Bit-Plane Complexity Segmentation Steganography: Implementation and Analysis

Philip Rodger

School of Computing Science

Sir Alwyn Williams Building

University of Glasgow

G12 8RZ

A dissertation presented in part fulfillment of the requirements of the Degree of Master of Science at the University of Glasgow

September 4th, 2019

Abstract

This project is focused on Steganography using Bit Plane Complexity Segments (BPCS). The technique was first described in 1999 (Kawaguchi & Eason, 1999). They then applied for a US patent to protect the technique (Kawaguchi & Eason, 2002). As it is now 20 years from the earliest filing date (May 21, 1999) the patent protection is now expired. Opening the technique to new implementations and modifications.

Previous implementations of BPCS-Steganography have focused exclusively on 8x8 pixel segment sizes, this project demonstrates the viability of arbitrary segment sizes. In addition to the original BPCS algorithm two variations are implemented. The resulting stego-image image quality is measured using Peak Signal to Noise Ratio.

Acknowledgements

I would like to thank Dr. Ron Poet for his guidance and time in supervising this project.

Contents

Chapter 1 Introduction

Steganography is the practice of embedding data into other data in such a way that the presence of the embedded data is not apparent. The embedded data is termed a “payload”, and the data it is concealed in is the “vessel”(Pfitzmann, 1996).

The goal of using Steganography is an outside observer should not be able to detect, visually or by automation, that the vessel contains a payload. An attacker of steganographic techniques may wish to detect, extract, or destroy the payload.

The obvious application is to hide some secret information so it can be transmitted or retrieved later without raising suspicion. However, the payload does not always need to be secret, it can be used to embed a unique identifier to the vessel known as a “digital watermark” (Liu & He, 2005). This can be used to enforce rights management to identify the source of copyright infringement or leak. The most important steganographic property for this application is robustness of the message to alterations in the vessel, rather than capacity.

To be used as a vessel the medium must contain redundant information that can be replaced by the payload data. The more redundant information, the more can be replaced. Media files - images, audio, video - particularly uncompressed formats, tend to be large and contain high proportions of data that can be replaced without a human noticing. Making them candidates as high capacity vessels(Amin, Salleh, Ibrahim, Katmin, & Shamsuddin, 2003).

The simplest form of steganography in image files is “least significant bit” Steganography(Gupta, Goyal, & Bhushan, 2012). In formats such as .bmp and .png each pixel is represented as combinations of red, green and blue. Each of these component colours are given 8 bits (1 byte) capable of representing 256 different intensities. The lower significant bits can often be replaced without a noticeable effect on the overall image. The argument made against this type of steganography is its limited capacity 10-15% of the vessel(Kawaguchi & Eason, 1999), and that in real images the least significant bits are not random(Lee, Bell, Huang, Wang, & Shyu, 2009).

Steganography and cryptography techniques can both used to secure information, however they are independent. Cryptography scrambles a message so that only the intended recipient can decode it however the ciphertext presence is not concealed. This can be a problem in countries where strong encryption is illegal or subject to key disclosure laws("Regulation of Investigatory Powers Act 2000 (RIPA) Section 51,").

Steganography can be used to hide the presence of a payload which is already encrypted. Ciphertext is typically indistinguishable from random noise to protect against distinguishing attacks(Katz & Lindell, 2014), this is a useful assumption for modeling payloads in Steganography.

Chapter 2 Bit-Plane Complexity Segmentation

The technique is based on the observation that the parts of an image that are most important for human recognition are simple (more uniform) parts. Conversely, selectively replacing the more complex regions ('Segments') with other similarly complex patterns can be done without noticeable effects on the resulting image. The replacement segments may hold a payload.

In order to better understand the technique this project follows the foundational 1998 paper by Eiji Kawaguchi and Richard O. Eason (Kawaguchi & Eason, 1999).

2.1 Bit-Plane

Bit planes are a way of splitting the original 24-bit pixels image into a series of 24 simple single bit pixel images.

To demonstrate this I used the commonly used Lena test image, Figure 1.

Figure 1: Lena Test Image Source (Roberts, 1962).

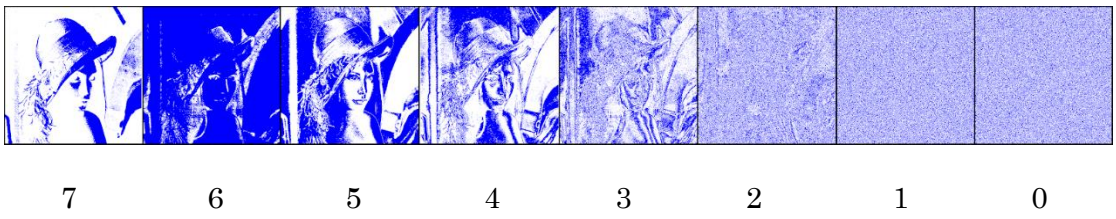

Figure 2: Blue Bit Plane Example

Figure 2, shows the image's blue channel into 8 separate bit images then visualised with blue ('1' or bit set) and white ('0' or off). Left is the most significant bit and right is least significant.

In RGB images each pixel is made by 3 bytes, one byte per colour. These bytes can therefore hold a value 0-255 inclusive (unsigned byte). In Figure 2, the byte representing the blue is separated into individual images representing each bit. The image becomes noisier as the significance of the bit-plane decreases.

While this works with standard binary code, the original BPCS paper proposes a different way of separating the byte into bit planes. The proposed problem with standard binary encoding is that represent adjacent values are often very different in their binary representation. For example:

127_{10} 128_{10}

$0111\ 1111_2$ $1000\ 0000_2$

While the difference between the represented values is only 1, 8 bits in the implementation are flipped. The number of bits that are changed is known as the hamming distance (Hamming, 1950)and the phenomenon of some adjacent values differing having a large hamming distance is called “the hamming cliff”. The effect causes noise in the lower bit-plane images.

To minimise this effect an alternate encoding known as “gray code” (Gray, 1953) is applied to the byte before separating into bit-planes (Kawaguchi, Endo, & Matsunaga, 1983). This minimises the hamming distance in adjacent values to 1 and avoid the hamming cliff effect.

To view the effect on the bit planes I compared the middle blue bit planes of the standard binary coded to the gray code side by side, Figure 3.

There is a noticeable reduction in the noise in the lower bit-planes when the colour intensity is first converted to gray code. This matches the observation in the original paper.

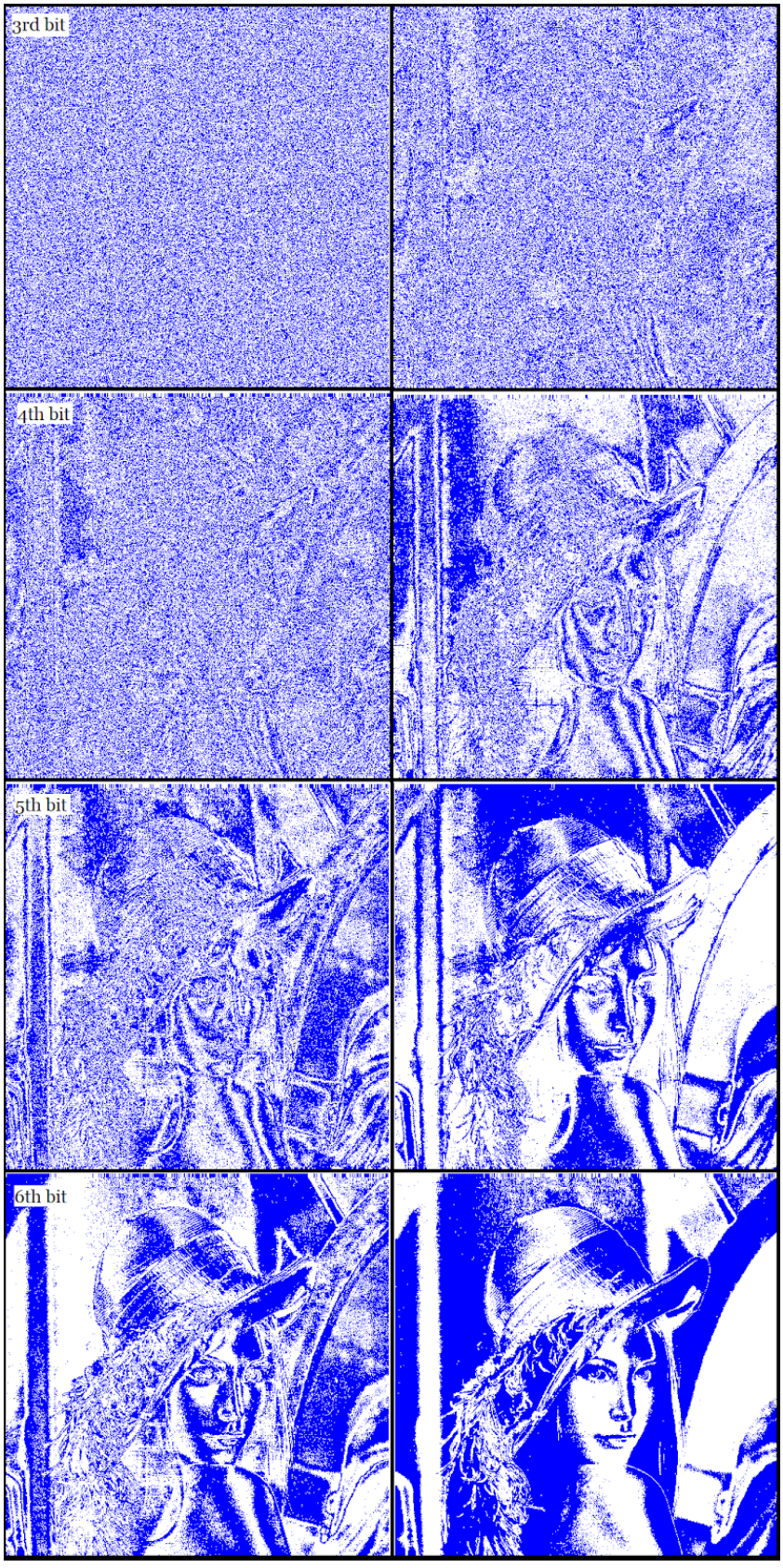

Figure 3: Side by side comparison, standard binary encoding(left), Gray encoded(right)

While the binary code provides more noisy regions for replacement by a payload the paper argues that these extra regions are artificially created by the hamming cliff effect and would be more noticeable if replaced.

2.2 Segments

BPCS considers parts of the bit-plane image in fixed sized windows, in the Kawaguchi-Eason paper these are chosen to be 8x8 blocks. Figure 4 shows an example of a randomly generated block.

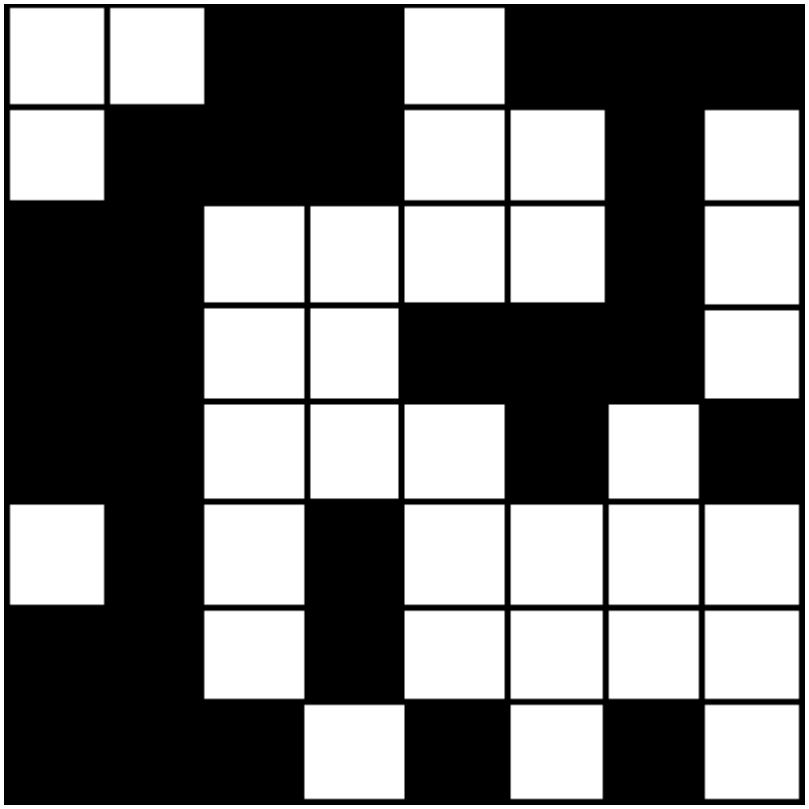

Figure 4: Randomly generated 8x8 bitmap

The original image can be broken down into many of these fixed sized segments and segments that meet a minimum cutoff for complexity are eligible for replacement.

2.3 Measuring Complexity

The black and white border complexity is used in the original paper. This sums the number of colour changes along each row and column in the segment. This can then be divided by the maximum number of changes to get a number between 0 and 1 inclusive.

As a small example, taking a 4x4 segment from the top-left of the above example image there are 8 transitions, Figure 5.

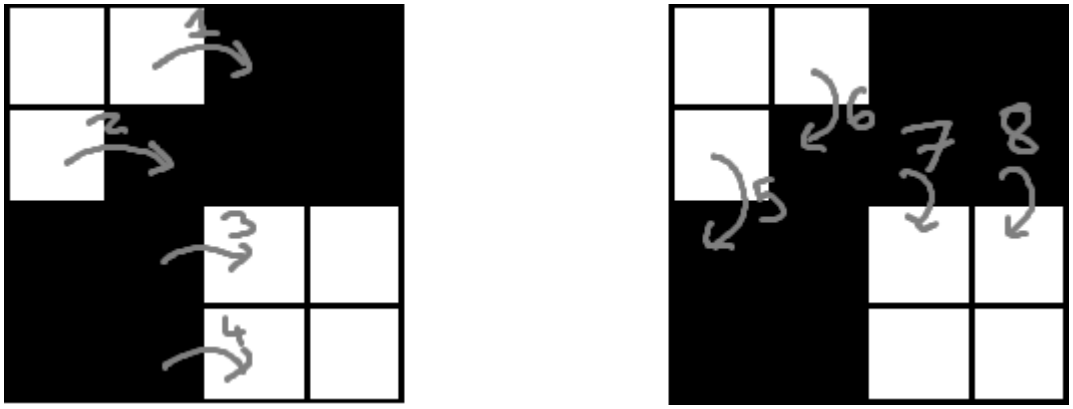

Figure 5: Border Complexity Measurement

The maximum complexity pattern under this definition is a checkerboard pattern and is shown in Figure 6. There are two possible checkerboard patterns and the other can be found by inverting the black and white bits.

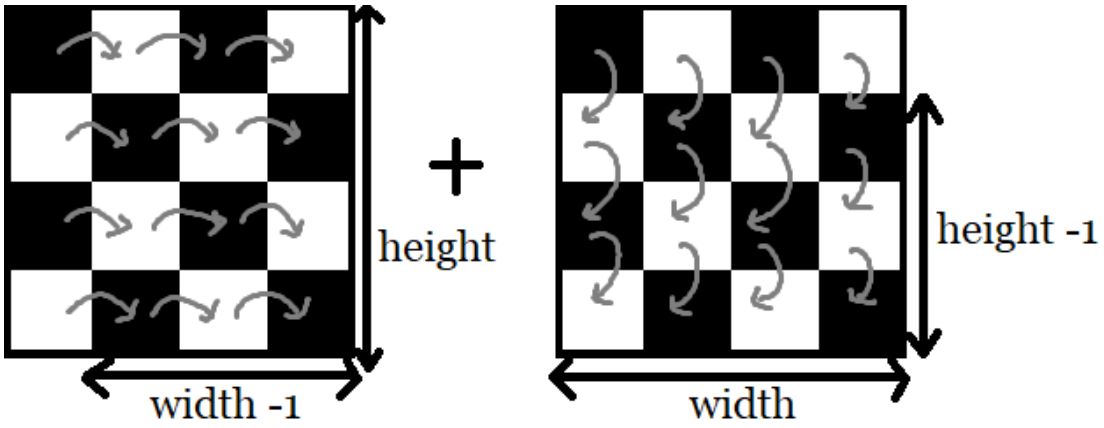

Figure 6: Maximum Complexity

The max complexity of any rectangular segment can therefore calculated by the equation:

```
Max Complexity = ((Width -1) x Height) + (Width * (Height -1))
```

The original paper's equation assumes a square segment size this was required alteration to support arbitrary segment dimensions than the 8x8.

In bit-map shown in Figure 6 therefore has a the complexity is 8/24 or 0.333..

As there was now a way of calculating the complexity of segments, the complexities of each bit-plane for the Lena image (Figure 1) can be plotted as a histogram, Figure 7.

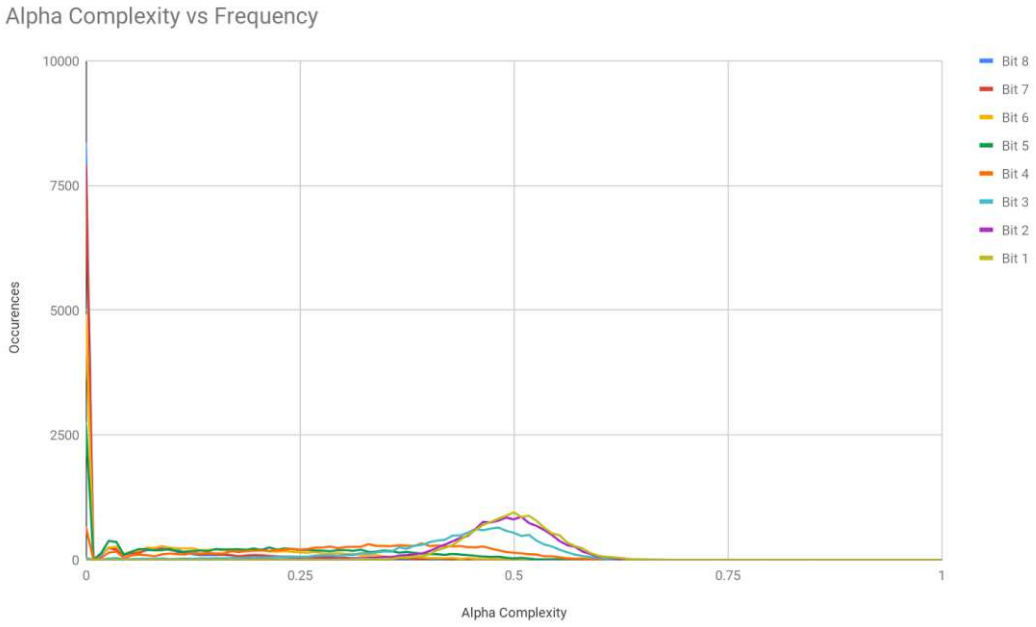

Figure 7: Complexity Histogram(8x8) broken down by bit significance.

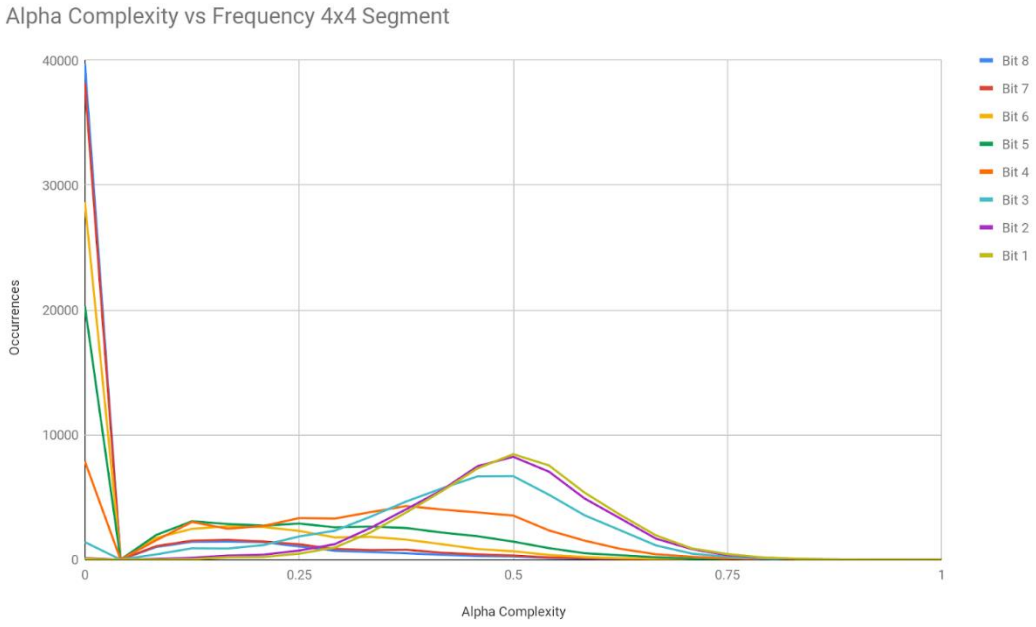

Figure 8: Complexity Histogram (4x4) broken down by bit significance.

The like bit-planes, e.g. least significant bit after grey code for red, green and blue are aggregated. Figure 8 shows 4x4 segments and is easier to see as there are fewer discrete possible categories. There is a 0-frequency complexity that corresponds to a single black-white change because this is not possible in a 2-dimensional segment.

The histograms show that the segments of more significant bit-planes are primarily of low complexity, while the less significant planes, especially the least significant bit appears normally distributed around 0.5 . This is the same distributions seen with random segments, Figure 9.

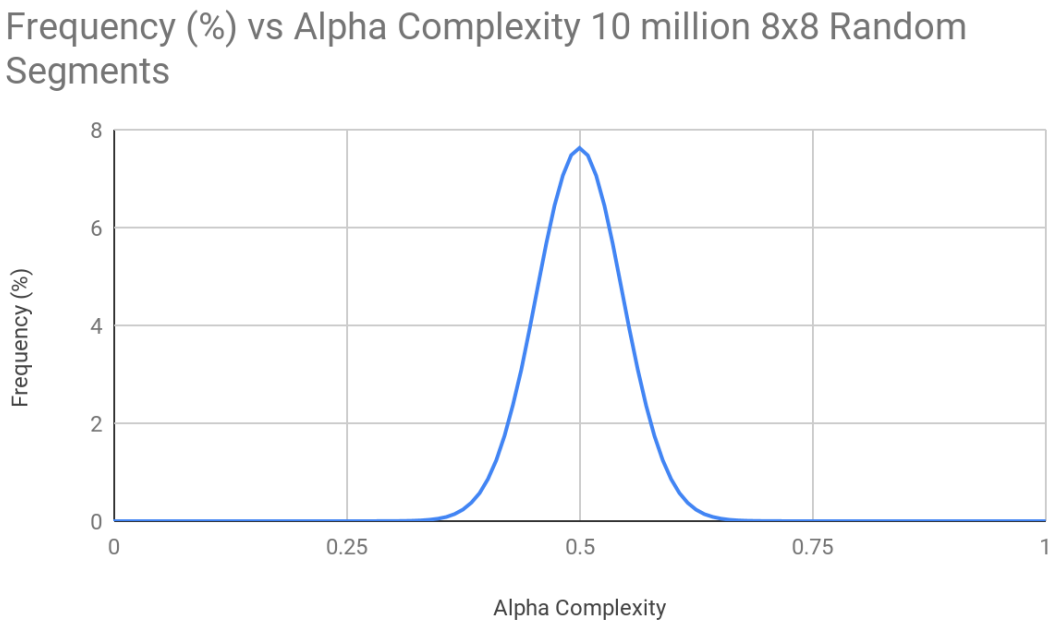

Figure 9: Complexity Histogram (8x8) of Randomly Generated Segments.

This pattern matches up with the observation in the bit plane visualization (Figure 2&3) that the lower bit planes are more random (noise-like).

Figure 10 show the histogram when the average is taken over all the bit-planes.

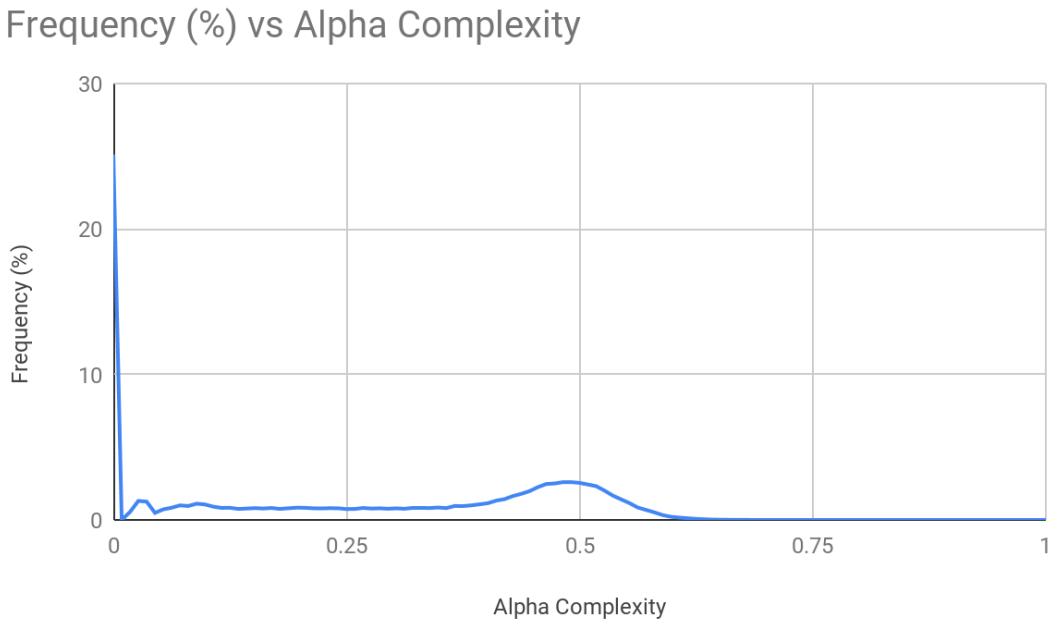

Figure 10: Complexity Histogram (8x8) over all Bit-Planes.

To embed the message, it much be broken into segment size pieces. Then segments of the cover image that meet a minimum complexity value (often 0.3) can be replaced with the payload segments.

2.4 Conjugation

With a way of identifying segments that can be replaced, the reverse (extraction) process must also be capable of identifying those same segments to get the payload out. A problem will occur if a payload segment is lower than the cut off chosen. When the program attempts to identify segments the payload segment will not be identified and not retrieved.

To solve this BPCS uses a process of conjugation on payload segments that don't meet the minimum complexity to guarantee it can be retrieved. A mapping of which segments have been conjugated must also be encoded into the image so it can be reversed, and the original payload retrieved.

Figure 11 shows the process performs bitwise XOR of a segment with a checkerboard pattern the result is a conjugate with a complexity equal 1 - complexity of the original. Applying the same process to the conjugate produces the original image.

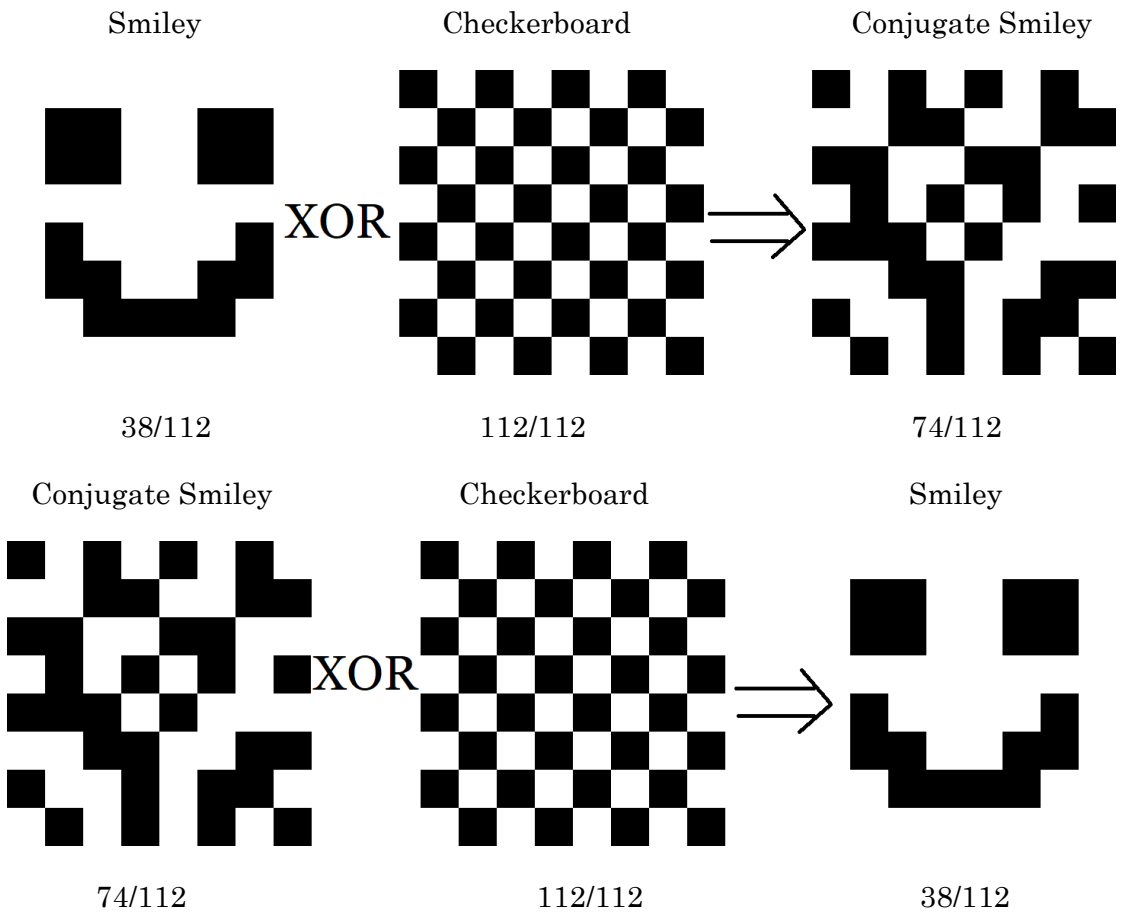

Figure 11: Conjugation example using simple smile image bit-map

Using this process if the complexity of a payload segment would be lower than the cutoff it can be conjugated first before being embedded. It is guaranteed that section will be recovered using the same cut off provided it is below 0.5 .

With the conjugation process implemented it is possible to simulate a random payload and view the effect on stego-images.

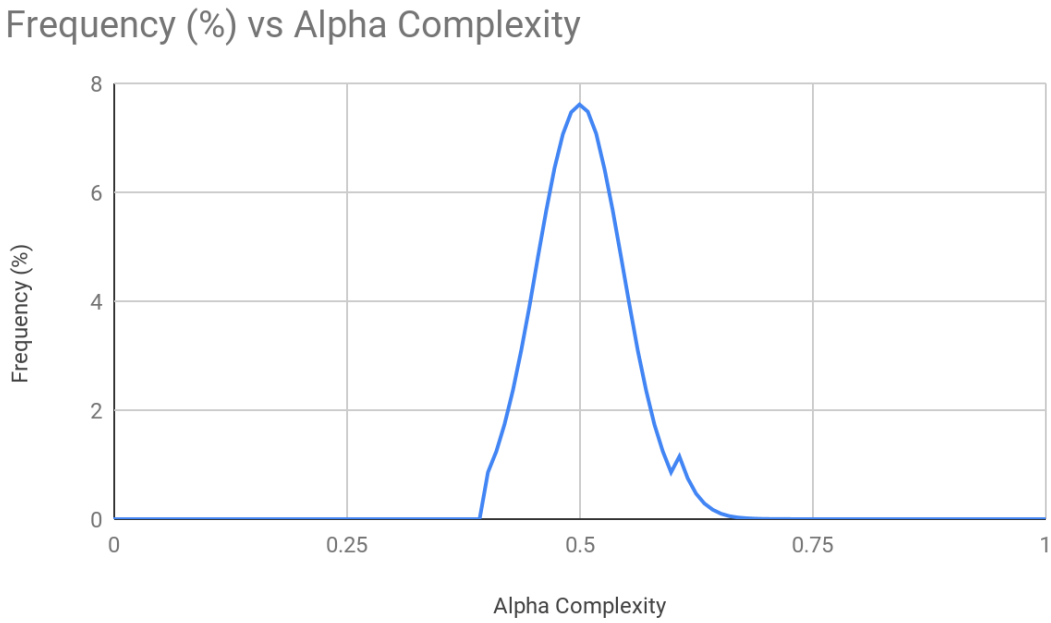

Figure 12: Effect of conjugation on complexity distribution of the payload

Figure 12, above, shows the effect of a 0.4 cut off and conjugation to 10 million 8x8 segments. Any segment below the cut off is conjugated and causing a spike at 1-Cutoff. The closer to 0.5 the cutoff is the sharper the peak. At 0.3 (common choice in literature) it is difficult to notice this spike.

2.5 Simulating Maximal Payload

Figure 13: Cutoff of 0.1 complexity over different segment sizes

The process of embedding a payload to full capacity can be simulated by replacing all the segments meeting the complexity threshold with random noise(Katz & Lindell, 2014). Figure 13 shows segments above 0.1 complexity replaced with random segments above the cutoff. This low complexity cut off was chosen to make the segment boundaries clear in the output image. The image used is 512 x 512 which is why segment boundaries can't be seen in the last image. Figure 14, shows that once the complexity cutoff is dropped below approximately 0.3 the effect on the output image is visually noticeable.

Figure 13: 8x8 Segment replacement with varying cutoffs

As the complexity cutoff is increased the number of segments replaceable drops and so the size of payload that can be embedded also drops. Because of the need for a conjugation map that must also be embedded in the image, not all the replaceable area can be used for the payload. I wanted to take this into account and compare a theoretical maximum payload between different segment sizes.

Figure 14 shows that this 0.3 cut off is also where it becomes difficult to see the effect of replacement for other segment sizes.

Figure 13: 0.3 cut off with varying segment sizes

This is an interesting result because when the segments are as large as image there is a similar capacity as the standard 8x8 segments without the noticeable artifacts / segment boundaries. This can be seen when compared side by side with the original, Figure 14. Particularly around the edges of the hat and shoulder in the 8x8 segment replacement image segment edges can be seen.

These results motivated the development of a way to visualize which parts of the bit planes were being replaced.

Figure 14: Comparison 0.3 complexity cut off

2.6 Segment Replacement Visualization

Because I had already created a way to create visualisations of bit-plane images I reused this but coloured the replaced segments with black. Figure 15 shows as expected from the complexity histogram, that majority of the replacements are at the lower significant bit-planes.

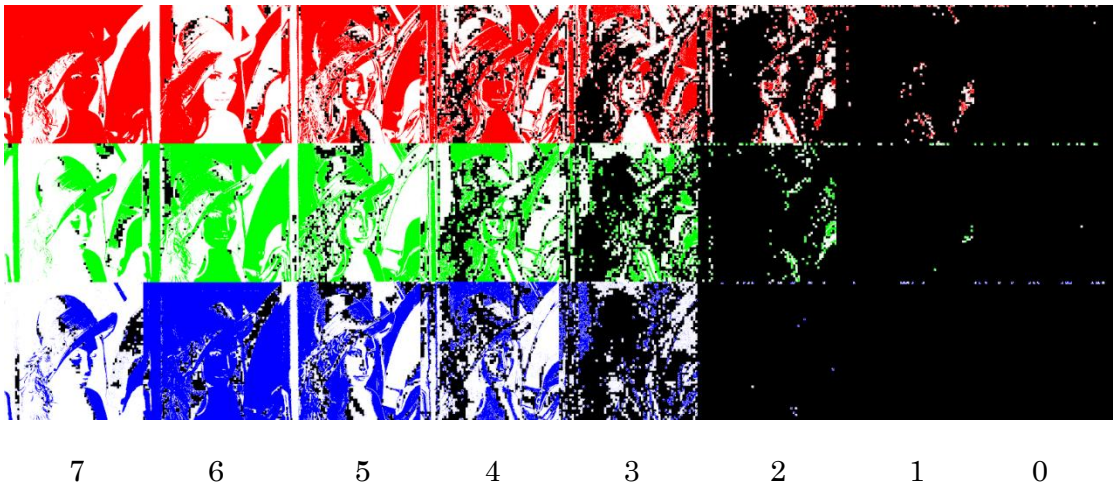

Figure 15: 8x8 Segments 0.3 cut off replacement bit-maps

Interestingly, when the segment size is increased to the size of the image BPCS-Steganography behaves similar to Least Significant Bit Steganography (Figure 16). According to the original paper, least significant bit is limited to 5-15% of the original image. That is true if only the very least bit is replaced, however if more of the bits are replaced much higher capacity can be embedded.

7 6 5 4 3 2 1 0

Figure 16: 512x512 segment size 0.3 cut off replacement bit-maps

Using BPCS to replace whole bit-planes has some benefits over simple Least Significant Bit replacement as it can expand the capacity depending on the image. It also does not leave artifacts seen in the smaller segment sizes of BPCS(Figure 17).

Figure 17: Identifying cause of artifacts using replacement image

The cause of the green artifact on the hat appears to be the replacement in the most significant bit plane of green. The original paper acknowledges that replacements in the higher planes are more noticeable and recommends using the lower plains before higher ones. More recent papers have suggested using more stringent cut offs at the higher planes and suggest that a weakness of BPCS is that edges in the image such as on the hat (Sun, 2015).

8x8 Segments Original Whole Bit Plane Segments

Figure 18: Comparison of Segment Size on another image (zoomed in section).

To confirm that these observations are not specific to the Lena image another comparison image was created (Figure 18) using the same parameters: 0.3 Complexity Cut Off used on Gray Code Bit-Planes. 8x8 segments replaced 32% with noticeable image artifacts, while the whole bit plane segments replaced 25% of the image (2 least significant bits).

2.7 Complexity Histogram

Visual detection is a way steganography can be broken, statistical approaches can also be used. One of these approaches may involve analyzing the complexity histogram.

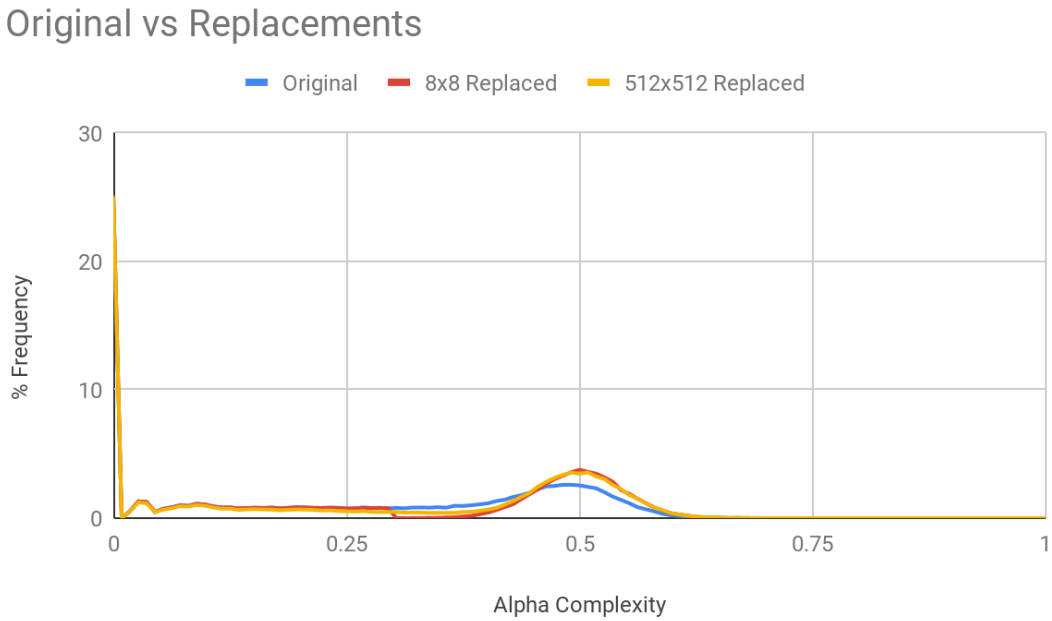

Figure 19: Lena image complexity before and after replacement above 0.3. A segment size is 8x8 was used for calculating the complexity plotted.

When the complexity distributions are plotted (Figure 19). The original image has a left skew around 0.5 segment complexity, however when 8x8 segment BPCS is applied anything above the cut off is forced into a non-skewed normal distribution around 0.5. There is a 'cliff' where the cut off happens.

When whole bit planes (512x512) are used the distribution is still less skewed however there is no cliff at the cutoff point.

2.8 Conjugation Maps

There are two ways to implement a conjugation map:

1. Internal: A reserved bit inside each segment indicating whether it has been conjugated.

2. External: Addresses of segments that are conjugated are stored after the payload.

The internal approach is very simple to implement, but it can be wasteful given that in a 8x8 segment it would take up 1/64th of the segment. The conjugation usually only needs to be done in a very small percentage of payload segments. This bit imbalance could be used to identify stego-images.

The external implementation can take up less space as it would require a way to uniquely address each segment which can be done in a space of:

$$\log_2 (\text{Number Of Segments Above Cut Off})$$

Considering the percentage of segments expected to require conjugated an estimate of payload size can be given assuming the conjugation map does not require conjugation itself, Figure 20.

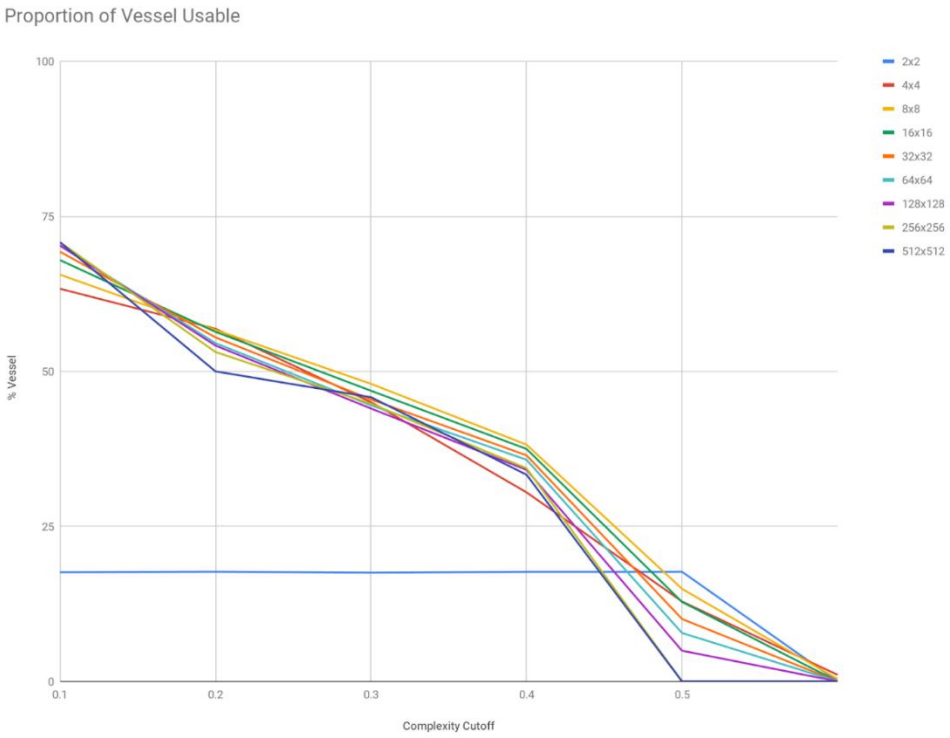

Figure 20: Lena image usable payload proportion over different thresholds and segment sizes.

An external conjugation map was not implemented in this project due to edge cases involving conjugated segments in the map itself. The first bit of each segment was reserved for internal conjugation mapping.

The original paper asserts that the using BPCS-Steganography replacement of approximately half the vessel image is possible without a noticeable effect (approx. 0.3 cut off). From this estimation the 0.3 cut off can be used for other segment sizes, 4x4 and over and have similar embedding capacity.

2.9 Embedding and Extraction of payload

To embed and extract a payload, the structure of segments and conjugation map was implemented.

In the very first segment, following the conjugation bit, is an address representing the end bit of the embedded file. The file data follows over segments until that end bit is reached. Because multiple files can be embedded into a single file e.g. a zip file format, there is no need to implement this or meta-data for the embedded files.

The implementation was confirmed by completing “round trips” (embedding a sample zip file and extracting it) ensuring it was unaltered over various segment sizes.

With a program capable of performing all the basic functions of BPCS Steganography over arbitrary segment sizes, improvements could be added.

Chapter 3 Adaption and Improvements

3.1 Encryption of Payload

The base BPCS algorithm can be used to hide the existence of a message, however there is no protection against extracting the payload to determine the message.

Java's Crypto Package was used to encrypt the payload with its implementation of symmetric AES block cipher using a SHA-256 hash of a user supplied key.

Compressed file formats such as ZIP tend to have a normal distribution, while uncompressed formats do not (Kawaguchi, 2015). There is no restriction on the type of file that can be embedded using this implementation, so having a block encryption step can ensure the payload does not contain any pattern that could be used to detect the steganography.

3.2 Peak Signal-to-Noise Ratio

To provide an objective quality estimate for the effect of BPCS-Steganography the Peak Signal-to-Noise ratio measure was used. The technique takes the original image as the signal and calculates the mean squared error of changed pixel values. As I used images in the RGB colour space the squared error (difference) from each channel was added to the total squared error. When calculating the mean the number for pixels were correspondingly multiplied by 3 to compensate for the extra comparisons.

This is then converted to a logarithmic decibel value, with a higher value meaning that the output image is closer to the original. From previous research on Peak Signal-to-Noise Ratio the values can be a reliable method for comparing the quality of the same image after alteration using related techniques (Huynh-Thu & Ghanbari, 2008).

3.3 Modified BPCS

An improvement to the BPCS algorithm was proposed which uses a different cut off complexity for each bit-plane (Shi & Li, 2010). The justification for this modification can be seen in Figure 17, where the replacements in more significant bit-planes replacement cause blocky artifacts in the output stego-image. The original paper (Kawaguchi & Eason, 1999) suggests replacing segments in the lower bit-planes first to counteract this effect. Using a higher threshold for higher bit-planes is a more robust approach.

Bit-Plane index	7	6	5	4	3	2	1	0
(Shi & Li, 2010) threshold	0.525	0.5	0.475	0.45	0.425	0.4	0	0
Altered threshold	-	-	0.475	0.45	0.425	0.4	0	0

Table 1: Threshold alpha complexity used for each bit-plane

Table 1 shows the thresholds used for the original (Shi & Li, 2010) and those use in my implementation of this algorithm. Complexity values at 0.5 and above for bit-planes 6 and 7 were omitted because if a payload segment requires conjugation its resulting complexity would equal 1-Complexity of the original. This would result in a segment that is less complex than the original and would not be recoverable in the extraction process.

Stego-images generated from this implementation do not make any change to the two most significant bit planes and utilizes fully the two least significant bit planes for embedding the payload.

3.4 Diagonal Complexity Definition

The definition of complexity in the standard BPCS algorithm is the number of transitions horizontally and vertically along the bit plane. No weight is given to diagonal transitions. An alternate complexity definition based on number of diagonal transitions was developed (Figure 21).

The maximum diagonal complexity of a segment is:

```
MaxComplexity = 2((width-1)(height-1))
```

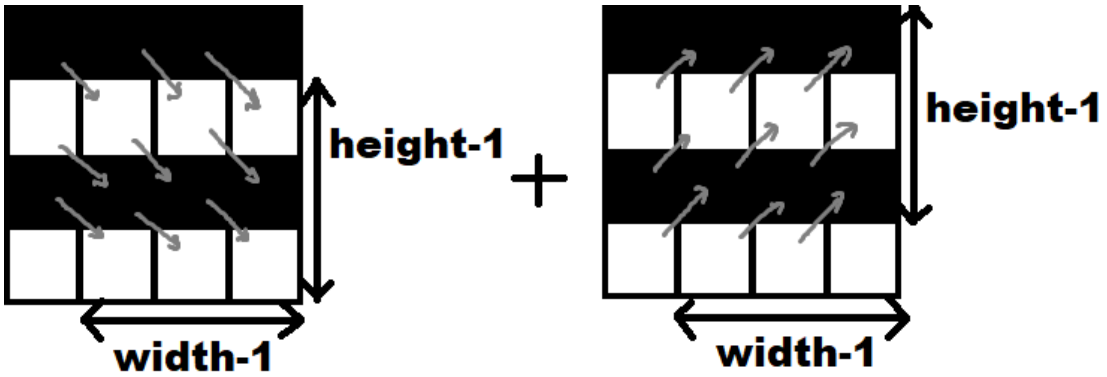

Figure 21: Diagonal Complexity Measurement

Conjugation with diagonal complexity behaves the same way alpha complexity (the original) and can be normalized to a number between 0 and 1 (Figure 22).

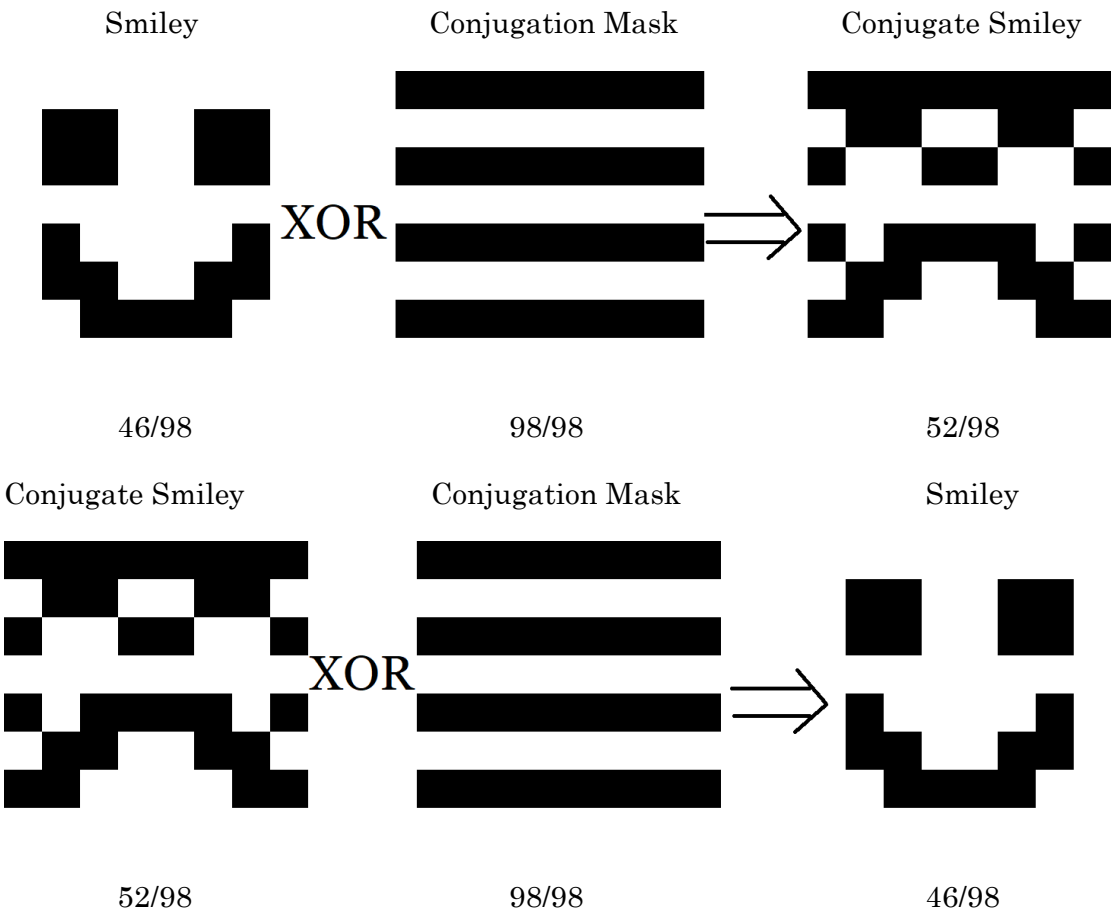

Figure 22: Conjugation example using simple smile image bit-map

3.5 Command Line Arguments

To provide a way to use the implemented steganographic algorithms without changing the source code, the program was modified to accept command line arguments. The compiled BPCS.jar file included in this project accepts some simple formatted arguments for .bmp and .png vessels. When producing a stego-image it will contain parameter info needed to extract and PSNR data.

Usage:

```
java -jar BPCS.jar <algorithim> mode=<mode> vessel=<vessel_path>
payload=<payload_path>
```

<algorithim> = “original” / “modified” / “diagonal”

<mode> = “embed” / “extract” / “roundtrip”

<payload_path> = payload to embed or extracted payload will be saved.

Optionals:

```
threshold=<double>  segmentwidth=<pixels>  segmentheight=<pixels>
key=<password>
```

Chapter 4 Software Design

The initial iteration of the program was to learn more about this type of steganography though implementing the base algorithm as described in the foundational paper (Kawaguchi & Eason, 1999). Much of this stage of the project was writing quick prototype code to test and verify many of the assumptions in that original paper and confirming that the arbitrary segment sizes could work.

The final iteration of this software has been designed to be modular. This is to assist implementing algorithms based on BPCS image steganography with arbitrary segment sizes.

The core BPCS algorithm was refactored with this goal towards the end of the project. In particular, the complexity definition can be replaced by implementing 3 simple methods:

1. A way to get the complexity of a given bitmap.
2. The maximum possible complexity for a given dimension.
3. The conjugation bitmap mask of a given dimension.

Consequently, this sped up the implementation of the alternate complexity definition using diagonal complexity. The resulting program can embed and extract a payload using this new definition while altering very little code. This modular approach was also used for the modified BPCS.

A UML class diagram for the final project is attached as Appendix A.

Because refactoring of the project continued until the end of the project the unit tests developed at earlier in the project became invalid. The final project does contain some functional tests that cover a large proportion of the code base, these involve embedding a sample payload into the lena image and extracting it again, ensuring the payload is not changed as well as testing the encryption key protects the payload.

Chapter 5 Conclusion and Future Work

The software developed during this project can embed and extracting payloads from images using Bit Plane Complexity Segmentation Steganography using arbitrary segment sizes. Where the user can select from three variants of the algorithm with option to encrypt the payload with a custom key was added to protect the contents of a payload if it is detected. When embedding payloads, the segment and the threshold/cutoff used can be customized. The software produced is for research purposes, rather than pubic use with settings and the resultant image quality included for convenience and analysis in the image file name.

Previous work have implemented the original(Kawaguchi & Eason, 1999) and modified(Shi & Li, 2010) BPCS algorithm with segment sizes of 8x8. This project has shown that is possible to use arbitrary segment sizes for these algorithms. These possess similar embedding capacities when compared to the the standard 8x8 segment size provided they were not too small (4x4 or larger), Figure 20.

When segment size is increased to the size of the whole image the lower bit planes are replaced similar to Least Significant Bit Steganography, however retains the analysis of BPCS to react to the vessel image.

In addition to varying segment sizes, an alternative complexity measure, diagonal complexity, is proposed for use in steganography. This project ran out of time to analyses whether this definition has any unique advantage to using the regular definition.

Changing of the segments size was seen to mask some of the histogram anomalies (Figure 19). Diagonal complexity also aids in diversifying the ways in which BPCS steganography can be performed makes it more expensive for an attacker faced with a steganography image to reliably detect it.

If this specific project continued the next step would be to verify and compare the different algorithms against each other in terms of PSNR values and capacity.

Future work in the area should focus on implementing more of the variations of BPCS such as I_BPCS (Shi & Li, 2010) to allow arbitrary segment sizes. This algorithm pseudo-randomly changes the threshold slightly for each segment from the bit-plane threshold used in Modified BPCS. The steganography key currently used to encrypt the payload could be used to seed a pseudo-random number generator for this purpose. This could also be used to reproducibly shuffle the segments meeting the threshold to distribute the payload across the whole image and to select which checkerboard pattern signifies conjugation is required.

Another interesting project could be to implement non-rectangular segment shapes. Bits closest to each other in a plane should theoretically be the best predictor of local complexity. Choosing a segment shape approximating a circle to minimize the distance between the bits could be viable. Perhaps a hexagonal shape to tightly pack segments together, honeycomb style.

References

Amin, M. M., Salleh, M., Ibrahim, S., Katmin, M. R., & Shamsuddin, M. (2003). *Information hiding using steganography.* Paper presented at the 4th National Conference of Telecommunication Technology, 2003. NCTT 2003 Proceedings.

Gray, F. (1953). United States Patent No. US2632058 United States Patent: U. S. Patent.

Gupta, S., Goyal, A., & Bhushan, B. (2012). Information hiding using least significant bit steganography and cryptography. *International Journal of Modern Education and Computer Science, 4*(6), 27.

Hamming, R. W. (1950). Error detecting and error correcting codes. *The Bell system technical journal, 29*(2), 147-160.

Huynh-Thu, Q., & Ghanbari, M. (2008). Scope of validity of PSNR in image/video quality assessment. *Electronics letters, 44*(13), 800-801.

Katz, J., & Lindell, Y. (2014). *Introduction to modern cryptography*: Chapman and Hall/CRC.

Kawaguchi, E. (2015). Complexity histograms of a non-compressed and a compressed file. Retrieved (01-09-2019) from http://datahide.org/BPCSe/comp-noncomp-hist-e.html

Kawaguchi, E., & Eason, R. O. (1999). *Principles and applications of BPCS steganography.* Paper presented at the Multimedia Systems and Applications.

Kawaguchi, E., & Eason, R. O. (2002). United States Patent No. 6473516: U. S. Patent.

Kawaguchi, E., Endo, T., & Matsunaga, J.-I. (1983). Depth-first picture expression viewed from digital picture processing. *IEEE transactions on pattern analysis and machine intelligence*(4), 373-384.

Lee, Y.-K., Bell, G., Huang, S.-Y., Wang, R.-Z., & Shyu, S.-J. (2009). *An advanced least-significant-bit embedding scheme for steganographic encoding.* Paper presented at the Pacific-Rim Symposium on Image and Video Technology.

Liu, J., & He, X. (2005). *A review study on digital watermarking.* Paper presented at the 2005 International Conference on Information and Communication Technologies.

Pfitzmann, B. (1996). *Information hiding terminology*, Berlin, Heidelberg.

Regulation of Investigatory Powers Act 2000 (RIPA) Section 51.

Roberts, L. (1962). *Picture coding using pseudo-random noise. IRE Transactions on Information Theory, 8(2), 145-154.*.

Shi, P., & Li, Z. (2010). *An improved BPCS steganography based on dynamic threshold.* Paper presented at the 2010 International Conference on Multimedia Information Networking and Security.

Sun, S. (2015). A new information hiding method based on improved BPCS steganography. *Advances in Multimedia, 2015*, 5.

Appendix A Design UML