Cairo University

Egyptian Informatics Journal

www.elsevier.com/locate/eij
www.sciencedirect.com

ORIGINAL ARTICLE

# Adaptive and non-adaptive data hiding methods for grayscale images based on modulus function

Najme Maleki, Mehrdad Jalali *, Majid Vafaei Jahan

*Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran*

Received 29 November 2013; revised 3 April 2014; accepted 2 June 2014
Available online 27 June 2014

**KEYWORDS**

Data hiding;
Non-adaptive steganography;
Adaptive steganography;
Modulus function;
Embedding capacity

**Abstract** This paper presents two adaptive and non-adaptive data hiding methods for grayscale images based on modulus function. Our adaptive scheme is based on the concept of human vision sensitivity, so the pixels in edge areas than to smooth areas can tolerate much more changes without making visible distortion for human eyes. In our adaptive scheme, the average differencing value of four neighborhood pixels into a block via a threshold secret key determines whether current block is located in edge or smooth area. Pixels in the edge areas are embedded by Q-bit of secret data with a larger value of Q than that of pixels placed in smooth areas. Also in this scholar, we represent one non-adaptive data hiding algorithm. Our non-adaptive scheme, via an error reduction procedure, produces a high visual quality for stego-image. The proposed schemes present several advantages. 1-of aspects the embedding capacity and visual quality of stego-image are scalable. In other words, the embedding rate as well as the image quality can be scaled for practical applications 2-the high embedding capacity with minimal visual distortion can be achieved, 3-our methods require little memory space for secret data embedding and extracting phases, 4-secret keys have used to protect of the embedded secret data. Thus, level of security is high, 5-the problem of overflow or underflow does not occur. Experimental results indicated that the proposed adaptive scheme significantly is superior to the currently existing scheme, in terms of stego-image visual quality, embedding capacity and level of security and also our non-adaptive method is better than other non-adaptive methods, in view of stego-image quality. Results show which our adaptive algorithm can resist against the RS steganalysis attack.

## 1. Introduction

Nowadays message transmission on the internet still has to face some problems such as data security, and copyright control. Therefore, we need secret communication schemes for transmitting message on the internet. Encryption may provide a safe way, which transforms data into a cipher-text via cipher algorithms. However, encryption makes the message unreadable, but making message suspicious enough to attract

* Corresponding author. Tel.: +98 9153143976.
E-mail addresses: mehrjalali@gmail.com, jalali@mshdiau.ac.ir (M. Jalali).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

Production and hosting by Elsevier

1110-8665 © 2014 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo University.
http://dx.doi.org/10.1016/j.eij.2014.06.001

eavesdropper's attention. For overcoming to this problem, we must utilize of data hiding techniques which hide the secret data behind a cover media. Steganography is a data hiding technique which embeds secret data into a cover media such as text, image, audio and video, so that the result does not notice any third's attention. In the past decade, steganography in image extremely has been studied. The image into which a massage is hidden is called a cover image and the result a stego-image. Application of the data hiding can be used in military, commercial and anti-criminal depended application, transmission of confidential documents between international governments and be anonymous in internet [1]. We can categorize steganography schemes into two types: non-adaptive and adaptive schemes. In non-adaptive schemes, the embedding capacity in each pixel of cover image is a fixed value without taking the image local texture into consideration. In other words, these methods do not consider the differences between adjacent pixels. Non-adaptive means that we did not consider the position, where each embedding change was done. In first category, a well-known steganographic method is the Least Significant Bit (LSB) substitution method, which embeds secret data by replacing k LSBs of a pixel with k secret bits directly [2]. Also for minimizing the image distortion, Chan–Cheng proposed a simple LSB algorithm based on optimal pixel adjustment [3]. In 2006, Zhang and Wang's algorithm [4] represented in $(2n + 1)$-ary notation system. In this method, just one pixel of n pixels into one group is increased or decreased by 1. Also in 2006, Mielikainen [5] proposed LSB matching algorithm for embedding secret message. Both of Zhang and Wang's and Mielikainen's schemes [4,5] had the limited capacity.

Since LSB-Based methods just modify the LSB of image pixels, that can be detected the present secret data easily by the proposed steganalysis algorithms, e.g. the well-known RS detector [6]. Hence, level of security in these methods is poor. Of other aspect, all Pixels in a cover image cannot tolerate equal values of changes without causing noticeable distortion. Such that, the 4-bit LSB methods can make the smooth areas of cover image very dirty. Thus, easily the hiding effects resulted from embedding notice by eavesdropper. Because, the changes occur in edge areas can be difficultly discerned to the human eyes, therefore adaptive methods for steganography have been presented [7–13] in which the amount of embedding data in pixels is variable. These schemes provide a more imperceptible result than those employed by simple LSB s substitution and other non-adaptive schemes. Wu–Tsai proposed a novel steganographic method that uses the different values between two neighboring pixels to estimate how many secret bits should be embedded [7]. Chang–Tseng used the side information of neighboring pixels for each input pixel to help the capacity estimation in edge and smooth areas [8]. In Zhang–Wang scheme, three neighbor pixels are employed to assess the size of secret message for each pixel in the original image [9]. Wu et al. proposed a novel steganographic method based on LSB substitution and PVD method. In their algorithm, the secret bits in pixels located into edge areas embed using of PVD algorithm and those in pixels placed into smooth areas, embed using 3-LSB substitution algorithm [10]. Yang–Weng proposed a multi-pixel differencing method that to determine how many secret bits should be embedded, that used of three different values in a four-pixel block [11]. For improving the stego-image quality in Wu–Tsai scheme, Wang et al. [12] presented a steganographic method which instead of the different values, that utilizes the remainder of two consecutive pixels to record the information of secret data [12]. Yang et al. [13] proposed an adaptive LSB steganographic method using the different values of two consecutive pixels based on k-bit modified LSB substitution method to discriminate between edge and smooth areas [13]. Weiqi and Sivaranjani in [14,15] have proposed the LSB matching revisited image steganography. In [14,15] for low capacities only edge regions of cover image have changed and the smooth regions remained constant and hence had preserved the statistical and visual features of cover image (because the regions located at the edges present more complicated statistical features and observation changes in these regions is hard and difficult). In Weiqi and Sivaranjani's schemes [14,15], the embedding regions can select according to the size of secret message and the difference between two corresponding pixel in the cover image. Those use the absolute difference between two adjacent pixels as the criterion for classification of edge and smooth regions of cover image. Manoj and others in [16] in 2011, represented an overview of image steganography and its applications with a basic image steganographic model. In this method [16], for maximizing the embedding capacity into each pixel, has been utilized of an adaptive encoding algorithm along with LSB insertion method. The method [16] used StegSan for implementation. Indeed, StegSan uses of an adaptive encoding algorithm to optimize the use of embedding space in a specific cover image. Using StegSan tool in [16] allowed the users to hide various large files so stego-image maintains good imperceptible. This technique has been implemented only on 4-LSB [16]. In [17] in 2011 presented an adaptive steganographic method based on just noticeable distortion (JND) profile measurement. To compute how much information can be embedded and also final value determination of the stego-pixel, different impact factors had used ((1) JND value of the target pixel, (2) a predefined embedding capacity control factor, (3) the contents of various length secret data bits). To preserve the visual features of cover image, method [17] like more adaptive methods embedded more secret data bits within edge areas.

In 2013 [18], is presented a new adaptive embedding scheme namely adaptive steganography by oracle (ASO). It is based on an oracle which is used to calculate the detectability map. This approach preserves both cover image and sender's database distributions during the embedding process, which improves the security. In addition, it offers to the sender the opportunity to choose the most reliable image(s), during his secret communication [18]. Also in 2013 [19], Yu and Wang represent an adaptive steganography algorithm in the sparse domain. Image blocks whose entropy is higher than the threshold Used in this paper, are selected for sparse decomposition as complex texture image regions. The secret message is embedded into the decomposition coefficients, and then the stego image is reconstructed with modified coefficients [19].

Three criteria are used to evaluate the performance of data hiding schemes: the embedding capacity, the visual quality of the stego-image and the security. However, existing data hiding schemes seldom consider all these factors in their methods. But in 2010, Lee and Chen [20] used a simple modulus function to imply all the performance factors listed above. However, in Lee–Chen scheme the embedding capacity into each image pixel was fixed and thus that was a non-adaptive method. In order to provide better stego-image quality, larger embedding capacity and increasing level of security, an adaptive method using neighborhood pixels differencing based on modulus function in [20] is presented in this paper. The average value of three different values in four-pixel into a block is utilized to distinguish between edge areas and smooth areas and to estimate how many secret bits will be embedded into the block. Readjust procedure will be applied to extract secret data exactly in destination and to minimize the hiding effects resulted from embedding in the embedding phase. Also a non-adaptive algorithm based on Lee–Chen's scheme [20] has represented. In this algorithm, 'error reducing procedure' causes to improve quality of the stego-image produced by Lee–Chen's.

The remainder of this paper is organized as follows. In Section 2, we represent three existing data hiding methods. In Section 3, the embedding and extracting algorithms of the proposed adaptive and non-adaptive methods are presented. In the next section, we compare the proposed schemes with Lee–Chen's [20]. The experimental results, comparisons between our methods with other adaptive and non-adaptive methods will be in Section 5. Finally, conclusions are given in Section 6.

## 2. Related work

We now describe briefly five existing data hiding schemes, namely Zhang and Wang's [4], Mielikainen's, [5], Wang et al.'s [12], Yang et al.'s [13] and Lee–Chen's [20].

In 2006, Zhang and Wang [4] represented one steganographic algorithms that used the $(2n + 1)$-ary notation system to embed each secret digit in a group with n pixels of cover image. The value $n$ is a parameter in their system [13]. This scheme exploits the modification direction which called the EMD method and each secret digit can have $(2n + 1)$ different values ($2n$ different styles of modification + the state in which no pixel is modified). For the embedding secret digit in EMD method, just one pixel of each pixel-group is increased or decreased by 1. In Zhang and Wang's algorithms, for each pixel-group of cover image is calculated a extraction function $f$ which is as a weighted sum $(g_i \times i,\ i = 1,2,\ldots,\mathrm{n})$ modulo $(2n + 1)$. Each secret digit in the $(2n + 1)$-ary notational system is Mapped to a pixel-group and that is not required for modifications, if secret digit $d$ is equal to $f$ of the cover pixel-group. But if $f \neq d$, then $m = d - f \bmod (2n + 1)$ is calculated. If $m \leqslant n$, then the value $g_m$ is increased by 1, otherwise the value of $g_{2n+1-m}$ is decreased by 1. (Suppose $g_1, g_2, \ldots g_n$ are the gray values of pixels in a group with $n$ pixel) [4].

In 2006, Mielikainen [5] has proposed a LSB matching scheme to the use of a binary function for hiding two bits of secret data within two pixels. Suppose $x_i$ and $x_{i+1}$ are a cover pixel pair and also $b_i$ and $b_{i+1}$ are two bits of secret data. In Mielikainen's scheme to embed secret message first, the value $LSB(\lfloor x_i/2 \rfloor, x_{i+1})$ is calculated and that is termed as function $f_l(x_i,x_{i+1})$. Then via comparing $b_i$ with $LSB(x_i)$ or comparison among $b_{i+1}$ with either $f_l(x_i,x_{i+1})$ or $f_l(x_{i-1},x_{i+1})$ is produced a series of the new embedding rules to determine stego values for a stego pixel pair $(x'_i, x'_{i+1})$. After the embedding of secret data in Mielikainen's algorithm, it may be even three cases for each of two pixels (increasing by 1, decreasing by 1 or no change). The maximum embedding rate of Mielikainen's scheme is only 1 bit for each pixel. Thus the embedding capacity of this scheme has limited [5].

In order to produce a better stego-image quality than to Wu–Tsai's [7], in 2008, Wang et al. [12] proposed a novel technique based on pixel-value difference and modulus function. For the embedding secret data, first a different value from two consecutive pixels and next via a modulus function, the reminder of the two consecutive pixels is computed. By using an original table rang, data can be embedded into the two pixels with altering their remainder. This method [12] significantly decreased the hiding effects appeared in the stego-image of Wu–Tsai's [7].

In Yang et al.'s scheme [13] the number of embedding bits is evaluated by the range in which different values of two consecutive pixels fall into. The range is divided into two levels, such as lower level and higher level. The embedding is performed by executing K-bit modified LSB substitution method, so that the value $k$ is decided via the level in which their different values belong to. The higher level used a larger value of $k$.

In Lee–Chen's scheme [20], the values R1, R2, v1, v2 are secret keys. Using two set-generation Hr(R1, v1) and Hc(R2, v2), are product two binary sets namely Kr and Kc with unique elements. The number of elements of Kr and Kc is $2^{\mathrm{v1}}$ and $2^{\mathrm{v2}}$, respectively. The numerical values of each element in Kr and Kc fall within the range $[0, 2^{\mathrm{v1}} - 1]$ and $[0, 2^{\mathrm{v2}} - 1]$, respectively where $\mathrm{R1} \in [1, 2^{\mathrm{v1}}!]$, $\mathrm{R2} \in [1, 2^{\mathrm{v2}}!]$ and $\mathrm{v1}, \mathrm{v2} \in \{0\} \cup N$. The bitstream secret data that denoted by S, are divided into many secret pieces $\mathrm{S_k}$, so that $\mathrm{S_k} = \mathrm{S_{k1}} | \mathrm{S_{k2}}$, where $\mathrm{S_{k1}}$ contains v1 bits and $\mathrm{S_{k2}}$ contains v2 bits and each pixel in cover image can carry (v1 + v2)-bit secret data. Via sets Kr and Kc can form a variant of a Cartesian product denoted as $\mathrm{Kr} \otimes \mathrm{Kc}$. Position $\mathrm{S_{k1}}$ into Kr (e.g., Kri) and position $\mathrm{S_{k2}}$ into Kc (e.g. Kcj) are determined. Next, index of the bitstream kri||kcj (e.g., $d$) into $\mathrm{Kr} \otimes \mathrm{Kc}$ is exploited. Next, for each cover pixel a pixel group $G = \{g_t | t = 1, 2, \ldots, n\}$ using a modulus operation is created, where $n = 2^{\mathrm{v1}+\mathrm{v2}}$. Than via index d, the secret piece $\mathrm{S_k}$ can be carried by the *dth* element in group G. Before the extracting process in the Lee–Chen's [20], firstly are generated two sets Kr and Kc using Hr(R1,v1) and Hc(R2,v2), respectively. This step is the same with the embedding process. Next, for each stego pixel create the pixel group G and determine the position information d so that the value of stego-pixel identical with $g_d$. Retrieve the *dth* element, which is the secret piece with (v1 + v2) bits, from $\mathrm{Kr} \otimes \mathrm{Kc}$. Repeat before steps until all the stego-pixels have been processed. Finally, concatenate all the pieces of secret data and return the secret information [20].

Lee–Chen's scheme [20] is better than of Zhang and Wang's and Mielikainen's schemes [4,5] of aspect security and capacity. Embedding rate in both Zhang and Wang's and Mielikainen's schemes are limited. Also the visual quality of the generated stego-image by Wang et al. [12] is better than other schemes [7,8]. But, the embedding algorithm of Wang et al.'s [12] needs extra steps for revising pixel values, when the problem of overflow and underflow occurs. Yang et al.'s scheme [13] produces the smallest distortion in the LSB-related embedding approaches. But both these methods [12,13] and also methods in [7] and [10] could not consider sufficiently the features of edge. Besides, the level of security in other methods [7,8,10,11] was in poor degree. Furthermore, security level in Lee–Chen's scheme [20] is high and detecting secret data for eavesdropper, because generating many permutations in producing sets Kr and Kc is difficult. Hence with the reasons mentioned above, we improved our non-adaptive and adaptive methods on Lee–Chen's algorithm [20]. But in Lee–Chen's scheme, the embedding capacity into each image pixel is fixed and thus Lee–Chen's scheme is a non-adaptive method. In order to provide better stego-image quality, larger embedding capacity and increasing level of security, an adaptive method with the high security, using neighborhood pixels differencing based on modulus function [20] is presented in this paper. Also, in order to improve visual quality of the produced stego-image by Lee–Chen's scheme, a non-adaptive algorithm based on Lee–Chen's algorithm [20] is used.

## 3. The proposed schemes

We conducted our methods based on Lee–Chen 2010 method [20]. The cover images have selected 8-bit grayscale images because, based on psycho visual redundancy in grayscale digital images, the pixels in edge areas than to smooth areas can tolerate much more changes without making perceptible distortion for human eyes and also a grayscale image needs lower space and time for transmission on the internet than to colored images. In our proposed methods, a cover image with size $M \times N$ is indicated as $I$ and each cover pixel is denoted as $y_i$. The bitstream secret message is denoted by S. The stego-image is denoted as $I'$, and $y''_i$ represents each stego-pixel. In our methods, firstly both two sets Kr and Kc are formed. Two set-generation functions Hr(R1,v1) and Hc(R2,v2) are used to generate two binary sets Kr = {Kri|$i$ = 1,2,…,$2^{v1}$} and Kc = {Kcj|$j$ = 1,2,…,$2^{v2}$}, where R1 ∊ [1,$2^{v1}$!], R2 ∊ [1,$2^{v2}$!]. Each binary element Kri in Kr is unique and its numerical value falls within the range $[0, 2^{v1} - 1]$. Similarly, each binary element Kcj in Kc is unique and its numerical value falls within the range $[0, 2^{v2} - 1]$. Kr and Kc have $2^{v1}$! and $2^{v2}$! possible permutations, respectively. The proposed adaptive algorithm is explained in Section 3.1 and the proposed non-adaptive algorithm is represented in Section 3.2.

### 3.1. The proposed adaptive scheme

There are five secret keys namely R1, R2, v1, v2, $T$ and $1 \leqslant v1$, $1 \leqslant v2$, $(v1 + v2) < 6$. The average different values of a four-pixel block are utilized to classify the block as a smooth area or an edge area. The range of average different value is partitioned into two different levels, smooth level and edge level. $Q$-bit of the secret data is embedded in Pixels located in the block, where $Q$ is decided by the level in which the average different values belong to. In the embedding process of secret data, according to the secret keys v1 and v2, the smooth level will use lower value v1 while the edge level uses greater value v1 + v2. The data embedding process is given in Section 3.1.1 and the extracting phase is described in Section 3.1.2.

#### 3.1.1. The embedding phase in proposed adaptive scheme

The cover image is partitioned into nonoverlapping four-pixel blocks. For each block, there are four neighboring pixels $P_{i,j}$, $P_{i,j+1}$, $P_{i+1,j}$, $P_{i+1,j+1}$ and their corresponding gray values are $y_0$, $y_1$,$y_2$,$y_3$, respectively. The detailed embedding steps are as follows.

| | |
|---|---|
| Input: | $I$, S, secret keys R1, R2, v1, v2 and $T$ |
| Output: | $I'$ |

Step 1: The same as what has been explained in Section 3, generate two sets Kr and Kc using Hr(R1,v1) and Hc(R2,v2), respectively. via sets Kr and Kc form a variant of a Cartesian product namely, Kr ⊗ Kc. Set Kr ⊗ Kc generates an ordered set of combinations of Kr and Kc with $2^{v1} \times 2^{v2} = 2^{v1+v2}$ elements (Eq. (1)). Each element of the variant Cartesian product of the two sets Kr and Kc is a binary string concatenation that combines the two binary strings of Kri and Kcj together to form one bitstream: Kri||Kcj and each element Kri||Kcj has a length of (v1 + v2) bits.

$$Kr \otimes Kc = \{Kri||Kcj|Kri \in Kr, Kcj \in Kc, i = 1, 2\dots, 2^{v1}, \\ = 1, 2, \dots, 2^{v2}\} \tag{1}$$

Step 2: Calculate the average different value $D$, which is determined by:

$$D = \tfrac{1}{3}\sum_{i=0}^{3}(y_i - y_{\min}) \\ y_{\min} = \min\{y_0, y_1, y_2, y_3\} \tag{2}$$

Step 3: Our method using threshold key value $T$ embeds secret data into two levels: the smooth-level and the edge-level. Addition to v1 and v2 keys, $T$ stands for a predefined threshold that can be used to control image distortion and the embedding rate. If $D \leqslant T$, $D$ belongs to 'smooth-level' and the block belongs to a smooth area, then $Q$ = v1. Otherwise, $D$ belongs to 'edge-level' and the block belongs to an edge area, then $Q$ = v1 + v2. We must satisfy the following conditions: $2v1 \leqslant T \leqslant 2v1 + v2$ and $1 \leqslant v1$, $(v1 + v2) < 6$.
Step 4: Determine whether current block belongs to 'Error Block'. If is, restart from Step2. Otherwise, continue to next step.

(*Definition* 1 Let $y_{\max}$ = max {$y_0$, $y_1$, $y_2$, $y_3$}, the block is called 'Error Block' if and only if: $D \leqslant T$, $(y_{\max} - y_{\min}) > 2 \times T + 2$. 'Error Block' is not used to embed secret bits).

Step 5: For each pixel $y_i$ in the block, according to level of that block, separated $S_Q$ = Q bits of secret data. For edge blocks, $S_Q$ divide into two pieces $S_{Q1}$ and $S_{Q2}$, where $S_{Q1}$ contains v1 bits and $S_{Q2}$ contains v2 bits. For smooth blocks, $S_Q$ divide into one piece $S_{Q1}$, where contains the same v1 bits.
Step 6: For edge blocks obtained the indices i and j using the conditions $S_{Q1}$ = Kri and $S_{Q2}$ = Kcj and for smooth blocks determine index i using the condition $S_{Q1}$ = Kri. (Kri and Kcj are *ith* and *jth* elements into Kr and Kc, respectively).
Step 7: For edge areas, bitstream Kri||Kcj into Kr ⊗ Kc can be indexed by Eq. (3) and for smooth blocks, bitstream Kri can be indexed by Eq. (4).

$$d = 2^{v2} \times (i - 1) + j \tag{3}$$

$$d = i \tag{4}$$

Step 8: Create a pixel group G using the following equation ($n = 2^{\mathrm{Q}}$).

$$f(y_i) = y_i \mathrm{mod} n \tag{5}$$

If α be the result of Eq. (5), then the pixel group $G = \{g_t|$-$t = 1,2,\ldots,n\}$ is an ordered set such as: $\{y_i - \alpha,\ y_i$ - $- \alpha + 1,\ldots,y_i,\ \ y_i + 1,\ldots,y_i + n - \alpha - 1\}$. Then derive the corresponding stego-pixel $y'_i$ from *dth* element of *G*: $y'_i = g_d$.

Step 9: his step is called '*error reducing procedure'* for minimizing perceptual distortion between cover and stego images. Also this step called *'readjust procedure'* to guarantee the same level that the average differencing value belongs to before and after secret data embedding. Hence, this step is necessary to extract secret data exactly in destination.

Let $y''_i = y'_i + L \times n,\ \ n = 2^{\mathrm{Q}},\ \ \mathrm{L} \in \{0, 1, -1\},\ \ 0 \leqslant i \leqslant 3$. Search ($y''_0$, $y''_1$, $y''_2$, $y''_3$) such that:

(1) $D''$ and $D$ belong to the same level, where:

$$D'' = \tfrac{1}{3}\sum_{i=0}^{3}(y''_i - y''_{\min}) \\ y''_{\min} = \min\{y''_0, y''_1, y''_2, y''_3\} \tag{6}$$

(2) The value of is $\sum_{i=0}^{3}(y''_i - y_i)^2$ minimized.
(3) The final stego-block ($y''_0$, $y''_1$, $y''_2$, $y''_3$) does not belong to 'Error Block'.
(4) After the replacement of ($y_0$, $y_1$, $y_2$, $y_3$) by ($y''_0$, $y''_1$, $y''_2$, $y''_3$) embedded 4 × Q-bit of secret data in the block.

Step 10: Repeat Steps 2–9 until the secret pieces have been embedded and obtained the stego-image *I'*.

*For instance*, we present one example with high embedding capacity which embeds 3-bit of secret data within each pixel in smooth area and 5-bit within each pixel in edge area. Suppose we have a block with four neighboring pixel values (155, 99, 184, and 140), and the secret data for embedding in cover image are '10001011000011011010'. Assume v1 = 3, v2 = 2, R1 = 30,301, R2 = 20 and *T* = 20. Before the embedding process, firstly Kr = {001, 110, 101, 010, 111, 100, 011, 000} can be generated using Hr(30301,3) and Kc = {00, 10, 11, 01} can be created using Hc(20,2). We calculate the average different values $D = (182/3) > T$, and thus current block has been placed in edge area and is embedded $Q = 5$ bits of secret data in each $y_i$, $0 \leqslant i \leqslant 3$ because v1 + v2 = 5. Hence, sum total 4 × 5 = 20 bits are embedded in the block. Next we separated four pieces containing 5-bit of secret data. Each 5-bit piece is further separated into two substrings: the 3-bit and the 2-bit substring, respectively. For first pixel into the block, e.g. $y_0 = 155$, the first piece '10001' is separated into the two substrings '100' and '01'. Then, we achieve $i = 6$ and $j = 4$ because the sixth element of Kr is '100' and fourth element of Kc is '01'. According to Eq. (3), we compute d using $2^2 \times (6–1) + 4 = 24$. Next, the pixel group G is created for the pixel value $y_0 = 155$ with $n = 2^{2+3} = 32$ via Eq. (3), as shown in Fig. 1, where $g_{28} = 155$. Finally, the stego-pixel $y'_0$ can be obtained from the *dth* element of G, i.e. $y'_0 = g_{24} = 151$. In the same way, can be obtained reminder the stego-pixel $y'_1 = 120$, $y'_2 = 161$, $y'_3 = 133$ and stego-block (151, 120, 161, 133). Readjust procedure is executed resulting in final stego-block (151, 88, 193, and 133). The average different values for this block obtained using $D = (213/3) > T$. Hence final stego-block not only has the equal level to cover block level but also has been minimized differences between cover and stego pixels. As an another example, with the same given information of before example, suppose cover block is (70, 79, 109, 106). We obtained $D = 28$. Then stego-block is produced (87, 88, 97, 101) and $D = 8$. After executing readjust process, result (55, 88, 97, 101) and $D = 40$. Thus the final stego-block will have the same level with cover block level.

### 3.1.2. The extracting phase in proposed adaptive scheme

Like the embedding process, Partition the stego-image into four-pixel blocks. The following steps are executed to extract the secret data.

| | |
|---|---|
| Input | A stego-image *I'*, secret keys v1, v2, R1, R2 and *T* |
| Output | A bitstream secret data |

Step 1: Generate two sets Kr and Kc using Hr(R1,v1) and Hc(R2,v2), respectively. We use the Cartesian product of Kr and Kc e.g. Kr ⊗ Kc, for blocks placed in edge areas and of Kr for blocks located in smooth areas.
Step 2: For each block ($P_{i,j}, P_{i,j+1}, P_{i+1,j}, P_{i+1,j+1}$), calculate the average different values D by Eq. (2).
Step 3: Use the threshold value *T* to figure out the level which D belongs to. If $D \leqslant T$ (e.g. smooth area) then $Q = $ v1, otherwise $Q = $ v1 + v2.
Step 4: Determine whether current block belongs to 'Error Block'. If not, continue to next step. Otherwise, restart from Step 2.
Step 5: For each pixel into the block create the pixel group G using Eq. (5) and determine the position information *d* because the stego-pixel $y''_i = g_d$, where $n = 2^{\mathrm{Q}}$.
Step 6: Extract the *dth* element, which is the secret piece with $Q = $ v1 + v2 bits, from Kr ⊗ Kc for the blocks placed into edge areas and the secret piece with $Q = $ v1 bits, from Kr for the blocks placed into smooth areas.
Step 7: Repeat Steps 2–6 until all the stego-blocks have been visited and then concatenate all the pieces of secret data.

*For instance*, we extract the embedding example (151, 88, 193, and 133), which is shown in the before subsection. Assume v1 = 3, v2 = 2, R1 = 30,301, R2 = 20 and *T* = 20. Kr = {001, 110, 101, 010, 111, 100, 011, 000} using Hr(30301,3) and Kc = {00, 10, 11, 01} by using Hc(20,2) are generated. We produce the variant Cartesian product Kr ⊗ Kc, which is {00100, 00110, 00111, 00101, 11000, 11010, 11011,…, 00011, 00001}. Because $D = (213/3) > T$, this block is placed in edge area and hence $Q = $ v1 + v2 = 3 + 2 = 5 bits have hided into each Pixel in the block. Sum total, 4 × 5 = 20 bits are embedded in current block. Let us consider third pixel into the block (e.g. $y''_2 = 193$). The pixel group G is created for

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 128 | 129 | 130 | 131 | 132 | 133 | 134 | 135 | 136 | 137 | 138 | 139 | 140 | 141 | 142 | 143 |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
| 144 | 145 | 146 | 147 | 148 | 149 | 150 | *151* | 152 | 153 | 154 | 155 | 156 | 157 | 158 | 159 |
| 17 | 18 | 19 | 20 | 21 | 22 | 23 | *24* | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |

**Figure 1** Pixel group created for value 155 for embedding secret data.

value 193 via Eq. (5) with $n = 2^{2+3} = 32$. The position of stego-pixel 193 in G is 2, because $d = (193 \bmod 32) + 1 = 2$. The piece of binary secret data '00110' can be extracted because '00110' is the second element of Kr ⊗ Kc. Similarly, has extracted the secret piece '10001' for $y''_0$, '01100' for $y''_1$ and '11010' for $y''_3$. Finally we achieve '10001011000011011010' which is the same secret data in the embedding example of before subsection.

### 3.2. The proposed non-adaptive scheme

In our non-adaptive algorithm, like Lee–Chen's [20], there are 4 secret keys namely v1, v2, R1, R2 so that v1,v2 $\in \{0\} \cup N$ ($N$ is Natural number sets). In our non-adaptive method, each pixel of cover image carries $Q$ = v1 + v2-bit of secret data. The embedding and extracting steps have been explained in Sections 3.2.1 and 3.2.2, respectively.

#### 3.2.1. The embedding phase in proposed non-adaptive scheme

| | |
|---|---|
| Input | $I$, S, secret keys R1, R2, v1 and v2 |
| Output | $I'$ |

Note that Eqs. (1), (3), and (5) in the below algorithm, have formerly been mentioned in the proposed adaptive scheme.

Step 1: Generate two sets Kr and Kc using Hr(R1,v1) and Hc(R2,v2), respectively. Then, like step 1 in our adaptive scheme, for sets Kr and Kc obtain a variant of a Cartesian product e.g. Kr ⊗ Kc, (Eq. (1)).
Step 2: According to keys v1 and v2, for each pixel $y_i$ in cover image separate $S_Q$ = (v1 + v2) bits of secret data. Then, $S_Q$ divides into two pieces $S_{Q1}$ and $S_{Q2}$, where $S_{Q1}$ contains v1 bits and $S_{Q2}$ contains v2 bits.
Step 3: Obtain the indices i and j using the conditions $S_{Q1}$ = Kri and $S_{Q2}$ = Kcj.
Step 4: Calculate the index $d$ into set Kr ⊗ Kc, using Eq. (3).
Step 5: Create a pixel group G using Eq. (5) and similar to step 8 of our adaptive method, where $n = 2^Q$ and $Q$ = v1 + v2. Then derive the corresponding stego-pixel $y'_i$ from *dth* element of $G$: $y'_i = g_d$.
Step 6: This step is called '*error reducing procedure'* for minimizing difference between cover and stego images. Let $y''_i = y'_i + L \times n$, $n = 2^Q$, $L \in \{0,1,-1\}$. Search $y''_i$, so that the value of $(y''_i - y_i)^2$ is minimized. After the replacement of $y_i$ by $y''_i$, has embedded Q-bit of secret data in the cover pixel.
Step 7: Repeat Steps 2–6 until the secret data have been embedded into all pixels and obtain the stego-image $I'$.

*For instance,* suppose v1 = 2, v2 = 2, R1 = 2, R2 = 24. Sets Kr and Kc have formed so that: Kr = {00,11,10,01} and Kc = {11,01,00,10}. Two cover pixels are 15 and 241 and secret data for the embedding are: '00010101'. Thus, each cover pixel carries v1 + v2 = 4 bits. Table 1 describes the embedding steps in detail, where $Q$ = v1 + v2 = 2 + 2 = 4, $n = 2^Q = 16$.

#### 3.2.2. The extracting phase in proposed non-adaptive scheme

| | |
|---|---|
| Input | a stego-image $I'$, secret keys v1, v2, R1 and R2 |
| Output | a bitstream secret data |

Note that Eq. (5) in the below algorithm, have formerly been mentioned in the proposed adaptive scheme.

Step 1: Generate two sets Kr and Kc using Hr(R1,v1) and Hc(R2,v2), respectively.
Step 2: Create the pixel group G using Eq. (5) where $n = 2^Q$ and $Q$ = v1 + v2. Determine the position information $d$ because the stego-pixel $y''_i = g_d$.
Step 3: Extract the *dth* element from set Kr ⊗ Kc, which is the secret piece with v1 + v2 bits.
Step 4: Repeat Steps 2 and 3 until all the stego-pixels have been visited and then concatenate all the pieces of secret data.

*For instance,* we extract the embedding example in our non-adaptive algorithm. The stego-pixels are 17 and 237. Set Kr ⊗ Kc is: {0011,0001,0000,0010,1111,1101,1100,...,0101, 0100,0110} and has $Q$ = v1 + v2 = 2 + 2 = 4, $n = 2^Q = 16$. Table 2 describes the extracting steps in detail. The last column of Table 2 is *dth* element of Kr ⊗ Kc.

## 4. Analysis and discussion

Generally, suppose $a,b \in \{0\} \cup N$ and $[(a \bmod b) = x]$, ($N$ is Natural number sets). The following equation is verified in each division:

$$(a - b)\bmod b = a\bmod b = (a + b)\bmod b = x \tag{7}$$

Therefore, according to used modulus function and without loss of generality of extracting phase in Lee–Chen scheme

**Table 1** Example details of the embedding steps in our non-adaptive scheme.

| Pixel | 4-bit | 2-bit | 2-bit | $d$ | $y'_i$ | (stego-pixel) $y''_i$ |
|---|---|---|---|---|---|---|
| 15 | 0001 | 00 | 01 | 2 | 1 | 1 + 16 = 17 |
| 241 | 0101 | 01 | 01 | 14 | 253 | 253–16 = 237 |

**Table 2** Example details of the extracting steps in our non-adaptive scheme.

| (stego-pixel) $y''_i$ | $d$ | The extracted secret section |
|---|---|---|
| 17 | (17 mod 16) + 1 = 2 | 0001 |
| 237 | (237 mod 16) + 1 = 14 | 0101 |

[20], 'readjust procedure' in our adaptive algorithm and 'error reducing procedure' in our non-adaptive algorithm can work correctly. In other words, these procedures do not change the embedded secret data. Because, according to Eq. (7) and step 5 of the extracting process in our adaptive algorithm and step 2 of the extracting process in our non-adaptive algorithm, each of three values ($y'_i - 2^Q$), ($y'_i$) and ($y'_i + 2^Q$) has the same reminder to $n = 2^Q$. Thus in destination, each of these three values causes to extract the same secret data. We now compare the proposed adaptive and non-adaptive schemes in this scholar with Lee–Chen's [20]. All these methods possess following common factors:

(1) Need little memory space. Only ($\text{v1} \times 2^{\text{v1}} + \text{v2} \times 2^{\text{v2}}$) bits of memory space are required for storing Kr and Kc.
(2) The problem of overflow or underflow does not occur, regardless of the nature of the cover pixels. Because, let us assume that the pixel intensity set is $\lambda = \{0, 1, 2, 3, \ldots, 255\}$ and is an ordered set of pixel values which dominates the pixel values of a 8-bit gray-scale image. But for each the produced pixel group G from each cover pixel has: $G \subseteq \lambda$ and each element of $\lambda$ falls into the range [0–255]. It must note which in the 'readjust procedure' of the proposed adaptive method and also in the 'error reducing procedure' in the proposed non-adaptive method, for preventing of overflow or underflow problem, decrease of $y'_i$ by n and increase of $y'_i$ by n may not be allowed if $y'_i < n$ and $y'_i > (255 - n)$, respectively. Hence in both these procedures, the problem of overflow and underflow will not be occurred.
(3) Detecting secret data is difficult. Because of existing many permutations ($2^{\text{v1}}$! for set Kr, $2^{\text{v2}}$! for set Kc and totally, $2^{\text{v1}}! \times 2^{\text{v2}}!$ for $\text{Kr} \otimes \text{Kc}$), an unauthorized user will face extreme difficulty in guessing the secret data.

But, our adaptive method is superior to Lee–Chen's scheme. Because, firstly the embedding capacity in Lee–Chen's scheme and also in our non-adaptive scheme, just via v1, v2 keys is scalable. A larger v1 or v2 can yield a greater embedding capacity and a smaller v1 or v2 can obtain a higher visual quality of stego-image. However, in Lee–Chen's non-adaptive scheme after determining v1 and v2 keys only could achieve to fixed embedding capacity, e.g. $M \times N \times (\text{v1} + \text{v2})$. But, in our adaptive and flexible method after adjusting v1 and v2, by means of various values of key T, the embedding rate as well as the image quality can be adjusted depending on the requirements of the practical applications. Accordingly, a larger v1 or v2 and a lower $T$ enhance the embedding rate whereas a lower v1 or v2 and a higher $T$ enhance stego-image quality. Because in predetermined values for v1 and v2 and according to step 3 in the embedding process of the proposed adaptive algorithm, a larger value $T$ causes more blocks locate in smooth areas and accordingly embed lower bits of secret data into pixels and enhances visual quality, whereas a lower $T$ presents the opposite. Consequently, our method is more scalable than Lee–Chen's [20].

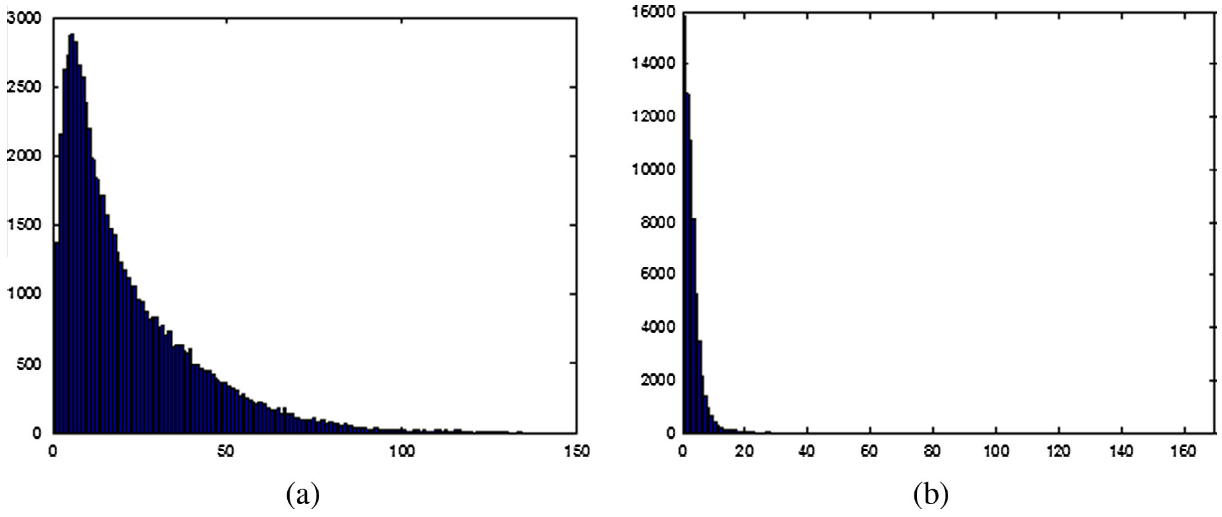
Secondly, the security in a data hiding system has provided via secret keys. In other words, extracting the secret data will be meaningless without knowing correct values of keys. The receiver must have the same set-generation functions Hr() and Hc() and appreciates the values of the secret keys v1, v2, R1, R2, T. Additionally, because of adding secret key T, the security of our method has been increased than Lee–Chen's scheme and hence detecting secret data will be more difficult. Thirdly, as the experimental results indicate, our adaptive method is better than Lee–Chen's scheme in terms of both embedding capacity and PSNR value.

In our adaptive scheme, has considered a block with $2 \times 2$ pixels due to a block of $2 \times 2$ pixels is neither too small nor too large to reflect the local complexity of an image. Also using a block with a larger size may increase the probability of degree revision and hence increases the distortion produced due to hidden data.

Hence, we have calculated the average different values into a block with $2 \times 2$ pixels. Fig. 2 indicates the histogram distributions of the average different values for two methods the complex Baboon image and the smooth splash image. The 'Baboon' image is complex, e.g. it has many edges, because the histogram distribution is scattered to a wide range of values. But the 'Splash' image is a smooth image, because the histogram distribution is limited in a narrow range. In our method, 'Error Block' is not used to embed secret bits [21]. Generally, there are significantly few error blocks in a cover image. So it will have a little effect on the capacity of our method, which can be almost ignored [21].

For example, let $T = 5$, a block with four-pixel values (178, 193, 178, and 178) belongs to *'Error Block'* because, $D = (15/3) \leqslant T = 5$ and $193–178 = 15 > 2 \times 5 + 2$. Suppose $\text{v1} = 2$, $\text{v2} = 2$ and sets Kr and Kc are the same with that example in our non-adaptive algorithm (Kr = {00,11,10,01}). Secret data are as follows: '01010010'. Each pixel into block carries $Q = \text{v1} = 2$ bits, because that places in a smooth area. In first, we generate stego-image (179, 195, 176, and 178) with $D = (24/3) > T = 5$. But the average different values in each of 81 choices of readjust process, do not be lower or equal to $T$ and following that, secret data are extracted falsely in destination. Hence, the error block is not used to embed secret data.

Also, as the comparison results show, our non-adaptive scheme is superior to Lee–Chen's scheme [20] in terms of stego-image visual quality. The error reducing procedure in our non-adaptive algorithm causes to minimum numeric distance between cover pixel values of the corresponding stego-pixel values. For justification of the error reducing procedure must explain which let ($y'_i - y_i = \text{dif}$) and [$\text{dif} > 2^{Q-1}$]. If ($\text{dif} > 0$) then $y''_i = y'_i - 2^Q$ and if ($\text{dif} < 0$) then $y''_i = y'_i + 2^Q$. Finally, if ($\text{dif} \leqslant 2^{Q-1}$) then for both ($\text{dif} > 0$) and ($\text{dif} < 0$) have: $y''_i = y'_i$. The worst case in Lee–Chen's



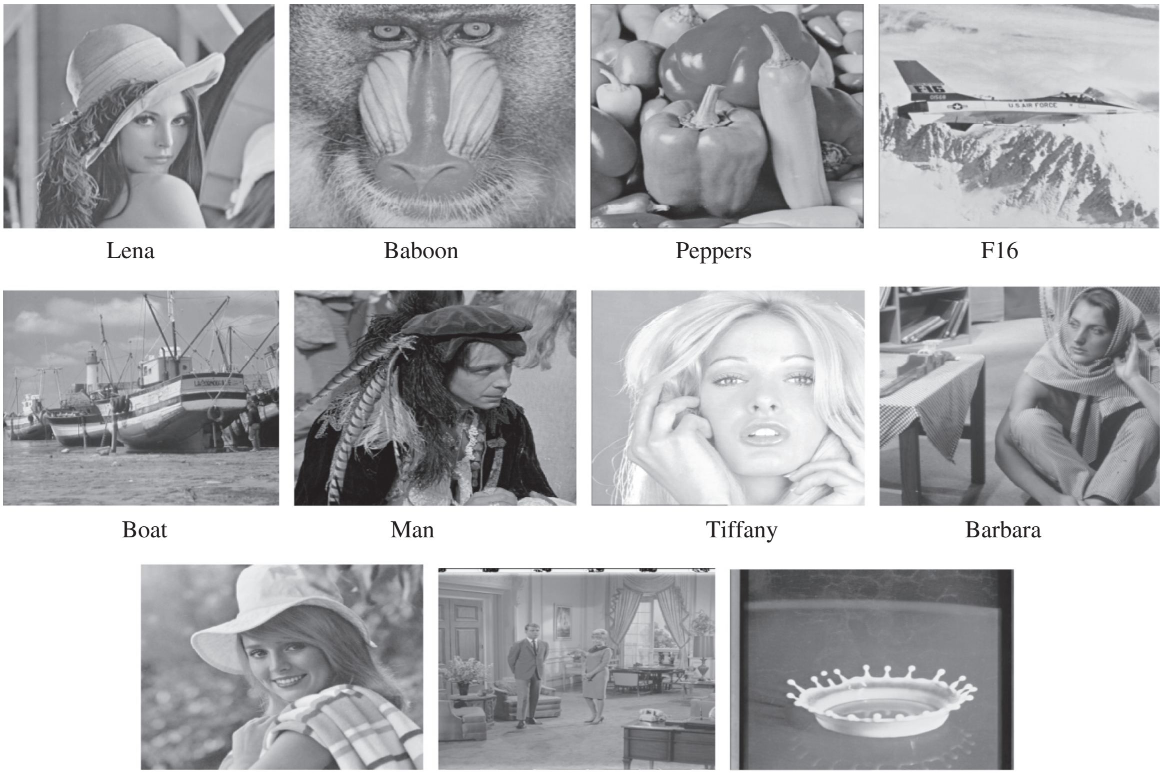
(a) (b)

**Figure 2** Histograms of block complexity for two cover images. (a: Baboon), (b: Splash). The $x$-axis depicts the average different values, while the $y$-axis indicates the relative numbers of the average different values.

algorithm [20] for pixel modification is $2^Q - 1$, while the worst case in the proposed non-adaptive algorithm, according to my explanation, is $2^{Q-1}$. Thus, the produced stego-image by our scheme is more similar to cover image and visual quality of the stego-image enhances.

## 5. Experimental results

Several experiments are performed to evaluate our proposed methods. Eleven grayscale images with size 512 × 512 are used in the experiments as cover images namely ‘Lena’, ’Baboon’, ’Peppers’, ’F16’, ’Boat’, ’Man’, ’Tiffany’, ’Barbara’, ’Elaine’, ’Couple’, ’Splash’, and are shown in Fig. 3. The proposed schemes have been implemented using the MATLAB 7.8.0.347 (R2009a) program on Windows XP platform. We used a series of pseudo-random numbers as the secret data to be embedded into the cover images and also utilized the peak signal-to-noise ratio (PSNR) value to evaluate the stego-images quality. The PSNR is defined as follows.



Lena Baboon Peppers F16

Boat Man Tiffany Barbara

Elaine Couple Splash

**Figure 3** Eleven cover images are used for the proposed schemes.

**Table 3** Experimental results with various parameters in our adaptive method (a and b).

| Cover images | 1–2, $T = 3$ | | 2–3, $T = 7$ | | 2–4, $T = 12$ | | 3–4, $T = 15$ | |
|---|---|---|---|---|---|---|---|---|
| | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR |
| *Panel a* | | | | | | | | |
| Lena | 412,416 | 47.74 | 591,544 | 43.92 | 595,120 | 41.63 | 812,484 | 39.50 |
| Baboon | 506,888 | 46.55 | 723,808 | 41.43 | 830,432 | 36.75 | 916,316 | 36.62 |
| Peppers | 415,480 | 47.56 | 573,000 | 44.47 | 580,208 | 42.34 | 808,024 | 39.69 |
| F16 | 468,053 | 46.98 | 586,244 | 44.12 | 603,256 | 41.43 | 817,948 | 39.31 |
| Boat | 457,536 | 47.06 | 641,828 | 42.71 | 646,400 | 40.00 | 830,332 | 38.80 |
| Man | 457,536 | 47.06 | 641,592 | 42.59 | 660,840 | 39.42 | 838,032 | 38.17 |
| Tiffany | 419,212 | 47.61 | 583,573 | 44.16 | 588,056 | 41.95 | 809,228 | 39.62 |
| Barbara | 446,016 | 47.27 | 648,980 | 42.71 | 710,760 | 38.64 | 873,100 | 37.62 |
| Elaine | 477,664 | 46.87 | 653,164 | 42.47 | 633,656 | 40.16 | 817,068 | 39.18 |
| Couple | 432,520 | 47.44 | 617,508 | 43.27 | 631,176 | 40.38 | 826,428 | 38.89 |
| Splash | 364,397 | 48.51 | 549,905 | 45.22 | 546,080 | 44.39 | 795,172 | 40.28 |

| Cover images | 2–5, $T = 19$ | | 3–5, $T = 21$ | | 4–5, $T = 28$ | |
|---|---|---|---|---|---|---|
| | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR |
| *Panel b* | | | | | | |
| Lena | 580,032 | 39.07 | 817,660 | 37.75 | 1,057,480 | 34.30 |
| Baboon | 852,068 | 32.22 | 984,400 | 32.29 | 1,118,320 | 31.93 |
| Peppers | 574,012 | 39.34 | 815,792 | 37.80 | 1,058,764 | 34.20 |
| F16 | 599,944 | 38.05 | 831,536 | 36.98 | 1,064,080 | 34.00 |
| Boat | 619,204 | 37.13 | 840,876 | 36.48 | 1,064,940 | 33.94 |
| Man | 639,676 | 36.08 | 852,696 | 35.47 | 1,068,640 | 33.02 |
| Tiffany | 572,600 | 39.62 | 813,700 | 38.09 | 1,056,668 | 34.43 |
| Barbara | 724,024 | 34.29 | 906,944 | 34.11 | 1,091,152 | 32.85 |
| Elaine | 569,500 | 39.56 | 808,212 | 38.39 | 1,053,116 | 34.53 |
| Couple | 610,184 | 37.34 | 834,704 | 36.60 | 1,061,764 | 33.95 |
| Splash | 545,980 | 42.26 | 799,708 | 39.33 | 1,053,912 | 34.52 |

$$PSNR = 10 \times \log_{10} \frac{255\times255}{\frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}(p_{i,j}-q_{i,j})^2}(dB) \tag{8}$$

here $p_{i,j}$ and $q_{i,j}$ denote the pixel values in row i and column j of the cover image with M × N size and the stego image, respectively. A high PSNR value indicates that the stego-image is very similar to the original image, whereas a low value indicates the opposite. Generally, distortion is indiscernible to the human eyes when PSNR is higher than 30 dB.

The overall embedding capacity in the proposed adaptive scheme is determined by Eq. (9), where the variables '*numedgeb*' and '*numsmoothb*' are the number of blocks located in edge and smooth areas, respectively. Also, the number of the embedding bits into each pixel (e.g. bit per pixel, bpp) in the proposed non-adaptive scheme is determined by Eq. (10), because, each pixel of cover image is embedded (v1 + v2) bits.

$$\text{capacity} = (\text{numedgeb} \times (v1 + v2) + \text{numsmoothb} \times v1) \times 4 \tag{9}$$

$$\text{capacity} = v1 + v2(bpp) \tag{10}$$

We have experimented using a series of v1 and v2 division with various threshold values. For example, 3–4 division $(v1 = 3, v2 = 1)$ with $T = 15$ means that each four pixels of the block with average different values placing into smooth area and edge area, will be embedded the 3-bit (e.g. v1 bits) and 4-bit (e.g. v1 + v2 bits), respectively. Table 3 a and b shows the results of our adaptive method in terms of embedding capacity and PSNR values. The PSNR values and the embedding capacities (in bits) are average values of the results executed by random bit streams many times.

Fig. 4 demonstrates the results of capacity-distortion for all cover images by our adaptive algorithm in the embedding rate (2–4) with various threshold values.

This shows which in a predetermined the embedding capacity by user via various threshold values, the embedding rate as well as the image quality can be adjusted for practical applications. Also Fig. 4 and the experimental results expose which in a predetermined the embedding rate, the smooth 'Splash'

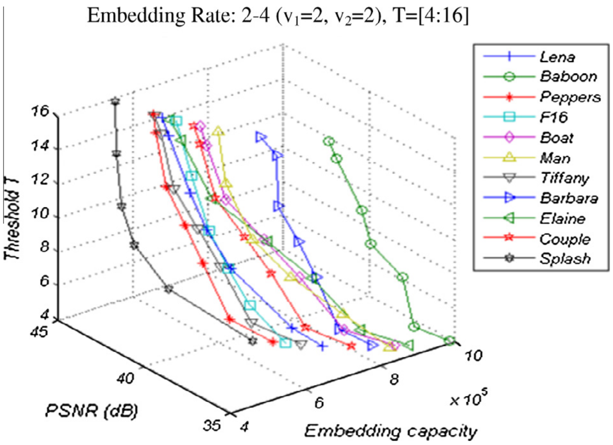


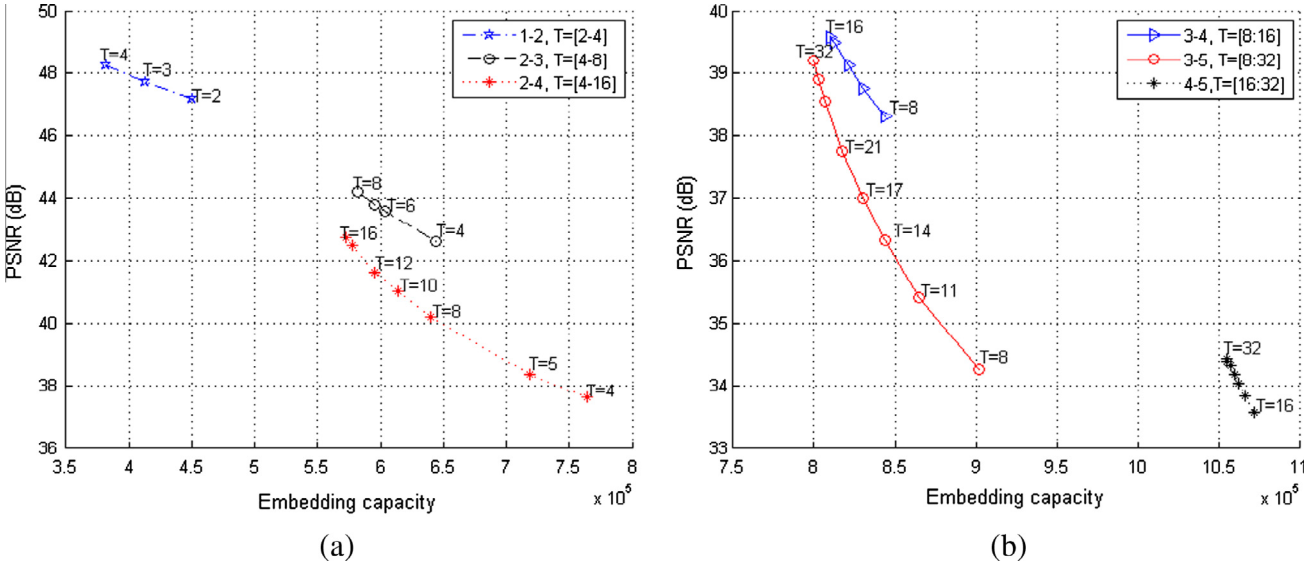
**Figure 4** Capacity–distortion for all the cover images in the embedding rate (2–4) by various thresholds in the proposed adaptive scheme ($T$ = [4–16]).

image produces higher quality and the complex ‘Baboon’ image provides higher capacity. Moreover for other embedding rates, as the embedding rate 2–4, we executed by various threshold values.

The results are similar to what Fig. 4 reveals. Fig. 5a and b shows capacity-distortion for Lena test image in the various embedding capacities by various threshold values. Figs. 4 and 5a and b verify that the proposed adaptive scheme is very scalable and flexible, so that a larger v1 or v2 and a lower *T* enhance the embedding rate whereas a lower v1 or v2 and a higher *T* enhance stego-image quality.

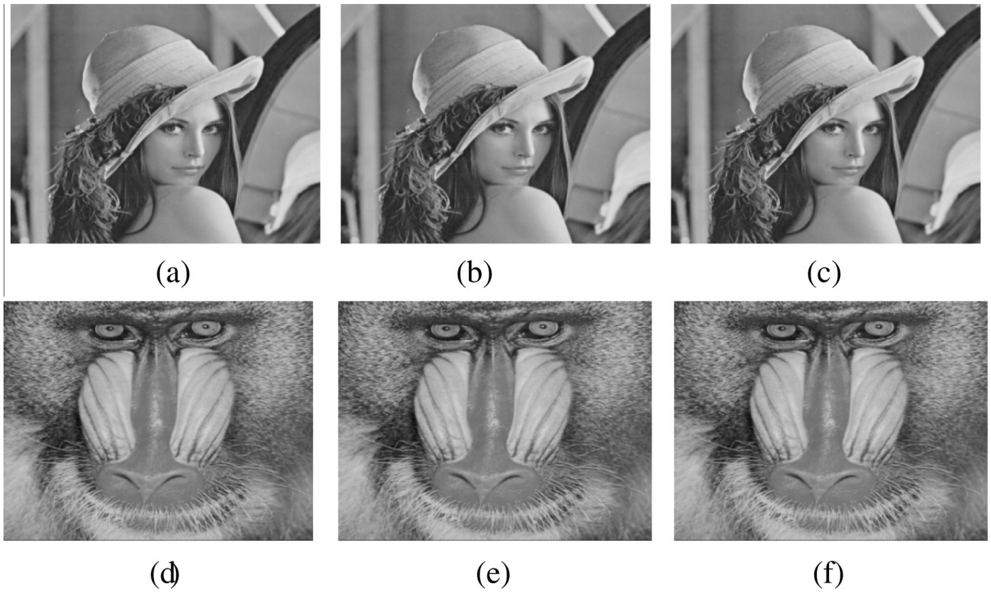
Fig. 6 indicates the stego-images produced by the proposed adaptive scheme for Lena and Baboon cover images. As the figures show, the distortions resulted from embedding are imperceptible to human vision.

Table 4 shows the comparisons of results of the embedding rate and image quality between Lee-Chen’s [20] and our adaptive scheme. (For instance, in columns 4 and 5 from Table 4 for Lee–Chen’s, the values of v1 and v2 have considered so that: v1 + v2 = 3 and in our adaptive scheme has been determined: v1 = 3 and v2 = 1). Hence in Lee–Chen’s scheme [20], each cover pixel embeds 3-bit of secret data but in our method, each pixel placed in smooth and edge block embeds 3-bit and 4-bit, respectively). In the embedding rate 2 bpp, our adaptive method for some images has lower PSNR values than Lee–Chen’s scheme, whereas for the higher embedding rates has



(a) (b)

**Figure 5** Capacity–distortion for Lena image in various embedding rates by various thresholds in the proposed adaptive scheme.



(a) (b) (c)

(d) (e) (f)

**Figure 6** (a) Original Lena image. (b) Stego-image in our adaptive scheme of Lena, 2–3 *T* = 8. (Embedded 581852 bits, PSNR = 44.22 dB). (c) Stego-image in our adaptive scheme of Lena, 2–4, *T* = 12. (Embedded 595120 bits, PSNR = 41.63 dB). (d) Original Baboon image. (e) Stego-image in our adaptive scheme of Baboon, 2–3 *T* = 8. (Embedded 712984 bits, PSNR = 41.60 dB). (f) Stego-image in our adaptive scheme of Baboon, 2–4 *T* = 12. (Embedded 830432 bits, PSNR = 36.75 dB).

**Table 4** Comparisons of the results between Lee–Chen's [20] and our adaptive method.

| Cover images | Lee–Chen's [20], 2-bit | | Our adaptive method 2–3, $T = 8$ | | Lee–Chen's [20], 3-bit | | Our adaptive method 3–4, $T = 16$ | | Lee–Chen's [20], 4-bit | | Our adaptive method 4–5, $T = 31$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR |
| Lena | 524,288 | 44.15 | 581,852 | 44.22 | 786,432 | 37.93 | 810,052 | 39.58 | 1,048,576 | 31.80 | 1,055,620 | 34.38 |
| Baboon | 524,288 | 44.15 | 712,984 | 41.60 | 786,432 | 37.94 | 909,804 | 36.76 | 1,048,576 | 31.85 | 1,108,708 | 32.21 |
| Peppers | 524,288 | 44.13 | 566,836 | 44.70 | 786,432 | 37.94 | 806,576 | 39.75 | 1,048,576 | 31.86 | 1,057,584 | 34.26 |
| F16 | 524,288 | 44.14 | 579,876 | 44.32 | 786,432 | 37.98 | 816,132 | 39.40 | 1,048,576 | 31.85 | 1,061,748 | 34.12 |
| Boat | 524,288 | 44.04 | 625,924 | 43.10 | 786,432 | 37.95 | 826,820 | 38.95 | 1,048,576 | 31.84 | 1,061,852 | 34.08 |
| Man | 524,288 | 44.13 | 628,056 | 42.88 | 786,432 | 37.91 | 833,912 | 38.32 | 1,048,576 | 31.82 | 1,065,496 | 33.14 |
| Tiffany | 524,288 | 44.15 | 575,491 | 44.40 | 786,432 | 37.91 | 807,040 | 39.74 | 1,048,576 | 31.84 | 1,055,268 | 34.50 |
| Barbara | 524,288 | 44.17 | 640,664 | 42.86 | 786,432 | 37.93 | 862,172 | 37.95 | 1,048,576 | 31.89 | 1,085,072 | 33.07 |
| Elaine | 524,288 | 44.12 | 634,044 | 42.88 | 786,432 | 37.90 | 811,760 | 39.40 | 1,048,576 | 31.83 | 1,051,984 | 34.59 |
| Couple | 524,288 | 44.12 | 606,492 | 43.54 | 786,432 | 37.94 | 823,096 | 39.01 | 1,048,576 | 31.85 | 1,058,792 | 34.10 |
| Splash | 524,288 | 44.16 | 544,496 | 45.47 | 786,432 | 37.98 | 794,680 | 40.32 | 1,048,576 | 31.87 | 1,053,448 | 34.55 |

**Table 5** Comparisons of the results between Yang et al.'s [13] and our adaptive method.

| Cover images | Yang et al.'s adaptive method, 2008 [13] (2–3) | | Our adaptive method, 2–3, $T = 8$ | | Yang et al.'s adaptive method, 2008 [13] (2–4) | | Our adaptive method, 2–4, $T = 8$ | | Yang et al.'s adaptive method, 2008 [13] (3–4) | | Our adaptive method, 3–4, $T = 8$ | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR | Capacity | PSNR |
| Lena | 575,188 | 43.95 | 581,852 | 44.22 | 626,088 | 36.96 | 640,024 | 40.18 | 837,332 | 36.28 | 843,692 | 38.31 |
| Baboon | 695,310 | 41.15 | 712,984 | 41.60 | 783,444 | 34.20 | 903,408 | 35.94 | 916,010 | 33.01 | 974,264 | 35.61 |
| Peppers | 561,236 | 44.49 | 566,836 | 44.70 | 598,184 | 38.11 | 609,560 | 41.14 | 823,380 | 37.17 | 828,892 | 38.83 |
| F16 | 568,184 | 44.38 | 579,876 | 44.32 | 612,080 | 39.62 | 635,792 | 40.30 | 830,328 | 37.80 | 841,856 | 38.46 |
| Boat | 624,284 | 42.69 | 625,924 | 43.10 | 723,880 | 36.36 | 728,384 | 38.16 | 886,028 | 35.53 | 887,656 | 37.09 |
| Tiffany | 566,992 | 44.23 | 575,491 | 44.40 | 609,696 | 37.52 | 627,392 | 40.53 | 829,136 | 37.10 | 837,288 | 38.53 |
| Barbara | 629,976 | 42.75 | 640,664 | 42.86 | 735,664 | 35.50 | 757,680 | 37.81 | 892,120 | 34.83 | 902,488 | 36.95 |
| Elaine | 621,052 | 42.57 | 634,044 | 42.88 | 717,816 | 34.17 | 745,672 | 37.76 | 883,196 | 33.52 | 895,252 | 36.82 |
| Couple | 581,753 | 43.70 | 606,492 | 43.54 | 645,862 | 38.20 | 689,152 | 38.93 | 840,470 | 36.87 | 868,408 | 37.55 |

the opposite. According to columns 6 and 7 from Table 4, Lee–Chen's scheme can carry a maximum of 4 bpp secret data with an acceptable quality and with the embedding 5 bpp secret data, the difference between the cover image and stego-image will be very high. Accordingly, the human eyes can distinguish the difference between them. But, our adaptive method can carry greater than 4 bpp, e.g. 5 bpp data into pixels located in edge areas, with the PSNR values greater of that in Lee–Chen's.

Tables 5 and 6 show the comparisons of the results between adaptive methods of Yang et al.'s [13] and Wang et al.'s [12] and our adaptive scheme in terms of embedding capacity and PSNR value. As a matter of fact, our adaptive scheme is superior to Lee–Chen [20], Wang et al. [12] and Yang et al. [13] schemes in three aspects, namely visual quality of stego-image, embedding capacity, level of security (because of existing five secret keys) and difficult detection of the embedded secret data (because of existing many permutations).

**Table 6** Comparisons of the results between Wang et al.'s [12] and our adaptive method.

| Cover images | Wang et al.'s adaptive method, 2008 [12] | | Our adaptive method 1–2, $T = 3$ | |
|---|---|---|---|---|
| | Capacity | PSNR | Capacity | PSNR |
| Lena | 409,752 | 44.15 | 412,416 | 47.74 |
| Baboon | 457,168 | 40.32 | 506,888 | 46.55 |
| Peppers | 407,256 | 43.28 | 415,480 | 47.56 |
| F16 | 421,080 | 42.14 | 468,053 | 46.98 |
| Boat | 421,080 | 42.14 | 457,536 | 47.06 |
| Man | 423,560 | 42.15 | 457,536 | 47.06 |
| Tiffany | 407,360 | 43.80 | 419,212 | 47.61 |
| Barbara | 442,560 | 42.34 | 446,016 | 47.27 |
| Elaine | 408,592 | 44.74 | 477,664 | 46.87 |
| Couple | 412,824 | 43.25 | 432,520 | 47.44 |
| Splash | 389,459 | 44.34 | 364,397 | 48.51 |

**Table 7** The results PSNR values by the embedding the same random massage between Lee–Chen's [20] and our non-adaptive method.

| Cover images | Lee–Chen non-adaptive method 2010 [20] | | | | Our non-adaptive method | | | |
|---|---|---|---|---|---|---|---|---|
| | 1-bit | 2-bit | 3-bit | 4-bit | 1-bit | 2-bit | 3-bit | 4-bit |
| Lena | 51.16 | 44.15 | 37.93 | 31.80 | 51.16 | 46.38 | 40.74 | 34.82 |
| Baboon | 51.14 | 44.15 | 37.94 | 31.85 | 51.15 | 46.36 | 40.72 | 34.81 |
| Peppers | 51.13 | 44.13 | 37.94 | 31.86 | 51.13 | 46.39 | 40.73 | 34.80 |
| F16 | 51.15 | 44.14 | 37.98 | 31.85 | 51.15 | 46.38 | 40.73 | 34.81 |
| Boat | 51.18 | 44.04 | 37.95 | 31.84 | 51.18 | 46.37 | 40.72 | 34.81 |
| Man | 51.12 | 44.13 | 37.91 | 31.82 | 51.12 | 46.36 | 40.73 | 33.97 |
| Tiffany | 51.15 | 44.15 | 37.91 | 31.84 | 51.16 | 46.38 | 40.75 | 34.89 |
| Barbara | 51.17 | 44.17 | 37.93 | 31.89 | 51.17 | 46.37 | 40.74 | 34.81 |
| Elaine | 51.13 | 44.12 | 37.90 | 31.83 | 51.13 | 46.36 | 40.73 | 34.80 |
| Couple | 51.17 | 44.12 | 37.94 | 31.85 | 51.17 | 46.35 | 40.65 | 34.75 |
| Splash | 51.15 | 44.16 | 37.98 | 31.87 | 51.15 | 46.36 | 40.75 | 34.82 |

The PSNR values in Lee–Chen's scheme [20] were similar to those in K-LSB substitution method [2] with the same embedding capacity into each pixel. Also, we compare our non-adaptive scheme with Lee–Chen's scheme. Table 7 indicates the comparison of the PSNR values between these two schemes in the same embedding capacity and also with the same random massage. As Table 7 shows, because of executing the error reducing process in ours, the PSNR values have enhanced. Thus, the produced stego-image by our scheme is more similar to cover image and stego-image visual quality enhances.

In the embedding 1-bit into each pixel of Table 7, our scheme produces the PSNR values similar to Lee–Chen's. Because, at a lower embedding capacity, for example: 1 bpp, the difference between cover-pixel values of its corresponding stego-pixel value is very small. Hence, the error reducing procedure in our non-adaptive scheme, will not have a great effect on minimizing this difference. But in higher embedding capacities, the difference between the value cover-pixels of its corresponding stego-pixel value is large and the error reducing process causes to decrease this difference and following that the PSNR value increases.

Finally, the proposed adaptive scheme is secure against the well-known RS steganalysis attack. The RS detector algorithm that is proposed by Fridrich et al. in 2001 [6] , can judge whether the stego-image is secure without visual distortion. In other words, the RS steganalysis can detect the presence of secret data within the stego-image. In Fridrich's algorithm [6], all the stego-pixels are categorized into three pixel groups: the regular group ($R_m$ or $R_{-m}$), the singular group ($S_m$ or $S_{-m}$) and the unusable group. If the relative number of $R_m$ is equal to that of $R_{-m}$, and the relative number of $S_m$ is equal to that of $S_{-m}$, the stego-image will pass the RS detector. Otherwise, the stego-image will be verified as a suspicious image and it reveals the presence of the secret data. In the detection results, we utilized of recommended masks $M = [0 1 1 0]$ and $-M = [0 -1 -1 0]$. Table 8 shows the RS steganalysis results. As Table 8 demonstrates, the relative number of $R_m$ and $R_{-m}$ and $S_m$ and $S_{-m}$ is much similar together. Consequently, the RS steganalysis detector cannot detect the presence of secret data into the stego-images. Accordingly, we can solidly represent that the proposed adaptive scheme is able to resist against the RS detector attack.

**Table 8** Statistics of RS steganalysis 2001 [6] for 11 stego-images embedded via varying secret data in bits.

| Stego | Capacity | $R_m$ | $R_{-m}$ | $S_m$ | $S_{-m}$ |
|---|---|---|---|---|---|
| Lena | 810,052 | 22,858 | 22,796 | 16,913 | 16,849 |
| Baboon | 909,804 | 25,642 | 25,585 | 22,921 | 23,018 |
| Peppers | 806,576 | 25,507 | 25,806 | 18,731 | 18,576 |
| F16 | 816,132 | 25,716 | 25,972 | 18,973 | 18,722 |
| Boat | 826,820 | 22,769 | 22,883 | 17,782 | 17,813 |
| Man | 833,912 | 25,307 | 26,408 | 21,406 | 20,518 |
| Tiffany | 807,040 | 24,846 | 24,488 | 17,646 | 17,637 |
| Barbara | 862,172 | 23,131 | 23,182 | 18,823 | 18,765 |
| Elaine | 811,760 | 23,900 | 23,898 | 19,705 | 19,652 |
| Couple | 823,096 | 24,005 | 24,236 | 18,177 | 17,991 |
| Splash | 794,680 | 24,893 | 24,793 | 17,098 | 16,948 |

## 6. Conclusions

In this study firstly we developed an adaptive steganographic scheme that uses average differencing value of four neighborhood pixels and modulus function. Our adaptive scheme is based on the concept of human vision sensitivity, so that it is more difficult to notice changes at the edge regions of cover image than those in smooth regions. Accordingly, the number of bits to be embedded into each block is variable and determined by the correlation between neighborhood pixels into its block. In our adaptive scheme, the average differencing value of a four-pixel block and a threshold secret key $T$ are factors for detecting the edge or smooth areas, according to the local complexity of a cover image. The number of bits to hide in the edge blocks is greater than of those to hide in smooth blocks. Then, we represent a non-adaptive scheme which produces minimum distortion for the stego-image. In both our adaptive and non-adaptive algorithms, problem of overflow and underflow will not be occurred. Experimental results indicate that the proposed adaptive algorithm significantly is superior to the currently existing scheme, in terms of stego-image visual quality, embedding capacity, level of security and our non-adaptive algorithm is better of aspect producing stego-image with higher visual quality. Finally, the detection results for our adaptive scheme verified that the RS steganalysis algorithm [6] has disablement in detecting the present of secret data into the stego-images. Hence, the proposed adaptive scheme can resist against the RS steganalysis attack. Because of property of *mod* function, without loss of generality of the mathematical rule "$[(a - b)\ mod\ b] = [a\ mod\ b] = [(a + b)\ mod\ b]$" for each $a,b \in \{0\} \cup N$ and according to the represented comments in analysis and discussion section, the readjust and error reducing procedures in both adaptive and non-adaptive methods cause the data extraction process to be done correctly. Our safe methods can be adjusted, depending on the requirements of the application concerned. Also, these allowed the users to hide various large secret messages, while at the same time maintain the general appearance of any cover image used. In the proposed methods in this scholar, detecting secret data is extremely difficult for malicious, because of the large permutations. Also with our schemes having several secret keys, their security level is high. However our adaptive method produces good result, and it can be in future works to beside the metrics addressed in this scholar, weave other adaptive steganographic method to achieve a stego-image with higher quality.

## References

[1] Hartung F, Kutter M. Information hiding – a survey. Proc IEEE 1999;87:1062–78.

[2] Bender DW, Gruhl NM, Lu A. Techniques for data hiding. IBM Syst J 1996;35:313–6.

[3] Chan CK, Cheng LM. Hiding data in images by simple LSB substitution. Pattern Recognit 2004;37(March):469–74.

[4] Zhang X, Wang S. Efficient steganographic embedding by exploiting modification direction. IEEE Commun Lett 2006;10(11):781–3.

[5] Mielikainen J. LSB matching revisited. IEEE Signal Process Lett 2006;13(5):285–7.

[6] Fridrich J, Goljan M, Du R. Reliable detection of LSB steganography in color and grayscale images. In: Proceedings of ACM workshop on multimedia and security; 2001. p. 27–30

[7] Wu DC, Tsai WH. A steganographic method for images by pixel-value differencing. Pattern Recognit Lett 2003;24:1613–26.

[8] Chang CC, Tseng HW. A steganographic method for digital images using side match. Pattern Recognit Lett 2004;25:1431–7.

[9] Zhang X, Wang S. Steganography using multiple-base notational system and human vision sensitivity. IEEE Signal Process Lett 2005;12:67–70.

[10] Wu HC, Wu NI, Tsai CS, Hwang MS. Image steganographic scheme based on pixel-value differencing and LSB replacement methods. Proc Inst Elect Eng, Vis Images Signal Process 2005;152(5):611–5.

[11] Yang CH, Weng CY, A steganographic method for digital images by multi pixel differencing. In: Proceedings of international computer symposium, Taipei, Taiwan, R.O.C.; 2006. p. 831–6.

[12] Wang CM, Wu NI, Tsai CS, Hwang MS. A high quality steganography method with pixel-value differencing and modulus function. J Syst Software 2008;81:150–8.

[13] Yang CH, Weng CY, Wang SJ, Sun HM. Adaptive data hiding in edge areas of images with spatial LSB domain systems. IEEE Trans Inf Forensics Security 2008;3(3):488–97.

[14] Luo Weiqi, Huang Fangjun, Huang Jiwu. Edge adaptive image steganography based on LSB matching revisited. IEEE Trans Inf Forensics Security 2010;5(2).

[15] Sivaranjani Mrs, Semi Sara mani Ms. Edge adaptive image steganography based on LSB matching revisited. J Comput Appl (JCA) 2011;IV(1):1–3.

[16] Meena Manoj Kumar, Kumar Shiv, Gupta Neetesh. Image steganography tool using adaptive encoding approach to maximize image hiding capacity. Int J Soft Comput Eng (IJSCE) 2011;1(2):7–11.

[17] Hsiao Ju-Yuan, Chang Chieh-Tse. An adaptive steganographic method based on the measurement of just noticeable distortion profile. Original Res Article Image Vision Comput 2011;29(2–):155–66.

[18] Kouider S, Chaumont M, Puech W. Adaptive steganography by oracle (ASO). Multimedia and Expo (ICME), 2013 IEEE International Conference; 15–19 July 2013.

[19] Yu C, Wang J. An image adaptive steganography algorithm based on sparse representation and entropy. Sci Computer Appl 2013.

[20] Lee CF, Chen HL. A novel data hiding scheme based on modulus function. J Syst Software 2010;83:832–43.

[21] Liao X, Wen QY, Zhang J. A steganographic method for digital images with four-pixel differencing and modified LSB substitution. J Vis Commun Image R 2010;1–8.