© 2015 IJSRSET | Volume 1 | Issue 3 | Print ISSN : 2395-1990 | Online ISSN : 2394-4099
Themed Section: Engineering and Technology

Integer Wavelet Bit-Plane Complexity Segmentation Image Steganography

Srinivasa[*1], Choodarathnakara A L[2], Venugopal C K[3], Chethan K S[4], Varun S Gangoor[5]
[*1,2,3]Dept. of Electronics & Communication Engineering, GEC, Kushalnagar, Kodagu, Karnataka, INDIA
[4]Dept. of Electronics & Communication Engineering, PESIT, Bangalore, Karnataka, INDIA
[5]Department of Electronics & Communication Engineering, MCE, Hassan, Karnataka, INDIA

ABSTRACT

Steganography is a technique to hide secret information in some other data (called vessel) without leaving any apparent evidence of data alteration. All of the traditional steganographic techniques have limited information-hiding capacity. They can hide only 10% (or less) of the data amounts of the vessel. This is because the principle of those techniques is either to replace a special part of the frequency components of the vessel image or to replace all the least significant bits of a multi valued image with the secret information. In this paper, an image steganography system is proposed, in which the data hiding (embedding) is realized in bit planes of sub band wavelets coefficients obtained by using the Integer Wavelet Transform (IWT). To increase data hiding capacity while keeping the imperceptibility of the hidden data, the replaceable IWT coefficient areas are defined by a complexity measure used in the Bit-Plane Complexity Segmentation Steganography (BPCS). This technique to hide secret information is based on the property of human vision system and not on a programming technique. Its information hiding capacity can be as large as 50% of the original image data. The proposed system can recover the hidden message in a lossless manner if the communication channel is ideal. The extracted message contains some erroneous bits since the communications channel is not ideal. document provides some minimal guidelines (and requirements) for writing a research paper. Issues related to the contents, originality, contributions, organization, bibliographic information, and writing style are briefly covered. Evaluation criteria and due dates for the research paper are also provided.

Keywords: BPCS, Bit rate, BPP, Data rate, IWT, IIWT

I. INTRODUCTION

Steganography is the art and science of writing hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message [10]. It is a form of security through obscurity. The word steganography is of Greek origin and means "concealed writing". It includes the concealment of information within a carrier. The carrier could be any medium used to convey information, including wood or slate tablets, tiny photographs or word arrangements. In digital steganography, electronic communications may include steganographic coding inside of a transport layer, such as a document file, image file, program or protocol. This paper describes a method of steganography for hiding large volumes of data using digital images as carriers.

In steganography, data is hidden inside a vessel or container that looks like it contains only something else. A variety of vessels are possible, such as digital images, sound clips, and even executable files. In recent years, several steganographic programs have been posted on internet home pages. Most of them use image data for the container of the secret information [4]. Some of them use the least significant bits of the image data to hide the data. Some other programs make use of the sampling error in image digitization. However, all those steganographic techniques are limited in terms of information hiding capacity. They can embed only 5-15% of the vessel image at the best. Therefore, current steganography is more oriented to water marking of computer data than to secret person-person communication applications.

In this paper, a new technique to hide secret information in a colour image is implemented. This is not based on a programming technique, but is based on the property of human vision system. Its information hiding capacity can be as large as 50% of the original image data. This could open new applications for steganography leading to a more secure internet communication. Digital images are categorized as either binary (black-and-white) or multi-valued pictures despite their actual colour. We can decompose an n-bit image into a set of n binary images by bit-slicing operations. Therefore, binary image analysis is essential to all digital image processing. Bit slicing is not necessarily the best in the Pure-Binary Coding system (PBC), but in some cases the Canonical Gray Coding system (CGC) is much better.

II. BPCS – STEGANOGRAPHY

The Human Visual System is not sensitive to modifications of noisy data in an image. The BPCS Image Steganography technique utilizes this fact and embeds the secret data in an image in the form of noisy data. For the extraction of the hidden data, all the noisy components from the image are extracted. The noisy data thus obtained is converted back to the original file format.

Resource File = Secret message
Container Image = Cover Image

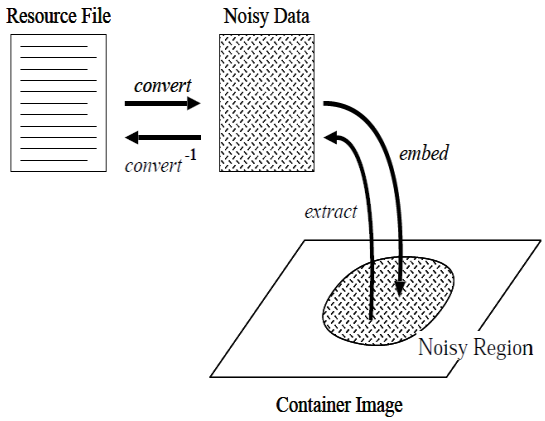


Figure 1: BPCS Steganography

A. Complexity

The method of steganography outlined in this paper make use of more complex regions of an image to embed data [6]. There is no standard definition of image complexity. We adopted a black and-white border image complexity.

The length of the black-and-white border in a binary image is a good measure for image complexity. If the border is long, the image is complex, otherwise it is simple[11]. The total length of the black-and-white border equals to the summation of the number of color-changes along the rows and columns in an image. For example, a single black pixel surrounded by white background pixels has the boarder length of 4. We will define the image complexity *'a'* by the following:

$$a = k \qquad 1$$

The above equation gives the maximum possible Black-White changes in the image, where, 'k' is the total length of black-and-white border in the image. So, the value ranges over $0<a<1$. This is defined globally, i.e., 'a' is calculated over the whole image area. It gives us the global complexity of a binary image. However, we can also use for local image complexity (e.g. an 8*8 pixel-size area). We will use such 'a' as our local complexity measure.

Informative images are simple, while noise-like images are complex. However, this is only true in cases where such binary images are part of a natural image. In this section we will discuss how many image patterns are informative and how many patterns are noise-like. We will begin by introducing a "conjugation" operation of a binary image.

B. Conjugation of a Binary Image

Let P be a 2N * 2N size black-and-white image with black as the foreground area and white as the background area. W and B denote all-white and all-black patterns, respectively. We introduce two checkerboard patterns Wc and Bc, where Wc has a white pixel at the upper-left position, and Bc is its complement, i.e., the upper-left pixel is black. We regard black and white pixels as having a logical value of "1" and "0", respectively. P, W, B, Wc, Bc and P* are interpreted as follows. Pixels in the foreground area have the B pattern, while pixels in the background area have the W pattern. Now we define P* as the conjugate of P which satisfies:

- The foreground area shape is the same as P.
- The foreground area has the Bc pattern.

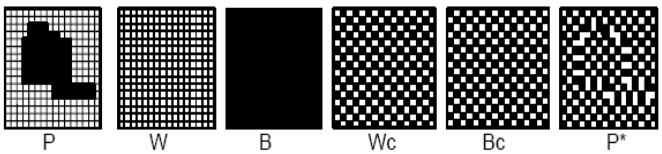


Figure 2: Representation of White and Black Patterns

The background area has the Wc pattern. Correspondence between P and P* is one-to-one, onto. The following properties hold true and are easily proved for such conjugation operation.

$$P^* = P \oplus W_c$$
$$(P^*)^* = P$$
$$P^* \neq P \qquad 2$$

The most important property about conjugation is the following. Let α (P) be the complexity of a given image P, then we have.

$$\alpha(P^*) = 1 - \alpha(P) \qquad 3$$

It is evident that the combination of each local conjugation (e.g., 8x8 area) makes an overall conjugation (e.g., 512 x 512 area) says that every binary image pattern P has its counterpart P*. The complexity value of P* is always symmetrical against P regarding a = 0.5. For example, if P has a complexity of 0.7, then P* has a complexity of 0.3.

C. Bit-Plane Complexity segmentation

RGB (Red Green Blue) format is the simplest form of an image in the computer. These RGB components also known as channels collectively form one pixel and an image is the collection of these pixels. For a true 24-bit bitmap image, each of the RGB components takes one byte of memory. Each RGB component value ranges from zero (0) to 255 where zero represents darkest shade of the color and 255 represent brightest shade of this color [8]. All other colors can be generated with the combinations of these ranges. A 4 by 4 sample image is given below. Each pixel is a combination of red, green and blue values. Their integer values are given in Table1.

Figure 3: Test image (4 x 4)

TABLE 1 RGB VALUES

Column	R	G	B
0	175	247	09
	167	225	30
	164	217	38
	155	197	29
1	09	223	247
	30	206	225
	38	199	217
	33	169	184
2	80	09	247
	80	30	255
	91	38	217
	85	35	201
3	202	69	252
	190	32	251
	165	04	225
	134	03	184

This image has total 16 pixels and each pixel has three components having one-byte for each component. Therefore the total size of the image is 48 bytes. Alternatively the value of each RGB component can be represented in binary format too. The table 2 shows the binary equivalent values for the above 4x4 RGB View.

TABLE 2 BINARY REPRESENTATION OF RGB VALUES

Pixel	R	G	B
0	10101111	11110111	00001001
	10100111	11100001	00011110
	10100100	11011001	00100110
	10011011	11000101	00011101
1	00000001	11011111	11110111
	00011110	11001110	11100001
	00100110	11000111	11011001
	00100001	10101001	10111000
2	01010000	00001001	11110111
	01010000	00011110	11100001
	01011011	01011011	11011001
	01010101	00100011	11001001
3	11001010	01000101	11111100
	10111110	00100000	11111011
	10100101	00000100	11100001
	10000110	00000011	10111000

Following steps are followed for the constructions of the binary planes.

Step 1: Formation of Channel Matrix

First of all, the selected channel (R in this case) is picked from all the pixels of the image. The channel matrix contains the N elements where N is the total number of pixels in the image.

TABLE 3 'R' CHANNEL MATRIX

R0	R1	R2	R3
10101111	00000001	01010000	11001010
10100111	00011110	01010000	10111110
10100100	00100110	01011011	10100101
10011011	00100001	01010101	10000110

Step 2: Get Corresponding Bits

In next step, we get the corresponding i^{th} bits from each of the channel to construct a plane. These bits are picked out using the same sequence in which the channel itself is allocated in the image. The height and width of a binary plane (as there are only) 1's and 0's in plane) is the same as the height and width of the original image (4x4). Using this fact, we can easily calculate the total number of bits in any of the plane.

Step 3: Formation of 'N' Binary Planes

Applying the same procedure described in the step 2, the following N planes are constructed. N is the number of bits per RGB component.

TABLE 4: PLANES EXTRACTED FROM 'R' CHANNEL

Plane 1				Plane 2			
1	0	0	1	0	0	1	1
1	0	0	1	0	0	1	0
1	0	0	1	0	0	0	0
1	0	0	1	0	0	1	0
Plane 3				**Plane 4**			
1	0	0	0	0	0	1	0
1	0	0	1	0	1	1	1
1	1	1	1	0	0	0	0
0	0	1	0	1	0	1	0
Plane 5				**Plane 6**			
1	1	0	1	1	0	0	0
0	1	0	1	1	1	0	1
0	0	1	0	1	1	0	1
1	0	0	0	0	0	1	1
Plane 7				**Plane 8**			
1	0	0	1	1	1	0	0
1	1	0	1	1	0	0	0
0	1	1	0	0	0	1	1
1	0	0	1	1	1	1	0

We can represent these planes visually as under, where one black block represents a 1 and white block represents a 0.

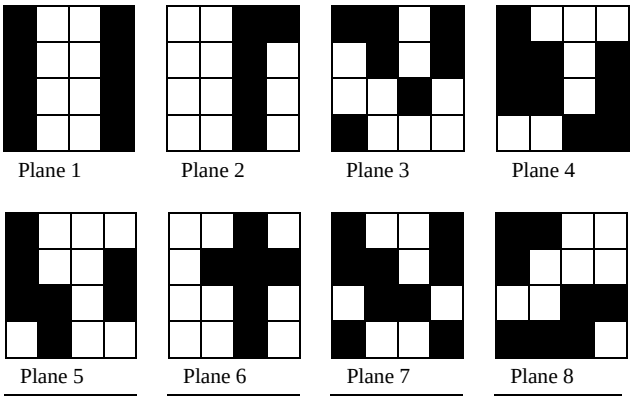


Figure 4: Visual Representation of Planes

D. Complexity Of Binary Planes

All the binary planes share the information of a particular channel (R in this example). It is the property of the natural images that this sharing of the information among the planes is not unique. Some of the planes share more information of the channel and some don't. The planes having more information are called simple or informative planes and planes having less information are called complex or noisy planes. The complex planes have more ratio of noise as compared to the simple planes. There are a few popular methods to measure the complexity of the binary planes [9]. These are discussed below.

E. Block Complexity Measures

The new embedding method ABCDE is based on the principle introduced in BPCS. A resource file is converted into noisy data, and is replaced with the pixel data in noisy regions of a container image. We have to locate noisy regions appropriately to embed the resource file secretly. If not, informative regions of the container image would be disordered by the embedded resource file and noticeable changes would be left after embedding. In BPCS, the noisy region of an image is located on each bit-plane as small pixel blocks those have noisy patterns. Each bit-plane of a container image is regularly divided into small square binary pixel blocks as illustrated in Figure 5. A binary pixel block can be regarded as one in a noisy region if it has a complex black-and-white pattern. Only such complex blocks are used for embedding.

On embedding, the blocks on the lowest bit-plane (the LSB plane) is used first. The blocks in a container image are examined one by one from those on the LSB plane through up to those on the highest bit-plane (the MSB plane). A resource file is embedded piece by piece as a complex block is found on a bit-plane. This way of embedding is preferable, because changes in lower bit-planes would not spoil the quality of a container image greatly.

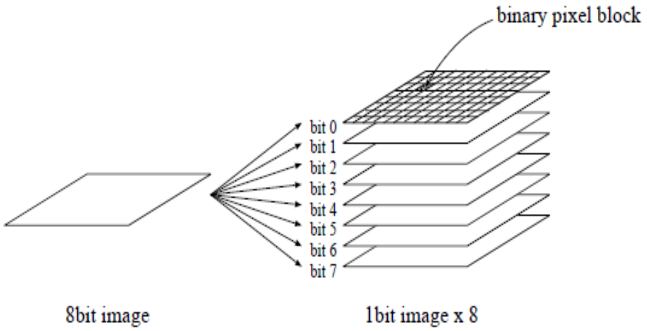


Figure 5: Binary Pixels on Bit-Planes

F. Black and White border Complexity Measure

We find that a block is simple when it is entirely oral most entirely in black or white. Figure 6 shows an 8x8 block. It looks simple. There are (8-1) x 8 x 2 = 112 pixel borders inside, but only eight out of them lie between black and white pixels. The others lie between the same colour pixels, i.e., either between two white pixels or between two black pixels. A block can thus be regarded as simple when it has a few black-and-white borders in it. This observation leads us to the definition of a complexity measure of a block based on the total length of black-and-white borders in it, which is employed in BPCS [6, 8]. Suppose that k out of M pixel borders lie between black and white pixels in a block, the complexity measure is then given by,

$$\alpha = \frac{k}{M} \qquad 4$$

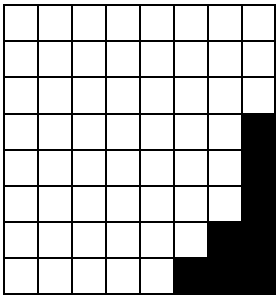


Figure 6: A Simple Block

For example, $\alpha = 8/112$ for the block in Figure 6. If *'α'* of a block is large, the block has many black-and-white borders inside, and we can regard it as complex. On the other hand, if *'α'* of a block is small, the block must be simple. The range of this measure is [0, 1]. A threshold value α_0 is introduced to discriminate complex blocks from simple ones. A block B is regarded as complex if $\alpha(B) \geq \alpha_0$. Figure 7(a) and 7(b) show a simple block and a complex block in respect of α. The values of *'α'* are 20/112 and 72/112 respectively.

The black-and-white border complexity measure **α** is easy to understand and usually works well for classifying blocks into complex ones and simple ones. This measure is, however, not always applicable. Even when a block contains many black-and-white pixel borders, it does not necessarily mean that the block is complex.

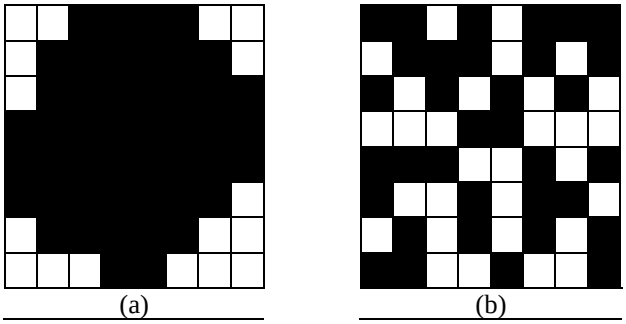


Figure 7: A simple block and a complex block in respect of α

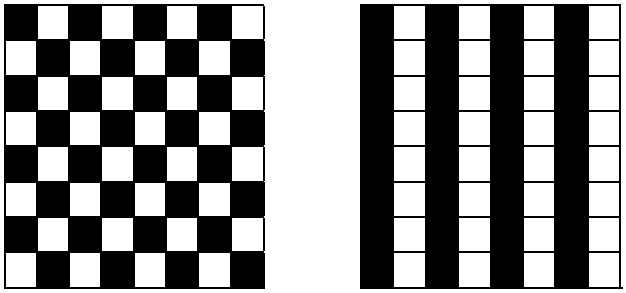


Figure 8: Blocks those are not complex

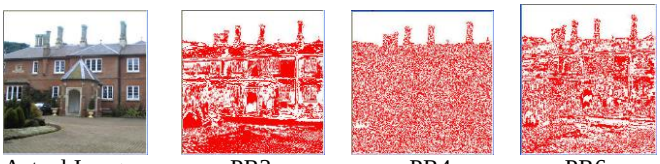


Figure 9: Complex representation of an image

Since bit-planes of a container image are divided regularly into blocks regardless of their contents, some of the blocks may be on the boundary of a noisy region and an informative region. An example of such a block is shown in Figure 9. The black-and-white border complexity measure **α** may indicate such a block is complex. Embedding a resource file in such blocks in a container image would, however, cause the noisy region to grow. As a result, remarkable changes would be left on the container image. We thus notice that the black-and-white border complexity measure cannot always be applicable for finding complex blocks. Two new complexity measures are proposed in this paper to properly discriminate complex blocks from simple ones. They are called the run length irregularity and the border noisiness. If a block has large run-length irregularity and large border noisiness at the same time, we can regard it as complex one. These complexity measures are presented in the following sections.

G. Length of Black And White Border

This method uses the four-connectivity neighbourhood phenomena. The total length of the black and white border is the total number of color changes for each of the bit values. A black pixel has the total length 4 if it is surrounded with the 4 white pixels. In this case, it is assumed that image is always square where width is equal to the height. The images used for experimentation must have the size of 2N, where N ranges from 2 to 12. The following formula is used for the calculation of the complexity.

$$\mathbf{C = k/2 * 2N * (2N - 1)} \quad (5)$$

where, k is the total sum of color changes within the plane, N is the height or width of the image and C is the complexity of the plane ranging from 0 to 1.

According to the above measurement the value for complexity is 0 for planes having no colour change, i.e., whole black or white plane. The maximum complexity measure is 1 in case of check box planes, as it has the maximum colour changes.

TABLE 5 BIT COMPLEXITY VALUES OF PLANE 3

1,3	1	2
2,0	0	--
2,1	0	--
2,2	1	4
2,3	0	--
3,0	1	3
3,1	0	--
3,2	0	--
3,3	0	--
Sum of Color Change(s)		9

For example, the complexity of the plane 3 is calculated below, we have used the bit value '1' as a base for calculation of complexity. We use the equation 4 to calculate the complexity of the selected plane 3.

C = 15/2 * 24 * (24 – 1)
C = 15/2 * 16 * (16 – 1)
C = 15/2 * 16 * 15
C = 15/480
C = 0.03125

So the complexity of the plane 3 is 0.031. In the same way we can also calculate the complexity of the remaining planes too.

H. Number of Connected Areas

The number of connected areas can also be used as the basis for the calculation of the complexity of a plane. Consider the following plane which has 4 connected areas - 2 for black and 2 for white.

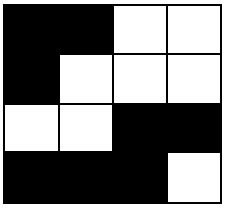


Figure 10: Plane 8

If we denote the connected areas as 'Ca' then the following formula is used to calculate the complexity of the plane using the "connected areas" method where,

N = Height or Width of the image
Ca = Number of the connected areas
C = Complexity of the plane.
C = Ca/2N * 2N
C = 4/24 * 24
C = 4/256
C = 0.015625

The value of C ranges from

$$\frac{1}{(2N * 2N)} \leq C \leq 1 \quad (6)$$

According to this formula again the simplest plane is fully black or white and the complex one will be checker board style. It is interesting to see that for the same plane the value of complexity measure can be different after using the different method for its calculations.

I. Threshold Mechanism

Although using the formula 5, the complexity value of a plane obtained still needs to be decided whether the plane is informative or noisy. Generally the complexity value approaching to '0' means an informative plane and '1' means a noisy one. For this purpose, a threshold value is used to choose the noisy planes. This is a subjective value and selected by the user.

We can define a threshold value from 0 to 1. The planes having complexity value less are considered informative and planes having more complexity than threshold are considered noisy. It depends on the implementation of technique that what to do when complexity is equal to

the threshold. Normally such planes are also considered as noisy planes. Now planes can be replaced with message to be embedded in the image

III. INTEGER WAVELET TRANSFORM

A one dimensional discrete wavelet transform is a repeated filter bank algorithm. The input is convolved with a high pass filter and a low pass filter. The result of the latter convolution is a smoothed version of the input, while the high frequency part is captured by the first convolution. The reconstruction involves a convolution with the syntheses filters and the results of these convolutions are added. In two dimensions, we first apply one step of the one dimensional transform to all rows. Then, we repeat the same for all columns. In the next step, we proceed with the coefficients that result from a convolution in both directions. As shown in figure 11, these steps result in four classes of coefficients: the (HH) coefficients represent diagonal features of the image, whereas (HL & LH) reflect vertical and horizontal information. At the coarsest level, we also keep low pass coefficients (LL). We can do the same decomposition on the LL quadrant up to log 2 (min (height, width)).

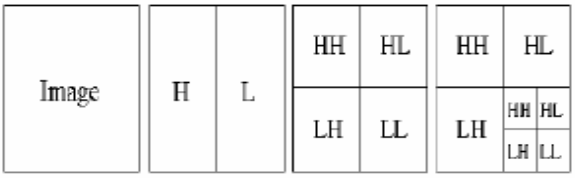


Figure 11: Two dimensional wavelet Transform

When the input data consist of sequences of integers (as in the case for images), the resulting filtered outputs no longer consist of integers, which doesn't allow perfect reconstruction of the original image [11]. However, with the introduction of Wavelet transforms that map integers to integers we are able to characterize the output completely with integers.

One example of wavelet transforms that map integers to integers is the *S-transform*. Its smooth (*s*) and detail (*d*) outputs for an index *n* are given in (7) & (8) respectively. At the first sight it seems that the rounding-off in this definition of *s*(*n*) discards some information. However, the sum and the difference of two integers are either both odd or both even. We can thus safely omit the last bit of the sum, since it equals to the last bit of the difference. The *S-transform* is thus reversible and its inverse is given in equations (9) & (10).

$$s(n) = [\, x(2n) + x(2n+1) \,] / 2 \quad (7)$$

$$d(n) = x(2n) - x(2n+1) \quad (8)$$

$$x(2n) = s(n) + [(d(n) + 1) / 2] \quad (9)$$

$$x(2n+1) = s(n) - [(d(n) / 2] \quad (10)$$

However, we need to redefine those equations in 2D in order to be applied on images and hence be useful in our implementation. In this section, we will define the construction of the 2D *S-transform*. Suppose that the original image (*I*) is *Y* pixels wide and *X* pixels high. Denote the colour shade level of pixels located at position *i* and *j* by $I_{i,j.}$ Generally, the 2D *S-transform* can be computed for an image using equations (11), (12), (13) and (14).

Of course the transform is reversible, i.e., we can exactly recover the original image pixels from the computed transform coefficients. The inverse is given in equations (15), (16), (17) & (18). Note that the transform results in four classes of coefficients: (*A*) the low pass coefficients, the (H) coefficients represent *horizontal* features of the image, whereas (*V*) & (*D*) reflect *vertical* and *diagonal* information respectively. During the transform we ignore any odd pixels on the borders.

$$A_{i,j} = (I_{2i,2j} + I_{2i+1,2j}) / 2 \quad (11)$$

$$H_{i,j} = I_{2i,2j+1} - I_{2i,2j} \quad (12)$$

$$V_{i,j} = I_{2i+1,2j} - I_{2i,2j} \quad (13)$$

$$D_{i,j} = I_{2i+1,2j+1} - I_{2i,2j} \quad (14)$$

$$I_{2i,2j} = A_{i,j} - (H_{i,j} / 2) \quad (15)$$

$$I_{2i,2j+1} = A_{i,j} + [(H_{i,j} + 1) / 2] \quad (16)$$

$$I_{2i+1,2j} = I_{2i,2j+1} + V_{i,j} - H_{i,j} \quad (17)$$

$$I_{2i+1,2j+1} = I_{2i+1,2j} + D_{i,j} - V_{i,j} \quad (18)$$

Where, $1 \le i \le X/2,\ 1 \le j \le Y/2$

The availability of the integer to integer wavelet transform makes it possible to implement lossless coding in the transformed domain. Furthermore, the successive approximation property of the wavelets also finds its application in progressive transmission of images through the internet together with lossless coding [13, 14]. Integer to integer wavelet transform proposed in this is implemented using a lifting scheme, which is one of the approaches used for floating point wavelet transformation for faster & straightforward computation. It is demonstrated that with this lifting scheme, the integer to integer wavelet transform is nothing but a rounding process during the lifting & dual lifting stages. The process is described schematically in the Figure 12.

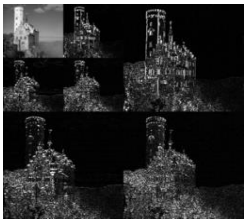


Figure 12: Integer Wavelet Transformed Image

In high bit-rate data hiding we have two primary objectives: the technique should provide the maximum possible pay load and the embedded data must be imperceptible to the observer. We stress on the fact that steganography is not meant to be robust [12, 16]. Any modifications to the file, such as conversions between file types and / or standard image processing, is expected to remove the hidden bits from the file. Fundamentally, data payload of a steganographic scheme can be defined as the amount of information, can hide within the cover media. As with any method of storing data, this can be expressed as a number of bits, which indicates the max message size that might be inserted into an image. If we assumed that the coloured image contains *XY* pixels, then every sub-band of its wavelets transform will contain 3*(XY/4) coefficients. So, the data payload of the proposed algorithm can be expressed using equations (18) & (19).

$$\textit{Data payload} = 3*4(XY/4)*N \text{ bits} \quad (19)$$

$$\textit{Payload Percentage} = [(3*4\ (XY/4)*N/8) / 3XY] *100\% = (N/8*100)\ \% \quad (20)$$

Discrete Wavelet Transform (DWT) has been widely used in many signal and image processing because its spatial-frequency relationship. The principle drawback of the DWT for some applications is that the coefficients in the decomposed sub bands are real values and then some modifications (including the machine precision) to these values cause quantization errors at the reconstruction stage, and in consequence, a perfect reconstruction cannot be achieved. The integer Wavelet Transform (IWT) was proposed to overcome this problem in 1996.

IWT

$$d_{1,l} = s_{0,2l+1} - round\left[\frac{1}{2}(s_{0,2l} + s_{0,2l+2})\right] \quad (21)$$

$$s_{1,l} = s_{0,2l} + round\left[\frac{1}{4}(d_{1,l-1} + d_{1,l})\right] \quad (22)$$

IIWT

$$s_{0,2l} = s_{1,l} - round\left[\frac{1}{4}(d_{1,l-1} + d_{1,l})\right] \quad (23)$$

$$s_{0,2l+1} = d_{1,l} + round\left[\frac{1}{2}(s_{1,2l} + s_{l,2l+2})\right] \quad (24)$$

IWT and its inverse are given by Equations 21, 22 and 23, 24 respectively.

A. **FUNCTIONS OVERVIEW**

The 2 main functions performed in this paper are:

- Hiding data inside an image - EMBED
- Extracting data from the image – EXTRACT

B. EMBED

Embedding of data into the image comprises of 3 major processes as shown in Figure 13.

- **Sub-band Decomposition Using IWT:** Using Integer Wavelet Transforms (IWT) the large amount of data is compressed. This ensures that the image is decayed and stored.
- **Data Hiding using BPCS:** After compression of the data, the compressed data is stored as a secret message using BPCS. This process involves the following steps:
  - Bit Plane Decomposition & Block Segmentation
  - Complexity measurement of each block
  - Complexity measurement of secret message
  - Replace complex image-data block to message block
- **Inverse Integer Wavelet transform:** This module is used to assemble stego-image to its original form with the secret data hidden in it.

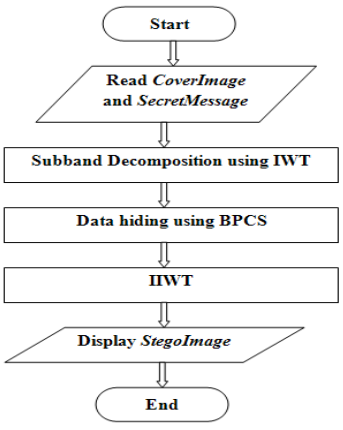


Figure 13: Flowchart for Data Hiding Process

C. EXTRACT

As seen in the Figure 14, extracting data from the image comprises of 2 major processes.

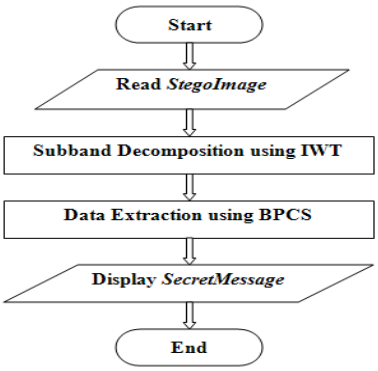


Figure 14: Flowchart for Data Extraction Process

- **Sub-band Decomposition Using IWT**
- **Data Extraction using BPCS:** Similar to data hiding, this process involves the following steps:
  - Bit Plane Decomposition
  - Block Segmentation
  - Complexity measurement of each block
  - Conjugation Map Extraction: The first complex blocks are corresponded to the conjugation map.
  - Hidden Data Extraction: The complex blocks are extracted and conjugated, if necessary to recover the secret message.

IV. RESULTS & DISCUSSION

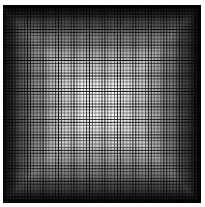
i. Type of the image: JPEG
Cover Image size: 9 KB
Size of the Secret text: 27.1 KB
Stego Image size: 56.7 KB
Size of the Secret text after extraction: 32.3KB

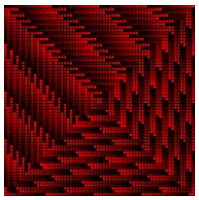

Before Embedding


After Embedding

- Height of the image = 250
- Width of the image = 250
- Number of pixels in the image = 62500
- File Block size = 3469
- Cover image considered is a grayscale image.
- Upon embedding of the text file, only the red channel of the image is obtained.
- Size of cover image is lesser than secret text file.
- Clear differences seen between the cover image and the stego image.
- The secret text is successfully embedded and extracted from the grayscale image.

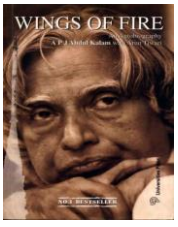
ii. Type of the image: PNG
Cover Image size: 21.4 KB
Size of the Secret text: 27.1 KB
Stego Image size: 74.3 KB
Size of the Secret text after extraction: 27.8 KB


Before Embedding

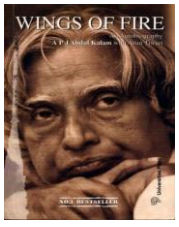

After Embedding

- Height of the image = 320
- Width of the image = 320
- Number of pixels in the image = 102400
- File Block size = 3469
- Cover image considered is a grayscale image.
- Upon embedding of the text file, only the red channel of the image is obtained.
- Size of cover image and secret text are comparable.
- Clear differences seen between the cover image and the stego image.
- The secret text is successfully embedded and extracted from the grayscale image.

iii. Type of the image: JPEG
Cover Image size: 29.1 KB
Size of the Secret text: 179 KB
Stego Image size: 394 KB
Size of the Secret text after extraction: 263 KB



Before Embedding



After Embedding

- Height of the image = 500
- Width of the image = 327
- Number of pixels in the image= 163500

- File Block size = 22912
- Size of the cover image is lesser than the secret text file.
- Secret text is successfully hidden in the cover image.
- No visible differences seen between the cover image and the stego image. However, upon careful observation we conclude that the stego image is not exactly similar to the cover image.
- Hidden data is successfully extracted from the stego image

iv. Type of the image: JPEG
Cover Image size: 10.4 KB
Size of the Secret text: 27.1 KB
Stego Image size: 39 KB
Size of the Secret text after extraction: 28.8 KB


Before Embedding After Embedding

- Height of the image = 120
- Width of the image = 160
- Number of pixels in the image = 19200
- File Block size = 3469
- Size of cover image is lesser than secret text file.
- Secret text is successfully hidden in the cover image.
- Clear differences seen between the cover image and the stego image.
- Hidden data is successfully extracted from the stego image.

v. Type of the image: JPEG
Cover Image size: 18.8 KB
Size of the Secret text: 75.4 KB
Stego Image size: 137 KB
Size of the Secret text after extraction: 82.4 KB


Before Embedding After Embedding

- Height of the image = 320
- Width of the image = 240
- Number of pixels in the image = 76800
- File Block size = 9652
- Size of the cover image is much lesser than that of the secret text file.
- Image failed to hold the entire data tried to be hidden in it.
- Only a part of the secret text is successfully hidden in the cover image.
- Clear differences seen between the cover image and the stego image.
- Hidden data is successfully extracted from the stego image.

V. CONCLUSION

BPCS – Image Steganography system can be customized for each user. The steganography method presented here can also be combined with some cryptography method to keep the data non-decipherable even if it were detected. It guarantees secret internet communication providing a high information hiding capacity. For a true colour image it was found to be around 50%. It protects against eavesdropping on the embedded information.

VI. REFERENCES

[1] Silvia Torres-Maya, Mariko Nakano-Miyatake, Héctor Perez-Meana, "*An Image Stegnography Systems based on BPCS and IWT*", Proceedings of the 16th IEEE International Conference on Electronics, Communications and Computers (CONIELECOMP 2006) 0-7695-2505-9/06

[2] Ms. K Ramani, Dr. E V Prasasd, Dr. S Varadarajan, " *Stegnography using BPCS to the integer wavelet transformed image*", IJCSNS International Journal of Computer Science and Network Security, VOL.7 No.7, July 2007.

[3] Yeshwanth Srinivasan, *"High Capacity Data Hiding System using BPCS Steganography"*, Texas Tech University, December 2003.

[4] Kawaguchi, E., Endo, T. and Matsunaga, J., "*Depth-first picture expression viewed from digital picture processing*", IEEE Trans. On PAMI, vol.5 page 354-373, 1988.

[5] Kawaguchi E. and Taniguchi, R., "*Complexity of binary pictures and image thresholding – an application of DF Expression to the thresholding problem*", Proceedings of 8th ICPR, vol.2 page 1221-1225, 1986.

[6] R. C. Gonzalez, R. E. Woods, *"Digital Image Processing"*, Prentice Hall of India, New Delhi, 2008

[7] E. Kawaguchi and Richard O. Eason, "*Principle and Applications of BPCS – Steganography*", SPOE's International Symposium on Voice, Video and Data Communications, vol.5, 1998.

[8] Jain, Anil K., *"Fundamentals of Digital Image Processing"*, Prentice Hall, Englewood Cliffs, NJ, 1989.

[9] K. Nozaki, M. Nimmi, Richard O. Eason and E. Kawaguchi, "*A large capacity steganography using color BMP images*", Proc. ACCV'98.

[10] Smitha Joseph, "*Implementation of the Two Dimensional Integer Wavelet Transform for transmission of image*", In Proc. Of IEEE International Conference on Acoustics, Speech and Signal Processing, vol.3, page 1749-1752, IEEE May 2001.