A Novel Reversible Data Hiding Technique based on Histogram Shifting and Efficient use of Location Map

Nice Mathew
PG Scholar,
Department of CSE(PG),
Sri Ramakrishna Engineering College, Coimbatore

A. Grace Selvarani, M.E., Ph.D
M.E., Professor & HOD,
Department of CSE(PG),
Sri Ramakrishna Engineering College, Coimbatore

ABSTRACT
In this paper a reversible data hiding method which uses a novel algorithm based on histogram sifting is proposed. The distortion that occurs due to embedding can be completely removed during the data extraction. The proposed method uses double embedding to increase the reversibility. Location map information is embedded into the image itself, so that the reversibility can be easily achieved. The lower bound of PSNR for the proposed method is 51.73dB for 20000 bits. Moreover, the proposed method is quite simple and efficient. Experimental results show that the new method performs better than existing methods.

Keywords
Reversible data hiding, histogram shifting, double embedding, shifting and embedding functions, location map.

1. INTRODUCTION
Today the growth in the information technology, especially in computer networks such as Internet, Mobile communication, and Digital Multimedia applications has opened new opportunities in scientific and commercial applications. But this progress has also led to many serious problems such as hacking, duplications and malevolent usage of digital information. Steganography finds its role in attempt to address these growing concerns. Steganography means hiding a secret message (the embedded message) within a larger one (source cover) in such a way that an observer cannot detect the presence of contents of the hidden message. With the use of steganographic techniques, it is possible to hide information within digital audio, images and video files which is perceptually and statistically undetectable.

Data hiding is an art of data cover up in which the presence of embedded messages cannot be detected. Digital images are often used for carrying data in many data hiding techniques because they are often delivered using the Internet. If a digital image is served as a message carrier, the image for carrying data is called a cover image, and the image that contains data is called a stego image.There are two common methods of embedding: Spatial embedding in which messages are inserted into the LSBs of image pixels, and Transform embedding in which a message is embedded by modifying frequency coefficients of the cover image (result is called the stego-image). An information-hiding system is characterized using four different aspects: capacity, security, perceptibility and robustness.

For most image data hiding methods, the host image is permanently distorted and it cannot be restored from the marked content. But in some applications such as medical image sharing, multimedia archive management, and image trans-coding, any distortion due to data embedding is intolerable and the availability of the original image is in high demand. To this end, a solution called "reversible data hiding" (RDH) is proposed, in which the host image can be fully restored after data embedding. RDH is a hybrid method which combines various techniques to ensure the reversibility. Its feasibility is mainly due to the lossless compressibility of natural images.

Depending on different image features, data hiding can be classified into three domains namely spatial domain [1], [3], [8], compression domain [6], [11] and frequency domain [2], [5]. The main idea of image processing in spatial domain is to handle the pixels directly by varying pixel values. The most common method is the least significant bit (LSB) substitution [1], [10]. Secret messages are embedded by modifying the last few bits of a pixel value. Human's eyes are not sensitive in this kind of pixel fine tuning. Histogram modification method shifts the pixels between two points of the histogram of an image to slightly modify the pixel grayscale values for embedding data into the image. Novel techniques described in [8], [12] constructed a histogram of difference values between original pixels and predicted pixels to enhance embedding capacity and reserve the image quality. Multilevel histogram modification technique [12] is able to enlarge the payload of secret message in a more flexible and adaptive way.

Image interpolation [3], [9] is usually used to generate a high-resolution image from its low-resolution image. The interpolated pixels result to some difference values comparing with its original image and data hiding is applied by taking this advantage. There are a number of simple interpolation methods, such as nearest neighbour, bilinear, and nearest neighbour mean interpolation.

2. RELATED WORKS
2.1 Ni et al's method
In the traditional histogram modification data hiding, the amount of every pixel values of the cover image is counted and then a histogram of that is constructed. This scheme was first applied by Ni et al. in 2006 [7]. For a grayscale cover image C with size of M×N, let a grayscale value with the maximum number of pixels in C as the peak point (P) and let a grayscale value with no pixels or the minimum number of pixels in C as the zero point (Z). Assume P < Z, then histogram modification method shifts the values within the range [P +1, Z −1] of the histogram to the right-hand side by one unit thus leaving the grayscale value (P + 1) empty. A bit b which is extracted from the secret data will be embedded, and the output stego-image is S is obtained. The rule to hide the secret bit b is as shown in Eq. (1) and (2).

For $0 \le i \le M-1$, $0 \le j \le N-1$,

$$C'(i,j) = \begin{cases} C(i,j) \ if \ C(i,j) \le P \\ C(i,j) + 1 \ if \ Z > C(i,j) > P \end{cases} \quad (1)$$

$$S(i,j) = \begin{cases} C'(i.j) \ if \ C'(i,j) = P \ and \ b = 0 \\ C'(i.j) + 1 \ if \ C'(i,j) = P \ and \ b = 1 \\ C'(i,j) \ otherwise \end{cases} \quad (2)$$

2.2 Lee et al.'s Method

In [4], Lee et al.'s proposed a technique by using the histogram of difference image. This method outperforms Ni et al.'s by getting better both embedding capacity and visual quality. The spatial correlation of natural images is exploited in Lee et al.'s method and thus a more appropriate histogram is obtained. Compared with the ordinary one dimensional histogram, the difference-histogram is better for RDH since it is normal in shape and has a much higher peak point.

Take n = 2 and it can be defined

1) $S = \{(x, y) \in Z^2 : |x - y| = 1\}$ and $T = Z^2 - S$.

2) For $(x, y) \in T$, greyscale value can be changed by applying Eq.(3)

$$g(x, y) = \begin{cases} (x,y), if \ |x-y| < 1 \\ (x+1,y), if \ x-y > 1 \\ (x,y+1), if \ x-y < -1 \end{cases} \quad (3)$$

3) For $(x, y) \in S$ and $m \in \{0, 1\}$, data bits can be embedded by applying Eq.(4).

$$f_m(x, y) = \begin{cases} (x+m,y), if \ x-y = 1 \\ (x,y+m), if \ x-y = -1 \end{cases} \quad (4)$$

2.3 Honsinger's method

This method [13] is applied in the spatial domain. For authentication it uses modulo 256 addition for embedding the hash value of the original grayscale image. The embedding formula is , Iw= (I+W) mod 256 in which I denotes the original image, Iw represents the marked image, and W=W(H(I),K) give the watermark, where H(I) denotes the hash function operated on the original image, and I and K are the secret keys for security. Because of the usage of modulo 256 addition, the overflow/underflow problem is prevented and the reversibility is achieved. Some annoying salt-and-pepper noise is generated owing to grayscale value flipping over between 0 and 255 in either direction during the modulo 256 addition.

3. PROPOSED SYSTEM

In the proposed system, initially divide the host image into non overlapping blocks of n pixels. Next step is to generate an n-dimensional histogram by counting the frequency of the pixel-value-array sized n of each divided block. Finally, the data embedding is achieved by modifying the resulting n-dimensional histogram. Here the pixel-value array is an element of Z^n. Now divide Z^n into two disjointed sets, one set is used to carry hidden data by expansion embedding, and the next set is simply shifted to create vacant spaces to ensure the reversibility.

The idea for histogram shifting based approach can be described as follows:

Let S and T be the two partitions of Z^n such that $S \cup T = Z^n$ and $S \cap T = \phi$.

Define two injective functions $g : T \rightarrow Z^n$, $f : S \rightarrow Z^n$ such that the sets g(T) and f(S) are disjointed with each other. Here "shifting function", g is used to shift pixel values whereas the "embedding function", f is used to embed data.

Then, each block with value $\mathbf{x} \in T$ will be shifted to $g(\mathbf{x})$, and the block with value $\mathbf{x} \in S$ will be expanded to $f(\mathbf{x})$ to carry one data bit. After data embedding also, the sets $g(T_s)$ and $f(S_e)$ are disjointed with each other, but the two sets $TS_u = T_{u,o} \cup S_{u,o}$ and $g(T_s) \cup f(S_e)$ may be overlapped.

The underflow/overflow problem is inevitable in RDH. To deal with this, the above defined sets T and S can be further processed. Let

$A_n = \{\mathbf{x} = (x_1, \ldots, x_n) \in Z^n : 0 \le x_i \le 255\}$
be the set of all pixel-value-arrays of length n of gray-scale image.

By definitions, each block with value $\mathbf{x} \in T_s$ will be shifted and each block with value $\mathbf{x} \in S_e$ will be expanded to carry one data bit. The block with value $\mathbf{x} \in T_{u,o} \cup S_{u,o}$ will remain unchanged, because it cannot be shifted or expanded due to underflow/overflow. The use of shifting and embedding functions to embed data can be described briefly as below.

Consider a pixel block of length n whose value is $\mathbf{x} \in A_n$.

1) If $\mathbf{x} \in T_s$, its value is simply shifted to $g(\mathbf{x}) \in A_n$ as the block does not carry any hidden data.

2) If $\mathbf{x} \in S_e$, the new pixel value is $f(\mathbf{x}) \in A_n$. In this situation, one data bit is embedded into each block. As the sets $g(T_s)$ and $f(S_e)$ are disjointed with each other, decoder can distinguish the blocks used for shifting from those for embedding data.

3) If $\mathbf{x} \in T_{u,o}$, $g(\mathbf{x})$ is not an element of A_n and to prevent underflow/overflow, do nothing with $\mathbf{x}$.

4) If $\mathbf{x} \in S_{u,o}$, it is clear that $f(\mathbf{x})$ is not an element of A_n and the block will remains unchanged. The set $T_{u,o} \cup S_{u,o}$ may overlap with $g(T_s) \cup f(S_e)$. So a location map is used to record the locations of blocks whose values belong to $T_{u,o} \cup S_{u,o}$.

The reverse of the embedding steps is done during extraction. Inverse shifting function (g^{-1}) and embedding function (f^{-1}) can be used in the extraction procedure.
To embed the data into the image shifting function (g) and embedding function (f) is used. Along with the data, auxiliary information required for the extraction of the data is also added into the image. The decoder can read the auxiliary information from the marked image and it can be used to extract hidden information from the image.

3.1 Data embedding

Data embedding is the process of hiding some useful information into a cover media. The embedding procedure contains several steps. First, after dividing the host image into non-overlapping blocks, the blocks are divided into three parts to get I_1, I_2 and I_3. Then, by using shifting and embedding functions defined, embed the hidden data into I_1 and I_3. After that, by using LSB replacement, embed the location map which records the underflow/overflow locations into I_1. Before replacing LSBs, the original LSBs of I_1 should be recorded into a LSB sequence. Finally, embed the LSB sequence into I_2 by using shifting and embedding functions.

Here, the partition of three parts is used to solve the underflow/ overflow problem by embedding the location map into the host image. The part I_1 is double embedded the hidden data is embedded first (using shifting and embedding functions) and then the location map (using LSB replacement).

The detailed data embedding procedure is described as below.

Step 1: Divide the host image into k non-overlapping blocks $\{X_1,\ldots, X_k\}$ so that each Xi contains n pixels. Assume the value of Xi is $\mathbf{x}_i \in A_n$.

Step 2: Define the binary sequence location map LM of length k: LM(i) = 0 if $\mathbf{x}_i \in T_s \cup S_e$, and LM(i) = 1 if $\mathbf{x}_i \in T S_u$. Denote k' = ⌈log2(k + 1)⌉, where ⌈*⌉ is the ceiling function. Take l = #{i : LM(i) = 1} which is the amount of underflow/overflow blocks. Now define a binary sequence LM_c of length l_c = (l + 1)k' to record all underflow/overflow locations.

1) $(LM_c(1),\ldots, LM_c(k'))$ is the binary representation of l underflow/overflow blocks.

2) $(LM_c(jk'+1),\ldots, LM_c(jk+k'))$ is the binary representation of i, where i is the j -th index such that LM(i) = 1 for each $j \in \{1,\ldots, l\}$.

Step 3: Divide the k image blocks into three parts to get I_1, I_2 and I_3.
1) $I_1 = \{X_1,\ldots, X_{k1}\}$ where $k_1 = \lceil lc/n \rceil$.

2) $I_2 = \{X_{k1+1},\ldots, X_{k1+k2}\}$ such that it has exactly l_c embeddable blocks. Here, a block is called an embeddable block if its value belongs to S_e.

3) $I_3 = \{X_{k1+k2+1},\ldots, X_k\}$ is the set of remaining blocks.

Step 4: Embed the hidden data bits into I_1 and I_3, i.e., for any i $\in \{1,\ldots, k1, k1 + k2 + 1,\ldots, k\}$.

1) If $\mathbf{x}_i \in T_s$, replace the value of X_i by $g(\mathbf{x}_i)$.

2) If $\mathbf{x}_i \in S_e$, replace the value of X_i by $f(\mathbf{x}_i)$ to embed data.

3) If $\mathbf{x}_i \in T S_u$, do nothing with X_i as they are underflow/overflow blocks.

Step 5: Record LSBs of the first l_c pixels of I_1 to get S_{LSB} which is a binary sequence. Then replace these LSBs by the sequence LM_c defined earlier.

Step 6: Embed the sequence S_{LSB} into I_2 using shifting and embedding functions in the same way as Step 4. As the length of S_{LSB} is l_c, S_{LSB} can be embedded exactly into the embeddable blocks of I_2. Finally, the marked image is obtained.

The important part of this procedure is Step 4 and the partition in Step 3 is to deal with the underflow/overflow problem. There is another commonly used way to encode the location map: in Step 2, one can get LM_c by losslessly compressing LM. However, in HS-based RDH algorithms, there are usually only a few blocks which may cause underflow/overflow condition. Then prefer to record those problematic locations rather than compressing the location map as the latter solution is time-consuming.

3.2 Data extraction

Data extraction is the process of separating the hided message from the marked image. The data extraction procedure also contains several steps.

First, divide the marked image blocks into three parts to get I_1, I_2 and I_3. Then, location map information can be obtained by reading LSBs of I_1. Next, according to the location map and by using shifting and embedding functions, determine the LSB sequence (defined in Step 5 of data embedding) by extracting data from I_2, and then replace the LSBs of I_1 by the extracted LSB sequence. Finally, extract the embedded data bits from I_1 and I_3. Notice that, using shifting and embedding functions, the image restoration can be realized simultaneously with the data extraction procedure. The detailed data extraction procedure can be described as below.

Step 1: Divide the marked image into k non-overlapping blocks $\{Y_1,\ldots, Y_k\}$. Assume the value of Y_i is $\mathbf{y}_i \in A_n$.

Step 2: Determine the amount of problematic locations, l, by reading LSBs of the first k = $\lceil \log_2(k+1) \rceil$ pixels. Then read LSBs of the first l_c = (l + 1)k' pixels to get the sequence LM_c. After that the location map LM is obtained. At last, with $k_1 = \lceil l_c/ n \rceil$, LM, and by identifying embeddable blocks, same partitions I_1, I_2 and I_3 is obtained.

Step 3: Extract data bits from I_2 and recover original pixel values of I_2, i.e., for any $i \in \{k_1 + 1,\ldots, k_1 + k_2\}$.
1) If LM(i) = 0 and $\mathbf{y}_i \in g(T_s)$, the original pixel value is $g^{-1}(\mathbf{y}_i)$ and contains no embedded bits.

2) If LM(i) = 0 and $\mathbf{y}_i \in f(S_e)$ holds, the original pixel value is $f^{-1}(\mathbf{y}_i)$ and extract the embedded data bit.

3) If LM(i) = 1, the original pixel value is $\mathbf{y}_i$ itself and there is no data is embedded.

The sequence S_{LSB} defined in Step 5 of data embedding process is extracted in this step.

Step 4: Replace LSBs of the first l_c pixels of I_1 with S_{LSB}.

Step 5: Extract the embedded hidden data and recover original pixel values in I_1 and I_3 in the same way as Step 3. The hidden data is extracted and the original image is restored in this way.

The embedding and extraction process should follow the steps described above. Different algorithms can be developed by changing the shifting and embedding functions.

3.3 A novel Shifting and Embedding function

A number of different shifting and embedding functions can be defined to improve the performance of data hiding. A novel method for improving the performance can be described as below. For a 3 × 3 block $\mathbf{x} = (x_1,\ldots, x_9)$, take the following linear predictor value with non uniform weight to predict x_5.

$$x_5' = 1/16(x_1 + x_3 + x_7 + x_9) + 3/16(x_2 + x_4 + x_6 + x_8)$$

The prediction-error is defined as $e_5 = x_5 - x_5'$. Utilizing smooth pixels for reversible data embedding whereas ignoring the noisy ones will significantly reduce the embedding distortion in data hiding. Then take the following function

$$C(\mathbf{x}) = max\{x_1, \ldots, x_4, x_6, \ldots, x_9\} - min\{x_1, \ldots, x_4, x_6, \ldots, x_9\} \quad (5)$$

to measure the local complexity of pixel x_5. Now use an integer-valued parameter s to select smooth pixels.

For a selected integer t > 0, take $t_l = \lceil t/2 \rceil$ and $t_r = \lfloor t/2 \rfloor$
Tthen define shifting and embedding functions as given in Eq. (6) and (7).

1) $S = \{\mathbf{x} \in Z^9 : -t_l \leq e_5 < t_r, C(\mathbf{x}) < s\}$ and $T = Z^9 - S$.

2) For $\mathbf{x} \in T$, define

$g(\mathbf{x}) =$

$$\begin{cases} (x1,..,x4,x5+tr,x6,..,x9) if\ e5 \geq tr\ and\ C(x) < s \\ (\boldsymbol{x1},..x4,x5-tl,x6,..x9)\ if\ e5 \leq tl\ and\ C(x) < s \\ x,\ if\ C(x) \geq s \end{cases} \quad (6)$$

3) For $\mathbf{x} \in S$ and $m \in \{0, 1\}$, define

$$f(\mathbf{x}) = (x_1, \ldots, x_4, x_5 + \lceil e_5 \rceil + m, x_6, \ldots, x_9). \quad (7)$$

To minimize the embedding distortion, the parameter s is first fixed as its maximum value 256 and the parameter t is selected as the smallest possible integer such that it is capable to embed the required payload.

4. EXPERIMENTAL RESULTS

Peak signal-to-noise ratio, often abbreviated PSNR, is an expression for the ratio between original image and noise. It is most commonly used to measure the quality of reconstruction. PSNR measurement is most easily defined via the mean squared error (MSE) value. The mean squared error (MSE) for practical purposes allows comparing the "true" pixel values of our original image to our degraded marked image. The MSE represents the average of the squares of the "errors" between original image and noisy image. The error is the amount by which the values of the original image differ from the noisy image. Given a noise-free m×n image I and its noisy approximation K, MSE is defined as in Eq. (8) [14].

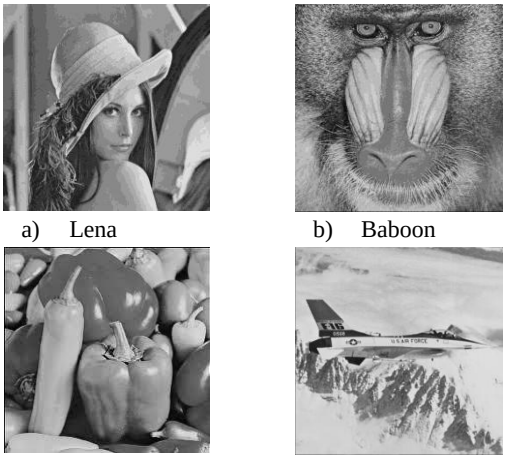
$$MSE=\frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}[I(i,j)-K(i,j)]^2 \quad (8)$$

The PSNR can be defined as given in Eq. (9):

$$PSNR=20\log_{10}\left(\frac{MAXf}{\sqrt{MSE}}\right) \quad (9)$$

where MAX_f is the maximum possible pixel value of the image. When the pixels are represented using 8 bits per pixel, this is 255. In the absence of noise, the two images I and K are identical, and thus the MSE becomes zero. In this case the PSNR is undefined.

The test images selected for evaluation are shown in Fig:1



a) Lena b) Baboon

c) Peppers d) Airplane

Fig 1: Test images

The algorithm is evaluated by comparing it with some already existing works. Four test images are taken for comparing the algorithms. The PSNR value obtained for different algorithms for each test image can be shown in tables 1 and 2.

Table 1. Comparison of PSNR(in dB) for the proposed method with some existing methods for an EC of 10000 bits.

Test image (512x512)	Ni et al's method	Lee et al's method	Hong	Proposed system
Lena	46.53	58.20	58.75	59.49
Baboon	45.32	54.03	53.05	53.96
Air plane	49.68	61.26	62.45	64.04
Peppers	47.81	56.12	56.19	57.73

Table 2. Comparison of PSNR(in dB) for the proposed method with some existing methods for an EC of 20000 bits.

Test image (512x512)	Ni et al's method	Lee et al's method	Hong	Proposed system
Lena	43.21	55.82	55.92	55.29
Baboon	41.19	53.40	50.39	51.73
Air plane	46.48	58.97	59.72	61.83
Peppers	45.92	53.65	53.79	56.07

To increase the embedding capacity further, other data hiding techniques can be also used in conjunction with histogram shifting method. Other common techniques are image interpolation, difference expansion, integer transform etc.

5. CONCLUSION

A novel data hiding technique based on histogram shifting is proposed in the paper. The procedure for embedding and extraction proposed here is having greater embedding capacity compared to other state-of-the-art works. Also the location map information stored along with the image can be

used to reduce the underflow/overflow problem. Future developments are expected by introducing new shifting and embedding algorithms. This method can be also used in conjunction with other techniques. Combining different techniques certainly increases the embedding capacity. Also algorithms are expected to introduce which can embed data in colour images.

6. REFERENCES

[1] Chan C. K., Cheng L. M. 2004 Hiding data in images by simple LSB substitution. Pattern Recognit 37(3):469–474

[2] Chan Y. K., Chen W. T., Yu S. S., Ho Y. A., Tsai C. S., Chu Y. P. 2009 A HDWT-based reversible data hiding method. J Syst Softw 82(3):411–421

[3] Jan S. R., Hsu S. J., Chiu C. F., Chang S. L. 2011 An improved data hiding method using image interpolation. In: 2011 seventh international conference on intelligent information hiding and multimedia signal processing, pp 185–188

[4] Lee S. K., Suh Y. H., and Ho Y. S. 2006 Reversible image authentication based on watermarking. Proc. IEEE Int. Conf. Multimedia Expo: 1321–1324.

[5] Langelaar G. C., Lagendijk R. L. 2001 Optimal differential energy watermarking of DCT encoded images and video. IEEE Trans Image Process 10(1):148–158

[6] .Lu Z. M., Pan J. S., Sun S. H. 2000 VQ-based digital image watermarking method. Electron Lett 36(14):1201–1202

[7] Ni Z., Shi Y. Q., Ansari N., Su W. 2006 Reversible data hiding. IEEE Trans Circuits Syst Video Technol 16(3):354–362

[8] Pawar P. H., Jondhale K. C. 2012 Histogram-based reversible data hiding using block division. In: 2012 IEEE international conference on advanced communication control and computing technologies (ICACCCT), pp 295–299

[9] Yalman Y., Akar F., Erturk I. 2010 An image interpolation based reversible data hiding method using R-weighted coding. In: 2010 13th IEEE international conference on computational science and engineering, pp 346–350

[10] Yang C. H. 2008 Inverted pattern approach to improve image quality of information hiding by LSB substitution. Pattern Recognit 41(8):2674–2683

[11] Yang C. H., Wang W. J., Huang C. T., Wang S. J. 2011 Reversible steganography based on side match and hit pattern for VQ-compressed images. Inf Sci 181(11):2218–2230

[12] Zhao Z., Luo H., Lu Z. M., Pan J. S. 2011 Reversible data hiding based on multilevel histogram modification and sequential recovery. AEÜ, Int J Electron Commun 65(10):814–826.

[13] Honsinger C. W., Jones P., Rabbani M., and Stoffel J. C., 2001 Lossless Recovery of an Original Image Containing Embedded Data, U.S. Patent 6 278 791 B1

[14] Salomon, David 2007. Data Compression: The Complete Reference (4 ed.). Springer. p. 281. ISBN 978-1846286025.