Journal of Applied Sciences

ISSN 1812-5654
Brownian Motion of Binary and Gray-Binary and Gray Bits in Image for Stego

Rengarajan Amirtharajan and John Bosco Balagurun Rayappan
Department of Electronics and Communication Engineering,
School of Electrical and Electronics Engineering, SAstra University,
Thanjavur, Tamil Nadu, India

Abstract: Computers have invaded all premises of the human world, starting from a grocery store to a missile launching center. Because of the omnipresence of computers, it becomes more and more difficult everyday to secure the confidential information from misuse. The fairly common technique of cryptography has been proved inadequate in recent years. Steganography, a contemporary yet an age-old technique to hide secret data into an unsuspected cover media like an image, thereby preventing the recognition of the very presence of secret data, is an alternative. In this study, an improved image steganographic approach is proposed. This method reduces the mean square error (MSE) by localizing the error-reduction process to every row. The error reduction is performed by selective embedding of the actual secret, its binary complement, gray-coded version or inverted gray-coded version. Of the four versions, the version giving the least MSE is embedded on a row-by-row basis. This method reduces the MSE by a factor of 1.8 and boosts the peak signal to noise ratio (PSNR) by a 0.25 db and considerably increases the security.

Key words: Binary gray encoding, inverted pattern approach, optimum pixel adjustment process, steganography

INTRODUCTION

Power has taken several incarnations ever since its genesis. Right from Stone Age to the present day, power has kept its possessor at the summit of hegemony and thus it is the most sought after commodity. In the electronic epoch power has manifested itself in the form of classified and critical information. Since the human race has succumbed to enticing power to such an extent that iniquity today is skyrocketing, there is a need to protect information from falling into the wrong hands and to prevent clandestine and unscrupulous activities. Providentially, the advancements in technology have beget many techniques to maintain the veracity and variability of the crucial information giving rise to an entire discipline called information hiding. Information hiding is stratified into several subsets namely cryptography (Schneier, 2007), steganography and watermarking (Stefan and Fabin, 2000; Zaidan et al., 2010).

Cryptography (Schneier, 2007) is the art of writing esoteric information in an occult fashion thereby rendering it scruptable only to the authorized receiver. In contrast to cryptography which focuses on keeping the contents of a message secret, steganography (Stefan and Fabin, 2000; Zaidan et al., 2010) focuses on keeping the very existence of a message secret. Steganography is implemented in digital audio (Zhu et al., 2011), video (Al-Frajat et al., 2010) and images (Amirtharajan and Balaguru, 2009, 2010, 2011, Amirtharajan et al., 2012, Bender et al., 1996) of which image steganography has gained much appreciation and commendation in the recent past. In image steganography the vital information is assembled in a cover image with assiduous efforts resulting in a stego image. The embedded secret information is imperceptible to the human eye thereby rendering the image impregnable (Yang, 2008).

In the available literature many researchers proposed an assortment of approaches to information hiding. These methodologies have different characteristics like capacity, imperceptibility and robustness (Amirtharajan and Balaguru, 2009, 2010, 2011; Kumar et al., 2011). These characteristic are inevitable for different applications, such as secret communication (Stefan and Fabin, 2000), copyright protection (Wang and Lin, 2004; Yen and Tsai, 2008) and tampering detection or integrity check (Lin et al., 2005).

Information hiding techniques could be categorized into two types: methods in the spatial domain and methods in the frequency domain. In the spatial domain approach, the secret messages are embedded by directly
injecting secret data in the image pixels (Chan and Cheng, 2001, 2004; Wang et al., 2001; Chang et al., 2003; Yang, 2008; Thien and Lin, 2003). Whereas in the later case, the frequency domain approach the image is first transformed into its frequency domain (Amirharajan and Rayappan, 2012a, b; Chang et al., 2002) then the secret messages are embedded in the transformed coefficients.

The major concern is about the objective of transmitting secret data, the stego method should possess high capacity, high quality and imperceptibility. More number of research papers have been intended for this theme and performs the embedding operations in the spatial domain either using raster scan or random scan (Amirharajan and Balaguru, 2009, 2010; Amirharajan et al., 2011, 2012b, 2012a; Yen and Lin, 2010). A detailed survey on Information hiding till 1999 is available by Petitcolas et al. (1999). A complete survey on image steganography could be found by Chedad et al. (2010) and on random image steganography and steganalysis in Amirharajan et al. (2012) three more survey on Field Programmable Gate Array (FPGA) for steganography, middle ware for cryptography/ steganography and Orthogonal Frequency Division Multiplexing (OFDM)+Code Division Multiple Access (CDMA)+stego for secure communication is available by Rajagopalan et al. (2012), Janakiraman et al. (2012a) and Themmozi et al. (2012), respectively. There are three kinds of approaches called LSB-based (Chan and Cheng, 2001, 2004; Wang et al., 2001; Chang et al., 2003; Yang, 2008, Thien and Lin, 2003; Amirharajan and Balaguru, 2009, 2010), PVD-based (Wang et al., 2008; Amirharajan et al., 2010) and mod-based (Chan and Cheng, 2004; Thien and Lin, 2003; Wang et al., 2008) are commonly available in literature and sometimes it could be combined to offer both capacity, imperceptibility and to improve the security (Chang et al., 2003; Hnood et al., 2010a, b; Xiang et al., 2011; Lin et al., 2005; Janakiraman et al., 2012b; Zaidan et al., 2010, 2011 and Zanganah and Ibrahim, 2011). The counter attack on steganography called steganalysis are detailed (Xia et al., 2009; Qin et al., 2009). A detailed review on steganalysis is reported by Qin et al. (2010).

In LSB-based approaches, secret data are embedded by directly substituting the least-significant-bits (LSBs) with equal bits of the secret for each pixel. Furthermore, techniques based on pixel-value differencing (PVD) modify the difference value between a pair of pixels to fit the value of the embedded secret. Finally, mod based approaches which use the modular operation, are similar to k-bit LSB-based approaches if the modulus is $2^k$.

Motivated by this study, a simple and effective stego method has been proposed to improve the stego image quality and to introduce cryptic effect while embedding.

PRELIMINARY RELATED WORKS

Chan and Cheng (2004) proposed an LSB-based hiding scheme using an optimal pixel adjustment process (OPAP). Their method adjusts each pixel after the message is embedded to improve the quality of the stego object and their experimental results showed that their method yielded quicker results. Yang (2008) proposed new LSB-based approach, named as the Inverted pattern (IP) LSB substitution approach. Later this method combined with OPAP called IPL-KSB to improve the quality of the stego image. In this study, we have adapted a new LSB-based approach based on Yang (2008), named as the inverted pattern binary and gray (IPBG) LSB substitution approach, to further highlight the quality of the stego-image. Before secret messages are embedded, some secret messages are transformed by inverting operation and some secret messages are not. A simple strategy is used to judge whether a section of messages is inverted and a bit string named as the IPKey is used to record these inverting actions. Also, we combine the concept of the OPAP with our approach to improve image quality further. The experimental results show that the proposed approach results in a better image quality than that of the optimal LSB substitution approach (Wang et al., 2001; Chan and Cheng, 2001; Thien and Lin, 2003), the OPAP LSB substitution approach (Chan and Cheng, 2004) and inverted pattern approach (Yang, 2008).

In a normal LSB substitution the RGB (red blue green) image is converted in to gray image and then last few least significant bits of gray image are selected according to key length $k$ and the message which is to be embedded is converted to series of ASCII values of the characters in the message and then to binary. Message is then stored in the cover according to the method of embedding. The series of operations done in LSB substitution are as follows:

Let $C$ be the original 8-bit grayscale cover-image of $M \times N$, pixels represented as:

$$C = \{x_{ij} | 0 \leq I < M, 0 \leq j < NC, x_{ij} \in \{0, 1, 2, ..., 255\}\}$$

$D$ be the $n$-bit secret data represented as:

$$D = \{d_{i} | 0 \leq I < n, d_{i} \in \{0, 1\}\}$$

Suppose that the $n$-bit secret data $D_{i}$ (decimal representation) is to be embedded into the $k$-rightmost LSBs of the cover-image $C$:

$$S = C - C \mod 2^k + D_{i}$$
Here, S is the Stego object, C cover object and \( D_j \) is the decimal equivalent of the secret data.

In the extraction process, given the stego-image S, the embedded messages can be readily extracted without referring to the original cover-image. The \( k \) LSBs of the selected pixels are extracted and lined up to reconstruct the secret message bits. Mathematically, the embedded message bits D can be recovered by:

\[ D_j = S \mod 2^k \]

The OPAP simply improves the stego object after embedding the secret data, either by adding or subtracting \( 2^k \) without affecting the rightmost \( k \) secret data bits in the stego cover.

THE PROPOSED METHOD

Embedding: A schematic diagram of the proposed method is given in Fig. 1 and 2. Initially the secret data or message is encrypted using Data Encryption Standard DES (Schneier, 2007), is a symmetric key cryptography algorithm. The cover image is split into separate rows. The order of rows considered for embedding data is chosen using a Pseudo random number generator with a chosen seed. For each row, a try is made to embed data, inverse data, gray code of data and the inverted gray code of data. The encoded form of the confidential information on the selected row which offers minimum Mean Square Error (MSE) is chosen and fixed for the same. This binary/inverted binary/gray and inverted gray data pattern is stored as IPKEY. Thus, on an average MSE is reduced to a greater extent. The Stego image, IPKEY and the seed are communicated.

Mathematical model for row wise, inverted pattern LSB embedding

- **General formulae:** 1's complement of a No.:

\[ \bar{x} = (2^k - 1) - x \]

Where:

\( k \) = No. of bits

\( x \) = Number to be inverted in bits

\( \bar{x} \) = 1's complement of the number

For example, take a 4 bit binary representation of a number '2' [0010] as \( x \) here:

\( k = 4 \) so \( \bar{x} = (16-1) - 2 = (16-1) - 2 \)

\( \bar{x} = 13[1101] \) complement of [0010] '2'

- **LSB embedding:**

\[ S_i = C_i \times C_i \mod 2^k + m_i \]

Where:

\( k \) = No. of bits to be embedded

\( C_i \) = Cover pixel

\( S_i \) = Stego pixel

\( m_i \) = k – bit message block in decimal

For example, let \( k = 4 \). \( C_i = 16[00010000] \) and \( m_1 \) is '2' [0010] as \( m_1 \):

\( S_i = 16 \times 16 \mod 2^4 + 2 = 16 \times 16 \mod 16 + 2 = 18 \)

\( S_i = 8[00010010] \)

![Fig. 1: Proposed schematic diagram for embedding](image-url)
**LSB recovery:**

\[ m_i = S_i \mod 2^k \]  \hspace{1cm} (3)

where, symbols are same as Eq. 2:

Let \( S_i = 18[00010010] \)

To Extract the last 4 [since \( k = 4 \)] bits, we have:

\[ m_i = 18 \mod 2^4 = 18 \mod 16 = 2[0010] \]

**General IDEAS**

Four flavours of secret data:

- Plain data-m \((i,j)\)
- Inverted data \(\bar{m} \) \((i,j)\)
- Grey Coded data-g \((i,j)\)
- Inverted Grey Coded data-g' \((i,j)\)

**R rows:** In each cover image, there are \( R \) No. of rows, each of same length \( D \), where:

\[ R \times D = M_r \times N_c \]  \hspace{1cm} (4)

where, the \( M_r \times N_c \) are dimensions of the cover image. Each row is denoted as \( r_i \), where, \( i \in N \) and \( i \in R \) i.e., Set of rows = \( \{ r_i, \forall i \in N \text{ and } i \in R \} \)

- Each row \( r_i \) is in turn a matrix, denoted as:

\[ r_i = [r_{i1}, r_{i2}, \ldots, r_{ik}] \text{ where } i \in N \text{ and } i \leq R \]  \hspace{1cm} (5)

In other words, each row has \( D \) pixels.

**Message data (secret) to be embedded [k bit length]:**

\[ m(i, j) \text{, where:} \]

\[ i = \text{Row identifier} \]

\[ j = \text{Pixel inside a row} \]

The complement of \( m(i, j) \) is denoted as \( \bar{m}(i, j) \)

**Embedding procedure:** Let the cover image be \( C \) with \( M_r \times N_c \) pixels.

Let it be divided into \( R \) blocks named \( r_1, r_2, \ldots, r_R \) each having equal number of pixels \( D \):

\[ R \times D = M_r \times N_c \]  \hspace{1cm} (6)

Also, \( r_i = [r_{i1}, r_{i2}, \ldots, r_{ik}] \), where \( i \in N \) and \( i \leq R \)

Let 'k' be the number of LSBs to be replaced in cover pixels.

Let the secret message be a matrix \( M \), where each elements of \( M \) is made up of \( k \) bits. Then we can denote the message to be embedded in the \( i \)th row, \( j \)th pixel as \( m(i, j) \). Let \( s(i, j) \) denote the stego value of \( j \)th pixel in the \( i \)th row, when message \( m(i, j) \) is embedded in cover pixel \( r_j \).

Alternatively \( \bar{m}(i, j) \) is embedded instead of \( m(i, j) \) then the stego pixel is denoted as \( \bar{s}(i, j) \):

\[ \bar{s}(i, j) = r_j - r_j \mod 2^k + m(i, j) \text{ (Applying Eq. 2)} \]  \hspace{1cm} (7)

\[ s(i, j) = r_j - r_j \mod 2^k + \bar{m}(i, j) \text{ (Eq. 1)} \]

\[ s(i, j) = r_j - r_j \mod 2^k + g(i, j) \]

\[ s(i, j) = r_j - r_j \mod 2^k + g'(i, j) \]
If we consider R blocks of stego image as \( s_1, s_2, \ldots, s_R \). Then, \( s_i = s(i) \) or \( \bar{s}(i) \) or \( s'_i \) (i) or \( s''_i \) where MSE is minimum and \( s(i) = \{s(i,j), j \in N \text{ and } j \leq D\} \). Key matrix is denoted as:

\[
K = [K_1, K_2, \ldots, K_n]
\]

where, \( K \) is chosen based on the following conditions:

01 = if \( \bar{s}(i,j) \) is embedded
10 = if \( g(i,j) \) is embedded
11 = if \( g'(i,j) \) is embedded

**Retrieval procedure:** Key matrix is denoted as:

\[
K = [K_1, K_2, \ldots, K_n]
\]

The preliminary, unprocessed message \( m_i \) can be extracted from pixels in stego image as: \( m_i = s(i,j) \mod 2^d \) from Eq. (3) the actual message \( m(i,j) \) can be extracted by processing \( m_i \) as follows:

- \( m(i,j) \) is chosen from the following conditions based on \( K \):
- \( m(i,j) = 0 \) if corresponding \( K \) is 00
- \( (2^k-1) \times m(i,j) \), if \( K \) is 01
- \( g^1(i,j) \) if \( K \) is 10 (if \( g^1 \) denotes inverse of grey code function)
- \( g'(i,j) = (2^k-1) \times g^1(i,j) \) (if \( K = 11 \))
- \( (g'^1 \) denotes inverse of inverted grey code function)

**WORST CASE MSE**

The worst case MSE for a block with \( D \) pixels is defined as:

\[
\text{MSE}_{\text{w}}(i) = \frac{1}{D} \sum_{j=0}^{D-1} (s(i,j) - d(i,j))^2
\]

MSE for \( i^\text{th} \) row, when \( m(i) \) (actual data) is embedded, is given as:

\[
\text{MSE}(i) = D^{-1} \sum_{j=0}^{D-1} (s(i,j) - d(i,j))^2
\]

When inverted data \( \bar{s}(i) \) is embedded, then MSE for the same parameters is denoted as:

\[
\text{MSE}(i) = D^{-1} \sum_{j=0}^{D-1} (\bar{s}(i,j) - d(i,j))^2
\]

When grey coded data \( g(i) \) is embedded, then MSE for the same parameters is denoted as:

\[
\text{MSE}_{g}(i) = D^{-1} \sum_{j=0}^{D-1} (g(i,j) - d(i,j))^2
\]

When inverse grey coded data \( g'(i) \) is embedded, then MSE for the same parameters is denoted as:

\[
\text{MSE}_{g'}(i) = D^{-1} \sum_{j=0}^{D-1} (g'(i,j) - d(i,j))^2
\]

According to the embedding procedure, minimum MSE is chosen. The minimum MSE for a row is defined as:

\[
\text{MSE}_{\text{min}}(i) = \min \{ \text{MSE}(i), \text{MSE}_{g}(i), \text{MSE}_{g'}(i), \text{MSE}_{\bar{s}} (i) \}
\]

\[
\text{MSE}(i) = \sum_{j=0}^{D-1} (s(i,j) - d(i,j))^2 + \sum_{j=0}^{D-1} (\bar{s}(i,j) - d(i,j))^2
\]

\[
= D^{-1} \sum_{j=0}^{D} (2^k - 1)^2 \text{ (since sum of all } s, \bar{s}, s_g \text{, and } s' \text{ components):}
\]

\[= (2^k - 1)^2 \text{ (as given in the IP paper)}
\]

\[= (2^k - 1)^2 = \text{MSE}_{\text{w}}(i) \quad (9)
\]

We know that, if any \( n \) numbers \( x_1, x_2, x_3, \ldots, x_n \) add up to produce a total \( T \), then:

\[
\text{Min} \{ x_1, x_2, x_3, \ldots, x_n \} \leq (T/n) \quad (10)
\]

Thus, applying (10) in (9), we get:

\[
\text{MSE}_{\text{min}}(i) \leq \frac{1}{2} \text{MSE}_{\text{w}}(i) \text{ for all } i \in R
\]

Thus, we get MSE for any block to be less than or equal to \( \frac{1}{2} \) of the worst case MSE.

Random k-bit Adaptive Embedding

**Inputs:**

- Sampled Cover Image C
- Secret data bit stream M
- Key E for Encryption
Outputs:

- Stego Image (S), containing embedded secret data
- KEY (Used for recovery)

Algorithm for embedding:

- **Step 1:** Encrypt the secret data (M) using DES (Data Encryption Standard) with key E
- **Step 2:** Let \( P = \) length of secret data stream M (in number of bits) got from Step-1
- **Step 3:** Split the cover image C into separate rows. Let \( N = \) Total number of rows
- **Step 4:** Generate a array PRN of N pseudo-random numbers in the range \([0,N-1]\) where each No. occurs only once. Let the seed be stored in a text file
- **Step 5:** Invert the bit array M to give \( \bar{M} \). Encode the bit array M using Grey Code to give G and invert G to give \( \bar{G} \)
- **Step 6:** Let \( i = 1 \) (Here, \( i \) is the row counter)
- **Step 7:** Select PRNG[i]'th block and perform the following operations

Selective embedding

- a. Let \( r = \) pixel index array (for traversal)
- b. For ( \( j = 1 \) to length (r) ) do (Here \( j \) is the pixel counter)
  
  Replace k LSBs of jth pixel of the selected block with k bits from M to give \( O[i,1] \)
  
  c. Compute MSE
  
  d. For ( \( j = 1 \) to length (r) ) do (Here \( j \) is the pixel counter)
  
  Replace k LSBs of jth pixel of the selected block with k bits from M to give \( O[i,2] \)
  
  e. Compute MSEGray
  
  f. For ( \( j = 1 \) to length (r) ) do (Here \( j \) is the pixel counter)
  
  Replace k LSBs of jth pixel of the selected block with k bits from \( \bar{G} \) to give \( O[i,3] \)
  
  g. Compute MSEGray
  
  h. If MSE is greatest

**KEY[i] = "00"**

Else if MSEGray is greatest

**KEY[i] = "01"**

Else if MSE is greatest

**Assign MSE[i] = Minimum MSE**

- Choose STEG[i] as the value of \( O \) for which MSE is minimum
- \( P = P-k. \) (Reduce length as k bits have been embedded)
- If \( P>0 \) then assign \( i = i+1 \). Else, goto step-8 (that is check whether message is finished)
- If \( i>N \) then goto step-8 (check whether EOF is reached for cover image)
- **Step 7:** F. Goto
- **Step 8:** Save the array STEG as the stego image array S
- **Step 9:** Save S into a image file and KEY in a text file
- **Step 10:** Communicate S,KEY and seed used to generate PRN

Recovery process: The same Pseudo random number sequence is generated using the received seed. Using the KEY, the pattern is identified for different rows. Recovery modules are run to recover the secret. The result is then decrypted using DES to get the message back.

Random k-bit adaptive recovery

- **Inputs:**
  
  - Stego Image (S), containing embedded secret data
  - Key E for decryption. KEY in text file from embedding process
  - Seed (to generate Pseudo Random Number Generator PRNG)

- **Output:**
  
  - Secret data bit stream M

Algorithm for extraction

- **Step 1:** Split the stego image S into separate rows. Let \( N = \) Total number of rows
- **Step 2:** Generate a array PRN of N pseudo-random numbers in the range \([0,N-1]\) where each No. occurs only once
- **Step 3:** Let \( i = 1 \) (Here, \( i \) is the row counter)
- **Step 4:** Select PRNG[i]'th row and perform the following operations:

- Get Message M using retrieval
- B. If **KEY[i,1] = "01"**
  
  \( M[i] = \bar{M} [i] \)
  
  Else if **KEY[i,1] = "10"**
  
  \( M[i] = \text{MGray}[i] \)
else if KEY[i,1] = "11"
M[i] = M[Gray][i]
else
M[i] = M[i]

- Assign i = i+1 (increment row count)
- If i>N goto step-5 else goto Step-4

- **Step 5:** Decrypt M using DES and write it to text file as output

**RESULTS AND DISCUSSION**

In this present implementation Lena, Baboon, Gandhi and Temple 256×256 pixel Images have been considered by varying k = 1, 2, 3 and 4 bit LSB embedding, then stego image quality has been improved with OPAP. The effectiveness of the proposed system has been estimated by computing the MSE and PSNR of the Stego object with cover object.

The MSE is calculated by using the equation:

$$\text{MSE} = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (X_{ij} - Y_{ij})^2$$  \hspace{1cm} (11)

where, M and N denote the total number of pixels in the horizontal and the vertical dimensions of the image $X_{ij}$ represents the pixels in the original image and $Y_{ij}$ represents the pixels of the stego-image.

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE for simple LSB substitution Lena</th>
<th>k bit embedding</th>
<th>PSNR for simple LSB substitution Lena</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>Binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.497101</td>
<td>2.210863</td>
<td>9.042145</td>
<td>36.60521</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>Inv. binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.502899</td>
<td>2.197872</td>
<td>8.976959</td>
<td>36.90599</td>
</tr>
<tr>
<td>Gray</td>
<td>Gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.499008</td>
<td>2.509048</td>
<td>9.85718</td>
<td>41.16984</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>Inv. gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.500992</td>
<td>2.488876</td>
<td>9.86731</td>
<td>40.88161</td>
</tr>
<tr>
<td>Best</td>
<td>Best</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.465515</td>
<td>2.082016</td>
<td>8.578262</td>
<td>34.84331</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE after OPAP process Lena</th>
<th>k bit embedding</th>
<th>PSNR after OPAP process Lena</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>Binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.497101</td>
<td>1.495239</td>
<td>5.536330</td>
<td>21.59154</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>Inv. binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.502899</td>
<td>1.500183</td>
<td>5.463287</td>
<td>21.65492</td>
</tr>
<tr>
<td>Gray</td>
<td>Gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.499008</td>
<td>1.505264</td>
<td>5.505615</td>
<td>21.61382</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>Inv. gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.500992</td>
<td>1.499008</td>
<td>5.498413</td>
<td>21.53495</td>
</tr>
<tr>
<td>Best</td>
<td>Best</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.465515</td>
<td>1.458481</td>
<td>5.36801</td>
<td>20.98838</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE for simple LSB substitution Baboon</th>
<th>k bit embedding</th>
<th>PSNR for simple LSB substitution Baboon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>Binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.502014</td>
<td>2.185147</td>
<td>9.108154</td>
<td>36.10493</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>Inv. binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.497986</td>
<td>2.231379</td>
<td>9.056458</td>
<td>36.02116</td>
</tr>
<tr>
<td>Gray</td>
<td>Gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.500198</td>
<td>2.515854</td>
<td>9.962066</td>
<td>40.31307</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>Inv. gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.498802</td>
<td>2.485123</td>
<td>9.90790</td>
<td>40.33501</td>
</tr>
<tr>
<td>Best</td>
<td>Best</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.467408</td>
<td>2.090928</td>
<td>8.618286</td>
<td>34.82237</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE after OPAP Process Baboon</th>
<th>k bit embedding</th>
<th>PSNR after OPAP Process Baboon</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>Binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.502014</td>
<td>1.497986</td>
<td>5.538086</td>
<td>21.48482</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>Inv. binary</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.497986</td>
<td>1.498291</td>
<td>5.4776</td>
<td>21.47478</td>
</tr>
<tr>
<td>Gray</td>
<td>Gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.500198</td>
<td>1.509262</td>
<td>5.525238</td>
<td>21.52657</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>Inv. gray</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.498802</td>
<td>1.49379</td>
<td>5.480499</td>
<td>21.71319</td>
</tr>
<tr>
<td>Best</td>
<td>Best</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0.467408</td>
<td>1.459976</td>
<td>5.374146</td>
<td>20.97417</td>
</tr>
</tbody>
</table>

The Peak Signal to Noise Ratio (PSNR) is calculated using the equation:

$$\text{PSNR} = 10 \log_{10} \left( \frac{I_{max}^2}{\text{MSE}} \right) \text{dB}$$  \hspace{1cm} (12)

where, $I_{max}$ is the intensity value of each pixel which is equal to 255 for 8 bit grey scale images. Higher the values of PSNR better the image quality. The cover image is given in Fig. 3a and the corresponding stego images for k = 1 in Fig. 3b, k = 2 in Fig. 3c, k = 3 in Fig. 3d, k = 4 in Fig. 3f and the proposed stego results in Fig. 3g for k = 4, MSE, PSNR is given in Fig. 4 and 5, respectively.

In the case of simple LSB embedding for full embedding capacities 256×256 bits for k = 1, 256×256×2 bits for k = 2 and so on. While using secret data in binary format alone for k = 4 the MSE is 36.60, inverted binary is 36.90, gray is 41.17, inverted gray is 40.88 and the proposed is 34.84. The proposed method MSE by adapting quantum of 64 pixels further reduces it to 32.81. These values are shown in Table 1-3.

The corresponding PSNR value of the proposed method improved to 35.03826 dB which is for better than Chan and Cheng (2004) method PSNR of 34.8 dB. The corresponding MSE value of the proposed method reduced to 20.38 which is for better than Chan and Cheng (2004) method MSE of 21.6 and Yang (2008).

Image steganography is successfully implemented using a novel encoding method in which various bit
Fig. 3(a-g): (a) Cover Images Lena, Baboon, Gandhi and Temple, (b) $k = 1$, (c) $k = 2$, (d) $k = 3$, (e) $k = 4$, (f) $k = 4$ Proposed 256 Stego Images Lena, Baboon, Gandhi and Temple and (g) $k = 4$ Proposed 64 Stego Images Lena, Baboon, Gandhi and Temple
Table 2: Comparative MSE values for full embedding capacity on Lena and Baboon by splitting into 64 pixels as one block

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE for simple LSB substitution Lena image</th>
<th>MSE for simple LSB substitution Baboon image</th>
<th>PSNR for simple LSB substitution Lena image in dB</th>
<th>PSNR for simple LSB substitution Baboon image in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>0.4971 2.2100 9.0421 36.6052</td>
<td>0.4997 2.1849 9.0795 36.6056</td>
<td>Binary 51.1663 44.0867 38.5689 32.4953</td>
<td>Binary 51.1440 44.7365 38.5502 32.5606</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>0.5028 2.1987 8.9769 36.9659</td>
<td>0.5003 2.2141 9.0668 35.8387</td>
<td>Inv. binary 51.1159 44.7809 38.5985 32.4508</td>
<td>Inv. binary 51.1382 44.6787 38.5562 32.5873</td>
</tr>
<tr>
<td>Gray</td>
<td>0.4990 2.5090 9.8571 41.1694</td>
<td>0.4999 2.4795 9.8605 40.5180</td>
<td>Gray 51.1497 44.1357 38.1932 31.9850</td>
<td>Gray 51.1418 44.1871 38.1878 32.0543</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>0.5009 2.4888 9.8673 40.8816</td>
<td>0.5001 2.5086 9.9822 39.9203</td>
<td>Inv. gray 51.1325 44.1707 38.1888 32.0155</td>
<td>Inv. gray 51.1345 44.3727 38.7289 34.7953</td>
</tr>
<tr>
<td>Best</td>
<td>0.4381 1.4943 8.0310 32.8100</td>
<td>0.4356 1.4926 8.0661 32.1796</td>
<td>Best 51.7645 45.2319 39.0830 32.9707</td>
<td>Best 51.7403 45.2469 39.0642 33.0550</td>
</tr>
</tbody>
</table>

MSE after OPAP process Lena image

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE for simple LSB substitution Lena image</th>
<th>MSE for simple LSB substitution Baboon image</th>
<th>PSNR for simple LSB substitution Lena image in dB</th>
<th>PSNR for simple LSB substitution Baboon image in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>0.4971 1.4952 5.5365 21.5915</td>
<td>0.4997 1.4990 5.5185 21.5684</td>
<td>Binary 51.1663 46.3837 40.6984 34.7879</td>
<td>Binary 51.1440 46.3782 40.7126 34.7926</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>0.5028 1.5001 5.4632 21.6549</td>
<td>0.5003 1.5061 5.4880 21.3119</td>
<td>Inv. binary 51.1159 46.3693 40.7562 34.7752</td>
<td>Inv. binary 51.1382 46.3522 40.7367 34.8146</td>
</tr>
<tr>
<td>Gray</td>
<td>0.4999 1.4990 5.4679 21.5678</td>
<td>0.4999 1.5041 5.5056 21.6138</td>
<td>Gray 51.1497 46.3546 40.7272 34.8734</td>
<td>Gray 51.1418 46.3876 40.7526 34.7927</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>0.5001 1.4990 5.4984 21.5549</td>
<td>0.5001 1.5041 5.5061 21.4189</td>
<td>Inv. gray 51.1325 46.3727 40.7289 34.7953</td>
<td>Inv. gray 51.1404 46.3581 40.7223 34.8228</td>
</tr>
<tr>
<td>Best</td>
<td>0.4381 1.4943 8.0305 32.8123</td>
<td>0.4356 1.4926 8.0661 32.1796</td>
<td>Best 51.7645 45.2319 39.0830 32.9707</td>
<td>Best 51.7403 45.2469 39.0642 33.0550</td>
</tr>
</tbody>
</table>

MSE after OPAP process Baboon image

<table>
<thead>
<tr>
<th>k bit embedding</th>
<th>MSE for simple LSB substitution Lena image</th>
<th>MSE for simple LSB substitution Baboon image</th>
<th>PSNR for simple LSB substitution Lena image in dB</th>
<th>PSNR for simple LSB substitution Baboon image in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>Binary</td>
<td>0.4997 1.4990 5.5185 21.5684</td>
<td>0.4997 1.4990 5.5185 21.5684</td>
<td>Binary 51.1440 46.3782 40.7126 34.7926</td>
<td>Binary 51.1410 46.3782 40.7126 34.7926</td>
</tr>
<tr>
<td>Inv. binary</td>
<td>0.5003 1.5061 5.4880 21.3119</td>
<td>0.5003 1.5061 5.4880 21.3119</td>
<td>Inv. binary 51.1382 46.3522 40.7367 34.8146</td>
<td>Inv. binary 51.1382 46.3522 40.7367 34.8146</td>
</tr>
<tr>
<td>Gray</td>
<td>0.4999 1.4990 5.4679 21.5678</td>
<td>0.4999 1.5041 5.5056 21.6138</td>
<td>Gray 51.1497 46.3546 40.7272 34.8734</td>
<td>Gray 51.1418 46.3876 40.7526 34.7927</td>
</tr>
<tr>
<td>Inv. gray</td>
<td>0.5001 1.5041 5.5061 21.4189</td>
<td>0.5001 1.5041 5.5061 21.4189</td>
<td>Inv. gray 51.1404 46.3581 40.7223 34.8228</td>
<td>Inv. gray 51.1404 46.3581 40.7223 34.8228</td>
</tr>
<tr>
<td>Best</td>
<td>0.4356 1.4162 5.1879 20.2704</td>
<td>0.4356 1.4162 5.1879 20.2704</td>
<td>Best 51.7403 46.6194 40.9809 35.0622</td>
<td>Best 51.7403 46.6194 40.9809 35.0622</td>
</tr>
</tbody>
</table>

Table 3: Comparison of MSE values with other methods for full embedding capacity in Lena, Baboon, Gandhi and Temple

<table>
<thead>
<tr>
<th>K bit embedding</th>
<th>MSE for Lena image</th>
<th>MSE for Baboon image</th>
<th>MSE for Gandhi image</th>
<th>MSE for Temple image</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple LSB</td>
<td>0.4971 2.2100 9.0421 36.6052</td>
<td>Simple LSB</td>
<td>Proposed [256]</td>
<td>Proposed [64]</td>
</tr>
<tr>
<td>Best</td>
<td>0.4655 1.9493 8.0310 32.8100</td>
<td>Best</td>
<td>Proposed [256]</td>
<td>Proposed [64]</td>
</tr>
<tr>
<td>Chen and Cheng</td>
<td>0.4971 1.4952 5.5365 21.5915</td>
<td>Chen and Cheng</td>
<td>Proposed [256]</td>
<td>Proposed [64]</td>
</tr>
<tr>
<td>Thien and Lin</td>
<td>0.5003 1.4990 5.4084 21.5549</td>
<td>Thien and Lin</td>
<td>Proposed [256]</td>
<td>Proposed [64]</td>
</tr>
<tr>
<td>Yang</td>
<td>0.4655 1.4942 5.4370 20.9467</td>
<td>Yang</td>
<td>Proposed [256]</td>
<td>Proposed [64]</td>
</tr>
<tr>
<td>Proposed [64]</td>
<td>0.4655 1.4958 5.3638 20.9985</td>
<td>Proposed [64]</td>
<td>Proposed [64]</td>
<td>Proposed [64]</td>
</tr>
</tbody>
</table>

Fig. 4: Comparative MSE values for full embedding capacity on Lena

representations namely binary, inverted binary, gray and inverted gray are employed. Here the secret data, encoded in all the four representations is embedded in a row of the cover image and the MSE is calculated exclusively for each of the four encoding bit representations. Of the representations the one that yields the least MSE is adopted for the respective row. In this way all the four forms of representation are used in each row and the form resulting in the least MSE and PSNR is espoused and the results are given in Fig. 4 and 5.

Finally a key is formulated using a code to depict the bit representation format employed in each row which

436
Fig. 5: Comparative PSNR values for full embedding capacity on Lena

again is arcane thereby protecting the stego image from malicious aggressors.

Considering an image of dimensions 256×256, key bits per row is 2. Therefore, in order to account for 256 rows, we get 256×2 = 512 bits. These 512 bits of data form a secret key array. Thus, we can define key-to-data ratio as 512/(256×256×8) = 512/(65536×8) = 0.00098 = 0.098%. Furthermore one more experiment has been carried out to improve the quality of the stego image by splitting each row into 4 quantum units of 64 pixels. The results are encouraging with slight increase in the key length.

Since the embedding depends upon the Least Mean Square Error which is dynamically determined by the combination of cover image pixels and secret data bits, any attack to recover the data without using the secure key becomes impossible.

**COMPLEXITY ANALYSIS**

- The DES cryptography system introduces a complexity of $2^{26}$
  - For 256×256 pixel image, total number of rows will be 256
  - These 256 rows can be selected in a random manner in 256! Ways
  - In each row one embedding technique out of four is chosen
  - If we are embedding k bits in each pixel then
  - The total complexity = $2^{26} \times 256! \times 4^8/k$
  - So total complexity in this case will be $2^{26} \times 256! \times 4^8/k$

**For proposed 64 method:**

- The DES cryptography system introduces a complexity of $2^{24}$
  - For 256×256 pixel image, total number of rows will be 256
  - These 256 rows can be selected in a random manner in 256! Ways
  - Each row is grouped into 4 blocks of 64 pixels.
  - For each block a particular technique is selected out of 4
  - If we are embedding k bits in each pixel then
  - Total complexity is $2^{26} \times 256! \times 4^8/k$
  - If we select the blocks in each row in a random manner, there will be 4 blocks and we can select it in 4! Ways and if we select the pixels in a block in a random manner then
  - Total complexity is $2^{26} \times 256! \times 4^4 4!^8/k$
  - This security level estimation reveals the of the proposed stego against hackers

**CONCLUSION**

By simultaneously serving two ultimate requirements of security, i.e., greater imperceptibility (least MSE) and high complexity (cryptic effect created by the choice of row-wise embedding), the proposed technique promises un-tampered transmission and authorized use of secret data. Usage of nominal key length reduces the cost associated with the transport of key over a secure channel. To summarize the key points in this paper:

- An improved image steganographic method has been proposed, implemented and tested called Brownian motion of Binary and Gray-Binary and Gray Bits in Image for stego
  - It is a variation of Yang (2008) method with additional choices of Gray code and inverted Gray code along with binary and inverted binary. This provides four choices for the data to be embedded. Thus, it further reduces the effective Mean Square Error to make the stego image more imperceptible and also gives cryptic effect
  - A mathematical model has been developed to justify the work
  - The worst case Mean square error is derived $\text{MSE}_{\text{worse}} \leq (1/2) \text{MSE}_{\text{worse}}$ and the results are discussed in detail
  - This method reduces the MSE by a factor of 1.8, without compromising the data embedding capacity
and marginal improvement in imperceptibility. (In Information hiding with respect to magic triangle capacity, imperceptibility and Robustness). The proposed method will not consider robustness, because robustness will come for watermarking definitely not for spatial domain steganography

- Security analysis has been made to highlight its firmness against hackers
- Total complexity is $2^n*256!*4*4!/8/k$
- The work tested for 10 cover images, due to large data values, only four frequently used cover images are given in the result & discussion. Table 3 highlights the superiority of the proposed method with available literature
- Usage of nominal key length reduces the cost associated with the transport of key over a secure channel about 0.098% of the embedded text

**REFERENCES**


