ACA Based Image Steganography

Anindita Sarkar[1], Amitava Nag[1], Sushanta Biswas[2], and Partha Pratim Sarkar[2]

[1] Academy of Technology, West Bengal University of Technology / Hoogly 721212, India
[2] Department of Engineering and Technological Studies, University of Kalyani / Kalyani 741 235, India

***** Corresponding Author: Amitava Nag

Received July 14, 2013; Revised July 29, 2013; Accepted August 12, 2013; Published October 31, 2013

** Regular Paper*

***Abstract*:** LSB-based steganography is a simple and well known information hiding technique. In most LSB based techniques, a secret message is embedded into a specific position of LSB in the cover pixels. On the other hand, the main threat of LSB-based steganography is steganalysis. This paper proposes an asynchronous-cellular-automata(ACA)-based steganographic method, where secret bits are embedded into the selected position inside the cover pixel by ACA rule 51 and a secret key. As a result, it is very difficult for malicious users to retrieve a secret message from a cover image without knowing the secret key, even if the extraction algorithm is known. In addition, another layer of security is provided by almost random (rule-based) selection of a cover pixel for embedding using ACA and a different secret key. Finally, the experimental results show that the proposed method can be secured against the well-known steganalysis RS-attack

***Keywords*:** Information Hiding, Steganography, Steganalysis, RS-attack, Asynchronous cellular automata

1. Introduction

In recent years, the increasing popularity of digital media and communication networks have resulted the huge transmission of secret information over the Internet. On the other hand, the transmission of secret information through an open communication channel in the form of digital media can be replicated and tampered with easily by a malicious user. Therefore, protecting the content of sensitive and secure data from malicious users in an Internet environment has become a significant issue. Cryptography [1, 3] is a very old technique to scramble a message so it cannot be understood by unauthorized users. On the other hand, this can naturally raise the curiosity level of an eavesdropper. Therefore, it would be wiser to cleverly embed a secret message in another media so a secret message can be concealed from everyone. This idea forms the basis for steganography [4], which is a branch of information hiding by camouflaging secret information within covert carriers to avoid observation. Steganography is the art of hiding the presence of secret communication by embedding secret messages into innocent, innocuous looking cover documents, such as digital images [2, 6-9, 12], videos [15, 16], sound [13, 14] or documents [17, 18]. The stego-medium is the result of hiding a secret message in a cover-medium. Images provide excellent carriers for hidden information. Many different techniques have been introduced to embed messages in images.

The most common approaches for steganography in images are Least Significant Bit (LSB) modification [7, 8], where binary secret bits are hidden at specified position of the cover pixel. The basic drawback of this method is that the secret bits can be retrieved easily by malicious users because it does not require any calculations. In 2006, Mielikainen improved LSB replacement using a LSB matching revisited (LSBMR) method [8]. A binary function and four embedding rules were applied in LSBMR to hide two secret bits into a pair of pixels of a cover image. Zhang and Wang proposed a novel steganographic method to fully exploit the modification direction of LSBMR [2]. In [19], a reversible histogram transformation function (RHTF) steganography scheme based on a secret key was proposed. In 2011 [20], D.-C. Lou and C.-H. Hu discovered two vulnerabilities: “zero points” and “double frequency” of the RHTF proposed by [19]. These two vulnerabilities can distinguish the stego-images easily from cover images and brake the value of the secret key.

On the other hand, steganalysis is the science and art of detecting the possible existence of a secret message

embedded in a stego-image using steganography [4, 5]. Two popular steganalysis are Chi-squared ($\chi 2$ detection) [11] and Regular Singular(RS) attack [10], which are applied to detect the existence of a hidden message by finding statistical abnormalities in stego-media caused by message embedding.

2. Related Work

In this section, the RS steganalysis technique and the RHTF (Reversible Histogram Transformation Function) based LSB (Least Significant Bits) steganography are reviewed including Asynchronous Cellular Automata (ACA).

2.1 RS attack

RS-attack steganalysis proposed by Fridrich et. al. [10] is used to examine whether an image is a cover image or a stego-image and also gives the percentage probability of embedding. Currently, it is recognized as the most powerful and leading steganalysis technique for stego-images by LSB-based steganography. The principle of RS-attack is as follows:

Step 1: Select an m-tuple Mask M with values {-1, 0, 1}.

Step 2: Assign m adjacent pixels $x_1, x_2.....,x_n \in$ {0 to 255} to set $G_c= (x_1, x_2.....,x_n)$

Step 3: The smoothness of pixel group G_c is determined using the discrimination function f as

$$(x_1, x_2, \dots \dots \dots, x_n) = \sum_{0}^{n-1} |x_{i+1} - x_i| \qquad (1)$$

Step 4: An invertible mapping F_M, called flipping function is defined on [0 to 255] to flipping the pixel value according to M, i.e. F_1 for positive M and F_{-1} for negative M as,

$$F_1(x) = \begin{cases} x-1, & if x \bmod 2 = 1 \\ x+1, & if x \bmod 2 = 0 \end{cases} \qquad (2)$$

$$F_{-1}(x) = F_1(x+1) - 1 \qquad (3)$$

Step 5: If $F_M(G_c)$ is the result of flipping all the pixel values of group G_c and $f(F_M(G_c))$ is the result of a $F_M(G_c)$ input to the discrimination function, f, define three types of groups Regular (R), Singular(S) and Unusable (U) according to the following rules:

Regular Groups : $G_c \in R \Rightarrow f(F_M(G_c)) > f(G_c)$

Singular Groups: $G_c \in S \Rightarrow f(F_M(G_c)) < f(G_c)$

Unusable Groups: $G_c \in U \Rightarrow f(F_M(G_c)) = f(G_c)$

The same grouping was also applied to the negative Mask i.e. –M.

Step 6: Step 1 to Step 5 is repeated up to half of the total number of pixels and up to the total number of pixels of the stego- image.

Step 7: Similarly, for half of the total number of pixels, calculate $R_M(p/2),S_M(p/2),R_{-M}(p/2),S_{-M}(p/2)$. Similarly, for the total number of pixels, calculate $R_M(1-p/2),S_M(1-p/2),R_{-M}(1-p/2),S_{-M}(1-p/2)$, where p is the unknown length of the message in a stego-image (in percent of pixels).

Step 8: Obtain the value of x from the following quadratic equation,

$$2(d_1 + d_0)x^2 + (d_{-0} - d_1 - d_1 - 3d_0)x + d_0 - d_{-0} = 0 \qquad (4)$$

where

$$\begin{cases} d_0 = R_M\left(\frac{p}{2}\right) - S_M\left(\frac{P}{2}\right) \\ d_1 = R_M\left(1-\frac{p}{2}\right) - S_M\left(1-\frac{P}{2}\right) \\ d_{-0} = R_{-M}\left(\frac{p}{2}\right) - S_{-M}\left(\frac{P}{2}\right) \\ d_{-1} = R_{-M}\left(1-\frac{p}{2}\right) - S_{-M}\left(1-\frac{P}{2}\right) \end{cases} \qquad (5)$$

Step 9: p is estimated from x as $p = \frac{x}{\left(x - \frac{1}{2}\right)}$

2.2 Review of RHTF-based LSB steganography

In 2011, Lou et al. proposed RHTF-based LSB steganography on selected images to resist RS attack by grouping the pixels and adjusting the secret-key dynamically [20].

2.2.1 Embedding Algorithm

Step 1: Divide the cover image I_c into n groups of size $\frac{I_c}{n}$

Step 2: Generate n number of secret keys, a_j, using the following key generation algorithm

(a) generate a key by increasing a_j as $a_j = a_{j-1} +1$, until $a_j = a_U$.

(b) generate a key by decreasing a_j as $a_j = a_{j-1} +1$, until $a_j = a_L$.

(c) continue Steps 2 (a) and (b) until j = n.

where a initial value (a_1), upper bound(a_U) and lower bound (a_L) are predefined.

Step 3: Apply the compressing transformation technique to each cover pixel P as follows:

$$P_1 = P - \left\lfloor \frac{P}{a_j + 1} \right\rfloor$$

Step 4: Replace the LSB of$P_1$by a secret bit and is obtained.

Step 5: Apply the following formula to produce a stego pixel

$$P_s = P_1 - \left| \frac{P_1}{a_j} \right|$$

2.2.2 Extracting Algorithm

Step 1: Divide the stego-image I_s into n groups of size $\frac{I_s}{n}$.

Step 2: Generate n number of secret keys, a_j, using the following key generation algorithm

(a) generate a key by increasing a_j by $a_j = a_{j-1} + 1$, until $a_j = a_U$.

(b) generate a key by decreasing a_j by $a_j = a_{j-1} + 1$, until $a_j = a_L$.

(c) continue Steps 2 (a) and (b) until j = n.

where the initial value (a_1), upper bound (a_U) and lower bound (a_L) are predefined.

Step 3: Apply the compressing transformation technique to each cover pixel P as follows:

$$P_1 = P_s - \left\lfloor \frac{P_s}{a_j + 1} \right\rfloor$$

Step 4: Extract the LSB of P_1 as a secret bit

2.3 Asynchronous Cellular Automata

The cellular automata (CA) [21, 22], as proposed by Wolfram, consists of a lattice of cells, each of which stores a discrete variable at time t, which refers to the present state of the CA cell. The next state of a cell is affected by its current state and the current states of its neighbors at time t. In a 1-dimensional two-state 3-neighborhood (self, left and right neighbors) CA, the next state of each cell is expressed as:

$$S_{t+1}(i) = f(S_t(i-1); S_t(i); S_t(i+1))$$

where f is the next state function; $S_t(i-1)$, $S_t(i)$ and $S_t(i-1)$ are the present states of the left neighbor, self and right neighbor of the i^{th} CA cell at time t. The decimal equivalent of the 8 outputs is called `rule'. A two-state 3-neighborhood dependency has 2^8 (256) CA rules. The collection of the states of all cells ($S1_t$; $S2_t$, …, Sn_t) at time t is called a CA state on that time. If the left most and right most cells are the neighbors of each other (i.e. $S_t(0) = S_t(n)$ and $S_t(k) = S_t(1)$, where k = n+1, for CA with n cells), the CA are the periodic boundary CA. On the other hand, in the null boundary CA, $S_t(0) = S_t(k) = 0$ (null). Here, only the null boundary condition is considered.

If all the cells of CA update their states simultaneously, CA are synchronous. On the other hand, in asynchronous CA, the cells are updated independently. Therefore, with Asynchronous Cellular Automata (ACA), any number of ACA cells may be updated in a single time step [21].

The ACA are reversible [22] if all the CA states are cyclic, otherwise they are irreversible. The reversibility guarantees that each CA state has a unique predecessor and successor. Rule 51 has a special property in that it always allows a cell to flip its previous state when updated. The reversibility is possible using Rule 51.

3. The Proposed Scheme

In this section, the proposed image steganography technique using Asynchronous Cellular Automata (ACA) will be discussed. In the proposed method, first the cover image is decomposed into several blocks of the same size and the pixels are selected randomly inside the block using ACA. Next, the secret bits are embedded by selecting a random position from the 0^{th} LSB to the 5^{th} LSB in the cover pixel.

3.1 Generation of update pattern to construct a cycle

An update pattern [21] is generated for obtaining a cycle of length 2^i ($1 \le i \le n$) by updating a single cell at a time, where n is the number of ACA cells.

To get a cycle of length 2^i($1 \le i \le n$) of an ACA with n cells using rule 51, form a sequence of i cells, to be updated, arbitrarily. Generation of update pattern is always start with an arbitrary state. Update $(2^{j-1})^{th}$ state by updating j^{th} cell ($1 \le j \le i$) of the sequence to generate the next state. Repeat the update of the j^{th} cell after each 2^j state, where j < i. However, update the i^{th} cell (i.e. the last cell) again after 2^{i-1} state to get a cycle of length 2^i.

From the above rule, the update pattern U=<U_1, U_2, … U_n > is generated from the sequence SEQ=<q_1, q_2,…,q_n> using the following equation:

$$U_k = q_j, \text{where } k = \begin{cases} k + 2^j & if\ j < n \\ k + 2^{j-1} & if\ j = n \end{cases} \quad (6)$$

Example: The aim is to design a cycle of length $16(=2^4)$ for a 4-cell null-boundary rule 51 ACA. To generate the cycle, some cells need to be updated. This update pattern is generated from a sequence of 4 values of different position of the cells. Here, the sequence is taken as SEQ=<4, 1, 3, 2>. Table 1 lists the cell positions and their duration of repetition, and Table 2 shows the process of the change in states.

According to the previous rule, cell position 4 (where j=1) is to be updated $(2^{j-1})^{th}$ i.e. $2^{1-1}= 2^0= 1^{st}$ given state and it will be taken repeatedly after $2^j= 2^1=2$states. Next, where j=2 and the position of the cell is 1. Therefore, the first state that will be updated in cell position 1 is 2^{j-1}th, i.e. 2^{nd} state. Again, the next state updated in its cell position 1 is $2+2^j= 2+2^2$i.e.6^{th} state. When j=3 and position of the cell is 3, the first state updated in position 3 is the $2^{3-1}=4^{th}$ one. The update using the 3^{rd} cell position repeats after each 2^3(as j=3) = 8 states. The next state updated by 3(cell position) is the 4(the 1^{st} state updated in 3^{rd} cell)+8=12^{th} state. The last cell position is 2 where j=4. Therefore, the $(2^{4-1})^{th}$ = 8^{th} state is updated first in its 2^{nd}cell position. Because it is the last cell position, it will be repeated after 2^{4-1} = 8states.

Table 1. Duration of the repetition for the sequence.

Value ofj	Positions of cells	Duration of repetition
1	4	2^1=2
2	1	2^2=4
3	3	2^3=8
4	2	2^{4-1}=8

Table 2. Process of a change of state based on the update pattern.

State no.	Cell positions to update the state	States 4321	Decimal equivalent
1	4	0100	4
2	1	1100	12
3	4	1101	13
4	3	0101	5
5	4	0001	1
6	1	1001	9
7	4	1000	8
8	2	0000	0
9	4	0010	2
10	1	1010	10
11	4	1011	11
12	3	0011	3
13	4	0111	7
14	1	1111	15
15	4	1110	14
16	2	0110	6
1	4	0100	4 (repeat next cycle)

3.2 Generation of a Stego image using ACA

A stego-image can be generated with the help of ACA using the following steps:

Step 1: Select an M × N gray scale image I_c as a cover image and divide it into several non-overlapping blocks B_i of size $2^m \times 2^n$, where $1 \le m, n \le 4$ and i = 1, 2, ……., $\frac{M \times N}{2^m \times 2^n}$. The secret bits are embedded randomly in 2^{m+n} different pixels of each block, B_i, in the cover where pixels are selected randomly by applying ACA rule 51 and the update pattern. An (m + n) bits value R_1 is taken as the initial state of (m + n)-cell ACA to apply rule 51 for the generation of 2^{m+n} positions in each block B_k of the cover image, where R_1 can be represented as a (m + n) bit binary as follows:

$$R_1 = \sum_{i=1}^{m+n} b_{9-i}^1 \times 2^{i-1} \quad where \quad b_i \in \{0.1\} \tag{7}$$

Here R_1 is chosen as the first pixel for the insertion of the first secret bit in block B_1. To generate 2^{m+n} positions, an update-pattern V=<v_1, v_2,…..v_N> of length $N=2^{m+n}$ is needed. In this case, a sequence SEQ_B_1<q_1, q_2,…,q_{m+n}> of length (m + n) provided in the key is used to generate the update-pattern, V=<v_1,v_2,…..v_N> for the first block of the cover image B_1. The second position R_2 in the first block B_1 is generated from R_1 using rule 51 by flipping the bit in the v_1 position of R_1 as follows:

$$R_2 = \sum_{\substack{i=1 \\ i \neq 9-v_1}}^{m+n} b_{9-i}^1 \times 2^{i-1} + \overline{b_{v_1}^1} \times 2^{8-v_1} \tag{8}$$

In the case of the next block, B_2, the SEQ_B_2=<q_1, q_2,…,q_{m+n}> is changed using the cyclic rotation scheme as $q_{i+1}=q_i$ and $q_1=q_{m+n}$ and SEQ_B_2 is generated. From this new sequence, a new update pattern V=<v_1,v_2,…..v_N> is generated, which is used for block B_2. In this way, for each block B_k, a new SEQ_B_k and a new update pattern are generated.

Therefore, the update-pattern in block B_k generates position R_j in B_k as follows

$$R_j = \sum_{\substack{i=1 \\ i \neq 9-v_1}}^{m+n} b_{9-i}^{j-1} \times 2^{i-1} + \overline{b_{v_j}^{j-1}} \times 2^{8-v_j} \tag{9}$$

where $\bar{b}$ represents the compliments of bit b.

Step 2: For a 2^{m+n} number of update pattern, a sequence (SEQ_B_k) of length (m + n) is needed. Let secret message M be transformed into a bit-stream, in which each secret bit $m \in \{0,1\}$ and $B_j(x,y)$ represents a pixel located at coordinates (x,y) in the j^{th} block of the cover image I_c. The secret bits are embedded into pixel $P=B_j(x,y)$ by calculating the coordinate (x,y) from a random value R_j as follows:

$$x = \frac{R_j}{2^m} + 1 \quad and \quad y = mod\left(R_j, 2^n\right) + 1 \tag{10}$$

The pixel P can be represented in binary form as follows:

$$P = \sum_{i=1}^{8} b_i * 2^{i-1} \quad , where \quad b_i \in \{0.1\} \tag{11}$$

Step 3: To insert the secret bits, first with the help of a new sequence SEQ_P of size (m+n), another update-pattern U=<u_1,u_2,…..u_{256}> is generated. Embedding position pos in pixel P is calculated using the following

equation:

$$pos = \begin{cases} u_j - 1 & if\ u_j \le K \\ mod(u_j, 8) & otherwise \end{cases} \quad (12)$$

where K represents the embedding position from LSB in a pixel and 5≤ K ≤ 0. K≠6 or 7, because K=6 or 7 represents the first or second significant bit from the left (MSB), respectively, in a pixel. Therefore, embedding in those positions may cause massive distortions of the stego-image.

4: Let, m be a secret bit that is embedded in the pos position by

$$Q = \begin{cases} P & if\ b_{pos} = m \\ P_1' & Otherwise \end{cases} \quad (13)$$

where Q is a stego-pixel in location (x, y), b_{pos} represents the bit at position pos in pixel P and P_1' is modified value of P, which is calculated from P1 by following steps:

Step 4(a): When m ≠b_{pos}, P is modified to P_1 by Eq. (5) as follows:

$$P_1 = \sum_{\substack{i=1, \\ i \neq pos}}^{8} b_i * 2^{i-1} + m * 2^{8-pos}, \quad where\ \ b_i \in \{0.1\} \quad (14)$$

Step 4(b): P_1 is modified to P_1' in such a way that the bit at position pos in P_1 remains untouched and $|P_1' - P_1|$ is a minimum.

The sequence SEQ_P is also varied from one block to another. For the first block of the image SEQ_P is used, which is provided by the secret key. On the other hand, in the next block, every element q_i of the sequence SEQ_P is changed using Eq. (15)

$$q_i = mod(q_i, 8) + 1 \quad (15)$$

The updating of SEQ_B_k and SEQ_P before using every block provides the randomness of choosing the pixels for embedding inside the block as well as choosing the bit position inside every pixel in that block. Fig. 1 presents the process of embedding.

Steps 3 and 4 are used to embed a single bit into selected cover pixel. To insert multiple bits per pixel, steps 3 and 4 need to be executed multiple times using the different sequence SEQ_P to avoid collision.

An example of the proposed embedding algorithm using a 4 × 4 block B_i of the cover image is given below:

$$B_i = \begin{bmatrix} 83 & 88 & 86 & 90 \\ 178 & 97 & 98 & 102 \\ 63 & 78 & 79 & 105 \\ 176 & 186 & 178 & 220 \end{bmatrix}$$

The following Table 3 describes how the pixels for insertion are chosen from a 4×4 block of pixels using the above embedding algorithm. To achieve this purpose, the update pattern, V=< 4, 1, 4, 3, 4, 1, 4, 2, 4, 1, 4, 3, 4, 1, 4, 2>, is generated from the SEQ_B =<4, 1, 3, 2>. The first state 0100 is chosen randomly. The states are updated according to the update pattern. From the decimal equivalent R_j of the newly generated state, the position of the pixel (x, y) can be determined easily by calculating $x = \frac{R_j}{2^m} + 1$ and $y = mod(R_j, 2^n) + 1$, where m=n=2 for a 4×4 block. The way the pixels are chosen from the block is almost random and provides a layer of security.

Table 3. Generation of the pixel. location(x,y) based on the update pattern (V).

Update Pattern (V)	States (4 3 2 1)	Decimal equivalent (Rj)	Coordinate (x,y)inside a 4×4 Block	Pixel at position(x, y)
v1=4	0100	4	(2,1)	178
v2=1	1100	12	(4,1)	176
v3=4	1101	13	(4,2)	186
v4=2	0101	5	(2,2)	97
v5=4	0001	1	(1,2)	88
v6=1	1001	9	(3,2)	78
v7=4	1000	8	(3,1)	63
v8=3	0000	0	(1,1)	83
v1=4	0100 (will be used to generate the next state)	2	(1,3)	86
In a similar manner the other pixels are chosen in the block.				

Let the message bits be inserted into the chosen pixels in any of the 5 bits from its LSB (indicates k=5). The position of embedding inside every pixel is chosen from a new update pattern, U={3, 5, 3, 6, 3, 5, 3, 8, 3, 5, 3, 6, 3, 8} generated from another sequence SEQ_P={3, 5, 6, 8} using the formula generated from the update pattern from a given sequence. If the value of the update pattern is u_r<=k, the embedding position pos=u_r -1. If u_r >k the embedding position becomes mod(u_r, k). After inserting the message bit inside every pixel, the stego pixel is modified without changing the inserted bit so that the difference between the cover and stego pixel is a minimum. The position of embedding the message bit is chosen almost randomly, which provides another layer of security.

After embedding the message bits stream 1111110110001010 on B_i, the produced stego-block B' is also shown as follows:

Table 4. Generation of insertion position in a selected pixel based on update pattern (U).

Update Pattern (U)	Pixel at(x.y) in 4X4 block	Binary Equivalent of pixels 76543210	Message bit	Insertion position(for k=5)	After Embedding 76543210	Decimal equivalent	Modified value	Difference between cover and stego-image pixel
u1=3	P1=178	10110010	1	u1<k, pos1=u1-1 =2	10110110	182	Q1=180 (10110100)	|P1-Q1|=2
u2=5	P2=176	10110000	1	u2=k, pos2=u2-1=4	10110000	176	Q2=176 (10110000)	|P2-Q2|=0
u3=3	P3=186	10111010	1	u3<k, pos3=u3-1=2	10111110	190	Q3=188 (10111100)	|P3-Q3|=2
u4=6	P4=97	01100001	1	u4>k, pos4=u4%5=1	01100011	99	Q4=98 (01100010)	|P4-Q4|=1
u5=3	P5=88	01011000	1	u5<k, pos5=u5-1=2	01011100	92	Q5=87 (01010111)	|P5-Q5|=1
u6=5	P6=78	01001110	1	u6=k, pos6=u6-1=4	01011110	94	Q6=80 (01010000)	|P6-Q6|=2
u7=3	P7=63	00111111	0	u7<k, pos7=u7-1=2	00111011	59	Q7=64 (01000000)	|P7-Q7|=1
u8=8	P8=83	01010011	1	u8>k, pos8=u8%k=3	01011011	91	Q8=79 (01001111)	|P8-Q8|=2
Further message bits 10001010 are inserted in random inside the other pixels chosen randomly from the block.								

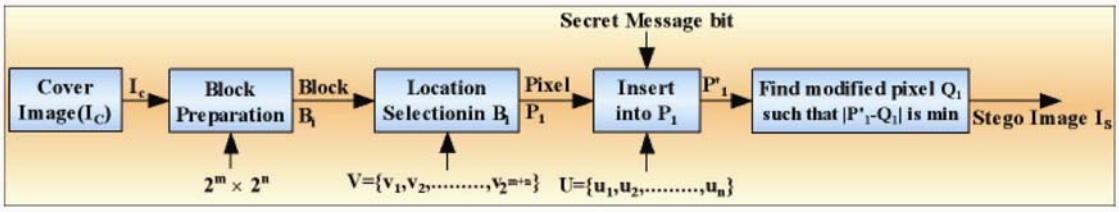

Fig. 1. Diagram of the proposed Embedding scheme.

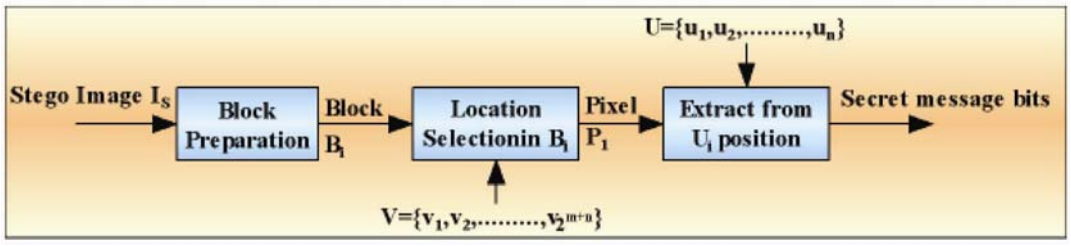

Fig. 2. Diagram of the proposed Extraction scheme.

$$B' = \begin{bmatrix} 79 & 87 & 86 & 89 \\ 180 & 98 & 98 & 102 \\ 64 & 80 & 79 & 105 \\ 176 & 188 & 180 & 224 \end{bmatrix}$$

Further message bits are inserted in the next block B_{i+1}. Before embedding the message bits in the pixels of blockB_{i+1}, the SEQ_B=<q_1, q_2, q_3, q_4> is changed using the formula $q_{i+1}=q_i$ and $q_1=q_{m+n}$. Therefore, the new SEQ_B is <2, 4, 1, 3> and the new update-pattern generated from the new SEQ_B is <2, 4, 2, 1, 2, 4, 2, 3, 2, 4, 2, 1, 2, 4, 2, 3>. SEQ_P=<q_1, q_2, q_3, q_4> is also updated by using q_i=mod(q_i,8)+1 and the new SEQ_P is<4, 6, 7, 1>.

3.4 Extraction of the Secret Message

In the extraction phase, first the stego-image is divided into blocks B_i. Using ACA and with the help of a stego-key ($V = < v_1,v_2...,v_2^{m+n} >$), the locations for the extraction of secret bits in B_i are selected and secret bits are extracted from the selected pixels using ACA and another set of secret keys ($U = < u_1,u_2...,u_2^{m+n} >$), as shown in Fig. 2.

First the key is extracted from the stego image. This key consists of SEQ_B_i, SEQ_P, m, n, a random number and k. From SEQ_B_i, the update pattern V=<$v_1,v_2,…..v_N$> can be generated easily, where $N=2^{m+n}$ using Eq. (6). The binary representation of the random number is the first state of ACA, which is used to generate the next states

using the update pattern, V. From the decimal value of every state, the coordinate of pixels (x, y) can be generated easily, which is chosen for embedding inside every block of size, $2^m \times 2^n$ using Eq. (10). The update pattern $U=<u_1,u_2,\ldots u_N>$ can also be found. Every value of the update pattern U determines the embedding position of the message bit inside the pixel. This bit position is generated using Eq. (12). Now the message bit is extracted and stored in a file.

EXAMPLE: Let the stego block be

$$B' = \begin{bmatrix} 79 & 87 & 86 & 89 \\ 180 & 98 & 98 & 102 \\ 64 & 80 & 79 & 105 \\ 176 & 188 & 180 & 224 \end{bmatrix}$$

From the key, the $SEQ\_B_i =< 4, 1, 2, 3>$, SEQ_P={3, 5, 6, 8}, m=2, n=2, the random number R=4 and k=5 are obtained.

The random number indicates it to be the first state of ACA. The received data of m and n indicates the block size is $2^2 \times 2^2$.Therefore,the number of pixels inside the block is 16 and also the total number of states is 16. Therefore, the first can be represented as 0100(binary equivalent of 4 in four bits). Using Eq. (6) and SEQ_B_i, the generated updated pattern is V=< 4, 1, 4, 3, 4, 1, 4, 2, 4, 1, 4, 3, 4, 1, 4, 2>

From SEQ_P, another update pattern U=<3, 5, 3, 6, 3, 5, 3, 8, 3, 5, 3, 6, 3, 8> is obtained. With the help of this update pattern U, the position for the extraction of secret bits inside a pixel is generated. Finally the secret bits are extracted from this selected position. Table 6 gives an example of the extraction of a secret message bits with the help of update pattern U.

4. Performance Evaluation

4.1 Security Analysis by RS attack

The RS attack steganalysis can retrieve the probability of embedding even when random embedding is used. This is achieved by grouping the pixels into Regular, Singular and Unusable groups. To show that RS attack cannot work in the proposed method, the following mathematical analysis is used, taking a 4 × 4 matrix as an example from the stego image obtained by applying the proposed method.

Select the mask (M) = [0 1 1 0]

r_1 = no. of regular groups for M

And Negative mask(-M)= [0 -1 -1 0]

r_2 = no. of regular groups for (–M)

Here we assume, N=4, initially set $r_i = s_i = 0$, $1<=i<=2$

s_1 = no. of singular groups for M

s_2 = no. of regular groups for (–M)

$$I' = \begin{bmatrix} 79 & 87 & 86 & 89 \\ 180 & 98 & 98 & 102 \\ 64 & 80 & 79 & 105 \\ 176 & 188 & 180 & 224 \end{bmatrix}$$

Table 5. Generation of the pixel location(x, y) for extraction based on the update pattern (V).

Update Pattern (V)	States 4321	Decimal equivalent (R_j)	Coordinate (x,y) inside a 4X4 Block	Pixel at position (x,y)
v_1=4	0100	4	(2,1)	180
v_2=1	1100	12	(4,1)	176
v_3=4	1101	13	(4,2)	188
v_4=2	0101	5	(2,2)	98
v_5=4	0001	1	(1,2)	87
v_6=1	1001	9	(3,2)	80
v_7=4	1000	8	(3,1)	64
In a similar manner, the other pixels are chosen in the block.				

Table 6. Generation of the extraction position in a selected pixel based on the update pattern (U).

Update pattern (U)	Embedded position	Pixel at (x,y)	Binary representation 76543210	Extracted Message bit
u1=3	u1<k,pos1=u1-1 =2	180	10110100	1
u2=5	u2=k,pos2=u2-1=4	176	10110000	1
u3=3	u3<k,pos3=u3-1=2	188	10111100	1
u4=6	u4>k,pos4=u4%5=1	98	01100010	1
u5=3	u5<k,pos5=u5-1=2	87	01010111	1
u6=5	u6=k,pos6=u6-1=4	80	01010000	1
u7=3	u7<k,pos7=u7-1=2	64	01000000	0
Further message bits are extracted is this way				

For total group/2:
In iteration 1, G_c={79,87,86,89} and f(G_C) = 12
$F_M(G_C)$ = {79,88,87,89} and f($F_M(G_C)$)=12
Therefore f($F_M(G_C)$) = f(G_C)
Hence, **it belongs to the Unusable Group.**
Again $F_{-M}(G_c)$= {79,87,86,89} and f($F_{-M}(G_c)$) = 12
Therefore f($F_{-M}(G_c)$) = f(G_C)
Hence, **it belongs to the Unusable Group**

In iteration 2, G_c={180, 98, 98, 102 } and f(G_C) = 86
$F_M(G_C)$ = {180, 99, 99, 102} and f($F_M(G_C)$) = 84
Therefore f($F_M(G_C)$) < f(G_C)
Hence, **Singular Group** set s_1**= 1**
Again $F_{-M}(G_c)$ = {180, 97, 97, 102} and
f($F_{-M}(G_c)$) = 88
Therefore f($F_{-M}(G_c)$) > f(G_C)
Hence, **Regular Group** set r_2**=1.**

RM = r_1/total group = 0
RM' = r_2/total group = 1/2
SM = s_1/total group = 1/2
SM'= s_2/total group = 0
d_0 = (RM – SM) = (0 – 1/2) = – 1/2
d_{-0}= (RM' – SM') = (1/2 – 0) = 1/2

For total group:-
From Iteration 1, r_1 = 0,r_2 = 1
From Iteration 2,s_1 = 1, s_2 = 0

In iteration 3, G_c={ 64, 80, 79, 105 } and f(G_C) = 43
$F_M(G_C)$ = { 64, 81, 80, 105 } and f($F_M(G_C)$) = 43
Therefore f($F_M(G_C)$) = f(G_C)
Hence, **it belongs to Unusable Group**.

Again $F_{-M}(G_c)$ = { 64, 80, 79, 105 } and f($F_{-M}(G_c)$) = 43
Therefore f($F_{-M}(G_c)$) = f(G_C)
Hence, **it belongs to Unusable Group**.
In iteration 4, G_c={176, 188, 180, 224 } and f(G_C) = 64
$F_M(G_C)$ = {176, 187, 179, 224 } and f($F_M(G_C)$) = 64
Therefore f($F_M(G_C)$) = f(G_C)
Hence, **it belongs to the Unusable Group**.

Again $F_{-M}(G_c)$ = {176, 189, 181, 224 } and f($F_{-M}(G_c)$) = 64
Therefore f($F_{-M}(G_c)$) = f(G_C)
Hence, **it belongs to the Unusable Group**.

RM = r_1/total group = 0/4 = 0
RM' = r_2/total group = 1/4=1/4
SM = s_1/total group = 1/4 = 1/4
SM'= s_2/total group = 0/4 = 0

d_1 = (RM – SM) = (0 – 1/4) = – 1/4
d_{-1}= (RM' – SM') = (1/4 – 0) = 1/4

$$2(d_1 + d_0)x^2 + (d_{-0} - d_{-1} - d_1 - 3d_0)x + d_0 - d_{-0} = 0$$

$$2\left(-\frac{1}{4}-\frac{1}{2}\right)x^2 + \left(\frac{1}{2}-\frac{1}{4}+\frac{1}{4}+\frac{3}{2}\right)x - \frac{1}{2}-\frac{1}{2} = 0$$

$$3x^2 - 4x + 2 = 0$$

x= imaginary

Hence the probability of embedding,
$P = x / (x - 0.5)$ = **imaginary**

4.2 Experimental Results

Some standard gray scale images of size 512×512 are used as a cover image. To measure the quality of the stego images, the Peak Signal to Noise Ratio (PSNR) was applied to compare the visual quality between the cover image and stego-image. The definition of PSNR is given as follows:

$$PSNR\,(dB) = 20\log_{10}\frac{255}{\sqrt{MSE}} \tag{16}$$

MSE is the mean squared error between the original image and modified image, which is defined as

$$MSE = \frac{1}{M \times N}\sum_{x=1}^{M}\sum_{y=1}^{N}\left(I(x,y) - I'(x,y)\right)^2 \tag{17}$$

In the experimental results, eight commonly images were used as cover images, as shown in Table 7.

Table 8 lists the PSNR values of LSB, RHTF-based LSB and the scheme for 90% embedding. The average PSNR of the proposed method is better than the LSB and RHTF method.

4.3 RS-Attack Analysis

Fig. 3 shows the RS-attack result of the gray stego-Lena image with a size of 512×512 using the proposed method. The X-axis represents the percentage of embedding and the Y-axis represents relative percentage of the regular and singular groups with masks M = [0 1 1 0] and –M = [0 -1 -1 0]. Fig. 3 shows that the expected R_m value is similar to the R_{-m} value, and S_m= S_{-m}. The other stego-images generated by the proposed method were also tested with similar results to those shown in Fig. 3. Therefore, the proposed method is secure against the RS-steganalysis.

4.4 Calculation of Key Size

The cover image is divided into blocks of size $2^m \times 2^n$, where $1 \le m \le 4$ and $1 \le n \le 4$. Therefore, the value of m and n will be a part of the key. The values 1 to 4 can be mapped directly with 0 to 3, and every value can be represented in 2 bits. Therefore, for m and n, 4 bits are needed. Initially a random number is required as an initial state, the number of bits used to represent that random number is (m + n) bits. In SEQ_B, there are (m + n) sequence values; every sequence value in SEQ_B requires $[\log_2(m+n)]$ bits. Therefore, a total of $[(m+n).\log_2(m+n)]$ bits are needed for (m + n) sequence values. In the sequence SEQ_P there are also (m + n) sequence values. SEQ_P is used to select the bit position

Table 7. PSNR of eight different images with respect to the embedding rate in terms of the bits per pixel(bpp).

Images		Capacity of embedding (bpp)				
		0.5	0.75	1.0	1.5	2.0
Tiffany	K=4	53.0643	51.3153	50.1357	45.1975	42.0724
	K=5	51.2550	49.4763	48.1287	43.6652	40.3183
	K=6	47.5028	45.8151	44.1074	41.1318	38.0245
Pepper	K=4	52.8415	51.1672	49.9950	45.2086	42.1273
	K=5	50.9742	49.3115	47.7845	43.1573	40.0824
	K=6	46.1063	44.3569	43.4214	39.8818	36.5914
Lena	K=4	53.1908	51.4407	50.2357	44.5394	41.3941
	K=5	51.5786	49.5805	48.2170	42.5554	39.4727
	K=6	47.4	45.3827	44.2758	38.7966	36.2157
Airplane	K=4	52.5783	50.8871	49.7259	44.1436	41.2177
	K=5	51.1342	49.4264	47.8885	42.7353	39.5897
	K=6	46.2242	44.9591	43.8837	39.094	36.3064
Mandrill	K=4	52.6136	50.8173	49.6581	43.933	40.8918
	K=5	49.7014	48.1922	47.1100	42.3766	39.2843
	K=6	46.5623	44.9552	43.7676	38.9306	36.1775
Boat	K=4	51.9553	50.1586	48.8298	44.8736	41.891
	K=5	49.8079	48.0559	46.8304	42.9637	39.9814
	K=6	46.1652	44.2376	43.1139	39.2067	36.4177
Elaine	K=4	53.5678	51.7789	50.5209	44.6184	41.6148
	K=5	52.8277	51.1112	49.9117	42.7815	39.8877
	K=6	50.6431	48.8389	47.6348	39.9078	36.7099
Couple	K=4	53.9081	52.133	50.8685	45.0399	42.038
	K=5	53.2187	51.4273	50.1182	43.2219	40.2287
	K=6	50.75	49.2077	47.9721	39.573	36.574

Table 8. PSNR of LSB,RHTF based LSB and proposed method for 90% embedding.

over-images (512×512)	LSB	RHTF based LSB [20]	Our method (K=4)
	PSNR (dB)	PSNR (dB)	PSNR (dB)
Airplane	50.2441	50.2273	50.4535
Baboon	49.9268	49.9983	50.1105
Barbara	50.7814	50.1712	51.0286
Boat	51.1182	51.1892	51.4029
Couple	51.1192	51.1715	51.3454
Goldhill	50.4200	50.5482	50.7195
Lena	51.1189	50.7136	50.9632
Man	50.6599	50.3456	50.8553
Peppers	50.2590	50.2675	51.3580
Stream	50.9532	51.0419	51.4000
Average	50.6600	50.5674	50.9637

for insertion inside the pixel. As the pixel value must be between 0 and 255, each of them can be represented in 8 bits. Therefore, the range of sequence values is 1 to 8, which can be mapped with 0 to 7. Each value of the sequence can be represented in 3 bits. Therefore, (m + n) values of SEQ_P requires 3.(m + n) bits. Accordingly, the key size is:

$$4 + (m+n) + (m+n).\log_2(m+n) + 3.(m+n)$$
$$= 4 + 4(m+n) + (m+n).\log_2(m+n)$$

The minimum key size is 14 and the maximum is 60. The size of the key of the proposed method was varied from 14 bits to 60 bits. On the other hand, the size of RHTF-LSB [20] was32 bits only. A variable key size makes the method more robust against the retrieval of

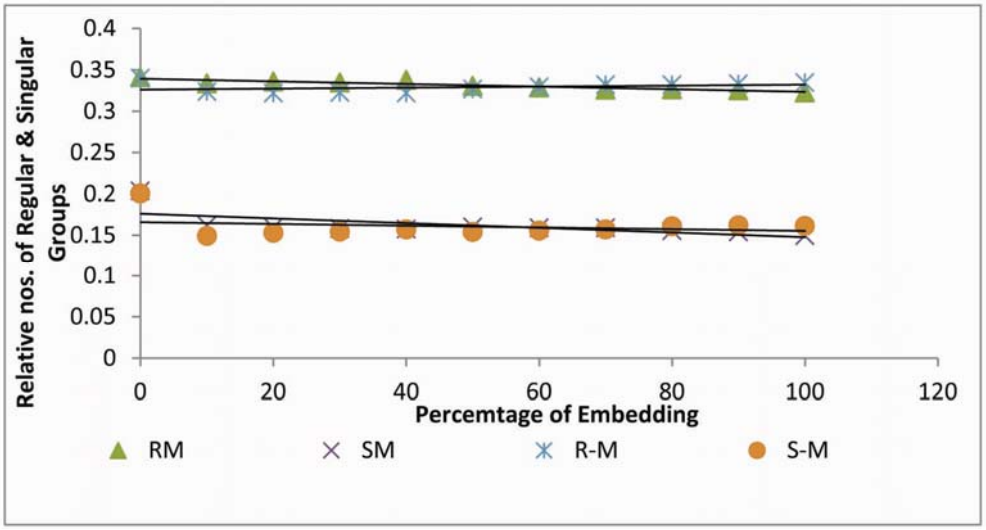

Fig. 3. RS diagram of a gray stego-Lena image with a size of 512×512 using the proposed method.

Table 9. Size of the Stego-key.

Embedding Scheme	Size
RHTF-LSB [20]	32 bits
Proposed Scheme	$4 + 4(m+n) + (m+n).\log_2(m+$ Min: 14 Max: 60

hidden information. Table 9 lists the key-size of the proposed method and RHTF-LSB scheme.

5. Conclusion

This paper proposes a secure and efficient LSB-based steganographic method by ACA to embed secret bits into any type of cover image. The main advantage of this method is that it does not require cover selection and provides sufficient security by creating dynamic secret-key using ACA. The experimental results of the proposed method showed that a large amount of secret data can be embedded into the cover images without creating noticeable distortions. Security analysis also showed that the proposed method can resist RS-attack steganalysis.

References

[1] William Stallings, Cryptography and Network Security – Principles and Practice, 4th ed. Pearson Education Pvt. Ltd., Indian, 2004.

[2] X. Zhang and S. Wang, Efficient steganographic embedding by exploiting modification direction, IEEE Communications Letters, vol 10, no. 11, pp. 1–3, 2006. Article (CrossRef Link)

[3] National Bureau of Standard(U.S), "Data Encryption Standard (DES)", *Federal Information Processing Technical Information Service, Springfield,* VA,1997. Article (CrossRef Link)

[4] A. Chedded, J. Condell, K. Curran and P. M. Kevitt, "Digital Image Steganography: Survey and Analysis of current methods", *Signal Processing* 90,727 – 752, 2010. Article (CrossRef Link)

[5] A. Nissar and A.H.Mir, "Classification of Steganalysis Techniques: A study", *DigitalSignal Processing* 20, 1758 – 1770, 2010. Article (CrossRef Link)

[6] H. Luo, F.-X. Yu, H. Chen, Z.-L. Huang, H. Li, P.-H. Wang, "Reversible data hiding based on block median preservation", *Information Science*, 181 (2) (2011) 308– 328. Article (CrossRef Link)

[7] W. Luo, F. Huang, J. Huang, "Edge adaptive image steganography based on lsb matching revisited", *IEEE Transactionon Information Forensics and Security*, 5 (2) (2010) 201–214. Article (CrossRef Link)

[8] J. Mielikainen, "LSB Matching Revisited", *IEEE Signal Processing Letters* 13 (5) (2006) 285–287. Article (CrossRef Link)

[9] Chin-Chen Chang, Hsien-Wen Tseng, "A steganographic method for digital images using side match",Pattern Recognition Letters, Volume 25, Issue 12, September 2004, Pages 1431-1437. Article (CrossRef Link)

[10] J. Fridrich, M. Goljan, R. Du, "Reliable detection of LSB steganography in color and grayscale images", *Proceedings ACM Workshop Multimedia and Security*, 2001, pp. 27–30. Article (CrossRef Link)

[11] A. Westfeld, A. Pfitzmann, "Attacks on steganographic systems", *Proceedings of the 3rd International Workshop on Information Hiding*, Dresden,Germany, 1999, pp. 61–76. Article (CrossRef Link)

[12] H. Wang & S.Wang, "Cyber warfare: steganography vs.steganalysis", *Communication of the ACM*, 47(10), 76–82. Article (CrossRef Link)

[13] R. Petrovic, J.M. Winograd, K. Jemili, E. Metois, "Data hiding within audio signals", Proceedings of the 4th International Conference on Telecommuni-

cations in Modern Satellite, Cable and Broadcasting Services, 1999, pp. 88–95. Article (CrossRef Link)
[14] P. Shah, P. Choudhari, S. Sivaraman, "Adaptive wavelet packet based audio steganography using data history",*Proceedings of the 2008 IEEE Region 10 and the 3rd International Conference on Industrial and Information Systems,* 2008, pp. 1–5. Article (CrossRef Link)
[15] A.A. Hanafy, G.I. Salama, Y.Z. Mohasseb, "A secure covert communication model based on video steganography", *Proceedings of the 2008 IEEE Military Communications Conference*, 2008, pp. 1–6. Article (CrossRef Link)
[16] B. Wang, J. Feng, "A chaos-based steganography algorithm for H.264 standard video sequences", *Proceedings of the 2008 International Conference Communications, Circuits and Systems*, 2008, pp. 750–753. Article (CrossRef Link)
[17] T.-Y. Liu, W.-H. Tsai, "A new steganographic method for data hiding in Microsoft Word documents by a change tracking technique", *IEEE Transaction on Information Forensics and Security* 2 (1) (2007) 24–30. Article (CrossRef Link)
[18] M.H. Shirali-Shahreza,M. Shirali-Shahreza, "A new approach to Persian/Arbic text steganography", *Proceedings of Fifth IEEE/ACIS International Conference on Computer and Information Science,* 10 -12 July 2006, pp. 310–315. Article (CrossRef Link)
[19] A.R.S. Marçal, P.R. Pereira, "A steganographic method for digital images robust to RS steganalysis", *International Conference on Image Analysis and Recognition*, *Toronto, Canada, Lecture Notes in Computer Science*, vol. 3656, 2005, pp. 1192–1199. Article (CrossRef Link)
[20] D.-C. Lou, C.-H. Hu, "LSB steganographic method based on reversible histogram transformation function for resisting statistical steganalysis", *Information Science* (2011), Article (CrossRef Link)
[21] Anindita Sarkar and Sukanta Das, On the Reversibility of 1-dimensional Asynchronous Cellular Automata. AUTOMATA 2011: 29-40. Article (CrossRef Link)
[22] Sarkar, A., Mukherjee, A., Das, S.: Reversibility in asynchronous cellular automata,Complex Systems,pp. 72 – 84. Article (CrossRef Link)

Anindita Sarkar is Assistant Professor of Information Technology at the Academy of Technology, India. She received her B.Tech and M.Tech degrees in Information Technology from University of Kalyani and Bengal Engineering and Science University, India, in 2009 and 2011, respectively Her research interests include Security and Asynchronous Cellular Automata. She is a member of the IEEE.

Amitava Nag obtained his M.Tech from University of Calcutta in the year 2005. He earned his B.Techfrom Dept. of Engineering & Technological Studies, University of Kalyani in the year 2003. He is presently working as an Assistant Professor in Academy of Technology, India and also working towards his PhD at the Dept. of Engineering & Technological Studies, University of Kalyani. He is a member of IEEE and CSI, India. His area of interest includes Cryptography and steganography.

Dr. S. Biswas obtained his Ph.D in engineering from Jadavpur University in the year 2004. He obtained his M.E from Jadavpur University and B.E from Bengal Engineering College (Presently known as Bengal Engineering and Science University, Shibpur) in the year 1994 and 1990 respectively. He is presently working as Scientific Officer (Associate Professor Rank) at the Dept. of Engineering & Technological Studies, University of Kalyani. He has more than 14 years of teaching experience. His area of interest includes, Artificial Neural Network, Image Processing, Frequency Selective Surfaces, Microstrip Antennas.

Dr. Partha Pratim Sarkar obtained his Ph.D in engineering from Jadavpur University in the year 2002. He has obtained his M.E from Jadavpur University in the year 1994. He earned his B.E degree in Electronics and Telecommunication Engineering from Bengal Engineering College (Presently known as Bengal Engineering and Science University, Shibpur) in the year 1991. He is presently working as Senior Scientific Officer (Professor Rank) at the Dept. of Engineering & Technological Studies, University of Kalyani. His area of research includes, Microstrip Antenna, Microstrip Filter, Frequency Selective Surfaces, and Artificial Neural Network. He has contributed to numerous research articles in various journals and conferences of repute. He is also a life Fellow of IETE.