Improving Data Hiding Capacity Using Bit-Plane Slicing of Color Image Through (7, 4) Hamming Code

Ananya Banerjee(✉) and Biswapati Jana

Department of Computer Science, Vidyasagar University, Midnapore 721102, West Bengal, India
{anaanya.2011,biswapatijana}@gmail.com

Abstract. Achievement of high-capacity data hiding with good visual quality is an important research issue in the field of steganography. In this paper, we have introduced RGB color image and bit-plane slicing for data hiding through Hamming code using shared secret key. We partitioned the color image into (3×3) pixel blocks and then decomposed into three basic color blocks. Again each color blocks are sliced up to four bit-plane starting from LSB plane. Now, a segment of three bits secret data is embedded within each bit-plane depending on a syndrome calculated using hamming code. As a result, 36 bits secret data can be embedded within (3×3) pixel block and achieve a high payload capacity with good visual quality compared with existing schemes.

Keywords: Steganography · Hamming code · Least significant bit (LSB) Bit-plane · Data hiding

1 Introduction

Steganography is the art and science of hidden data communication. Many data hiding schemes [1–10] are developed in last few decades. Some of them [6, 9, 10] are better in terms of security and imperceptibility. The data hiding schemes are useful in many application areas to solve the problem of ownership identification, copyright protection, authentication, verification, and more. The main aims of data hiding schemes are to ensure extraction of secret data and recovery of original object from stego media. On the other hand, data should stay hidden in stego image even if the eavesdropper tampered the stego or degrading through natural phenomenon like transmission resampling, compression or filtering, etc. The main drawbacks of data hiding schemes are not to provide a good solution in such cases. The degree of distortion will be high due to increase of data embedding capacity that should be balanced mathematically using spread spectrum. The data embedding in color image is considered to be more unsuspicious and secured, and less exploration has been done till today in this research area using hamming code.

Data hiding through matrix coding has been introduced by Crandall [2], and Westfield [7] implemented F5 algorithm based on matrix encoding using hamming code. Kim and Shin [8] suggested a data embedding procedure for halftone image. The

V. Bhateja et al. (eds.), *Information Systems Design and Intelligent Applications*, Advances in Intelligent Systems and Computing 672,
https://doi.org/10.1007/978-981-10-7512-4_36

scheme provides good capacity but poor visual quality. Based on matrix encoding, 'Hamming +1' method has been developed by Zhang et al. [3]. The embedding capacity is increased in 'Hamming +1' scheme by $\frac{k+1}{2^k}$ bpp. Chang [5] suggested an improved data embedding procedure using Hamming code which can hide (k + 1) bits of message in 2^k pixels with at most one change. Kim and Yang [9] developed 'Hamming + 3' data hiding scheme which can embed k + 3 bits within $2^k - 1$ pixels by at most two change. In 2016, Cao et al. [10] proposed their algorithm which can preserve stego image quality under high embedding capacity. The visual quality and data embedding capacity of Cao et al. are 37.90 dB PSNR and payload 3 bpp, respectively. Also Jana et al. [11] proposed reversible data hiding scheme through dual image with 53 dB PSNR and payload 0.14 bpp. After studying this literature, we have found that there is a chance to improve data embedding capacity and visual quality through Hamming code for color images. Here, we have proposed an improved information hiding scheme using (7, 4) Hamming code for color images. We have divided R, G, and B color pixels in bit-plane [12] starting from LSB to LSB-3 (up to four bit-plane) into (3 × 3) blocks and then applied hamming code-based data hiding scheme. In this scheme, 36 bits are embedded within 9 pixels which are more higher than other existing hamming code-based scheme, and in parallel, it maintains high visual quality.

2 Motivation and Objectives

1. Till date, data embedding algorithms are implemented and tested using grayscale images. But, we have implemented hamming code-based data hiding scheme for color images using basic color channel (that is R, G, and B) and their bit-plane.
2. We have introduced the concept of bit-plane. That means each R, G, and B color pixels are divided into four bit-plane starting from LSB to LSB-3.
3. Using Hamming code, Cao et al. [10] achieved data hiding scheme up to 3 bpp payload. But we have implemented our proposed algorithm for 4 bpp where the value of PSNR is always greater than 39 dB.

3 Proposed Method

I is considered as the cover image of size (M × N), and I′ is the marked image with data $D = \{d_1, \ldots, d_X\}$ are embedded, where $d_i \in \{0, 1\}$, $1 \leq i \leq X$. Here, H is a parity check matrix of the Hamming code. Let H is

$$H = \begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{vmatrix},$$

Embedding capacity is an important metric for data embedding. It is measured by how many secret bits can be embedded into a cover image. The embedding capacity is calculated as [10] $ER = L/M \times N\,bpp$, where L is the length of the secret message.

Before embedding the secret data, we take 36-bit secret key k_1 which is known to both the sender and the receiver, to encrypt the secret data bit using symmetric key encryption. We have taken each pixel block of size (3×3), and four bit-plane of each pixel is used to embed the data, which result in (3×3) $\times$ 4 = 36 bits of data (D_1) in one iteration. As an additional security measure, instead of choosing the cover image pixel block serially, we will use pseudo random number generator (PRNG) function with a secret predefined seed k_2 (which is only known to the sender and the receiver) to determine the next available block for embedding. Since this seed will be known to the sender and receiver only, the generated unique pattern of pixel block selection can be used in embedding and extraction process securely. The data embedding procedure is enlisted in Algorithm 1, and the data extraction procedure is depicted in Algorithm 2.

Algorithm 1: Data embedding process Input: Color cover image I ($M \times N$), secret data bits D, Hamming matrix H, secret key k_1, and seed value k_2

Output: A stego image I′ ($M \times N$).

Step 1: Collect random sequence of pixel blocks of size 3×3 from $I_{M \times N}$ using PRNG (k_2). Say the pixel blocks are $X_1, X_2, \ldots, X_{MN}$.
Step 2: Convert X_i into three separate RGB color blocks X_{iR}, X_{iG}, X_{iB}.
Step 3: Convert each X_i 's into binary form.
Step 4: Perform bit-plane slicing of each X_i 's up to four bit-plane starting from LSB that is $X_{iR(LSB)}$, $X_{iR(LSB-1)}$, $X_{iR(LSB-2)}$, $X_{iR(LSB-3)}$.
Step 5: Take $c = X_{iR(LSB)}$ and calculate the syndrome $S_1 = (H \times (c)^{T)T}$.
Step 6: Perform $D_1' = (D_1 \oplus k_1)$; k_1 = 36 bit length and D_1 is also same length
Step 7: Take three bits secret data $d_i = \{d_1, d_2, d_3\}$ from D_1' where $d_i \in \{0, 1\}$.
Step 8: Calculate $S_2 = (d_i \oplus S_1)$; if $S_2 = 0$, no change, otherwise flip a bit at the positional value of S_2 and generate H′.
Step 9: Compute $S_3 = (H' \oplus c)$ and store the data.
Step 10: Replace the matrix(c) with S_3 and update $X_{iR(LSB)}$.
Step 11: Repeat Step 4–10 using $X_{iR(LSB-1)}$, $X_{iR(LSB-2)}$, and $X_{iR(LSB-3)}$.
Step 12: Repeat Step 5–11 to embed secret data on X_{iG} and X_{iB} color blocks.
Step 13: Repeat Step 2–12 to embed secret data on each and every random sequence of (X_i's) of pixel blocks.
Step 14: Finally, after combining each stego block, we get stego image (I′) of size ($M \times N$).
Step 15: End.

Algorithm 2: Data extraction process Input: Stego image I′($M \times N$), Hamming matrix H, secret key k_1, and seed value k_2

Output: Original secret message D.

Step 1: Use PRNG with predetermined seed k_2 to determine the stego pixel of random sequence X'_i of size [3×3] from stego image I′.
Step 2: Separate RGB components into X'_{iR}, X'_{iG}, X'_{iB}.

Step 3: Convert into binary form of each X'_{iR}, X'_{iG}, X'_{iB}.
Step 4: Perform four bit-plane slicing of each X'_i 's starting from LSB, that is, $X'_{iR(LSB)}$, $X'_{iR(LSB-1)}$, $X'_{iR(LSB-2)}$, $X'_{iR(LSB-3)}$.
Step 5: Take $c' = X'_{iR(LSB)}$ and calculate the syndrome $S' = (H \times (c')^{T)T}$.
Step 6: Concatenate syndrome S' with data unit of D', that is, $D' = D' \| (S')$.
Step 7: Repeat Step 4–6 using X'_{iG} and X'_{iB}.
Step 8: Compute $D_i = D' \oplus k_1$.
Step 9: Repeat Step 2–8 using next random sequence of X_i block.
Step 10: Concatenate D_i 's, we get original secret message D.
Step 11: End.

The original color image is divided into R, G, and B color image as shown in Fig. 1. Then every color image is divided into (3 × 3) RGB color pixels. Then the secret bits are embedded within the image pixels. The stego (3 × 3) RGB color pixels are generated. After that, the color pixels are combined and formed the stego image of Lenna which is also (512 × 512) pixels.

3.1 Numerical Illustration

Example 3.1.1: Data embedding

1. Let I be a color image block with (3 × 3) pixel. $D = \{d_1, d_2, \ldots, d_{36}\} = \{0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0\}$. $k_1 = \{0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1,0, 1, 0, 1, 0, 1, 0\}$ and ER = 36/(3 × 3) = 4 bpp and the cover image pixels are as follows:

$$I_{3\times3} = \begin{vmatrix} 9277330 & 9276816 & 9277072 \\ 9276816 & 9343124 & 9343381 \\ 9211793 & 9409173 & 9409173 \end{vmatrix}$$

and $D' = D \oplus k_1 = \{0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0\}$

2. Divide into three RGB image pixel blocks as shown below.

$$R = \begin{vmatrix} 141 & 141 & 141 \\ 141 & 142 & 142 \\ 140 & 143 & 143 \end{vmatrix} \quad G = \begin{vmatrix} 143 & 141 & 142 \\ 141 & 144 & 145 \\ 143 & 146 & 147 \end{vmatrix} \quad B = \begin{vmatrix} 147 & 145 & 145 \\ 145 & 149 & 150 \\ 146 & 150 & 151 \end{vmatrix}$$

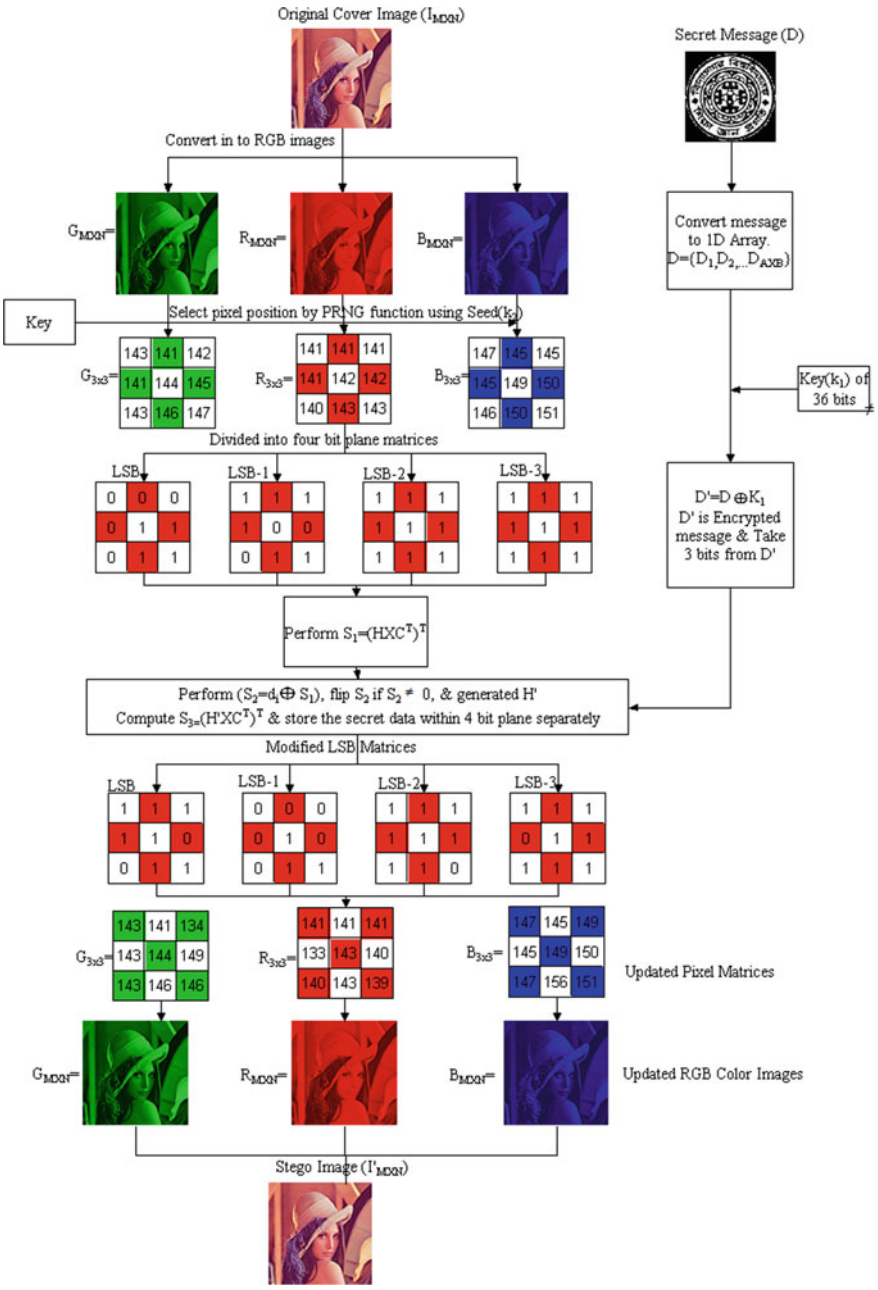


Fig. 1. Pictorial diagram of the proposed data hiding scheme

3. Take red image pixel block and transform into binary number matrix.

$$R = \begin{vmatrix} 10001101 & 10001101 & 10001101 \\ 10001101 & 10001110 & 10001110 \\ 10001100 & 10001111 & 10001111 \end{vmatrix}$$

4. Divide it into four bit-plane matrices starting from LSB.

$$R_{LSB} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 1 \end{vmatrix} \quad R_{LSB-1} = \begin{vmatrix} 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \end{vmatrix}$$

$$R_{LSB-2} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix} \quad R_{LSB-3} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

5. Read the LSB matrix and form a 1D matrix. c = [1 1 1 1 0 0 0 1 1]
6. Calculate the syndrome $S_1 = H \times (c)^T = \begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{vmatrix}$

$$\times \begin{vmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \end{vmatrix}^T = \begin{vmatrix} 1 \\ 0 \\ 1 \end{vmatrix}$$

7. Transpose the syndrome and XOR with the secret data bit, i.e., [1 0 1] $\oplus$ [0 1 1] = [1 1 0] which match with the fifth column of Hamming matrix.
8. Generate the code H′ = [0 0 0 0 1 0 0 0 0] and XOR with the original code c.

S_3 = [1 1 1 1 0 0 0 1 1] $\oplus$ [0 0 0 0 1 0 0 0 0] = [1 1 1 1 1 0 0 1 1].

9. Transform into a new LSB matrix.

$$R'_{LSB} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix}$$

10. Similarly, compute the LSB-1, LSB-2, and LSB-3 matrices as follows:

$$R'_{LSB-1} = \begin{vmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \quad R'_{LSB-2} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{vmatrix} \quad R'_{LSB-3} = \begin{vmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

11. Update all four modified binary matrices to their corresponding position in original red pixel matrix.

$$R'_{3\times3} = \begin{vmatrix} 10001101 & 10001101 & 10001101 \\ 10000101 & 10001111 & 10001100 \\ 10001100 & 10001111 & 10001011 \end{vmatrix} = \begin{vmatrix} 141 & 141 & 141 \\ 133 & 143 & 140 \\ 140 & 143 & 139 \end{vmatrix}$$

12. Similarly, get updated green and blue pixel matrices.

$$G'_{3\times3} = \begin{vmatrix} 143 & 141 & 134 \\ 143 & 144 & 149 \\ 143 & 146 & 146 \end{vmatrix} \quad B'_{3\times3} = \begin{vmatrix} 147 & 145 & 149 \\ 145 & 149 & 150 \\ 147 & 156 & 151 \end{vmatrix}$$

13. Finally, the stego image block will be $I'_{3\times3} = \begin{vmatrix} 9277330 & 9276816 & 9275028 \\ 8753040 & 9408660 & 9213333 \\ 9211794 & 9409179 & 9409179 \end{vmatrix}$

Example 3.1.2: Data extraction

1. The marked image sized I′ of size 3 × 3 is shown below

$$I' = \begin{vmatrix} 9277330 & 9276816 & 9275028 \\ 8753040 & 9408660 & 9213333 \\ 9211794 & 9409179 & 9409179 \end{vmatrix}$$

2. Divide into three RGB image pixel blocks.

$$R = \begin{vmatrix} 141 & 141 & 141 \\ 133 & 143 & 140 \\ 140 & 143 & 139 \end{vmatrix} \quad G = \begin{vmatrix} 143 & 141 & 134 \\ 143 & 144 & 149 \\ 143 & 146 & 146 \end{vmatrix} \quad \mathrm{B} = \begin{vmatrix} 147 & 145 & 149 \\ 145 & 149 & 150 \\ 147 & 156 & 151 \end{vmatrix}$$

3. Take red image pixel block and transform into binary numbers.

$$\begin{vmatrix} 10001101 & 10001101 & 10001101 \\ 10000101 & 10001111 & 10001100 \\ 10001100 & 10001111 & 10001011 \end{vmatrix}$$

4. Divide it into four bit-plane matrices starting from LSB.

$$R_{LSB} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix} \quad R_{LSB-1} = \begin{vmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix}$$

$$R_{LSB-2} = \begin{vmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{vmatrix} \quad R_{LSB-3} = \begin{vmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

5. Read LSB matrix and form a 1D matrix. c = [1 1 1 1 1 0 0 1 1]
6. Calculate the syndrome $S_1 = H \times (c)^T =$

$$\begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{vmatrix} \times \begin{vmatrix} 1 & 1 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \end{vmatrix}^T = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

7. Transpose the syndrome to get secret data bits d = [0 1 1]
8. Repeat the above steps until we do not get the secret data bits. Concatenate all the data bits to get the data, that is, D′ = {0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0}.
9. XOR the modified secret data with secret key k_1 to get the original secret data bits, that is, D = {0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0}

4 Experimental Result and Comparison

The scheme is implemented using NetBeans IDE 8.0 on standard color images to measure the performance. The standard cover images are collected from image database of SIPI [13].

The quality of the stego images is measured using mean square error (MSE) and peak signal to noise ratio (PSNR) [14, 15]. Figure 2 shows the stego image when ER = 4 bpp.

$$MSE = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} [I(i, j) - I'(i, j)]^2 \tag{1}$$

$$PSNR = 10 \log_{10} \frac{255^2}{MSE} (\text{dB}) \tag{2}$$

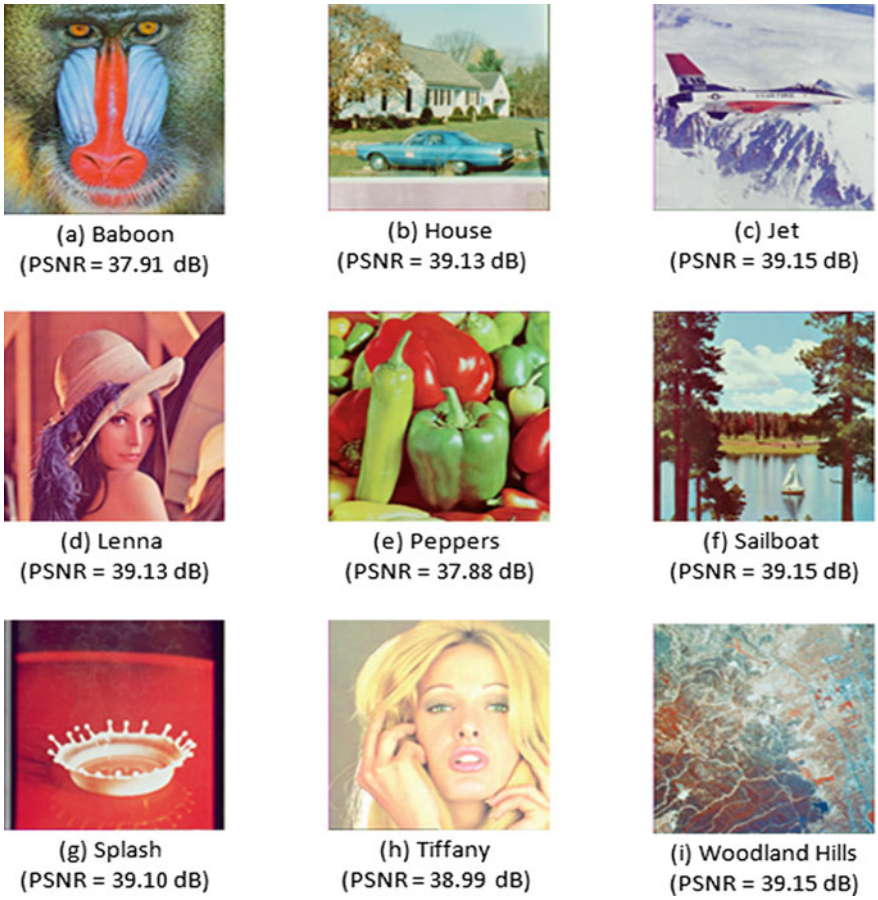


Fig. 2. Stego images of size (512 × 512)

The following tables represent the comparison of PSNR values of stego images generated by different methods with varying payload, and we achieve better PSNR value every time compared to existing methods (Tables 1, 2 and 3).

Table 1. Comparison of PSNR-ER with existing methods for ER = 1 bpp

	Lenna	Baboon	Tiffany	Peppers	Jet	Sailboat	Splash
Matrix encoding [2]	47.02	47.02	47.02	47.02	47.04	47.01	47.03
Nearest code [5]	47.02	47.02	47.03	47.02	47.04	47.01	47.03
Hamming +1 [9]	45.14	45.14	45.10	45.14	45.14	45.14	45.13
Cao et al. [10]	51.14	51.14	51.15	51.14	51.15	51.14	51.14
Proposed scheme	57.31	57.31	58.56	57.31	57.56	57.57	57.57

Table 2. Comparison of PSNR-ER with existing methods for ER = 2 bpp

	Lenna	Baboon	Tiffany	Peppers	Jet	Sailboat	Splash
Matrix encoding [2]	33.10	33.08	33.09	33.05	33.10	33.06	33.18
Nearest code [5]	33.11	33.07	33.09	33.06	33.10	33.07	33.18
Hamming +1 [9]	20.62	20.85	19.78	20.63	19.98	20.27	20.54
Cao et al. [10]	41.61	41.60	41.57	41.60	41.67	41.59	41.62
Proposed scheme	50.39	50.38	51.55	50.36	51.63	51.62	51.62

Table 3. Comparison of PSNR-ER with existing methods for ER = 3 bpp

	Lenna	Baboon	Tiffany	Peppers	Jet	Sailboat	Splash
Matrix encoding [2]	19.80	19.77	19.87	19.89	20.07	20.09	19.82
Nearest code [5]	19.80	19.77	19.87	19.89	20.08	20.09	19.81
Hamming +1 [9]	–	–	–	–	–	–	–
Cao et al. [10]	37.92	37.92	37.91	37.92	37.98	37.89	37.94
Proposed scheme	44.04	44.05	45.16	44.01	45.28	45.29	45.26

Table 4. Comparison of PSNR-ER, SSIM, and SD with existing methods for ER = 4 bpp

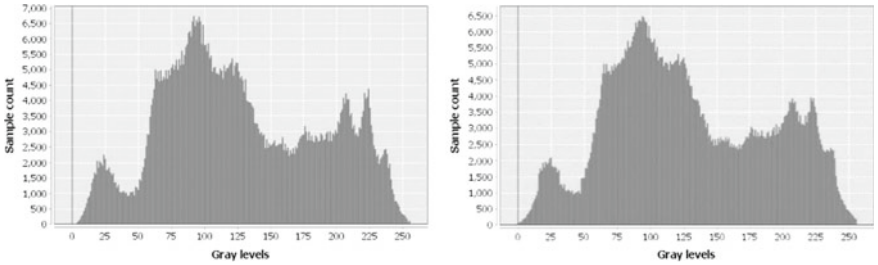
	Lenna	Baboon	Tiffany	Peppers	Jet	Sailboat	Splash
Proposed scheme	37.89	37.91	38.98	37.88	39.14	39.16	39.11
SSIM	0.93057	0.97676	0.93865	0.93022	0.9392	0.96248	0.9244
Standard deviation	0.08159	0.11102	0.175409	0.046307	0.0768	0.05825	0.02812

5 Security Analysis

Security analysis is an important factor of data hiding process. In this paper, we have used two levels of security to enhance our proposition from security perspective. First, we take a 36 bits secret key and encrypt the secret data bits using symmetric key encryption. As it is only known to the sender and receiver, the third party will not be able to decrypt it without knowing the secret key. In second level of security, we have taken a secret seed which is also known to the receiver and sender only. Using this seed, we generate a sequence of unique numbers with the help of PRNG function. We have taken the cover image pixel blocks according to the generated numbers. So

Table 5. RS analysis of stego image with ER = 4 bpp

	Lenna	Woodland Hills	House	Peppers	Jet	Sailboat
R_m	18137	17288	18109	17427	18294	17508
R_{m1}	19056	18152	19237	18151	19637	18123
S_m	14555	15772	14805	15396	14427	15509
S_{m1}	13868	15006	13983	14825	13362	15053
RS value	0.0491	0.0493	0.0592	0.0395	0.0736	0.0324



Histogram of Original Image (Lenna) Histogram of Stego Image (Lenna)

Fig. 3. Histogram of original and stego image

without knowing this seed, no one will be able to predict the number sequence.

We also verified our algorithm against some standard measurement like SSIM, standard deviation, RS analysis, histogram analysis. The structural similarity (SSIM) index is a method for measuring the similarity between two images. The SSIM index can be viewed as a quality measure of one of the images being compared, provided the other image is regarded as of perfect quality. From Table 4, it is observed that the SSIM values of all test images are nearer to 1. Standard deviation is used to measure the amount of variation between original and stego images. Here, we achieve SD neared to zero which means that the stego image and cover image are similar in nature. We also analyze our stego image through RS analysis [16]. In Table 5, it is shown that the values of R_m and R_{m1}, S_m and S_{m1} are almost equal and the ratio of R and S lies around 0.05, which is very small, so that we can conclude that our proposed scheme is secure against RS attack. Figure 3 represents the histogram of the original cover image and the stego image. It is shown that the shape of the histogram almost remains same after embedding high-capacity secret data. So we can say that our proposed method is robust against histogram attack.

6 Conclusion

In this paper, we introduced a novel secure data hiding scheme using Hamming code for RGB color image. Bit-plane slicing of the each RGB color cover image block is also introduced to increase data hiding capacity over grayscale image. So the data embedding rate is raised up to 4 bpp which is greater than other existing schemes. In our algorithm, PSNR is also high compared to existing schemes that mean we generate a better visual quality stego image. From security perspective, we introduced a shared secret key to find suitable bit pattern through XOR operation during data embedding as well as data extraction. The cover image block has been chosen in random location through PRNG of shared seed value which also enhances security in our proposed scheme. We have tested our stego image with RS analysis, histogram analysis, SSIM, SD method and observed that the proposed scheme is preferable for data embedding where visual quality and security constraint needs to be maintained for high payload. In

future, the scheme has been extended to enhance security, capacity, and quality in different domain for video-based steganography.

References

1. Wang, R. Z., Lin, C. F., & Lin, J. C. (2000), Hiding data in images by optimal moderately-significant-bit replacement. Electronics Letters, 36(25), 2069–2070.
2. Crandall R (1998), Some notes on Steganography, Posted on Steganography mailing list. http://os.inf.tudresden.de/westfield/Crandall.pdf.
3. Zhang, W., Wang, S., & Zhang, X. (2007). Improving embedding efficiency of covering codes for applications in steganography. IEEE Communications Letters, 11(8).
4. Huffman, W. C., & Pless, V. (2010). Fundamentals of error-correcting codes. Cambridge university press.
5. Chang, C. C., & Chou, Y. C. (2008, January). Using nearest covering codes to embed secret information in grayscale images. In Proceedings of the 2nd international conference on Ubiquitous information management and communication (pp. 315–320). ACM.
6. Liu, Y., Chang, C. C., & Chien, T. Y. (2017). A Revisit to LSB Substitution Based Data Hiding for Embedding More Information. In Advances in Intelligent Information Hiding and Multimedia Signal Processing (pp. 11–19). Springer International Publishing.
7. Westfeld, A. (2001, April). F5—a steganographic algorithm. In International workshop on information hiding (pp. 289–302). Springer Berlin Heidelberg.
8. Kim, C., Shin, D., & Shin, D. (2011, April). Data hiding in a halftone image using hamming code (15, 11). In Asian Conference on Intelligent Information and Database Systems (pp. 372–381). Springer Berlin Heidelberg..
9. Kim, C., & Yang, C. N. (2014). Improving data hiding capacity based on hamming code. In Frontier and Innovation in Future Computing and Communications (pp. 697–706). Springer Netherlands.
10. Cao, Z., Yin, Z., Hu, H., Gao, X., & Wang, L. (2016). High capacity data hiding scheme based on (7, 4) Hamming code. Springer Plus, 5(1), 175.
11. Jana, B., Giri, D., & Mondal, S. K. (2016). Dual image based reversible data hiding scheme using (7, 4) hamming code. Multimedia Tools and Applications, 1–23.
12. Banik, B. G., & Bandyopadhyay, S. K. (2017). Image Steganography Using BitPlane Complexity Segmentation and Hessenberg QR Method. In Proceedings of the First International Conference on Intelligent Computing and Communication (pp. 623–633). Springer Singapore..
13. University of Southern California, The USC-SIPI Image Database, 2015 http://sipi.usc.edu/database/database.php.
14. P. Gupta, et al., "A Modified PSNR Metric based on HVS for Quality Assessment of Color Images," Proc. of IEEE International Conference on Communication and Industrial Application (ICCIA-2011), Kolkatta (W.B.), India, no. 23, pp. 96–99, December, 2011.
15. P. Gupta, et al.,"A New Model for Performance Evaluation of Denoising Algorithms based on Image Quality Assessment," Proc. of (ACM ICPS) CUBE International Information Technology Conference & Exhibition, Pune, India, pp. 5–10, September, 2012.
16. Fridrich J, Goljan M, Du R (2001) Invertible authentication. In: Photonics West 2001-Electronic Imaging (pp. 197–208). International Society for Optics and Photonics.