USING INTEGER WAVELET TRANSFORMS IN COLORED IMAGE-STEGANOGRAPHY

M. F. Tolba[1], M. A. Ghonemy[1], I. A. Taha[2], and A. S. Khalifa[1]

[1]Faculty of Computer & Information Sciences,
Ain Shams University, Egypt.
mtolba@tcdata.net.eg,
amal_said_Khalifa@hotmail.com,
ghonimys@yahoo.com

[2]Military Technical College (MTC),
Cairo, Egypt.
ismail_taha@informatique.com.eg

Abstract: *Digital Steganography exploits the use of a host data to hide a piece of information in such a way that it is imperceptible to a human observer. Wavelet transforms that map integers to integers allow perfect reconstruction of the original image. Hence, we proposed an algorithm that embeds the message bitstream into the LSB's of the integer wavelet coefficients of a true-color image. The algorithm also applies a preprocessing step on the cover image to adjust saturated pixel components in order to recover the embedded message without lose. Experimental results showed the high invisibility of the proposed model even with large message size.*

Keywords*: Information hiding, steganography, integer wavelets, high bit-rate encoding.*

1. Introduction

The appearance of the Internet is considered to be one of the major events of the past years; information become available on-line, all users who have a computer can easily connect to the Internet and search for the information they want to find [1]. This increasing dependency on digital media has created a strong need to create new techniques for protecting these materials from illegal usage. One of those techniques that have been in practical use for a very long time is Encryption. The basic service that cryptography offers is the ability of transmitting information between persons in a way that prevents a third party from reading it [2].

Although, encryption protects content during the transmission of the data from the sender to receiver, after receipt and subsequent decryption, the data is no longer protected and is in the clear. That what makes steganography compliments encryption. Digital Steganography exploits the use of a host (container) data to hide or embed a piece of information that is hidden directly in media content, in such a way that it is imperceptible to a human observer, but easily detected by a computer. The principal advantage of this is that the content is inseparable from the hidden message [3].

In a blind image steganographic system, a message is embedded in a digital image by the stegosystem encoder which uses a key. The resulting stego-image is transmitted over a channel to the receiver where it is processed by the stegosystem decoder using the same key [1]. In general, if the channel is monitored by someone who is allowed to modify the information flow between the two parties, he is called an *active warden*; but if he can only observe it, he is called a *passive warden*. Generally, the terminologies used in this paper agree with those in [4].

The scientific study of steganography began in 1983 when Simmons stated the prisoner's problem [5]. During the past few years, there has been a lot of research on developing techniques for the purpose of placing data in still images. Some techniques are more suited to dealing with small amounts of data, while others to large amounts. Some techniques are highly resistant to geometric modifications, while others are more resistant to non-geometric modifications, e.g., filtering. Current methods for the embedding of messages into cover images fall into two main categories: High bit-rate data hiding and low bit-rate data hiding, where bit-rate means the amount of data that can be embedded as a portion of the size of the cover image. In this section, we will survey methods that explore both of these areas [6].

In low bit-rate encoding, we expect a high level of robustness in return for low bandwidth. The emphasis is on resistance against attempts of data removal by a third party. One technique that is referred to as Patch-work is based on a pseudorandom statistical process. Patchwork invisibly embeds in a host image a specific statistic, one that has a Gaussian distribution. Two sets of pixels, or patches, of the image are chosen, and then the algorithm works by modifying the same factor in both patches. Hence, this unique statistic indicates the presence or the absence of a signature [6]. A second method for low bit-rate data hiding in images is Texture Block Coding. This method is implemented by copying a region from a random texture pattern found in a picture to an area that has similar texture. Simple autocorrelation is used to expose the hidden information. Several techniques for data hiding in multimedia can be found in [8].

On the other hand, with high bit-rate methods are usually designed to have minimal impact upon the perception of the host image, but they do not tend to be immune to image modifications. In return, there is an expectation that relatively large amounts of data can be encoded. All high bit-rate methods can be made more robust through the use of error-correction coding, at the expense of data rate. So, high bit-rate codes are only appropriate where it is reasonable to expect that a great deal of control will be maintained over the images.

The most common and simplest form of high bit-rate encoding is the least significant bit insertion (*LSB*)[6]. This method embeds the message into one or more least significant bits of some selected pixels. Not every pixel is suitable for being changed. Changing the values of some pixels may result in a degradation of the quality of the original object. There are algorithms that can decide if a pixel may be changed by checking the variance in luminosity of the surrounding pixels that may be neither very low nor very high. [7]. The advantages of the LSB method include the ease of implementation and high message payload.

Other techniques include embedding the message by modulating coefficients in a transform domain [7], such as the Discrete-Cosine Transform (DCT), Discrete Fourier Transform, or Wavelet Transform. The transformation can be applied to the entire image or to its subparts. The embedding process is done by modifying some coefficients that are selected according to the type of protection needed. If we want the message to be imperceptible then the high range of frequency spectrum is chosen, but if we want the message to be robust then the low range of frequency spectrum is selected. Usually, the coefficients to be modified belong to the medium range of frequency spectrum, so that a tradeoff between perceptual invisibility and robustness is achieved [1].

Another technique depends on the introduction of high-frequency, low-amplitude noise and the use of direct sequence spread spectrum coding. This method combines techniques from spread spectrum communication, error-control coding, and image processing. The fundamental concept is that the data is embedded in the noise, which is added to the original image. Because the noise is low power and the decoding process is not perfect, a low-bit error-correcting code is incorporated [1].

In this paper, a new transform domain technique for embedding the secret information in the integer wavelet transform of the cover image is discussed. Hence, the paper is organized as follows: the next section explores the idea of integer wavelet transforms and redefines the 1D S-Transform into 2D. Then the proposed scheme is presented in section 3 which is divided into three other subsections each of which discusses a part of the model. The last section analyzes the performance of the proposed scheme and compares it with other famous embedding techniques.

2. Integer-to-Integer Wavelet Transforms

The wavelet domain is growing up very quickly. A lot of mathematical papers and practical trials are published every month. Wavelets have been effectively utilized as a powerful tool in many diverse fields, including approximation theory; signal processing, physics, astronomy, and image processing [9].

A one dimensional discrete wavelet transform is a repeated filter bank algorithm [10]. 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 1, these steps result in four classes of coefficients: the (*HH*) coefficients represent *diagonal* features of the image, whereas (*HG* and *GH*) 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)).

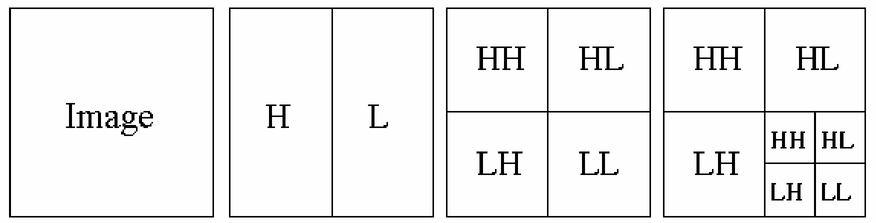


Fig. 1: A Two dimensional wavelet Transform

Since the discrete wavelet transform allows independent processing of the resulting components without significant perceptible interaction between them, hence it is expected to make the process of imperceptible embedding more effective. However, the used wavelet filters have floating point coefficients. Thus, 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. However, with the introduction of Wavelet transforms that map integers to integers we are able to characterize the output completely with integers [9, 10, 11].

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 (1a) and (1b) respectively. Note that the smooth and the detail outputs are the results of the application of the high-pass and the low-pass filters respectively [7, 9]. 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 [10]. The *S-transform* is thus reversible and its inverse is given in equations (2a) and (2b).

$$s(n) = \left\lfloor \frac{x(2n) + x(2n+1)}{2} \right\rfloor \tag{1a}$$

$$d(n) = x(2n) - x(2n+1) \tag{1b}$$

$$x(2n) = s(n) + \left\lfloor \frac{d(n)+1}{2} \right\rfloor \tag{2a}$$

$$x(2n+1) = s(n) - \left\lfloor \frac{d(n)}{2} \right\rfloor \tag{2b}$$

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 color 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 (3a), (3b), (3c), and (3d). 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 (4a), (4b), (4c), and (4d). 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*) and *(D*) reflect *vertical* and *diagonal* information respectively. During the transform we ignore any odd pixels on the borders.

$$A_{i,j} = \lfloor (I_{2i,2j} + I_{2i+1,2j})/2 \rfloor \tag{3a}$$

$$H_{i,j} = I_{2i,2j+1} - I_{2i,2j} \tag{3b}$$

$$V_{i,j} = I_{2i+1,2j} - I_{2i,2j} \tag{3c}$$

$$D_{i,j} = I_{2i+1,2j+1} - I_{2i,2j} \tag{3d}$$

$$I_{2i,2j} = A_{i,j} - \lfloor H_{i,j}/2 \rfloor \tag{4a}$$

$$I_{2i,2j+1} = A_{i,j} + \lfloor (H_{i,j}+1)/2 \rfloor \tag{4b}$$

$$I_{2i+1,2j} = I_{2i,2j+1} + V_{i,j} - H_{i,j} \tag{4c}$$

$$I_{2i+1,2j+1} = I_{2i+1,2j} + D_{i,j} - V_{i,j} \tag{4d}$$

where,

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

Note that the presented transforms are not computed using integer arithmetics, since the computations are still done with floating point numbers. However, the result is guaranteed to be integer [10] due to the use of the floor function and hence the invertetibility is preserved.

3. The Proposed Algorithm

The introduction of wavelet transforms that map integers to integers in the field of image steganography allowed the embedded message to be recovered without loss. We will apply the 2D *S-Transform* on each color plane of the colored cover image. Then the proposed algorithm stores the message bitstream in the least significant bits of the transform coefficients. This process obviously does not affect the integrality of the embedded coefficients.

The main problem encountered in implementing this algorithm was the error caused by hiding bits in coefficients that correspond to saturated pixel components. The embedding process may modify those coefficients making them exceed their maximum value (255). This range violation may result in loosing some parts of the embedded message. So, we proposed applying a pre-processing step on the cover image before the embedding process takes place. This step adjusts the saturated pixel components in a way to guarantee that they do not exceed their maximum value due to modifying their corresponding coefficients.

3.1 Cover Adjustment

Before the embedding process takes place we need first to apply a pre-processing step on the cover image. This is a very important step to preserve the over all invertability of the transform. That is, the embedding process may modify a coefficient that corresponds to a saturated pixel color component in such a way that makes it exceed its maximum value (255). In this case higher values will be clipped and the embedded message bits would then be lost. Hence, the original cover pixel components(*C(i, j, k)*) are adjusted according to the formula shown in equation (5), where *C'(i, j, k)* denotes the modified pixel component, *i, j* denote the spatial coordinates of the pixel and *k* represents the specific color component (Red, Green, Blue). The *N* value denotes the number of bits to be embedded in each coefficient.

$$C^{'}(i,j,k)=\begin{cases} C(i,j,k)-(2^{N}-1) & C(i,j,k)=255 \\ C(i,j,k) & otherwise \end{cases} \tag{5}$$

This adjustment guarantees that the reconstructed pixels from the embedded coefficients would not exceed the maximum value and hence the message will be recovered correctly. In addition, we can ensure that this step does not add any noticeable distortion to the resultant stego-image since effect of this adjustment is expected to be eliminated due to the embedding process.

3.2 The Embedding Module

Now, let us discuss the embedding process of the proposed algorithm. Of course, we need first to convert the secret message into a 1D bit stream. Of course the details of this step will depend on the particular message type. For example, in the case that the message is in text form, we can form the bit stream by simply converting the ASCII code of each character into an 8-bit binary representation, and then concatenating them as a sequence.

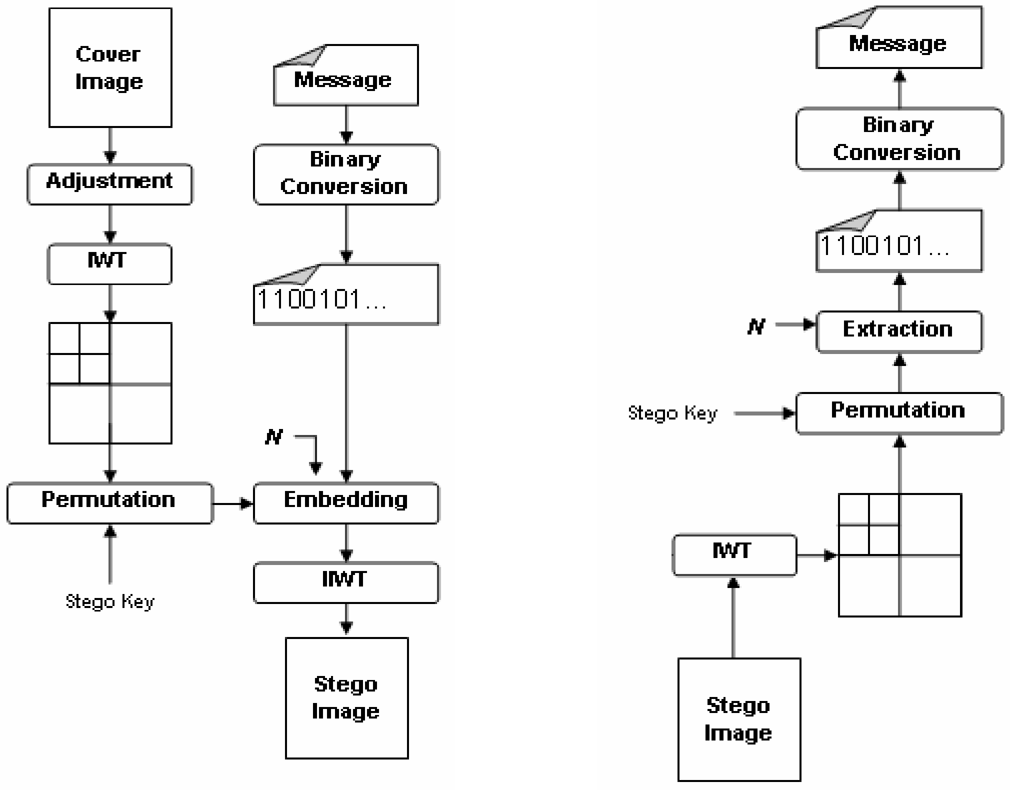
The next step that follows the cover adjustment is concerned with applying Integer Wavelet Transform (IWT) the on the cover image. However, the cover image is in true-color format (i.e. the image consists of three color planes: red, green and blue), so the wavelet transform is performed on each color plane separately. The embedding process stores (*N)* message bits in the least significant bits (LSB) of the IWT coefficients of the cover image. Furthermore, we have used the four sub-bands of the image transform for embedding. Of course, after the embedding process ends the stego image is produced by applying the *Inverse of the Integer Wavelet Transform* (*IIWT*) on the modified coefficients.

One thing is left to be considered. That is, we need to decide on the order by which the coefficients will be selected for embedding. We have employed the pseudorandom permutation as a secure selection scheme. The idea behind the permutation is that the permutation generator uses the stego key and produces as output different sequences of the set {1, 2, 3, …., length(Cover)} [1]. To build such a generator, Luby and Rackoff [12] proposed a pseudorandom function generator that is computationally feasible and secure. Nobody can guess the generated random sequence without knowing the secret key. This ensures that only recipients who know the corresponding secret key will be able to extract the message from a stego-object [13].

3.3 The Extraction Module

As shown in figure 2(b), the extraction process reverses the embedding operation starting from applying the IWT on each color plane of the stego image, then selecting the embedded coefficients, until extracting the embedded message bits from the *N* LSB's of the integer coefficients. Furthermore, the extracted bits are converted into its original digital form.

Obviously, the proposed scheme is blind since with the stego key only, the original cover-image is not needed to recover the embedded message from the received stego-image. In addition, the proposed scheme is considered secure. That is, without knowing the stego-key a passive warden can't extract the hidden message or even prove its very existence.



(a) **(b)**

Fig. 2: Block diagram for hiding binary data in integer wavelet coefficients of an image
(a) The embedding module, (b) The extraction module

4. Experimental Results

In high bit-rate data hiding we have two primary objectives: the technique should provide the maximum possible payload and the embedded data must be imperceptible to the observer. We stress on the fact that steganography is not meant to be robust. 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 [15].

Fundamentally, data payload of a steganographic scheme can be defined as the amount of information [3] it 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 colored image contains *XY* pixels, then every sub-band of its wavelet transform will contain 3*(*XY*/4) coefficients. So, the data payload of the proposed algorithm can be expressed using equations (6) and (7).

$$\text{Data payload} = 3 * 4(\mathrm{XY}/4) * \mathrm{N} \quad \text{bits} \tag{6}$$

$$\text{Payload percentage} = \frac{3 * 4(\mathrm{XY}/4) * \mathrm{N}/8}{3\mathrm{XY}} * 100\% = (\mathrm{N}/8 * 100)\,\% \tag{7}$$

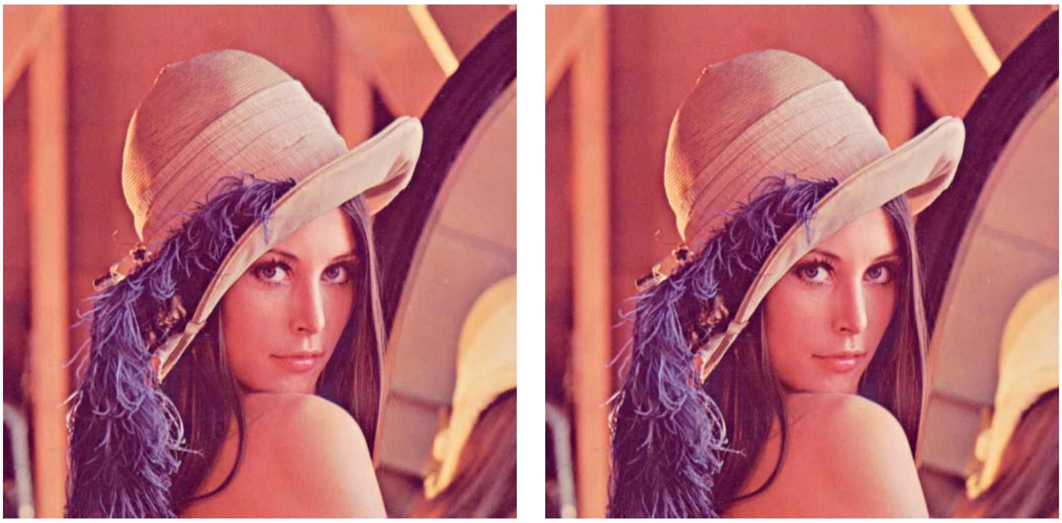
The question now is: how many bits per coefficient can be embedded while keeping an acceptable visual quality of the stego image? We tried to answer that question by embedding the maximum possible message for each value of *N* (using equation 6) where *N* takes a value between 1 and 8. Judging the visual quality of the resultant stego images in each case showed that the visual quality of the stego image is acceptable for embedding up to 4-bits per coefficient. So, substituting in equation (7) with *N*=4 results in an embedding capacity that represents 50% of the cover image size in terms of bytes.

Usually the invisibility of the hidden message is measured in terms of the Peak Signal-to-Noise Ratio (*PSNR*) defined in equation (8), where *p(x,y)* represents the color shade level of a pixel, whose coordinates are *(x,y)* in the original image, and $\tilde{p}(x,y)$ represents the same pixel in the distorted image. Equation (10) presents the Root Mean Square Error (*RMSE)* as a measurement criterion.

$$PSNR = 10\log_{10}\left(\frac{L^2}{MSE}\right) \qquad (8)$$

$$\mathrm{MSE} = \frac{1}{\mathrm{XY}}\sum_{\mathrm{x,y}}(\mathrm{p}(\mathrm{x,y}) - \tilde{\mathrm{p}}(\mathrm{x,y}))^2 \qquad (9)$$

$$\mathrm{RMSE} = \sqrt{\frac{1}{\mathrm{XY}}\sum_{\mathrm{x,y}}(\mathrm{p}(\mathrm{x,y}) - \tilde{\mathrm{p}}(\mathrm{x,y}))^2} \qquad (10)$$



(a) **(b)**

Fig. 3: Experimental result of the proposed method on colored images.
(a) Cover-image entitled Lena (512x512).
(b) Stego-image of the proposed method (*RMSE* = 0.0088 , *PSNR* = 73.91 dB)

Figure 3(a) shows the colored cover image entitled Lena and Figure 3(b) shows the resultant image after embedding the maximum possible message payload (≈ 96 Kb, 1 bit/pixel). The RMSE and PSNR measure for Figure 3(b) are 0.0088 and 73.91 dB respectively. Notice, the difference between the stego-images is barely distinguishable by the human eye.

To evaluate the performance of the proposed algorithm, several simulations have been performed in order to compare its performance with other existing schemes. Two types of LSB insertion methods were used, fixed size and variable size. The former embeds the same number of bits in each pixel of the cover image and the embedding capacity is 50% of the cover-image size. On the other hand, the variable-sized embedding method, the number of embedded LSBs depends on the local characteristics of the pixels to provide a higher capacity while keeping a better invisibility [14]. Unfortunately, those methods were developed for gray-scale images. So, we had to apply our method also on gray-scale images, although the proposed algorithm is developed mainly for colored images.

Table 1 shows the measured *PSNR* and *RMSE* for the images obtained from the two LSB methods as well as the proposed method using two cover images: *Lena* and *Maraho*. Each image is (512x512) in size as shown in Figure 4(a) and 4(b). In addition, we used Shakespeare's image (166x210) as a secret image (shown in figure 4(c)). The Table also show the maximum embedding capacity provided by each algorithm measured in bits per pixel (*BpP*).

Table 1: Experimental results of embedding using different methods

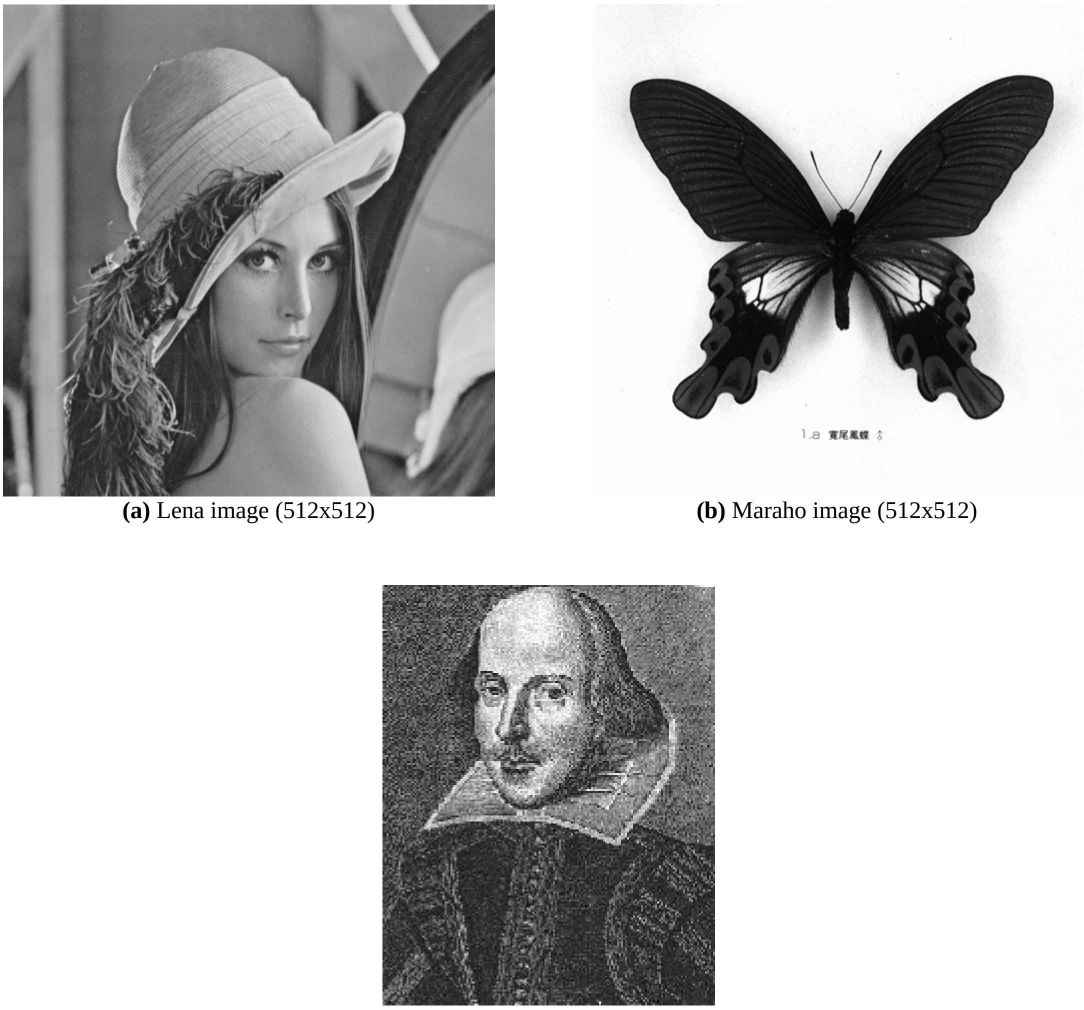
Embedding Method	**Max. Capacity**	**Lena**		**Maraho**	
		PSNR	RMSE	PSNR	RMSE
Fixed 4LSB	4 BpP	37.85 dB	3.263	36.79 dB	3.688
Adaptive LSB	4.03 BpP	37.65 dB	3.343	36.98 dB	3.608
Proposed method	4 BpP	39.36 dB	2.744	36.66 dB	3.746

According to the results listed in table 1, we can see that the adaptive LSB method proposed in [15] provides an embedding capacity a little bit more than 4 bits per pixel, while our proposed method (by analogy) can embed up to 4 bits per pixel. Furthermore, the proposed method provides better performance for the *Lena* image, while the other methods perform better for the *Maraho* image. This leads us to a very important conclusion. That is, our method is best suitable for images with a reasonable amount of detail and texture, while the adapted LSB algorithm is suitable for images with large smooth regions like the *Maraho* image.

5. Conclusions

Wavelet transforms that map integers to integers allow perfect reconstruction of the original image. The proposed algorithm deals with true-color images and applies the S-Transform on each color plane separately. The embedding process stores up to 4 message bits in each integer coefficient for all the transform sub-bands.

The algorithm preadjusts the original cover image in order to guarantee that the reconstructed pixels from the embedded coefficients would not exceed its maximum value and hence the message will be correctly recovered. The information capacity provided by the proposed algorithm can reach 50% of the original cover image size. Furthermore, experimental results showed that this scheme retains high quality of the stego-image over the existing LSB-based methods.



(a) Lena image (512x512)

(b) Maraho image (512x512)

(c) Shakespeare's image (166x210)

Fig. 4: The original gray-scale images used for comparison experiments

References

[1] Richard Popa, *An Analysis of Steganographic Techniques*, a working report for Faculty of Automatics and Computers- Department of Computer Science and Software Engineering at University of Timisoara, 1998.

[2] Neil F. Johnson and Sushil Jajodia, *Steganography: Seeing the Unseen*, IEEE Computer, February 1998, pp 26-34.

[3] Matt L. Miller Ingemar J. Cox, Jean-Paul M.G. Linnartz, *A review of watermarking principles and practices*, Published in "Digital Signal Processing in Multimedia Systems, Ed. K. K, Parhi and T. Nishitani, Marcell Dekker Inc., 461-485, (1999).

[4] Birgit Pfitzmann, *Information Hiding Terminology*, First Workshop of Information Hiding Proceedings, Cambridge, U.K. May 30 - June 1, 1996. Lecture Notes in Computer Science, Vol.1174, pp 347-350. Springer-Verlag (1996).

[5] G.J. Simmons, *The Prisoner's Problem and the Subliminal Channel*, In: Proceedings of CRYPTO '83. 1984.

[6] Walter Bender, Daniel Gruhl, Norishige Morimoto, Anthony Lu, *Techniques for Data Hiding* , IBM Systems Journal, Vol. 35, No. 3&4. Vol. 35 No. 3, 1996.

[7] Eugene T. Lin , Edward J. Delp , *A Review of Data Hiding in Digital Images*, In the Proceedings of the Image processing, Image Quality, and Image Capture Conference (PICS) , 1999.

[8] Walter Bender, Daniel Gruhl, Norishige Morimoto, Anthony Lu, *Techniques for Data Hiding* , IBM Systems Journal, Vol. 35, No. 3&4. Vol. 35 No. 3, 1996.

[9] A.R. Calderbank, Ingrid Daubechies, Wim Sweldens, Boon-lock Yeo, *Lossless image compression using integer to integer wavelet transforms*, in the international conference on image procrssing, Piscataway, NJ: IEEE Press, 1997, vol. I, pp 596-599.

[10] A. R. Calderbank, Ingrid Daubechies, Wim Sweldens, Boon-Lock Yeo, *Wavelet Transforms That Map Integers to Integers*, Applied and Computational Harmonic Analysis (ACHA), 1996.

[11] Ali Bilgin, Philip J. Sementilli, Fang Sheng, Michael W. Marcellin, *Scalable Image Coding Using Reversible Integer Wavelet Transforms*, 1999.

[12] Moni Naor, Omer Reingold, *On the construction of Pseudo Random Permutations : Luby- Rancoff Revisited*, Jornal of Cryptography, vol. 12, no. 1, 1999, pp. 29-66.

[13] Eugene T. Lin , Edward J. Delp , *A Review of Data Hiding in Digital Images*, In the Proceedings of the Image processing, Image Quality, and Image Capture Conference (PICS) , 1999.

[14] Yeuan-Kuen Lee and Ling-Hwei Chen, *A High Capacity Image Steganographic Model*, accepted by IEEE Proceedings Vision, Image and Signal Processing, 2000.

[15] Han-Yang Lo, Sanjeev Topiwala, Joyce Wang, *Wavelet Based Steganography and Watermarking*, Cornell University, Computer Science Department, 1998.

[http://dukiedoggie.tripod.com/cornell/wavelets/report.html]