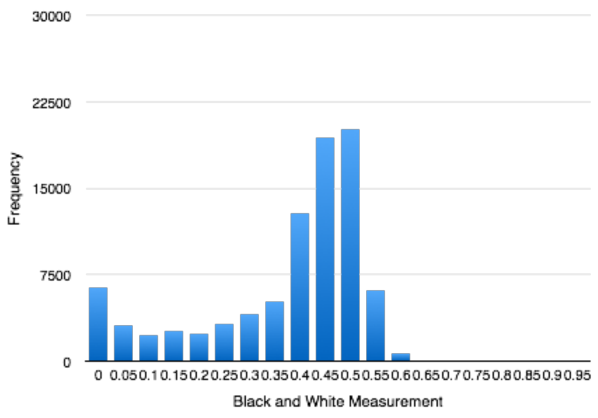
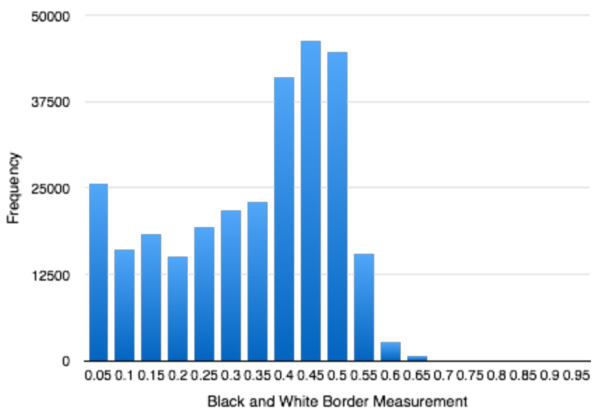


Figure 4.4: BPCS embedding of JP2 file created with external embedder



(a) The first 22 planes only



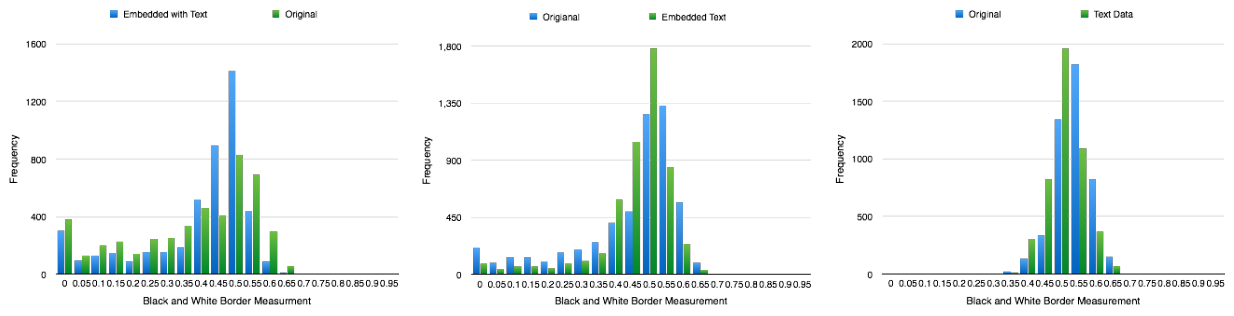
(b) All planes, without empty segments

Figure 4.5: Average JP2 measurements according to the Black and White Border metric

One of the more sophisticated steganalysis tools investigates the periodicity of the image by analysing the histograms of a visual complexity measure of the image and its components, with the aim of detecting statistical anomalies in the randomness of the image. In this instance the histogram of the black and white border measurement was used, with bins ascending in intervals of 0.05. Such analysis has revealed the presence of the payload within other implementations of BPCS; the original BPCS algorithm stored conjugation flags as a map in each plane [32], which proved to be easily detectable using the histogram of the least significant bit-plane [41]. This system avoids such detection by storing the conjugation flags in the last bit of each segment, but similar analysis can also be used to detect suspicious spikes in the histogram, resulting from frequent conjugation [40].

The first stage of this investigation sought to identify trends within the set of possible covers by computing the histograms of unaltered JPEG 2000 images (Figure 4.5). Analysing the resulting graphs revealed a general upwards trend below 0.5, followed by a sharp decline corresponding to a small number of segment measuring greater than 0.5. This trend fits with the research into the Black and White Border metric, which states that measurements above 0.5 contain no meaningful information perceivable by the HVS, being more akin to random noise [42]. Narrowing the focus of the histogram to the first 22 bit-planes produced a more extreme bias towards 0.5, with far fewer segments measuring in the range $[0..0.4]$. This is consistent with the notion of more significant bit-planes contributing more meaningful information, while noise-like areas are largely confined to the lower planes.

After establishing a baseline for the potential covers, the same process with repeated for stego-objects embedded with a variety of 20KB payloads. As expected, comparing the histograms of the unaltered covers with the stego-objects highlighted noticeable alterations, which could be used to detect the presence of embedded data if the original cover is available. Though this is true of any pure steganographic technique: any comparison which uses



(a) Airplane (b) Averages based on set (c) Mandrill

Figure 4.6: Text embedding

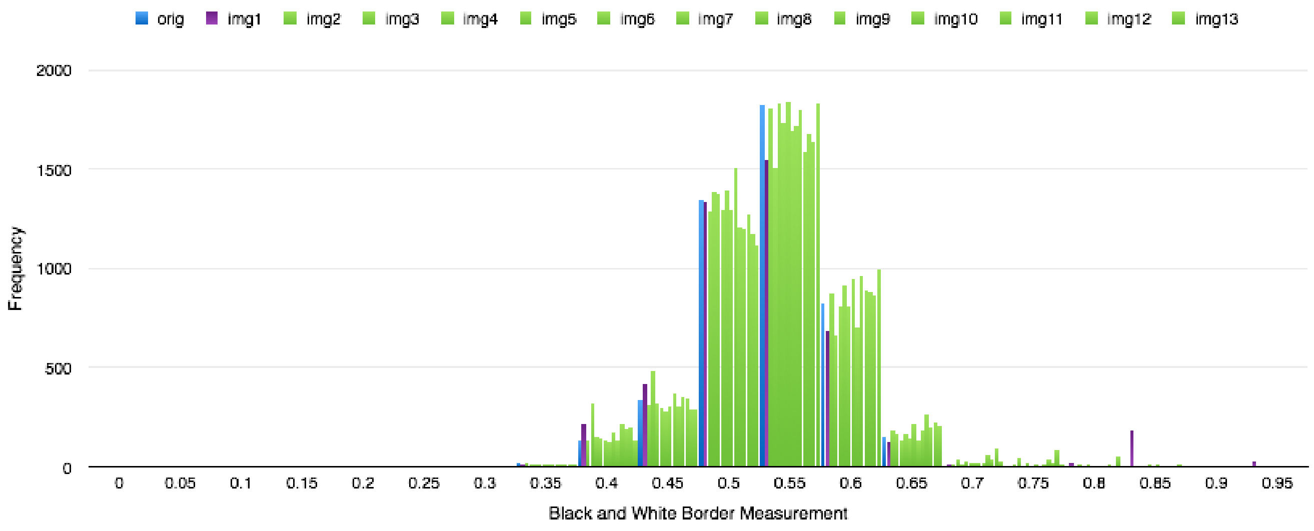


Figure 4.7: Embedding of image data within Mandrill

the original will reveal the presence of altered bits in the stego-image and extraction can swiftly follow. Pure-steganography relies on the original being kept out of the public domain to preserve the secrecy of the embedded data; anyone who has access to the cover can easily access the payload by a simple comparison. For this reason, it is assumed that Eve has no knowledge of the unaltered image and, instead, has to rely on the known properties of the set of covers, in this case the baseline established above.

Histograms of the stego-images embedded with text messages have a characteristic spike at the 0.5 bin (Figure 4.6), which can be used to identify not just the presence of embedded data but also the type of data. The spike is a major deviation from the baseline, as can be seen in figure 4.6b, which proves to be a reasonably reliable signature for this steganography technique. False-positives are possible with covers which exhibit uncharacteristic spikes, as was noted with the mandrill image (Figure 4.6c).

Image data was much harder to detect consistently, relying on properties of individual payloads. Generally, embedding image data resulted in slight changes distributed across the histogram, rather than causing noticeable spikes. The resulting graphs maintained the overall shape of the baseline and therefore anomalies where difficult to detect. Occasionally, segments in the stego-images of embedded data would measure above 0.6 which could then be used as markers for the presence of image data. The JPEG 2000 baseline clearly shows a minimal number of segments measure greater than 0.6 and therefore any images which exhibited spikes in this area were immediately suspicious. This was a rare occurrence, with the vast majority of stego-images causing false-negatives in this test; in the case of embedding within the mandrill image (Figure 4.7) only one stego-image out of 15 exhibited this telling marker.

Whether using text or image data, properties of the payload can betray the presence of the embedded data when

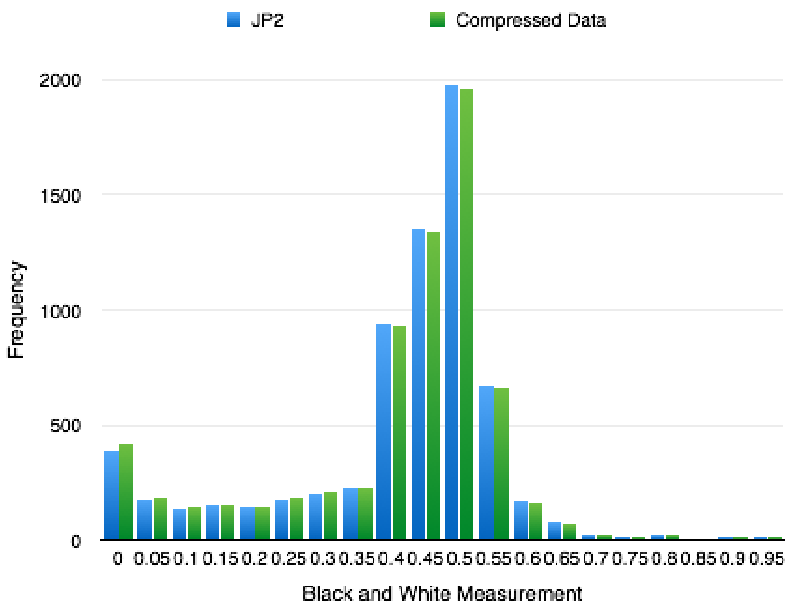


Figure 4.8: Black and White border measure of the JP2 average and post embedding of compressed data

subject to statistical analysis. This occurs because, as has been well established within this project, information isn't random; the periodicity of the data provides an easily detectable signature associated with the data type of the payload. In order to measure the information content of data, Shannon's entropy is used to determine the uncertainty within the data [55]. Shannon defined entropy as:

$$H(X) = \sum_i P(x_i) \log_2(P(x_i)) \tag{4.1}$$

Where X is the set of possible variables and $P(x)$ is the probability of one of those variables occurring. Using $\log_2$ measures entropy in bits [5].

As entropy is addictive, the entropy of a stego-object, s, can be defined in terms of the entropy of the cover, c, and the payload, p:

$$H(s) = H(c) + H(p) \tag{4.2}$$

Therefore in order to mitigate the possibility of detecting the embedded data based on statistical properties of the payload, $H(s)$ needs to be as close to $H(c)$ as possible, meaning $H(p)$ needs to be kept to within the limits of acceptable $H(c)$ values.

In order to reduce $H(c)$, and thereby increase the randomness of the payload, the data can be encrypted, compressed or both prior to embedding. These processes remove the characteristic periodicity of the data as they remove the informative capacity of the data, making it more akin to random. To prove this property, the statistical analysis was repeated once the images had been embedded with 20KB of compressed data. The results, shown in Figure 4.8, illustrate a lack of the telling signatures which were present for both image and text data. Therefore the JPEG 2000 system can resist statistical analysis of the informative capacity of the image, if the payload is sufficiently random.

All these tests were performed with the noise threshold set to 0 and the maximum embedding set to 22, but repeating the experiments with increasing values of both parameters didn't significantly alter the results. Therefore, these results are all consistent with the idea of confining embedding to below plane 22, and the ability to arbitrarily embed within these planes without needing to differentiate between noise-like and informative segments.

Chapter 5

Conclusion

This project aimed to determine the suitability of Kawaguchi and Eason's [32] Bit-Plane Complexity Segmentation steganography technique within JPEG 2000 lossy cover images. Following in the footsteps of Noda et al. [45], BPCS was implemented for JPEG 2000 by shifting embedding to the transform domain, allowing payload data to be stored in the quantised wavelet coefficients resulting from the Discrete Wavelet Transform. The resulting system generated surprising results, prompting an exploration of the properties of the stego-objects it produced. Consequently, a revised algorithm was developed which is a significant departure from existing BPCS literature.

Wavelet coefficients provide an ideal alternative to the spatial data used by Kawaguchi, because they are resilient to small changes and have scaled correspondence with regions in the spatial domain. The direct correlation between the transform and spatial domains suggested the resulting system would generate results comparable with Kawaguchi's original investigation. Instead, the system behaved unexpectedly, prompting scrutiny into the embedding process, with particular emphasis on the role of visual complexity.

Subjecting the system to popular steganalysis techniques revealed visual and statistical anomalies not present within an implementation of Kawaguchi's algorithm for lossless BMP covers. Initial visual analysis immediately betrayed the presence of embedded data due to the large scale defects introduced into the stego-object. These defects could be minimised, but not eradicated, by converting the image to Canonical Gray Code prior to embedding, as suggested by Kawaguchi, and avoiding the most significant bit-plane, which is analogous to the sign bit and can therefore be expected to introduce major alterations into the resulting image. Further investigation concluded defects would materialise with regularity in planes above 22, while those below wouldn't generate anomalies perceptible by the HVS. Plane 22 was a borderline case which would introduce defects dependent on the image itself. In addition it was noted that the informative capacity of the segments had no bearing below plane 22, a distinct departure from Kawaguchi's findings which are heavily dependent on noise-like regions in each plane.

The irrelevancy of the noise-like threshold was seen again when the stego-objects where exposed to statistical analysis of the randomness of each segment. By calculating a histogram of the black and white border measure of the original cover and the stego-image after embedding with a variant of payloads, it was observed that altering the threshold made little discernible difference to the final image. What statistical analysis did highlight was vulnerability of the algorithm to periodicity in the payload; text based payloads cause spikes in the histogram derived from their uniformity. This suggested payloads need to be akin to random data in order withstand steganalysis, which can be achieved by compressing the data prior to embedding. The histogram of stego-object with compressed payloads was indistinguishable from the set of possible values for JPEG 2000.

Together visual and statistical analysis revealed the unsuitability of BPCS for JPEG 2000 covers. However, it can be used as a starting point for developing a steganography technique tailored to the characteristic properties of JPEG 2000. The new algorithm (Algorithm 6) retains conversion to CGC and bit-plane segmentation, but bypasses the black and white border calculations, replacing them with a maximum embedding platform of plane

22. This revised system exhibits significant improvements on Noda's work, specifically an increase in payload capacity to 23% of the cover's size, an increase of 42% on Noda's results. It has been proved to be resistant to both visual and statistical analysis of the visual complexity of the stego-image. In addition it retains the resistance to structural attack from the original BPCS algorithm, by maintaining the integrity of the JPEG 2000 standard. Together these features illustrate a marked improvement on Noda's research and combine to create a steganogrpahy technique with increased suitability for embedding within JPEG 2000.

The creation of a steganography algorithm which provides a practical means of embedding within a lossy medium enables the possibility of using the Internet as a viable communication channel. Considering the prevalence of lossy images online and the resulting suspicion of lossless images [13], the environment of the Internet isn't conducive to preserving the secrecy of messages embedded with BPCS. But excitingly, the new algorithm produces stego-objects which fall inline with the norms of the Internet and can therefore be transmitted online without fear of discovery.

In the future, further investigation into the role of segmentation and the ideal block size could reveal further improvements and reduce the redundant space at the edges of the image. An additional avenue of exploration is the potential of adapting the algorithm to utilise the black and white border measure in a progressive manner for planes above 22. There is also the potential for investigating methods of increasing the robustness of the algorithm, such as limiting the embedding sub bands or utilising error correction codes.

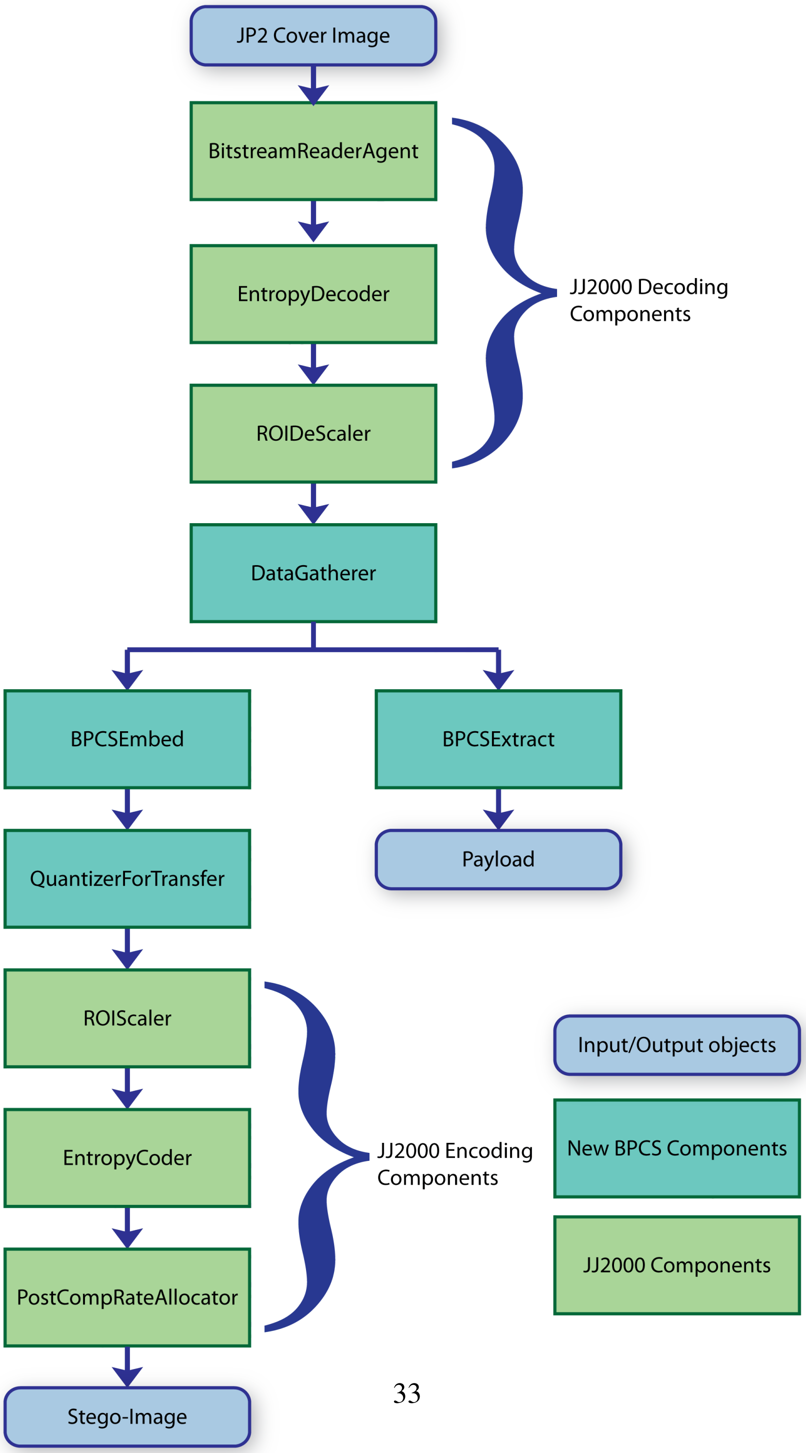
Algorithm 6: Revised Steganography for JPEG 2000

```
Convert image from PBC to CGC;
while Current Bit-Plane < 22 do
    foreach component in tile do
        foreach resolution level in component do
            foreach sub band in resolution level do
                get least significant bit-plane for sub band ;
                if current bit-plane < least significant bit-plane then
                    continue ;
                foreach block in sub band do
                    foreach 8x8 segment in block do
                        replace segment with payload segment ;
    Increment current bit-plane;
Convert image from CGC to PBP;
```

Appendices

Appendix A

BPCS in JJ2000



Appendix B

Test Images

This suite of images, accumulated from [51] and [61], were converted from their lossless PPM format to both BMP and lossy JPEG 2000 as test material for both systems.

(a) abe_natsumi

(b) airplane

(c) anonymous1

(d) baboon

(e) barbara

(f) brian_kernighan

(g) goldhill

(h) hara_fumina

(i) lenna

(j) megumi_oishi

(k) pepper

(l) reese_witherspoon

Appendix C

Plane 22 Comparison

A group of 16 respondents were provided with one of two sets of images containing a mix of stego and cover images, though never two of the same image. Each of the participants was asked to select whether they believed the image had been doctored in anyway based purely on visual inspection. Images were provided in a random order to prevent bias.

This particular subset of the test aimed to assess the boundary for maximum embedding. All the images on the left are unaltered JPEG 2000 covers, while the ones on the right are the stego-images resulting from embedding a compressed file approximately equal to the payload capacity of the image.

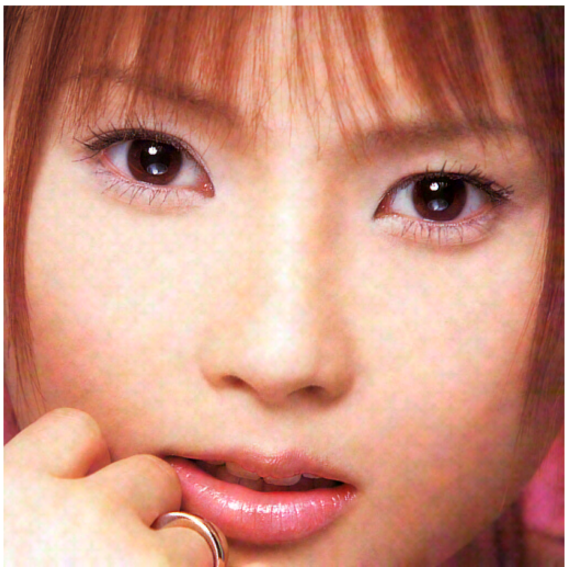
With the vast majority of the images below, a direct comparison between the original and stego-object reveals readily apparent defects, but without the original to compare to some of the images passed HVS analysis.

The results (summarised in Table C.1) suggest photographs of human faces do not make good cover images, with all bar one of the images containing a face being consistently labeled as doctored. 43% of the images were thought to be doctored less than 25% of the time.

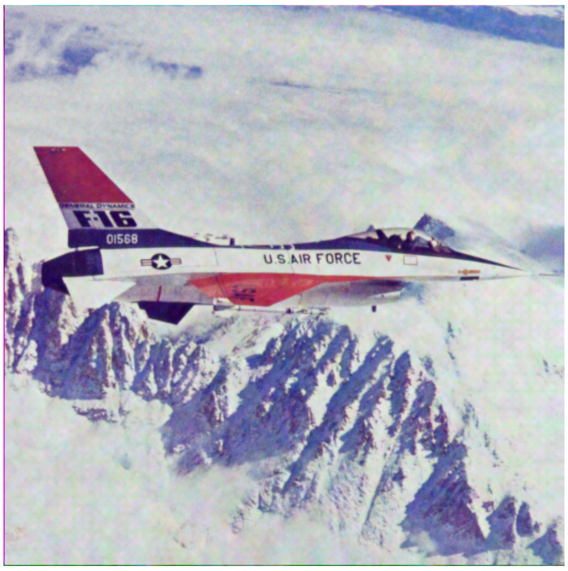
File Name	**% who believed image was doctored**
abe-natsumi	87.5
airplane	50
anonymous1	68.75
baboon	12.5
barbara	56.25
brian-kernighan	25
goldhill	68.75
hara-fumina	93.75
lenna	18.75
megumi-oishi	81.25
pepper	6.25
reese-witherspoon	62.5

Table C.1: Results of User test for Plane 22 comparison

(a) Original

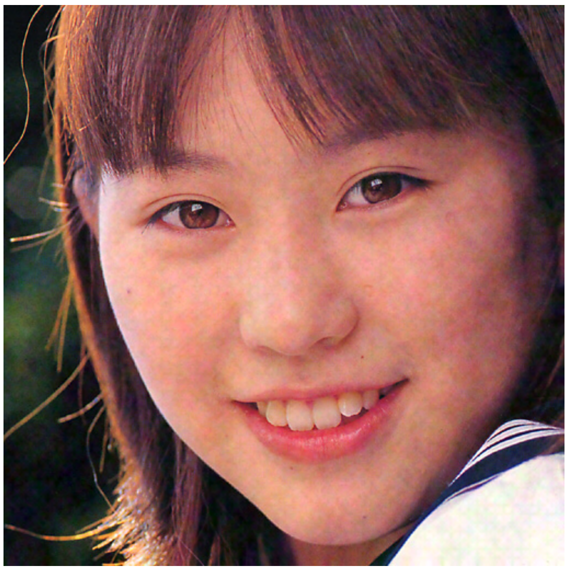

(b) Stego-object

(a) Original

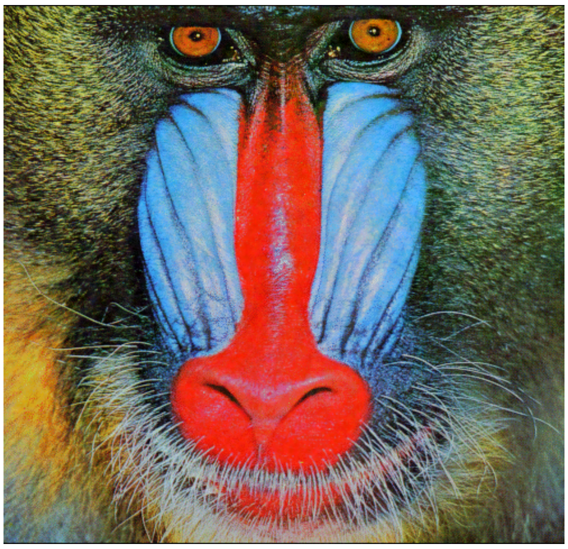


(b) Stego-object

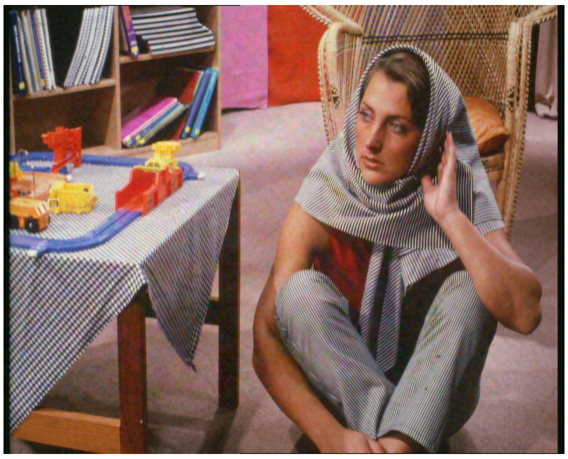
(a) Original


(b) Stego-object

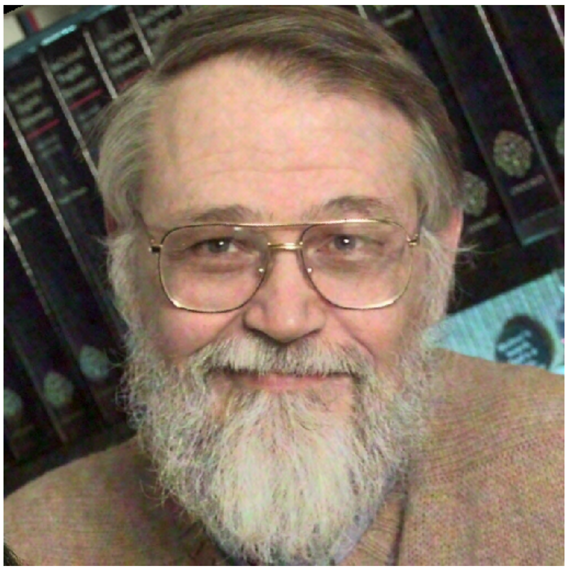
(a) Original


(b) baboon

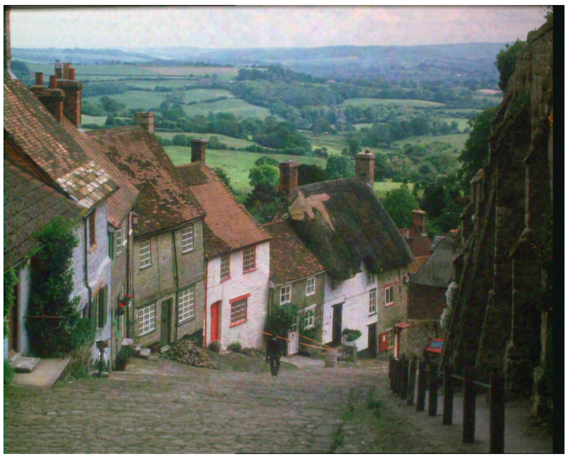
(a) Original


(b) Stego-object

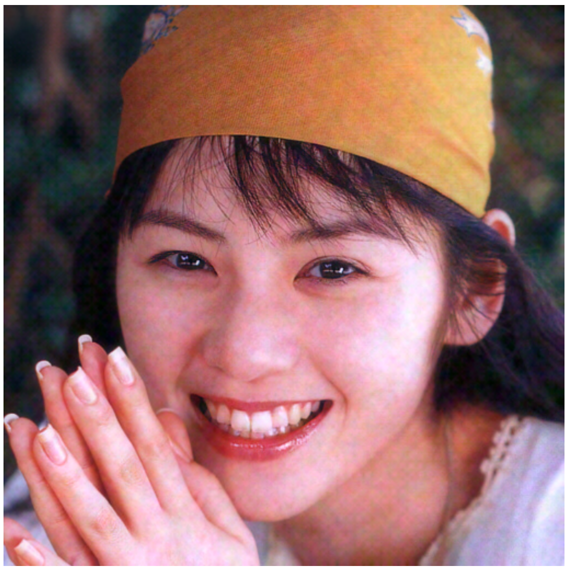
(a) Original


(b) Stego-object

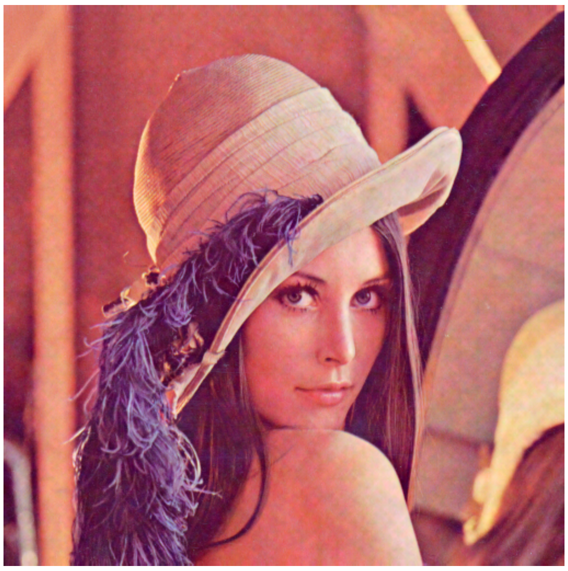
(a) Original

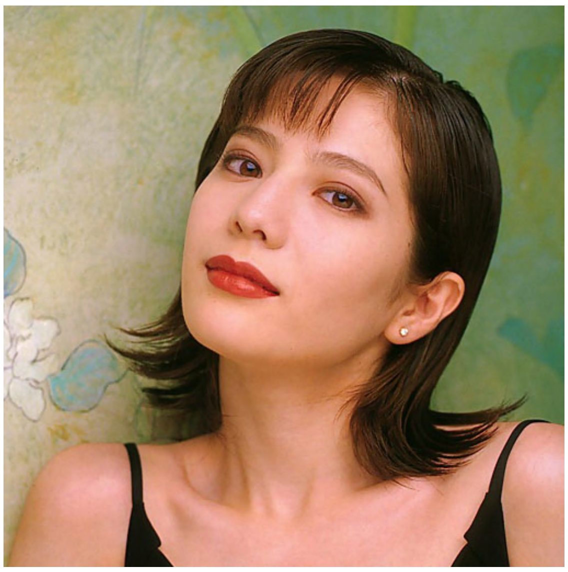

(b) Stego-object

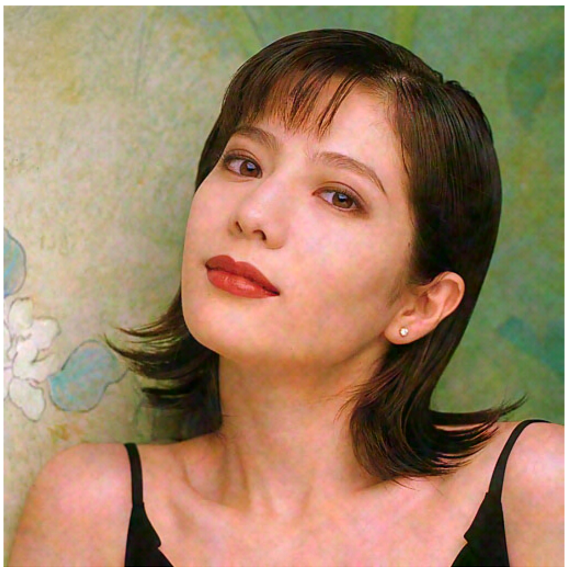
(a) Original

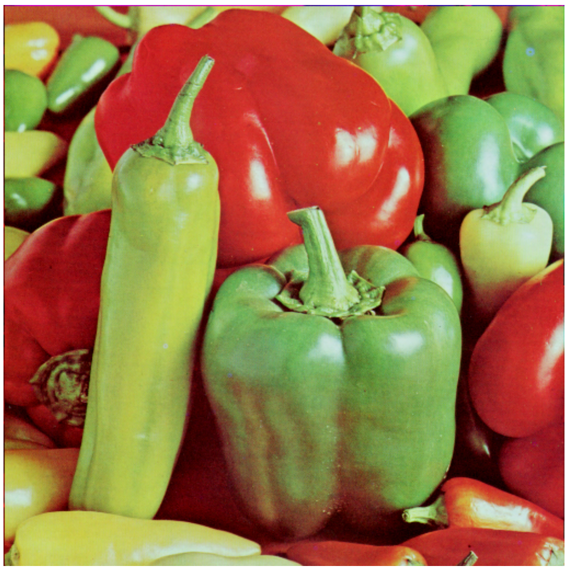

(b) Stego-object

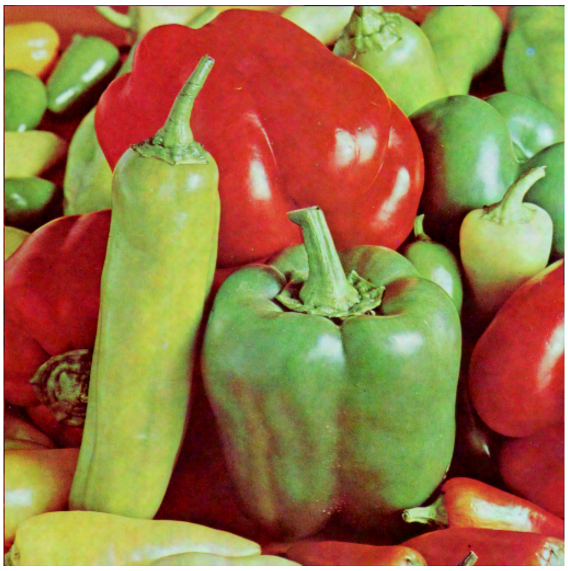
(a) Original

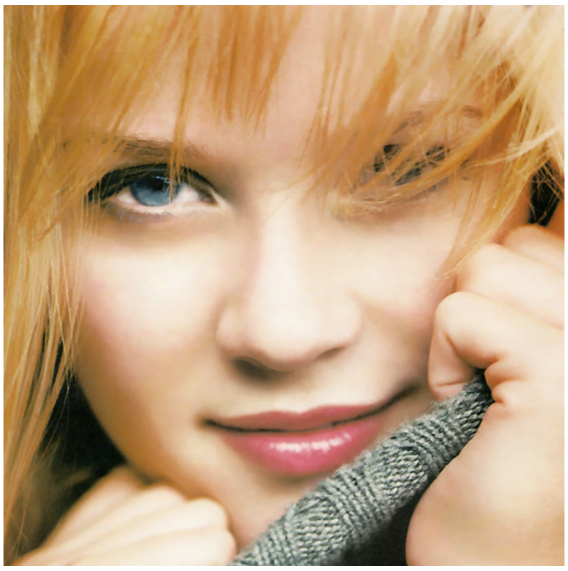

(b) Stego-object

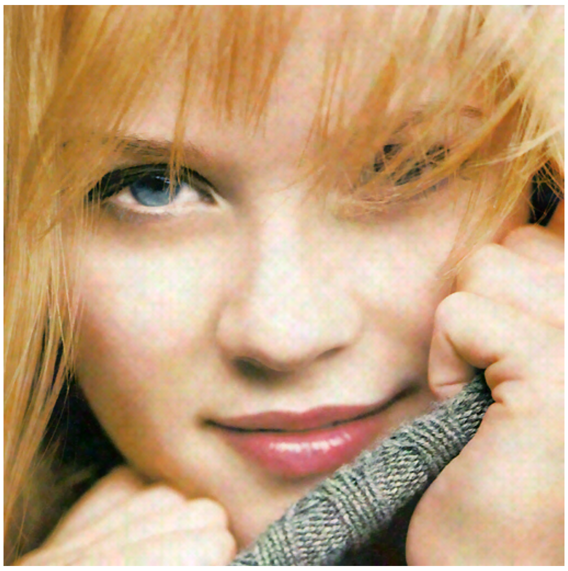

(a) Original


(b) Stego-object


(a) Original


(b) Stego-object


(a) Original


(b) Stego-object

Appendix D

Running the Programs

The distribution of this project contains three separate systems: the original JJ2000 codec; the implementation of BPCS for BMP images; and the implementation of BPCS for JPEG 2000 images.

For instructions on running the original JJ2000 codec, please refer to the manual included with the library.

D.1 BPCS in BMP

This module implements Kawaguchi's BPCS algorithm for BMP covers. It functions as both embedding and retrieval modules, depending on which parameters are set.

D.1.1 Single file mode

To run the the program for a single file, use the BPCSInBMP programme.

The following arguments are recognised:

```
-ext [on|off] (default = off)
        dictates whether to extract data or not
-i <filename>
        specifies the file in which to embed/extract. Must be a BMP file
-o <filename>
        specifies the output file, either the stego-object during embedding or the
-p <filename>
        specifies the payload file
-c [0..1]
        specifies the complexity threshold with which to embed
-console <filename>
        specifies the file to save stdOut
-u [on|off] (default = off)
        Prints usage information. If specified all other arguments (except 'v') are
-v [on|off] (default = off)
        Prints version and copyright information
-debug [on|off] (default = off)
        Determines whether to conduct debug operations
-gray [on|off] (default = on)
        Specifies whether to convert to Canonical Gray Code
```

```
-cap [on|off] (default = off)
        Returns an estimated capacity for the given cover image
```

These arguments are entered at the command line, such that:

```
      > java BPCSInBMP -i cover.bmp -p lorem.txt -o output.bmp -c 0.3
```

will embed lorem.txt into cover.bmp and produce the stego-object, output.bmp using the black and white border threshold of 0.3

To reverse the process, the following command would be used:

```
      > java BPCSInBMP -ext on -i output.bmp -o message
```

which would extract the original payload and save it, with appropriate file extension, to the file message.

D.1.2 Batch file mode

To run the the program as a batch test, use the BatchBPCS programme.

This module provides for batch testing of payloads and cover images. It takes a parameter string of the from:

```
        > java BatchBasicBPCS <inputs.txt> <payloads.txt>
```

Each parameter file should contain a list of paths to payloads/covers, one per line

D.2 BPCS in JPEG 2000

This module implements Kawaguchi's BPCS algorithm for JPEG 2000 covers.It functions as both embedding and retrieval modules, depending on which parameters are set.

D.2.1 Single File Mode

To run the the program for a single file, use the BPCSInJJ2K programme.

The following arguments are recognised:

```
 -u [on|off] (default = off)
      Prints usage information. If specified all other arguments (except 'v')
      are ignored
-v [on|off] (default = off)
      Prints version and copyright information
-verbose [on|off] (default = on)
      Prints information about the decoded codestream
-pfile <filename>
      Loads the arguments from the specified file. Arguments that are
      specified on the command line override the ones from the file.
      The arguments file is a simple text file with one argument per line of
      the following form:
        <argument name>=<argument value>
      If the argument is of boolean type (i.e. its presence turns a feature
      on), then the 'on' value turns it on, while the 'off' value turns it
      off. The argument name does not include the '-' or '+' character. Long
      lines can be broken into several lines by terminating them with '\'.
      Lines starting with '#' are considered as comments. This option is not
      recursive: any 'pfile' argument appearing in the file is ignored.
-i <filename>
      The file containing the JPEG 2000 compressed data.
-o <filename>
      This file to which the result of BPCS encoding will be written.
-payload <filename>
      This file contains the data to be embeded
-bpcsExt [on|off] (default = off)
      Specifies whether BPCS extraction should take place
-bpcs [on|off] (default = on)
      Specifies whether any form of BPCS should take place
-c <complexity threshold [0..1)>
      Specifies the complexity threshold to use for BPCS embedding
-console <filename>
      Specifies the file for printing stdOut
-gray [on|off] (default = on)
      Specifies whether to convert to Canonical Gray Code
-maxbp [0-32] (default = 21)
      Specifices which bit plane to use as the maximum
```

These arguments are entered at the command line, such that:

```
      > java BPCSInJJ2K -i cover.jp2 -p lorem.txt -o output.jp2 -c 0.0 -maxbp 21
```

will embed lorem.txt into cover.jp2 and produce the stego-object, output.jp2 using the maximum embedding plane of 21.

To reverse the process, the following command would be used:

```
      > java BPCSInJJ2K -bpcsExt on -i output.bmp -o message -maxbp 21
```

which would extract the original payload and save it, with appropriate file extension, to the file message.

D.2.2 Batch file mode

To run the the program as a batch test, use the BatchBPCS programme.

This module provides for batch testing of payloads and cover images. It takes a parameter string of the from:

```
        > java BatchBPCS <parameter-to-test> <listOfVariables.txt>
                         <OtherParameters.txt> <inputs.txt>
```

parameter-to-test is the letter code of the parameter to test (see above).

ListOfVariables is a text file containing a different variable to be assigned to the parameter-to-test per line.

OtherParameters is a text file containing the values of different parameters on separate lines, e.g:

```
-c
0.0
-maxbp
22
-gray
off
```

Inputs is a text file containing a list of paths to cover images, one per line.

Bibliography

[1] M. Adams. *The JPEG-2000 Still Image Compression Standard*. 2013.

[2] Aineias Tacticus. *How to Survive Under Siege*. Trans. by D. Whitehead. Bristol: Bristol Classical Press, 2002.

[3] M. Asad, J. Gilani, and A. Khalid. “An enhanced least significant bit modification technique for audio steganography”. In: *Computer Networks and Information Technology (ICCNIT), 2011 International Conference on*. 2011, pp. 143–147.

[4] W. Bender, D. Gruhl, N. Morimoto, and A. Lu. “Techniques for data hiding”. In: *IBM Systems Journal* 35 (1996), pp. 313–336.

[5] L. Brillouin. *Science & Information Theory*. Dover Publications, 2004.

[6] M. Chaney. *Whatever Happened to JPEG2000*. URL: http://www.steves-digicams.com/knowledge-center/whatever-happened-to-jpeg2000.html#b (visited on 03/06/2014).

[7] E. Cole. *Hiding in Plain Sight: Steganography and the Art of Covert Communication*. Indianapolis: Wiley, 2003.

[8] Communications and Remote Sensing Lab of the Université catholique de Louvain. *OpenJPEG*. URL: https://code.google.com/p/openjpeg/ (visited on 03/07/2014).

[9] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. New York: John Wiley & Sons, Inc., 1991.

[10] R. Cradall. *Some Notes on Steganography*. 1998. URL: http://www.di.unisa.it/~ads/corso-security/www/CORSO-0203/steganografia/LINKS%20LOCALI/matrix-encoding.pdf (visited on 03/22/2014).

[11] K. Dasgupta, J. K. Mondal, and P. Dutta. “Optimized Video Steganography Using Genetic Algorithm (GA)”. In: *First International Conference on Computational Intelligence: Modeling Techniques and Applications* (2013).

[12] R. Doran. “The Gray Code”. In: *Journal of Universal Computer Science* 13 (2007), pp. 1573–1597.

[13] J. J. Eggers, R. Bäuml, and B. Girod. “A Communications Approach to Image Steganography”. In: *in Proceedings of SPIE: Electronic Imaging 2002, Security and Watermarking of Multimedia Contents IV*. 2002, pp. 26–37.

[14] *Eyewitness*. URL: http://www.archives.gov/exhibits/eyewitness/flash.php (visited on 03/04/2014).

[15] J. Fridrich. *Steganography In Digital Media*. Cambridge: Cambridge University Press, 2010.

[16] J. J. Fridrich, M. Goljan, and D. Soukal. "Searching for the stego-key". In: *Security, Steganography, and Watermarking of Multimedia Contents* (2004), pp. 70–82.

[17] X. Gao, W. Lu, D. Tao, and X. Li. "Image quality assessment and human visual system". In: *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*. Vol. 7744. Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series. 2010.

[18] H. Georg Schaathun. "On Watermarking/Fingerprinting for Copyright Protection". In: *Innovative Computing, Information and Control* 3 (2006), pp. 50–53.

[19] M. Ghebleh and A. Kanso. "A robust chaotic algorithm for digital image steganography". In: *Communications in Nonlinear Science and Numerical Simulation* 19 (2014), pp. 1898–1907.

[20] P. Gramantik. *New iFrame Injections Leverage PNG Image Metadata*. 2014. URL: http://blog.sucuri.net/2014/02/new-iframe-injections-leverage-png-image-metadata.html (visited on 03/06/2014).

[21] F. Gray. "Pulse code communication". 2,632,058. 1953.

[22] M. Gupta and A. Gilbert. "Robust speech recognition using wavelet coefficient features". In: *Automatic Speech Recognition and Understanding, 2001. ASRU '01. IEEE Workshop on*. 2001, pp. 445–448.

[23] Herodotus. *Herodotus*. Ed. by J. M. Marincola. London: Penguin Classics, 2003.

[24] HTTP Archive. *Interesting Stats*. 2014. URL: http://httparchive.org/interesting.php (visited on 03/07/2014).

[25] Interagency Working Group on Cyber Security and Information Assurance. *Federal Plan for Cyber Security and Information Assurance Research and Development*. Virginia: NCO/NITRD, 2006.

[26] *ISO/IEC 15444-1: Information technology—JPEG 2000 image coding system—Part 1: Core coding system*. 2000.

[27] *ISO/IEC 15444-2:2004 Information technology – JPEG 2000 image coding system: Extensions*. 2004.

[28] *jj2000*. URL: https://code.google.com/p/jj2000/ (visited on 03/07/2014).

[29] Joint Photographic Experts Group. *ISO/IEC 10918: Information technology – Digital compression and coding of continuous-tone still images*. 1994.

[30] Kakuda. *JPEG 2000 Development Toolkit*. URL: http://www.kakadusoftware.com/ (visited on 03/07/2014).

[31] E. Kawaguchi and R. O. Eason. *Principle and Applications of BPCS-Steganography*. 1998. URL: http://web.eece.maine.edu/~eason/steg/SPIE98.pdf (visited on 03/04/2014).

[32] E. Kawaguchi and R. O. Eason. “Principles and applications of BPCS steganography”. In: *Proc. SPIE*. Vol. 3528. 1999, pp. 464–473.

[33] E. Kawaguchi, T. Endo, and J. Matsunaga. “Depth-first Picture Expression Viewed From Digital Picture Processing”. In: *IEEE Trans on PAMI* 5 (1988), pp. 373–384.

[34] J. Kelley. *Militants wire Web with links to jihad*. 2001. URL: http://usatoday30.usatoday.com/news/world/2002/07/10/web-terror-cover.htm (visited on 03/06/2014).

[35] A. D. Ker. “Steganalysis of Embedding in Two Least-Significant Bits”. In: *IEEE Transactions on Information Forensics and Security* (2007), pp. 47–54.

[36] A. D. Ker. “Steganalysis of LSB Matching in Grayscale Images”. In: *IEEE Signal Processing Letters* 12 (2005), pp. 441–444.

[37] A. Kerckhoffs. “La cryptographie militaire”. In: *Journal des sciences militaires* 9 (1883), pp. 5–83.

[38] G. Kipper. *Investigator’s Guide to Steganography*. Boca Raton: Auerbach Publications, 2003.

[39] M. J. Nichols and W. T. Newsome. “The Neurobiology of Cognition”. In: *Nature* 402 (1999), pp. C35–C38.

[40] M. Niimi, R. O. Eason, H. Noda, and E. Kawaguchi. “Intensity histogram steganalysis in BPCS-steganography”. In: *Security, Steganography, and Watermarking of Multimedia Contents* 3 (2001), pp. 555–564.

[41] M. Niimi, T. Ei, H. Noda, E. Kawaguchi, and B. Segee. “An attack to BPCS-steganography using complexity histogram and countermeasure”. In: *Image Processing, 2004. ICIP ’04. 2004 International Conference on*. Vol. 2. 2004, pp. 733–736.

[42] M. Niimi, H. Noda, and E. Kawaguchi. “An image embedding in image by a complexity based region segmentation method”. In: *Image Processing, 1997. Proceedings., International Conference on*. Vol. 3. 1997, pp. 74–77.

[43] M. Niimi, H. Noda, and E. Kawaguchi. “Steganography based on region segmentation with a complexity measure”. In: *Systems and Computers in Japan* 30.3 (1999), pp. 1–9.

[44] M. Niimi, H. Noda, and B. Segee. “A Study on Visual Attack to BPCS-Steganography and Countermeasure”. In: *Visual Content Processing and Representation*. Ed. by L. Atzori, D. Giusto, R. Leonardi, and F. Pereira. Vol. 3893. Lecture Notes in Computer Science. Berlin: Springer, 2006, pp. 29–36.

[45] H. Noda, J. Spaulding, M. Shirazi, M. Niimi, and E. Kawaguchi. “Bit-Plane Decomposition Steganography Combined with JPEG2000 Compression”. English. In: *Information Hiding*. Ed. by F. Petitcolas. Vol. 2578. Lecture Notes in Computer Science. Berlin: Springer, 2003, pp. 295–309.

[46] M. Nosrati, R. Karimi, and M. Hariri. “Reversible Data Hiding: Principles, Techniques, and Recent Studies”. In: *World Applied Programming* 2 (2012), pp. 349–353.

[47] R. Ouellette, H. Noda, M. Niimi, and E. Kawaguchi. “Topological-ordered color table for BPCS-steganography using indexed color images”. In: *Proc. SPIE* 3971 (2000), pp. 502–509.

[48] T. Pevný and J. Fridrich. “Novelty Detection in Blind Steganalysis”. In: *Proceedings of the 10th ACM Workshop on Multimedia and Security*. 2008, pp. 167–176.

[49] N. Provos and P. Honeyman. “Hide and Seek: An Introduction to Steganography”. In: *IEEE Security & Privacy* 1 (2003), pp. 32–44.

[50] J. Reeds. “Solved: The Ciphers in Book III of Trithemius’s Steganographia”. In: *Cryptologia* 22 (1998), pp. 291–317.

[51] R. Rostamian. *Image files for Math 625*. 2003. URL: http://www.math.umbc.edu/~rouben/2003-09-math625/images.html (visited on 03/24/2014).

[52] F. S. Russell. *Information Gathering in Classical Greece*. Ann Arbor: University of Michigan Press, 1999.

[53] D. Santa-Cruz, T. Ebrahimi, J. Askelöf, M. Larsson, and C. Christopoulos. “An analytical study of JPEG 2000 functionalities”. In: *Proc. of the SPIE’s 45th annual meeting, Applications of Digital Image Processing XXIII*. Vol. 4115. 2000, pp. 446–454.

[54] H. Shahadi and R. Jidin. “High capacity and inaudibility audio steganography scheme”. In: *Information Assurance and Security (IAS), 2011 7th International Conference on*. 2011, pp. 104–109.

[55] C. Shannon. “A Mathematical Theory of Communication”. In: *Bell System Technical Journal* 27 (1948), pp. 379–423.

[56] J. M. Shapiro. “Embedded Image Coding Using Zerotrees of Wavelet Coefficients”. In: *IEEE Transactions on Signal Processing* 41 (1993), pp. 3445–3462.

[57] G. J. Simmons. “The Prisoners’ Problem and the Subliminal Channel”. In: *Advances in Cryptology: Proceedings of CRYPTO ’83*. Plenum, 1983, pp. 51–67.

[58] J. Spaulding, H. Noda, M. N. Shirazi, and E. Kawaguchi. “BPCS Steganography using EZW lossy compressed images”. In: *Pattern Recognition Letters* 23 (2002), pp. 1579–1587.

[59] D. Taubman. “High performance scalable image compression with EBCOT”. In: *IEEE Transactions on Image Processing* 9 (2000), pp. 1151–1170.

[60] J. Trithemius. *Steganographia:Ars per occultam Scripturam animi sui voluntatem absentibus aperiendi certu*. 1621.

[61] M. Wakin. *Standard Test Images: Lena/Lenna*. 2003. URL: http://www.ece.rice.edu/~wakin/images/ (visited on 03/22/2014).

[62] P. Wayner. *Disappearing Cryptography: Information Hiding: Steganography & Watermarking*. 3rd ed. Burlington, MA: Elsevier Inc., 2009.

[63] A. Westfeld. “F5—a steganographic algorithm: High capacity despite better steganalysis”. In: *4th International Workshop on Information Hiding*. Vol. 2137. 2001, pp. 289–302.

[64] A. Westfeld and A. Pfitzmann. “Attacks on Steganographic Systems”. In: *Information Hiding. Third International Workshop*. Ed. by A. Pfitzmann. 2000, pp. 61–76.