International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:3, No:3, 2009

A Novel Steganographic Method for Gray-Level Images

Ahmad T. Al-Taani and Abdullah M. AL-Issa

***Abstract*—**In this work we propose a novel Steganographic method for hiding information within the spatial domain of the gray scale image. The proposed approach works by dividing the cover into blocks of equal sizes and then embeds the message in the edge of the block depending on the number of ones in left four bits of the pixel. The proposed approach is tested on a database consists of 100 different images. Experimental results, compared with other methods, showed that the proposed approach hide more large information and gave a good visual quality stego-image that can be seen by human eyes.

***Keywords*—**Data Embedding, Cryptography, Watermarking, Steganography, Least Significant Bit, Information Hiding.

I. INTRODUCTION

DATA hiding became an important field as the use of the Internet became popular. Data hiding is a young field and it is growing in an exponential rate [1],[2]. Steganography is used to hide information inside other. As derived from Greek, the word steganography literally means "Covered Writing". Steganography is the art and science of communicating in a way, which hides the existence of the communication, or it is the art of hiding information in ways that prevent the detection of hidden messages [3].

The main purpose of steganography is to hide a message in another one in a way that prevents any attacker to detect or notice the hidden message. The aim of this work is to develop a new method for hiding message in gray-scale images, mainly embedding text data in digital images.

In this work we present an efficient Steganographic approach for hiding information within a gray scale image. We have compared the new method with two well-known methods, PVD and GLM methods. Our results show the effectiveness of the proposed method compared with the other methods.

The rest of this paper is organized as follows. Section 2 introduces some important issues of Steganography and Data Hiding Methods. Section 3 discuses related work in the field of data hiding. Methods and materials are discussed in Section 4. Section 5 introduces the experimental results of the proposed method. Conclusions and future work are presented in section 6.

Ahmad T. Al-Taani is with the department of Computer Sciences, Yarmouk University, Irbid, Jordan. (Correspondence author: e-mail: ahmadta@yu.edu.jo).

Abdullah M. AL-Issa is with the Integrated Technology Group (ITG), Amman, Jordan (e-mail: noor_kamel@yahoo.com.).

II. STEGANOGRAPHIC AND DATA HIDING METHODS

A. Overview of Steganograhy

Steganography hides secret messages under the cover of a carrier signal so it cannot be seen or detected [3-5]. Steganography technique should generally possess two important properties: good visual/statistical imperceptibility and a sufficient payload. The first is essential for the security of hidden communication and the second ensures that a large quantity of data can be conveyed [6]. Two levels of protection can be done if the message is encrypted before hiding it, so it must be decrypted before reading it.

Invisible watermarking is treated as a subset of steganography [7]. The difference is that steganography conceals a message so that this hidden message is the object of the communication where in watermarking; the hidden message provides important information about the cover media, such as authentication or copyright.

Figure 1 shows the main components of a Steganography technique. Two main steps are shown in this Figure: Embedding the secret message inside the cover using a stegokey and extracting the secret message from the cover using the stegokey. Combining the embedded message into the cover makes a stegomedia. A stegokey is used to hide and extract the secret message. Only the holder of the stegokey can retrieve correctly the hidden secret message. Also, Steganography can be described using the following formula [3]: Cover media + embedded message + stegokey = Stegomedia

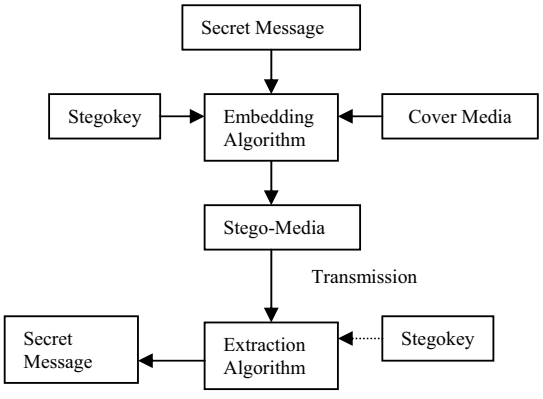

Fig. 1: Steganography Technique.

B. Data Hiding Methods

Many Steganography methods have been proposed to hide the secret data into the image. These data could be invisible inks, microdots, etc [7]. The most well known method to data hiding is called the least significant bit (LSB), which utilizes some least bits of pixels in the cover image to imbed secret data [8]. Some of the proposed approaches that based on the LSB method can be found in [9]. LSB is a simple approach to embedding information in an image. But image manipulation can destroy the hidden information in this image. Applying LSB technique to each byte of a 24-bit image, three bits can be encoded into each pixel, as each pixel is represented by three bytes. Applying LSB technique to each byte of an 8-bit image, only one bit can be encoded into each pixel, as each pixel is represented by one byte. For example, if we use 8-bit image to hide the letter A (has the binary value 10000011), we need eight pixels. Suppose the original eight pixels are:

(00100111) (11101001) (11001000) (00100111)
(11001000) (11101001) (11001000) (00100111)

Inserting the letter A (as a binary value) into these eight pixels will give the following (starting from left side):

(00100111) (1110100**0**) (11001000) (0010011**0**)
(11001000) (1110100**0**) (1100100**1**) (00100111)

Only the emphasized bits are changed. Care needs to be taken in the selection of the cover image, so that changes to the data will not be visible in the stego-image. Common images, like the *Mona Lisa* painting, should be avoided. Grayscale images are preferred because the shades change very gradually between palette entries. This increases the image's ability to hide information [10].

Masking and filtering techniques are mostly used in 24 bit and grayscale images. These hide information in a way similar to watermarks on actual paper and are sometimes used as digital watermarks. Masking images entail changing the luminance of the masked area. The smaller the luminance change, the less of a chance that it can be detected. Masking techniques are more suitable for use in lossy JPEG images than LSB insertion because of their relative immunity to image operations such as compression and cropping [11].

Recently, Rehab *et al.* [12] proposed a novel approach of image embedding consists of three main steps. First, the edge of the image is detected using Sobel mask filters. Second, the least significant bit LSB of each pixel is used. Finally, a gray level connectivity is applied using a fuzzy approach and the ASCII code is used for information hiding. The prior bit of the LSB represents the edged image after gray level connectivity, and the remaining six bits represent the original image with very little difference in contrast. The proposed method embeds three images in one image and includes, as a special case of data embedding, information hiding, identifying and authenticating text embedded within the digital images.

C. PVD Method for Gray-Level Image

The pixel-value differencing (PVD) method [5] segments the cover image into nonoverlapping blocks containing two connecting pixels and modifies the pixel difference in each block (pair) for data embedding. A larger difference in the original pixel values allows a greater modification. The hiding algorithm is described as follows:

1. Calculate the difference value d_i for each block of two consecutive pixels p_i and p_{i+1}.
   $d_i = p_i - p_{i+1} - (p_{i+1} - p_i)$.
2. Find the optimal R_i of the d_i, such that $R_i = \min(u_i - k)$, where $u_i \geq k, k = |d_i|$ and $R_i \in [l_i\ u_i]$.
3. Decide *t* bits of secret data which are hidden with each d_i, i.e. each block of two consecutive pixels is defined as $t = \log_2 w_i$ where w_i is the width of the R_i.
4. Read *t* bits binary secret data one by one according to Step 3, and then transform *t* into decimal value *b*. For instance, assume a binary secret data is 101, then $b = 5$.
5. Calculate the new difference value $d_i^{'}$ using:
$$d_i^{'} = \begin{cases} l_i + b, \text{for } d_i \geq 0; \\ -(l_i + b), \text{for } d_i < 0. \end{cases}$$
6. p_i and p_{i+1} are modified to hide *t* secret data by the following formula:
$$(p'_i, p'_{i+1}) = \begin{cases} (p_i - \lceil m/2 \rceil, p_{i+1} + \lfloor m/2 \rfloor) : d_i \in odd; \\ (p_i - \lfloor m/2 \rfloor, p_{i+1} + \lceil m/2 \rceil) : d_i \in even, \end{cases}$$
where $m = d'_i - d_i$. Finally, we compute the values of (p'_i, p'_{i+1}) which represent the secret data.
7. Repeat Steps 1-6, until all secret data are hidden into the cover image and the stego-image is obtained.

In the extraction phase, the original range table is necessary. It is used to partition the stego-image by the same method as used to the cover image. The extraction phase is implemented as follows:

1. Calculate the difference value $d_i^{'}$ between each two successive pixels for each block (p'_i, p'_{i+1}), from the following formula: $d_i^{'} = |p'_i - p_{i+1}^{'}| = -(p_{i+1}^{'} - p'_i)$.
2. Find the optimum R_i of the $d_i^{'}$ just as in Step 2 in the hiding phase.
3. Obtain b' by subtracting l_i from $d_i^{'}$. The b' value represents the value of the secret data in decimal.

4. Convert b' into binary then find number of bits (t) from the secret data, where $t = \log_2 \ w_i$ [9].

D. Gray Level Modification (GLM) Steganography

Gray level modification Steganography is a technique to map data (not embed or hide it) by modifying the gray level values of the image pixels. GLM Steganography uses the concept of odd and even numbers to map data within an image. It is a one-to-one mapping between the binary data and the selected pixels in an image. From a given image a set of pixels are selected based on a mathematical function. The gray level values of those pixels are examined and compared with the bit stream that is to be mapped in the image.

Initially, the gray level values of the selected pixels (odd pixels) are made even by changing the gray level by one unit. Once all the selected pixels have an even gray level it is compared with the bit stream, which has to be mapped. The first bit from the bit stream is compared with the first selected pixel. If the first bit is even (i.e. 0), then the first pixel is not modified as all the selected pixels have an even gray level value. But if the bit is odd (i.e. 1), then the gray level value of the pixel is decremented by one unit to make its value odd, which then would represent an odd bit mapping. This is carried out for all bits in the bit stream and each and every bit is mapped by modifying the gray level values accordingly.

Implementation of GLM Technique

To begin the process of embedding, we first select a set of pixels, which would be used for hiding the data. We will show that we need only 10 pixels because we have to store a bit stream with 10 elements.

For example, taking the bit stream 1001010101. The first element of the bit stream is 1, so we have to modify our selected pixel to represent 1. We propose to decrement the value of the selected pixel by one so as to make them odd; and now this odd pixel would represent a bit stream equivalent to 1. The overall concept can be summed up as follows:

- Select pixels according to an arbitrary function g (x, y).
- Modify the gray level values of the selected pixels to make them even by adding one. These even gray levels will represent 0 in a bit stream.
- To represent 1, modify the appropriate pixel by decrementing its gray level value by one.
- Thus, we can represent both 1s and 0s using pixels, which satisfy the condition of being odd or even.

So the bit stream 1001010101 can now be represented as shown in Figure 2. We see that the shaded pixels have an odd value. This is because we decremented the gray level value by one unit. So, these shaded pixels would represent 1s while the un-shaded pixels would represent 0s.

One important point needs to be kept in mind is we have to follow a serial order when modifying the pixels i.e. after selecting the pixels (and making them even); we map the proper bit stream with the proper pixel. What we mean by this is the first element of the bit stream should be mapped with the first selected pixel in the image. This is a main requirement otherwise it would not be possible to retrieve the data.

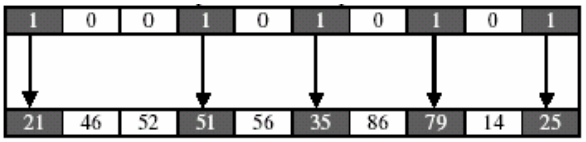

Fig. 2: Changing Gray level Values.

Now, we will discuss the retrieval process. The process of retrieval is completely opposite to that of embedding. We have the modified image and the secret function 2x+5%w. Using this function, we identify those pixels that have been used to embed data. The receiver knows the algorithm logic that an even value of gray level represents 0 while an odd value represents 1. Knowing this, the receiver can acquire the hidden data. The retrieval process is explained in the Figure 3. The receiver first picks up the selected pixels and map them to the respective binary data. Odd number is mapped to one while even number is mapped to zero. At this stage, the retrieving algorithm finishes i.e. the embedded data has been retrieved completely [13].

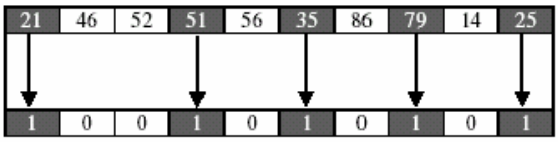

Fig. 3: The Retrieval Process.

III. Methods

In this section we discuss the proposed approach for hiding information within the spatial domain of the gray scale image. The proposed approach works by dividing the cover into blocks of equal sizes and then embeds the message in the edge of the block depending on the number of ones in left four bits of pixel. The proposed method consists of two main algorithms, the embedding algorithm which is needed by the sender and the extraction algorithm which used by the recipient in order to reed the message. The embedding algorithm proceeds in two steps:

Step 1: Determine the place of embedding in the image

1. Divide the cover into blocks of equal sizes 8x8.
2. Calculate the square root of median for each block.
   M=fix (sqrt (median))
3. Calculate the deference value for each two consecutive pixels p_i and p_{i+1} which is given by $d_i = |pi -(p_{i+1})|$.
4. IF $d_i >= M$ then embed Message in p_i and p_{i+1}
   (go to Step 2)

Step 2: Embedding algorithm:

1. Split each pixel into two equal parts (see Figure 4).
2. Count number of 1's in the most part and embed a secret message in the least part according to the corresponding number of bits in Table I.

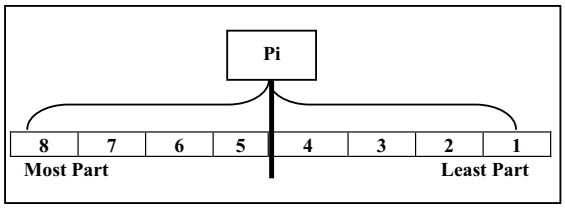

Fig. 4: Split Process.

TABLE I
OF 1'S IN THE MOST PART WITHIN EACH PIXEL.

# of 1's in the most part	# of Bits to be embedded
4 or 0	1 bit
2	2 bits
3 or 1	3 bits

The recipient uses the extraction algorithm in order to extract the secret message from the stego-image. Extracting secret message is done in the same way as in the embedded operation, depending on the value of the median:
M=fix (sqrt (median)). If the difference between each two adjacent pixels is more than the value of *M* then extract the message depending on the rule in Table 1.

IV. EXPERIMENTAL RESULTS

The proposed approach is tested on a database consists of 100 different images. In this section we present the experimental results of stego-image on three will known images: Lena, Baboon, and Pepper images. These images are shown in Figures 5-7. The quality of stego-image is still acceptable by human eyes. We present also a comparative study of the proposed methods with existing methods.

Cover image Stego-image Embedding data

Fig. 5: Lena Image (256x256)

Cover image Stego-image Embedding data

Fig. 6: Baboon Image (256x256)

Cover image Stego-image Embedding image

Fig. 7: Peppers Image (256x256)

We have analyzed our results according to PVD and GLM methods for each of the tested images. We also analyzed our results by computing Payload, Similarity measure and peak signal-to noise ratio (PSNR).

Payload: the size of date that could be imbedded within the cover image is shown in Table II.

TABLE II
MESSAGE SIZE FOR EMBEDDING.

Image	Image Size	Data Size (GLM)	Data Size (Proposed Algorithm)
Lena	128 x 128	2048	2493
	256 x256	8192	10007
	512 x512	32768	40017
	1024 x1024	131072	160604
Baboon	128 x 128	2048	2560
	256 x256	8192	10211
	512 x512	32768	40990
	1024 x1024	131072	163724
Peppers	128 x 128	2048	2443
	256 x256	8192	9767
	512 x512	32768	39034
	1024 x1024	131072	156308

It can be noticed from Table 2 that the proposed approach allows more data to be embedded than the GLM method. The experimental results showed that the PVD method is performed better than the proposed method in this respect.

Similarity measure: The similarity function used is:

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2}\sqrt{\sum ((y_i - \bar{y})^2}}$$

where:
X_i: the value of the pixels in the cover image
Y_i: the value of the pixels in the stego image
$\bar{x}$: the mean value of x_i.
$\bar{y}$: the mean value of y_i.

Table 3 shows the results of similarity function after embedding. After analyzing Table 3 we concluded that the proposed approach is performed better that the PVD method but not than the GLM method. It can be seen that the similarity ratio values were close to the GLM method. This can be noticed visually from Figures 8-10.

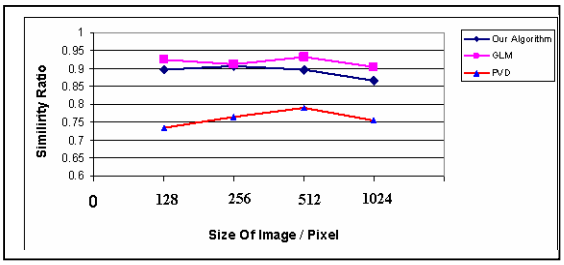

Fig. 8: Similarity Measure for Lena Image.

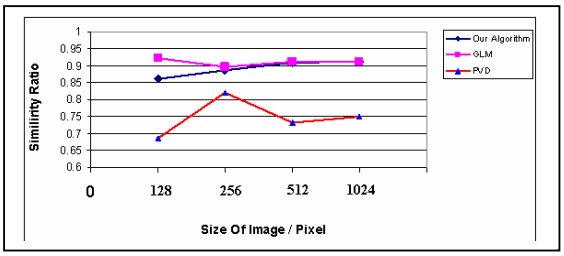

Fig. 9: Similarity Measure for Baboon Image.

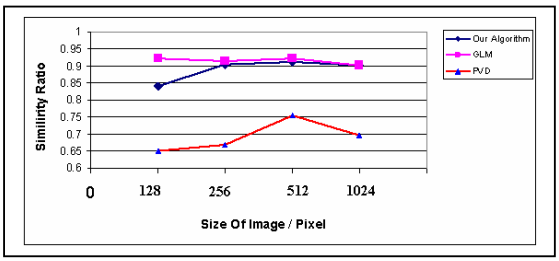

Fig. 10: Similarity Measure for Peppers Image.

Peak Signal -to Noise Ratio (PSNR):

PSNR measures the quality of the image by comparing the original image with the stego-image, i.e. it measures the percentage of the stegano data to the image percentage. The typical PSNR value is 45 [10]. We used the PSNR to evaluate the quality of the stego-image after embedding the secret message in the cover. The PSNR is computed using the following formulae:

$$MSE = \left[\frac{1}{N * N}\right]^2 \sum_{i=1}^{N}\sum_{j=1}^{N}(X_{ij} - \overline{X}_{ij})^2$$

$$PSNR = 10\log_{10}\frac{L^2}{MSE}db$$

Figures 11-13 show the PSNR after embedding using the GLM, PVD and the proposed algorithm. It can be seen from these Figures that the proposed method gave better results than other methods such that there is no difference between the original and the stegano-image.

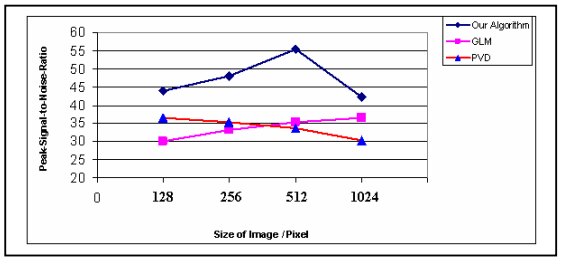

Fig. 11: PSNR for Lena Image.

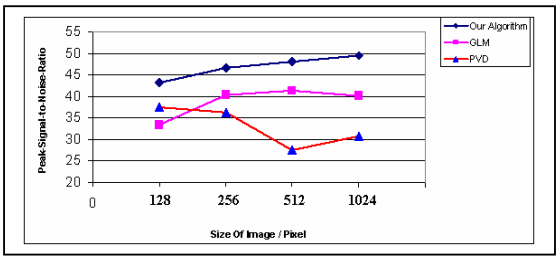

Fig. 12: PSNR for Baboon Image.

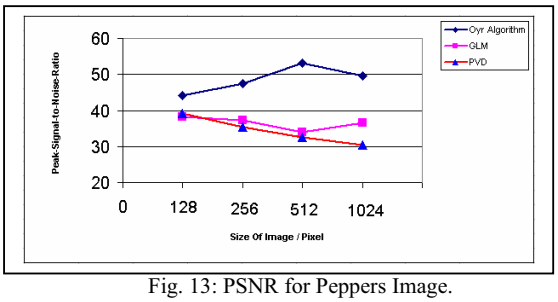

Fig. 13: PSNR for Peppers Image.

V. Conclusions And Future Work

Steganography can be used for a variety of reasons. Legitimate purposes include watermarking images for copyright protection. Digital watermarks are similar to Steganography in that they appear to be part of the original object and are not easily detectable by the casual eye. Steganography can also be used to tag notes to online images and is used to maintain the confidentiality of valuable information.

In this article we have proposed an efficient Steganography approach for hiding information within a gray scale image. We have compared the new method with two well-known methods, PVD and GLM methods. Experimental results showed the effectiveness of the proposed method compared with the other methods. In terms of data size PVD method was the best then the proposed method and GLM method was the last. The similarity measures proved that the GLM method was the best the proposed method gave close results to the GLM but the PVD method gave worst results.

Experimental results showed that the proposed method gave best values for the PSNR measure, which means that there is no difference between the original, and the stegano-images.

Future work will consider doing the following modifications to the proposed method:

1. Investigating the proposed method on color images.
2. modifying the proposed approach to embed image inside another image.
3. Preserve the secret message even if we do some transformations on the image like rotation, scaling compression.
4. Relate the encryption process with steganography in which we encrypt the message before embedding it inside the image in order to increase the security of the proposed method.

REFERENCES

[1] Pfitzmann, B, Information Hiding Terminology, Proceedings of the First International Workshop on Information Hiding, Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1996, 174 (1), 347-356.

[2] Johnson, N., and Jajodia, S., Exploring Steganography: seeing the unseen, IEEE Computer, 1998, 58(8), 26-34.

[3] Katzenbeisser, S., and Petitcolas, F., Information Hiding Techniques for Steganography and Digital Watermarking, Boston: Artech House.London, 2000.

[4] H. Noda, J. Spaulding, M. N. Shirazi, and E. Kawaguchi, Application of bit-plane decomposition steganography to JPEG2000 encoded images, IEEE Signal Processing Letters, 9(12), 410–413, Dec. 2002.

[5] D. C. Wu and W. H. Tsai, A steganographic method for images by pixel-value differencing, Pattern Recognition Letters, 24(9-10), pp.1613–1626, 2003.

[6] Xinpeng Zhang and Shuozhong Wang, Steganography Using Multiple-Base Notational System and Human Vision Sensitivity, IEEE Signal Processing Letters, 12(1), JANUARY 2005, 67.

[7] Ross, J., Fabien, A., on the limits of Steganography, IEEE Journal, 16(4): 474-481, 1998.

[8] S. Walton, "image authentication for a slippery new age," Dr. Dobb's Journal, vol. 20, no. 4, pp. 18 26, 1995.

[9] Na-I, W. A, Study On Data Hiding For Gray-Level And Binary Images, MSc Thesis, Graduate Institute of Networking and Computation Engineering, Chaoyang University of Technology, Taiwan, May 2004.

[10] S Craver, On Public-key Steganography in the Presence of an Active Warden, IBM Research Report RC 20931, 1997.

[11] E. Franz, A. Jerichow, S. Moller, A. Pftizmann, and I. Stierland, Computer Based Steganography in Information Hiding, Lecture Notes in Computer Science, Springer, 1996, Vol. 1174, 7-21.

[12] Alwan R. H., Kadhim F. J., and Al-Taani A. T., Data Embedding Based on Better Use of Bits in Image Pixels. International Journal of Signal Processing, 2 (1), 104-107, 2005.

[13] Potdar, V., and Chang, E., Gray Level Modification Steganography for Secret Communication. IEEE International Conference on Industrial Informatics, Berlin, Germany, 2004.

Ahmad T. Al-Taani is an associate professor of artificial intelligence at Yarmouk University, Jordan. He received his bachelor of science in computer science in 1985 from Yarmouk University, Jordan. He received his master of science in software engineering from National University, San Diego, California, U.S.A. in 1988. He received his PhD in computer vision from University of Dundee, U.K. in 1994. Dr. Al-Taani worked with the department of computer systems, Hail Community College, King Fahd University of Petroleum and Minerals, Saudi Arabia, from September 2000 to June 2003. He has several publications in the national and international journals and conference proceedings. He has also supervised a number of graduate students. His research interests includes: artificial intelligence applications, image processing, Arabic character recognition, direct machine translation, and Arabic web page classification. Dr. Al-Taani is a member in many societies and he is also a member of the Editorial Boards for many refereed journals. He is also a referee for many different national and international conferences.

Abdullah M. AL-Issa received his Master of Science in Computer Science from Al al-Bayt University, Jordan in 2006. He works as a Subject Matter Expert (SME) with the Integrated Technology Group (ITG), Amman, Jordan, His research interests include: steganography, watermarking, and E-learning.