BPCS Steganography

Mushfiq Mahmud

I. Introduction

1.1 Cryptography vs. Digital Steganography & Early Methods

Steganography is a method by which messages (in the form of pictures, audio, text, etc.) may be hidden or embedded into another media (picture, audio, text, etc.). It differs from general cryptographic techniques in the sense that one who is not privy to the secret would not be able to obviously figure out that something is hidden let alone the message in the first place. This paper will mostly emphasize on images as both the media to be embedded and also that will be embedded into.

There are many different ways to embed secret information inside another media, typically called the "vessel". The most popular is embedding a picture inside another picture since satisfactory results can be achieved whereby it would be impossible to distinguish the vessel before and after the secret image has been embedded. One preliminary method included replacing a specific band of the frequency spectrum of the bits in an image and then hiding the message in that band. Another method was to encode the least significant bits of the color values of an image. Although these methods have their own merits, they are limiting in the sense that they do not allow too much alterations to be done and therefore drastically reduces the size of the secret that can be sent. Early digital steganography methods could only hope to utilize about a maximum of 15% of the file size to store hidden information. For that reason, they were more suitable for less powerful applications such as digital watermarking rather than transferring secret messages with high security.

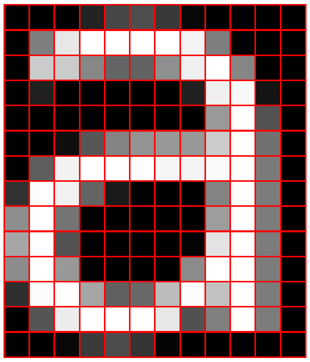
1.2 BPCS Steganography

This is where Bit Plane Complexity Segmentation (BPCS) Steganography comes in. BPCS Steganography proves to be more useful since it can use higher parts of the data inside a graphic to encode secret messages and therefore bigger messages can be embedded. It makes use of the inability of the human perception to distinguish between complex patterns when it is part of a much larger scale. Research shows that the human eye can generally only detect anomalies in a scene in areas of less complexity, i.e. areas where information and color are not very densely populated. BPCS Steganography takes advantage of this phenomenon and tries to embed information in a picture only where there exists complex bits of information or noise. It does so by substituting all the "noisy" areas in the bit-planes of the image to be embedded onto with secret data without significantly harming the quality of the image. This yields a much higher maximum storage size of about 50% of the vessel file size for storing embedded information

II. Concept

2.1 Bit planes

To first understand how BPCS Steganography works, we must consider how complex parts in any image are recognized. For this paper, we are only concerned with binary bitmap images (.bmp file extension). These are basically a two-dimensional array of byte (or bytes) that store information about the RGB color values of an image.



1.0	1.0	1.0	0.9	0.6	0.6	0.6	1.0	1.0	1.0	1.0	1.0
1.0	0.5	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0	1.0	1.0
1.0	0.2	0.2	0.5	0.6	0.6	0.5	0.0	0.0	0.5	1.0	1.0
1.0	0.9	1.0	1.0	1.0	1.0	1.0	0.9	0.0	0.0	0.9	1.0
1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.0	0.5	1.0
1.0	1.0	1.0	0.5	0.5	0.5	0.5	0.5	0.4	0.0	0.5	1.0
1.0	0.4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	1.0
0.9	0.0	0.0	0.6	1.0	1.0	1.0	1.0	0.5	0.0	0.5	1.0
0.5	0.0	0.6	1.0	1.0	1.0	1.0	1.0	0.5	0.0	0.5	1.0
0.5	0.0	0.7	1.0	1.0	1.0	1.0	1.0	0.0	0.0	0.5	1.0
0.6	0.0	0.6	1.0	1.0	1.0	1.0	0.5	0.0	0.0	0.5	1.0
0.9	0.1	0.0	0.6	0.7	0.7	0.5	0.0	0.5	0.0	0.5	1.0
1.0	0.7	0.1	0.0	0.0	0.0	0.1	0.9	0.8	0.0	0.5	1.0
1.0	1.0	1.0	0.8	0.8	0.9	1.0	1.0	1.0	1.0	1.0	1.0

Figure 1: 8×14 pixel size Bitmap Image with color information represented by numbers between 0.0 and 1.0 with black being 1.0. [Source: Pippin.gimp.org]

Figure 1 shows a bitmap image of the letter 'a'. The image is made up of pixels which, in this case, is wither black or white or a shade of grey. The pixels can each be described by a number as shown on the right. Here, the number is shown as a range from 0.0 to 1.0 with 1.0 being black. But this information is actually stored digitally in binary and in a general case, can store information for not only black or white images but also color images.

Each pixel can have 8 bits (or a byte) to store this information or up to 24 bits. We are going to concern ourselves with 8 bit images since they are simpler but the process for 24 bit and other images are the same. If we divide up each byte of data for each pixel grid element into its significant bits, we will have 8 bit planes each having the same significant bit for each pixel in a two-dimensional array. These bit planes are going to be crucial in BPCS Steganography since we are going to be dealing with one plane at a time.

2.2 Complexity

Since there are no specific definitions for complexity when it comes to BPCS Steganography, we are going to use a specific definition that the creators of BPCS Steganography made. It is called black-and-white border image complexity. We first take a bit plane and assign the colors black or white to each bit with black being 1 and white being 0.

We are going to determine the complexity by the length of borders between black and white bits. The image is going to be considered 'complex' if the border is long and simple if it is short. The way we are going to measure the border length is by looking at the amount of color changes as we go through each row and column in the plane. The color change is calculated by however many of the four bits that surround each bit (up, down, left, right) has the opposite value to it. As an example, a white bit surrounded by four black bits has a border length of 4. Figure 2 shows this visually. The final border length, k, is the sum of all these values.

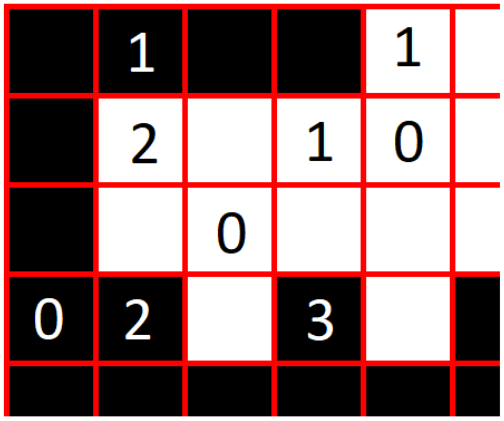


Figure 2: Calculation of border length for each bit. The numbers in the squares show the length of the border with respect to the number of color changes around its immediate vicinity (up, down, left, right).

The complexity α is defined as follows.

$$\alpha = \frac{k}{b}$$

Here, k is the final border length and b is the largest possible border length that the bit plane can have. For an $n \times n$ bit plane, $b = 4n(n-1)$. From here we can see, that the simplest bit plane that can be formed is a plane with all bits either 1's or 0's and the most complex is a checkerboard of 1's and 0's. Therefore, α can range from 0 to 1. That is

$$0 \leq \alpha \leq 1$$

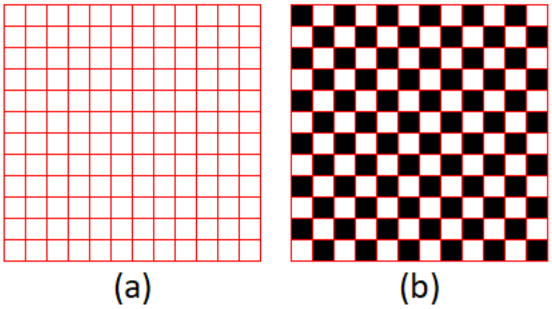


Figure 3: The simplest (a) bit plane with $k = 0, \alpha = 0$ and the most complex (b) bit plane with $k = 528,\ \alpha = 1$ for n=12

2.3 Conjugation

Conjugation is a process to increase complexity in cases of low complexity in order to be able to embed more information. We will see how this is applied but this section is reserved for the concept. Suppose we have a bit plane that looks like P in Figure 4. It has a certain amount of 1's and 0's. We want to find the conjugate $P^\star$ to P and the way the authors of BPCS Steganography have defined it to be is as follows.

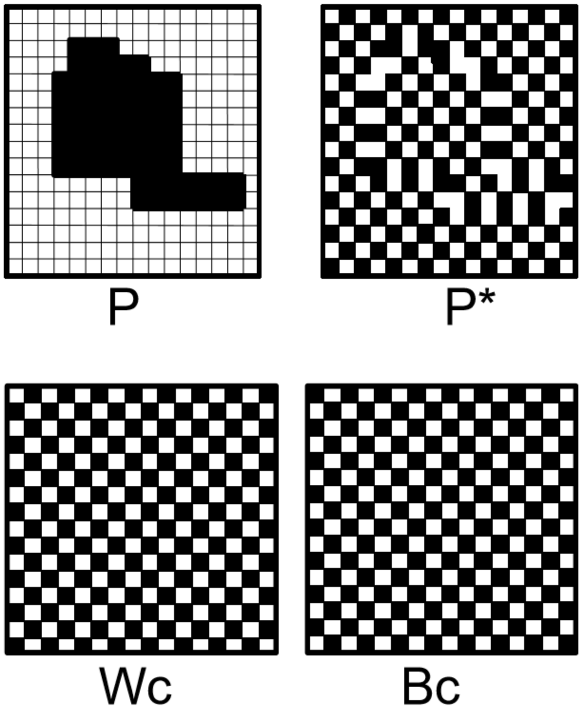


Figure 4: Conjugation of P to P. The checkerboard bit planes Wc and Bc are shown as well. [Source: Principle and applications of BPCS Steganography]*

We define two checkerboard patterns Bc and Wc as shown in Figure 4. Note that Wc and Bc are complements of each other. We then apply Wc to wherever there is a white bit in P and Bc to wherever there exists a black bit in P. This will yield $P^\star$ which will have a reciprocal complexity to P, i.e.

$P^{\star}$ can be conjugated back to P. Furthermore, it can be said that $P^{\star}$ is the Xor of P and Wc. This is written as:

$$(P^{\star})^{\star} = P$$

$$P^{\star} = P \oplus Wc$$

From all this, we can gather that

$$\alpha(P^{\star}) = 1 - \alpha(P)$$

This shows that the complexity value for $P^{\star}$ will also be complementary to that of P. This will prove to be very useful later.

2.4 Complexity Threshold

We need to be careful of the fact that the vessel image cannot be made to deteriorate in quality 'too much' otherwise it would be pointless to embed information since we would lose the secret aspect.

To do this, we must set limits on the value of the complexity. One such limit might be a lower limit on the complexity value that we cannot replace with embedded information. This is because the less complex valued bit planes (or, in other words, the simpler bit planes) would be more transparent as they are not complex or noisy enough to cover up the embedded information. That would be unwise. Let's call this value α_L.

Furthermore, we would also need another limit on the complexity value since we don't want to lose too much of the image. We will call this complexity limit α_U.

Taken together we need a complexity value α_0 that lies somewhere between these limit, i.e.

$$\alpha_L \leq \alpha_0 \leq \alpha_U$$

This value of α_0 will be called the threshold value for complexity.

III. Procedure & Experiment

3.1 Practical Procedure

The secret information to be embedded into the vessel image is prepared as follows:

- Break down the file into sub divisions having 8 bytes of data each.
- This results in having an 8×8 array of bits.
- These can be fed into the vessel image as noise figures to replace the original.

The embedding is done as follows:

- The vessel image is broken up into its bit planes. The bit planes can be separated into simple and complex (or noisy) areas using a complexity threshold α_0. A typical value is $\alpha_0 = 0.3$.
- The secret information file that has been broken up into planes can be fit into the vessel. This is done as a series of bytes.
- If a byte is not complex enough, determined according to the threshold value, the conjugate of the byte can be used to get a larger complexity. Otherwise, that byte can be used.
- Each of the secret byte information can be embedded into the noisy or complex areas of the bit planes of the vessel image, and the same can be done with each conjugated byte.

The inverse of this is done in order to decode or extract the secret.

3.2 Experimentation

The authors of the BPCS Steganography research have created their own software that implements it and therefore, I have presented their experiment and results here.

The vessel image selected is in Figure 6 (A). It was an 8x8 square image of size around 1,040,000 bytes. Embedded information that included an image (Figure 5), a few historical texts and a bunch of Shakespeare plays that all added up to 1,032,844 bytes but were compressed to make 441,318 bytes. Another image (Figure 7) was used to embed the same information (with one additional file embedded). This was done because the initial file had a lot of flat areas of similar color (such as in the walls). These areas were not suitable for embedding since they are not complex enough and would result in the result being noisy. So, the threshold was limited. But the later image could handle one more file since it didn't have as much flat space and was more complex. The results are shown in Table 1.

Figure 5 One of the secret embedded data that is hidden in Figure 6 (B) and 7 (B)

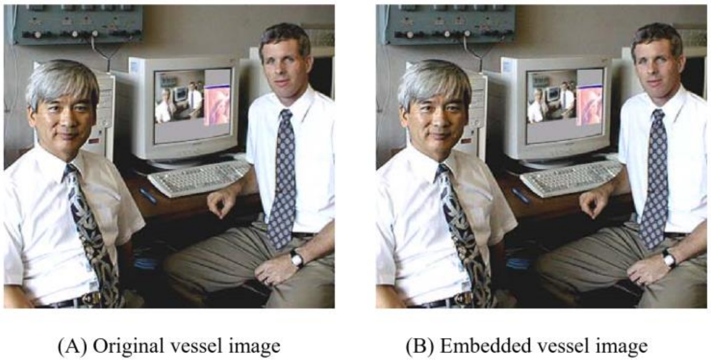
IV. Results

4.1 Results

Experiment	(1) Fig. 6	(2) Fig. 7
Vessel size (bytes)	1,040,000	936,114
Uncompressed Embedded Data Size (bytes)	1,032,844	1,212,744
Compressed Embedded Data Size (bytes)	441,318	505,502
Percentage of Original Vessel image (%)	42.40	54.00

Table 1 Experimental Results

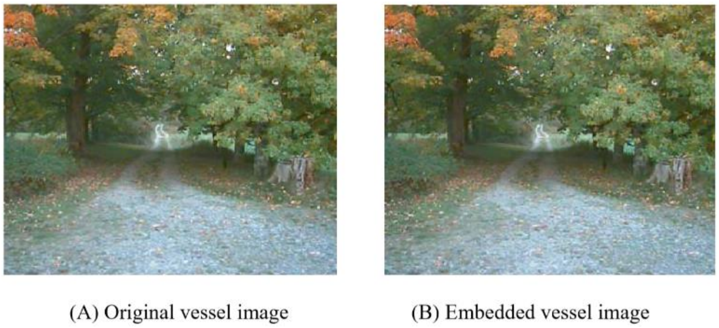
This shows an embedding percentage of around 50%! This is more than regular digital steganography could achieve and the result images look very similar to the original vessel images on a monitor screen.



(A) Original vessel image

(B) Embedded vessel image

Figure 6 Vessel image before and after embedding



(A) Original vessel image

(B) Embedded vessel image

Figure 7 Second vessel image with less flat areas for higher embedding

V. Conclusion

BPCS Steganography has shown to provide a lot of improvement over regular digital steganography. It has taken regular steganography from a point where it could hardly be used for digital watermarking purposes for files to a moderately complex and reliable secret keeping mechanism that works perfectly with the digital age of sharing files over the internet.

There have been other researches done on further advancing BPCS Steganography that were not explored in this paper. They include using wavelet transforms such as DWT (Discrete Wavelet Transform) or IWT (Integer Wavelet Transform) to change the values of the bit planes, thereby making it possible to have more information stored into any given image. There are obviously a lot of uses to digital steganography and since it is relatively a new form of encryption method, it is going to make great strides in advancing the security of the connected world.

VI. Reference

[1] E. Kawaguchi and R. O. Eason, "Principle and applications of BPCS-Steganography", *Proceedings of SPIE*, 1999.

[2] E. Kawaguchi, M. Niimi, H. Noda, R. O. Eason and K. Nozaki, "A Concept of Digital Picture Envelope for Internet Communication", *European Journal of Combinatorics*, 1998.

[3] J. Spaulding, H. Noda, M. Shirazi, M. Niimi and E. Kawaguchi, "BPCS Steganography Using EZW Encoded Image", *Digital Image Computing Techniques and Applications*, 2002.

[4] M. Ramani, D. Prasad and D. Varadarajan, "Steganography Using BPCS to the Integer Wavelet Transformed Image", *International Journal of Computer Science and Network Security*, vol. 7, no. 7, pp. 293-302, 2007.

[5] "Image Processing with gluas Introduction to pixel molding", *Pippin.gimp.org*, 2017. [Online]. Available: http://pippin.gimp.org/image_processing/images/sample_grid_a_square.png. [Accessed: 02- May- 2017].