Hiding Text Information in a Digital Image Based on Entropy Function

Nasser Hamad
Faculty of Information Technology, Arab American University, Palestine

Abstract: *The paper is concerned with hiding information into a digital image, specifically, an English text is used to be hidden into a digital grey-scale image. The purpose of this research is to embed a maximum text data size into the most suitable image selected among several images based on the binary entropy function measurements of both the text and image. The embedding process is constrained by minimising the bit error rate. Our results show that the binary entropy function can be considered as a powerful tool to select a proper image for a predetermined text under the BER constrain.*

Keywords: *Steganography techniques, flipping embedded text, binary entropy function.*

Received July 13, 2008; accepted September 25, 2008

1. Introduction

The scientific study of steganography in the open literature began in 1983 when Simmons [1] in his seminal work stated the problem in terms of communication in a prison. Steganography is the art of hiding information to prevent their detection by an unauthorized person [2, 3]. The digital information (bits or symbols) might be hidden in any digital object, either text or image. The hidden data should be embedded in image, e.g., without causing any kind of image degradation. Moreover, as the embedded text size goes large the number of image distortions is increased. Therefore, it is objective is to embed a text data into an image with minimum image degradation. That is, to embed a digital text in the Least Significant Bit (LSB) of the grey-scale image pixels in a way that the distortion in the image due to embedding, referred to as BER, is maintained minimum and always less than a predetermined threshold value. If the BER exceeds a threshold value, flipping all binary bits in a text message into its complement should be used to minimize the BER, such a flipping process is referred to as Flipping Embedded Text (FET).

The entropy of a certain message is defined as the average amount of information included in the message [4]. Accordingly, measuring the information entropy in both text and image, a first shot decision whether the previously selected pair of text and image is a proper pair or not is obtained. Thus, based on binary entropy function measurements, we picked up the proper text image pair that achieves the BER constraint.

Toward this end, we simulate the hiding problem as a Binary Symmetric Channel (BSC) [5], in which we represent the cover object *C* (the image before embedding) as a transmitted data, and stego object *S* (the image after embedding) as a received data. The embedding process is formulated as a noisy channel. In this context, to obtain a stego image *S* more closely to the cover image *C*, the entropy function is used to decrease the number of errors happened through embedding process. So, we expect that there is a relation between the entropy of the source information (cover image *C*) and the entropy of the noisy channel (text message *M*) that help us to determine if the source is suitable for the appropriate channel or not in the sense of errorless transmission (stego image *S* similar to *C*).

The entropy function is inversely proportional to probability of occurrence of an event [5]. That is, in a certain image that contains a large number of bits 0's (most likely black), one should expect a smaller BER occurs when using a text of higher distribution of 0's, henceforth. This concept drives us to consider the entropy measurements of both text and image. However, if the stego-object changes significantly, a third party may see that the information is being hidden and therefore could attempt to extract or destroy it [6]. In literature, there are two strategies to handle the image steganography techniques.

Spatial (Image) domain strategy: in image domain, the LSB technique is the most important and the easiest one to embed information in a cover image [7]. Popular steganographic tools that are based on LSB embedding vary in their approach for hiding information. Methods like Steganos and Stools use LSB embedding in the spatial domain, while others like Jsteg and OutGuess embed the message in the frequency domain [8]. With a well chosen image, one can even hide the message in the least as well as second to least significant bit and still not see the difference [7]. Some researchers hide the data in the 4-LSB. Alkhraisat in [9] used an algorithm to hide a maximum capacity of data in the cover image. His algorithm used the 4-LSB of the pixel representation for hiding the message.

Jsteg [8] uses the traditional LSB method, i.e., it replaces the LSB of the frequency domain of the cover image sequentially bit-by-bit. Which causes the secrete message becomes easy to extract and discover, which is not our case. Several techniques select the used pixels randomly, in this process, a chance of collision may happened, i.e., the same pixel may be selected twice in the random process [10].

Three closely related basic requirements in information hiding systems are capacity, security, and robustness [10]. Steganographic capacity which means "the amount of bits that can be hidden in an image using LSB technique without causing statistically significant modifications" is introduced by [11]. Their results are able to provide an upper bound on the capacity. However, our proposed technique uses the LSB algorithm and succeeds in decreasing the number of distortions.

Transform domain strategy: these strategies based on hiding information in more significant areas of the cover image making it more robust. Many transform domain methods are independent of the image format. Since such a strategy is out of our scope, for more details, consult [7, 12].

In section 2, we present our proposed system model, namely, text and image models. System capacity and performance is introduced in section 3. Numerical examples and simulation results are introduced in section 4. Finally, in section 5, we conclude our main results and present some future works

2. System Model and Analysis

In our system, we will consider a source that generates and hides the message M in an object, C. The resultant Stego Image, S, contains the hidden text. In our model the text message M should be flipped if the resultant BER > $\text{BER}_{\text{threshold}}$. The message should be embedded into the image using sequential algorithm employed in [10], in which the text message bits are sequentially (bit-by-bit) embedded into the image pixels.

We calculate the BER based on hamming distance measurements by choosing and comparing between the maximum, and minimum BER (BER_{max}), and (BER_{min}) obtained after the embedding process, respectively. Then, if $\text{BER}_{\text{max}} > 1 - \text{BER}_{\text{min}}$, the binary bits of the message should be flipped. A flag bit is generated to indicate whether the text message bits were flipped or not.

2.1. Text Model

We assume that a transmitter generates a text message contains all possible alphabets (characters). The source generates the message that contains M distinct characters. The i^{th} character, $1 \leq i \leq K$, denoted as m_i is generated by the source with probability $P(m_i)$, where K is the total number of characters from the source. Depending on the language itself, the characters are generated with different probabilities from the source, e.g., if we consider an English text message, due to its nature, the repetition of the character "*e*" differs from the repetition of, say, character "*q*". Accordingly, it is impractical to assume that the characters are generated with equally probable distribution [13], but, it is language dependent, that is to say, for $m_i \neq m_j$, we have, $P(m_i) \neq P(m_j)\ \forall^{\ i \neq j}$.

However, as the text characters are to be mapped to their ASCII codes or compressed to any codeword form, the source can be seen as a sequence of bits generator, where we assumed that the source generates a total number of bits, say, T. Thus, the generated alphabet is encoded into a vector t of binary bit stream of length T. One should note that, the Probability Density Function[1] (PDF) of the generated characters is a discrete non-uniform function, however, after mapping characters to their corresponding codes, the generated bits (0 and 1) have a discrete uniform p.d.f. of equal probability. Such a source is referred to as a Discrete Memmoryless Source (DMS) [14].

Accordingly, the vector t contains 0's and 1's of probabilities $P(0) = p$, and $P(1) = 1 - p$, respectively. The average amount of information, referred to as binary entropy function [5], is given by

$$H(p) = p \log_2 \frac{1}{p} + (1 - p) \log_2 \frac{1}{(1 - p)}. \tag{1}$$

The binary entropy function is shown in Figure 1 as a function of the probability p. We can see that the convex binary entropy function has its maximum value 1 when $p = 1 - p = 1/2$ and symmetrically decreases around p. The binary entropy function, moreover, has zero values if and only if $p = 0$ or 1.

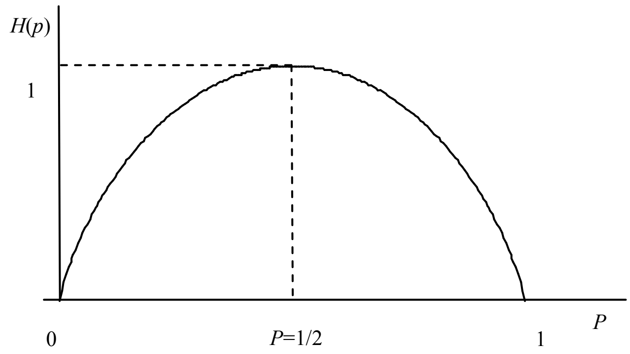

Figure 1. Binary entropy function of a discrete memoryless source.

The entropy function, $H(p)$, can be considered as a good measure of the statistical distribution of 0's and 1's within a specific text. Such a concept will be

[1] In literature, sometimes it is referred to as probability mass function (pmf).

considered when discussing the embedding of a binary mapped-text into a digital image.

2.2. Image Model

The image can be represented as a 2-dimensional function of two real variables $f(x,y)$, where x,y represent the spatial coordinates, and the function amplitude f given at any pair of coordinates is called the intensity or grey level of the image at that point. When x, y and the amplitude f are all finite and discrete quantities, we have a digital image [15]. A digital image consists of a rectangular map of the image's pixels. The number of bits used for each pixel is called the bit depth. For monochrome image, e.g., the bit depth is 1 bit (to represent black or white), and the bit depth of the grey image is 8 bits to display 256 different shades of grey.

No doubt that the image size increases with the number of pixels. Aiming at simplifying the matters, and due to the fact that the grey-scale images are considered a good type to hide data in spatial domain[2], we will consider the grey-scale image to hide the text file. The cover image C, can be seen as a 2-dimensional matrix of order $a \times b$. Each pixel position is determined by the spatial coordinates (x,y), where $0 \le x \le a$, and $0 \le y \le b$, the value of such a pixel is determined by the amplitude f of a decimal value encoded into 8-bits binary codeword that determines the image grey level.

Let us assume the cover image C is $a \times b$ pixels of total number of bits, $N = 8 \times a \times b$. As the text size is directly related to the image size, it is clear that the maximum text size is bounded by $N/8$, i.e., $T \le N/8$, of bits that can be hidden in the image pixels if we assume that the information is to be hidden only in the LSB of each pixel.

3. System Performance

3.1. System Capacity

Relying to the previous discussion, the maximum size of a text that can be embedded into the LSB of the image is T bits, but some of the LSB of the image pixels should be reserved to help the receiver to decode and extract the text from the received image. One bit was reserved as a flag bit to control the flipping process. Several bits (l bits) are reserved to represent the length of the text embedded in the image, in order to help the receiver to exactly find the bits of the message inside the image.

Lemma 1: the maximum system capacity (maximum data that can be embedded in the image) is $Q = T+l+1$.

Proof: Assume that the embedding is done in the LSB of each pixel, and define the decimal value of T in binary as $BT = \text{Bin}(T)$, to find the minimum number of bits that represents the text size T. Assume the number of bits that can represent the text length T is l, which is defined as the amount of information included in l, given by $l = \text{I}(l) = \log_2 (BT)$, were we consider BT in its decimal value. As T is represented by l bits, $T \le 2^l$. By taking $\log_2$ for both sides, we have, $\log_2 T \le l$. Thus, $l \ge$ Ceiling ($\log_2 T$), i.e., $l \ge \lceil \log_2 (T) \rceil$, where $\lceil x \rceil$ is the least maximum integer greater than x, and l is the minimum number of bits needed to represent the maximum text size used by specific image. Thus, the maximum system capacity is $Q = T + l + 1$.

3.2. System BER Performance

Through hiding information inside image pixels, we sometimes need to change the LSB in each pixel, and hence distortion of the image will occur. Such a distortion generates random errors in the original image, which we referred to as a BER. Simply, the BER is defined as the average number of bits in the LSB of each pixel that are changed through hiding the text information. Thus, our objective is to embed a high capacity of information into a cover grey-scale image, C, with minimum BER. We assume the LSB's of cover image C is q vector of binary bits of size Q where $Q = T \le N/8$, the i-th element in vector q, say $q_i \in \{0,1\}$ where $1 \le i \le Q$, Similarly, the secrete message M can be seen as a vector of binary bits, t where $t_i \in \{0,1\}$ and $1 \le i \le T$. Accordingly, the problem is completely simplified to hide a vector of bits t into a vector of bits q to formulate a vector of bits, say, s, as shown in Figure 2.

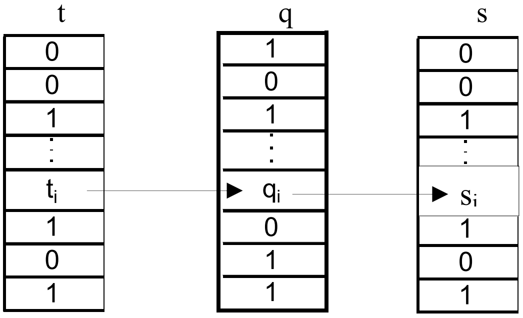

Figure 2. The embedding (transition) process of t into q to generate s.

Assume E_t and E_q represent the binary entropy functions of $p(0) = p$ in vectors t and q, respectively. We define the difference of entropies expressed as (L) between vector E_t and E_q and the difference of probabilities (β) between P_t and P_q as shown in Figure 3 below given as

$$L = | E_q - E_t |, \tag{2}$$

$$\beta = | P_q - P_t |. \tag{3}$$

Hereinafter, we will call β as the flipping parameter. That is, one can decide whether to flip or

[2] Data is hidden in the least significant bits of the image pixels

not to flip the bits in the text data based on the value of β.

Theorem 1: the BER due to the embedding process of a data text in a grey-scale image is minimized if the flipping parameter β is minimized.

Proof: the proof of theorem 1 is based on the discussion of the following cases:

Case 1: when E_t and E_q are in the same half, and they are coincide. In this case $L = \beta = 0$, and this represents the best case since we can guess that the entropy of the bit 0 distribution in both vectors are almost close to each other in the average. Such a result can be used to indicate that the number of 0's in text and image are approximately equal in average. The result obtained in this case is not to flip.

Case 2: if the two vectors are in the same half and they have a maximum L=1 and β = ½, this case represents the extreme case since the distribution of bits in both vectors are uncorrelated. One should note that flipping process does not make sense, because the flipping procedure will keep L = 1 and β = 1/2.

Case 3: if the two vectors are in the same half but not in the extreme case (case 2), we always have the absolute difference between P_t and P_q, β < ½, and the entropy difference L < 1. In this case, the distribution of bit 0 in the text and image are close to each other when the entropy difference is as minimum as possible, since the difference β becomes minimum. Therefore, we can achieve the best case as the two vectors close to each other until they are coincide.

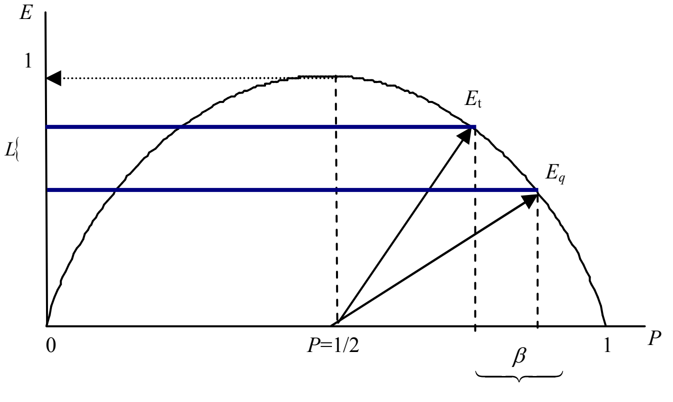

Figure 3. The binary entropy function shows the entropy difference L and probability difference β.

One should note that there is no need to flip the text vector, as flipping process in this case increases β which leads to uncorrelated distribution of bit 0 in both vectors.

Case 4: if the two vectors are not in the same half, and the entropy difference L = 0, this means that the distribution of 0's in the text and image are opposite of each other. In this case, $\beta \neq 0$, if $\beta \leq 1/2$ then the text vector should not be flipped, the flipping occurs only if β > 1/2, in this case E_t vector must be flipped Since the statistical average of bit 0 in both cases are completely opposite to each others. The flipping operation moves the vector E_t to $\bar{E}_t$ as shown in Figure 4 in the other half and it coincides over the image vector E_q which represents case 1, the best case, since the probability difference β is flipped and, hence, minimized to $\beta' = 0$.

Case 5: in this case the two vectors are in two different halves in which $0 < L < 1$, on the other hand, the probability difference $\beta \leq 1/2$, in this case there is no need to flip E_t because the statistical distribution of bit 0 in both vectors is still closed to each other. Furthermore, if $\beta > 1/2$, the text vector must be folded to a new location $\bar{E}_t$ in the second half because the statistical average of bit 0 in both cases are completely uncorrelated, on the other hand β is minimized as a result of flipping, i.e., the statistical distribution of bit 0 in $\bar{E}_t$ is correlated to E_q. Here, the flipping process does not affect L but it affects β, in other words L after flipping remains as it is and β is minimized and it is changed to β'.

Accordingly, in order to correlate the statistical distribution of bit 0 in both text and image, we need to minimize β, such a minimization is carried out by flipping process, which rotates the text vector E_t from the first half to a new position $\bar{E}_t$ in the second half.

As shown in Figure 4, when E_t is moved closely to E_q, β is decreased, and E_t is closing to E_q means that the distributions of bit 0 in both vectors become correlated.

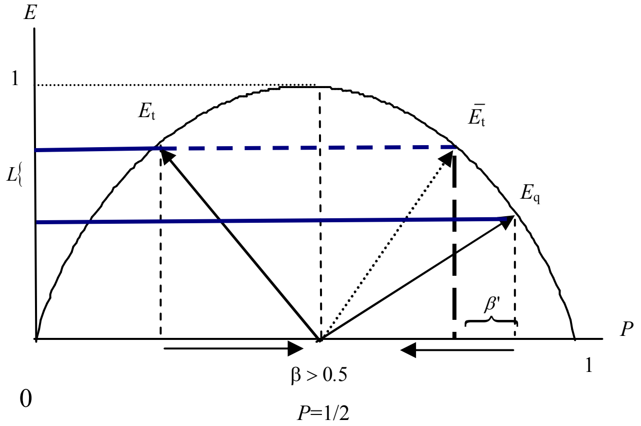

Figure 4. Minimizing β by folding E_t to $\bar{E}_t$.

Recall the entropy function, we assume $E_z(0)$ denotes either $E_t(0)$, $E_q(0)$, $E_s(0)$ which are the entropies of a bit 0 in vectors t , q and s, respectively, then

$$\begin{aligned} E_z(0) &= P_z(0)\log_2\frac{1}{P_z(0)} + P_z(1)\log_2\frac{1}{P_z(1)} \\ &= P_z(0)\log_2\frac{1}{P_z(0)} + (1-P_z(0))\log_2\left(\frac{1}{(1-P_z(0))}\right) \end{aligned} \qquad (4)$$

The results of calculation based on the entropy function measurements are given in the next section.

4. Simulation and Numerical Results

We adopt the sequential algorithm of [15] to construct the vector q from a given specific image. That is, assign a fixed size specific image that is used to formulate the q vector used to hide the text. In our simulation, we randomly have collected 50 images of size 256×256 pixels. The generated q vectors from these images are of average 8 KB size. Then, construct the vector t, serially bit-by-bit, from the encoded text of different sizes. We assume a text of 100 byte, then incremented by 100 byte every time. The entropy value increases as text size increases since the probabilities of 0's approach to 0.5 when the text size becomes large, so the entropy text approaches to 1. After that, E_t is calculated for 8 KB text, and 50 values of E_q were calculated for each image, from which the differences L and β are calculated. It is worthy to mention that the entropy does not show how 0's and 1's are arranged. Thus, we assist on the fact that the measurements here are independent on how the vectors are arranged.

We introduce several graphs to show how the entropy function behaves with size, entropy difference L, flipping factor, β, and BER. The BER without and with flipping the vector t is calculated using the weight of the hamming distance $w(d)$ vector divided by the length of t vector. That is to say BER = $w(d)/T$.

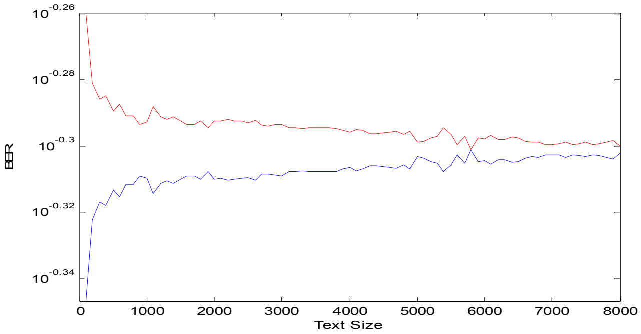

Figure 5. Text size versus BER for sequential algorithm with flip and no flip.

In Figure 5, we show the relation between the flipping and non-flipping techniques for different text size. The larger the text size, the more BER we have. With flip, it is noticed that flipping behaves better, especially, in the case of small size text, however, when the text size goes larger, we see that the flipped text and the unflipped one are almost the same. Figure 6 shows the relationship between L (the absolute difference between the entropy of text vector t and the entropy of image vector q), and the text size. As shown in the figure, the difference L decreases with text size almost exponentially. It is worthy to note that for small text size, the difference is noticeable and it becomes smaller with larger text. Aiming at explaining such a result, when L is small, this means that E_t and E_q are almost equal and L becomes 0 when E_t = E_q. Such a result completely coincide with our previous discussion and happened in large text size, since the distribution of 0's and 1's, on the average, becomes the same.

The entropy difference L and the BER relationship is shown in Figure 7, it is noticed that the BER increases as L increases and vice versa.

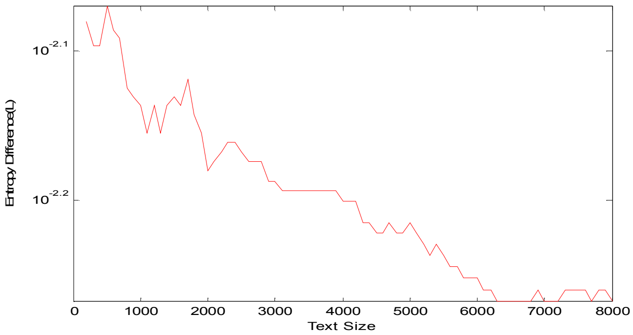

Figure 6. Text size with entropy difference between vector t and vectors q.

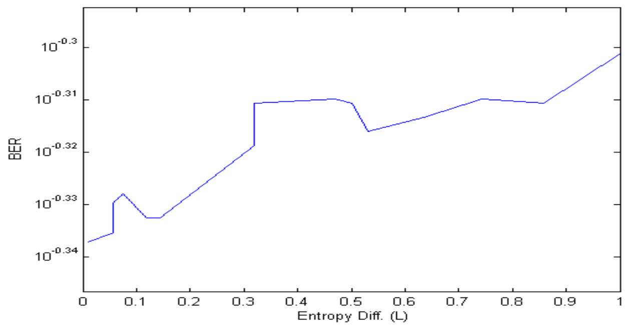

Figure 7. Entropy difference L and the BER.

Finally, Figure 8 shows the relation between the BER and the entropy difference L. In this case, we note that the BER is symmetric around the threshold value. If the BER goes large and exceeds the $\text{BER}_{\text{threshold}}$ the flipping process is necessary to keep BER as minimum as possible. From the figure, we can say that the flipping process always keeps the BER under $\text{BER}_{\text{threshold}}$.

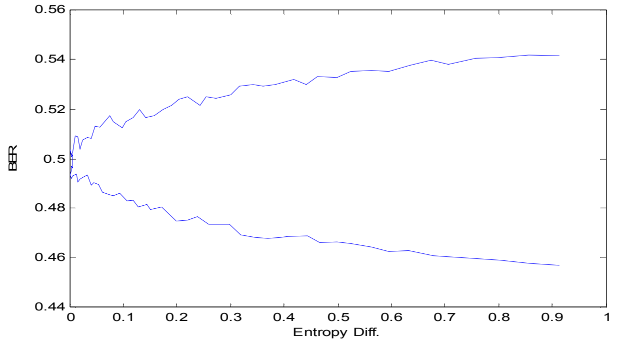

Figure 8. Entropy difference of fixed text and 50 images with BER.

5. Conclusions

In this paper, a binary text can be embedded into a grey-scale image using FET technique while keeping minimum BER based on a newly proposed technique, namely, the binary entropy function. Our main results can be concluded as: The FET technique is used to minimize the number of distortions and the artefacts occurred in the cover image through embedding process. There is a relation between the binary entropy function and the BER. Moreover, the BER increases with the entropy difference between the image entropy and the text entropy. We can say that the entropy measurement is useful in studying the distribution of bits 0's and 1's in both image and text as a first indication to determine whether the image is considered suitable for a specific text or not. We believe that more deep analysis is required to take the full benefits of the entropy and information theoretic concepts.

Acknowledgements

The author would like to thank Dr. Badie Sartawi for his encouragement to publishing the work and to Mr. Nizar Shihab for his help to simulate the proposed measurement techniques.

References

[1] Alkhraisat H., "4-Least Significant Bits Information Hiding Implementation and Analysis," *in Proceedings of Graphics, Vision and Image Processing Conference*, Egypt, pp. 19-21, 2005.

[2] Avcibas I., Memon N., and Sankur B., "Steganalysis Using Image Quality Metrics," *Computer Journal of IEEE Transactions on Image Processing*, vol. 12, no. 2, pp. 1132-1144, 2003.

[3] Chandramouli R. and Memon N., "Analysis of LSB Based Image Steganography Techniques," *in Proceedings of International Conference on Image Processing*, Greece, pp. 197-202, 2001.

[4] Fridrich J., Goljan M., and Soukal D., "Searching for the Stego Key," *in Proceedings of the International Society for Optical Engineering*, France, pp. 70-82, 2004.

[5] Gonzalez R., Woods R., and Eddins S, *Digital Image Processing Using MATLAB*, Prentice Hall, 2004.

[6] Hany F. and Siwei L., "Steganalysis Using Higher Order Image Statistics," *Computer Journal of IEEE Transactions on Information Forensics and Security*, vol. 1, no. 1, pp. 1-15, 2006.

[7] Morkel T., Eloff J., and Olivier M., "An Overview of Image Steganography," *in Proceedings of the 5th Annual Information Security Conference*, South Africa, pp. 44-51, 2005.

[8] Neil F. and Jajodia S., "Exploring Steganography: Seeing the Unseen," *Computer Journal of IEEE Computer Society*, vol. 31, no. 2, pp. 26-34, 1998.

[9] Norman A., *Information Theory and Coding*, McGraw Hill, 1986.

[10] Provos N. and Honeyman P., "Hide and Seek: An Introduction to Steganography," *Computer Journal of IEEE Security and Privacy*, vol. 1, no. 3, pp. 32-44, 2003.

[11] Shannon C., "Mathematical Theory of Communications," *Computer Journal of Bell System Technical*, vol. 27, no. 6, pp. 379-423, 1948.

[12] Shihab N. and Sartawi B., "Improving LSB Technique for Hiding Information in a Digital Image," *Master Thesis*, Al-Quds University, 2008.

[13] Simmons J., "The Prisoner's Problem and the Subliminal Channel," *in Proceedings of Crypto*, New York, pp. 51-67, 1984.

[14] Simon H., *Communication Systems*, Mc Grawhell, 2004.

[15] Thomas J. and Cover T., *Elements of Information Theory*, John Wiley & Sons, 1991.

Nasser Hamad received his PhD degree in electronics and telecommunication engineering from the University of Electro-Communications Tokyo, 2001. His research includes cellular mobile communication, information theory and coding, and digital communications. He leads and teaches modules at both BSc and MSc levels in electronics and communication engineering.