Lossless Data Hiding Based on Histogram Modification

Rajkumar Ramaswamy[1] and Vasuki Arumugam[2]
[1]Department of Electronics and Communication Engineering, RVS College of Engineering and Technology, India
[2]Department of Electronics and Communication Engineering, Kumaraguru College of Technology, India

Abstract: *Lossless data hiding is the technique of embedding data in an image and retrieval of the data with lossless reconstruction of original image. In this paper, we present a novel lossless data hiding scheme based on histogram modification. This technique is based on differences of adjacent pixels for embedding data and has more hiding capacity compared to existing methods. The number of message bits that can be embedded into an image equals the number of pixels associated with the peak point. Here, a histogram shifting technique is applied in order to prevent overflow and underflow problems of the pixels. For color images the hiding capacity is three times larger than the grayscale images. The performance of the algorithm has been evaluated for eight grayscale images and eight color images with hiding capacity (bits) and peak signal to noise ratio (dB) of the reconstruction as the parameters.*

Keywords: *Data hiding, histogram modification, histogram shifting, lossless reconstruction.*

Received May 6, 2010; accepted August 10, 2010

1. Introduction

Data hiding [8] is a scheme in digital media to embed any secondary data into original information. Data hiding has found a variety of applications such as access control, annotation and authentication, etc., [6, 9]. Data hiding is also found to be useful in sending secondary information in multimedia communication for achieving additional functionalities. A fundamental problem of all data hiding techniques is the embedding capacity, i.e., the number of bits that can be embedded into the original signal. Data hiding algorithms can be classified into one of the following two categories: irreversible data hiding and reversible data hiding. In irreversible algorithms the host signal cannot be completely recovered; these algorithms are not suitable for medical and military applications. In the reversible data hiding algorithms, original information can be completely recovered.

Awrangjeb [1] devised an invertible watermarking method by applying a lossless compression algorithm and encryption of bit planes. The space to hide information is found by compressing the bit plane that offers minimum redundancy. The main disadvantage of this algorithm is low hiding capacity and in noisy images the data becomes visible. DeVleeshouwer *et al*. [4] proposed a semi-fragile technique based on patchwork algorithm and modulo operation. In this method each message bit is associated with a group of pixels. Each group is equally divided into two pseudo random sets of pixels. These sets are represented as two zones, 'a' and 'b'. These zones have a close average value before embedding. Depending on the message bit to be embedded, their luminance values are incremented or decremented. Here, the histogram of the grayscale values of each zone is mapped to a circle. The weight is proportional to the occurrence of each luminance value. At the pixel level, rotation of vectors corresponds to the luminance shifts. The magnitude of the shift directly depends on the rotation angle. The main drawback of this method is that some groups of pixels do not allow data embedding and the algorithm is complex. Celik *et al*. [2] presented a lossless Generalized Least Significant Bit (GLSB) embedding scheme. This method uses generalization of LSB modification technique by compressing the quantization residues. In the embedding phase, host signal H is quantized and the residue r is calculated as:

$$r = H - Q_L(H) \tag{1}$$

The residue is compressed in order to create capacity for the payload w. The compressed residue and the payload are concatenated and embedded into the host signal by using the GLSB modification technique. The GLSB embedding process is given by:

$$H_w = H \mid (Q_L(H) + w) \tag{2}$$

Where H_w is the watermarked value of the signal and, L is the level of scalar quantization function of the value x.

$$Q_L(x) = L\left\lfloor \frac{x}{L} \right\rfloor \tag{3}$$

In the extraction phase, the watermarked signal is

quantized, the payload and the compressed residue are extracted. The extraction is the reverse of the embedding process given by:

$$w = H - Q_L(H) \tag{4}$$

Tian [7] devised a high capacity reversible data hiding technique called difference expansion. In the embedding process, in the first step, the original image is grouped into pairs of pixels. A pair consists of two neighboring pixel values. Here, integer transform is applied to each pair. If the grayscale pixel pair is *(x, y)*, then the integer average *l* and the difference *h* is calculated as given in equation 6:

$$l = \left\lfloor \frac{x + y}{2} \right\rfloor \tag{5}$$

$$h = x - y \tag{6}$$

The difference value *h* is calculated and ordered as a vector. In the next step, four disjoint sets are created [EZ, EN, CN, and NC] from *h* values.

- EZ: Contains all expandable values of h=0 and expandable values of h= -1.
- EN: Contains all expandable $h \neq$EZ.
- CN: Contains all changeable $h \neq$ (EZ U EN).
- NC: Contains non-changeable *h*.

The difference value *H* is expandable under the integer average value *l* if:

$$(2 \times h + b) \leq \min(2(255 - l), 2l + 1) \tag{7}$$

Where *b* is the message bit to be embedded. The integer average value is *l* if:

$$2 \times \left\lfloor \frac{h}{2} \right\rfloor + b \leq \min(2(255 - l), 2l + 1) \tag{8}$$

Third, a location map is created by using the selected expandable difference values. The location map is losslessly compressed by run length encoding and the compressed bit stream is denoted as £. In the next step, LSB (*h*) will be collected into a bit stream *C.* In the fifth step, the location map £, the original LSB's *C* and the payload *P* are embedded. The £, *C*, and *P* are combined together into a binary bit stream *B.* After embedding all the bits we apply inverse integer transform to obtain the original image, which is given by:

$$x = l + \left\lfloor \frac{h + 1}{2} \right\rfloor \tag{9}$$

$$y = l - \left\lfloor \frac{h}{2} \right\rfloor \tag{10}$$

The embedded bit stream *B* can be retrieved by collecting the LSBs of all changeable difference values. From *B*, the location map £ and the original LSB's *C* is decoded. The decoding process contains five steps. First, the difference values are calculated. Image pairing is done in the embedded image by using the same method as in embedding and integer transform is applied to each pair. Next, two disjoint sets of the difference values, CH and NC are created.

- CH: Contains all changeable *h.*
- NC: Contains all non-changeable *h.*

Then LSBs of all difference values are collected in CH and a binary bit stream *B* is formed. Next, the location map is decoded from *B* by using run length decoder. The decoder knows exactly the location in *B* where it is the last bit from the location map bit stream £. Ni *et al*. [5] introduced a reversible data hiding scheme based on histogram modification using pairs of peak and zero points. Let us consider a grayscale image of size M×N. The steps for embedding process will be as follows:

1. The histogram of the grayscale image is generated. In the histogram, a zero point *Z* and then a peak point *P* is calculated. A zero point corresponds to the grayscale value in which there is no pixel in the given image. A peak point corresponds to the grayscale value in which there are maximum numbers of pixels in the given image. Let us consider only one zero point and one peak point in the histogram. The objective of finding the peak point is to calculate the embedding capacity of the image, since the number of bits that can be embedded into an image is equal to the number of pixels which are associated with the peak point.
2. The whole image is scanned in any sequential order. For example, let the peak point *P* in the image be '*a*' and the zero point be '*b*'. The grayscale value of pixels between *(a+1)* and '*b*' is incremented by '1'. This step is equivalent to shifting the range of the histogram, to the right-hand side by one unit, leaving the grayscale value *(a+1)* empty.
3. The whole image is scanned once again in the same sequential order. Once a pixel with grayscale value of '*a*' is encountered, if the corresponding embedding bit in the message sequence is '1',then the pixel value is incremented by '1'. Otherwise, the pixel value remains unchanged. The watermarked image is of size M×N, each pixel grayscale value is in range of [0, 255]. The extraction process in the algorithm will be as follows.
4. Scan the marked image in the same sequential order as used in the embedding process.
5. If a pixel with its grayscale value *(P+1)* is encountered, a bit '1' is extracted. If a pixel with its grayscale value *P* is encountered, a bit '0' is extracted.
6. After the extraction of the message sequence, the original pixel values are restored and the original pixel values are restored and the original image is reconstructed.

In this paper, the input image is scanned in inverse s-order. Then we find pixel differences of the adjacent pixels and find the peak point in the difference values. The number of pixels at the peak point determines the embedding capacity of the image. Here we use histogram shifting technique in order to avoid the overflow and underflow situations. Then the message bits are embedded at the peak pixel values. At the receiver side the message bits are extracted and the original image is restored.

The remaining part of the paper is organized as follows. In section 2, a detailed explanation of the proposed algorithm for grayscale and color images with an example is given. In section 3, the experimental results of the proposed algorithm is tabulated and compared with the previous methods. Finally, section 4 concludes the paper.

2. Proposed Scheme

An efficient extension of the histogram modification technique is proposed in this paper. Here, the pixel differences between the adjacent pixels are calculated. In all images the neighboring pixels are highly correlated. The pixel difference values taken between adjacent pixels have a distribution that is packed near zero. The pixel difference of the adjacent pixels is expected to be very close to zero. Here, the histogram of the pixel difference has a Laplacian distribution as shown in Figure 1 and the maximum value is very close to zero. The count of the pixel value which is maximum in the pixel differences determines the hiding capacity of the image. The algorithm for the histogram modification based on pixel differences is given below.

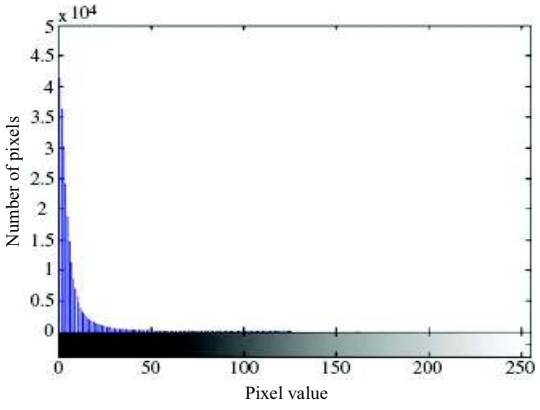


Figure 1. Distribution of pixel differences.

2.1. Algorithm for Embedding Message Bits in Grayscale Images

Let us consider an 8bit gray scale image H with N number of pixels and let X_i denote the gray scale value of the i^{th} pixel. The pixel values in the gray scale image are in the range of [0, 255]. The steps for histogram modification based on pixel differences are as follows:

1. Scan the image H in inverse s-order [3] as shown in Figure 2.

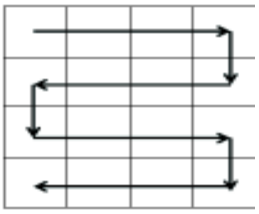


Figure 2. Inverse s-order scanning.

2. Calculate the pixel difference d_i between the pixels x_i and x_{i-1} by using:

$$d_i = \begin{cases} x_i, & if\ i = 0 \\ |x_{i-1} - x_i|, & otherwise \end{cases} \tag{11}$$

3. Determine the peak point P from the pixel differences.
4. If $d_i > P$, then shift each pixel by 1 unit using the following equation:

$$y_i = \begin{cases} x_i, if\ \ i = 0\ or\ d_i < P \\ x_i + 1, if\ \ d_i > P\ and\ x_i \geq x_{i-1} \\ x_i - 1, if\ \ d_i > P\ and\ x_i < x_{i-1} \end{cases} \tag{12}$$

Where y_i is the watermarked value of pixel i.

5. If $d_i = P$, modify the pixel x_i according to the message bit b [0,1]:

$$y_i = \begin{cases} x_i + b,\ if\ d_i = P\ and\ x_i \geq x_{i-1} \\ x_i - b,\ if\ d_i = P\ and\ x_i < x_{i-1} \end{cases} \tag{13}$$

Where b is a message bit to be embedded. After embedding the message bits, the embed ded image is scanned in the same inverse s-order at the receiving end to extract the embedded data.

6. The message bit b can extracted by:

$$b = \begin{cases} 0,\ if\ |y_i - x_{i-1}| = P \\ 1,\ if\ |y_i - x_{i-1}| = P + 1 \end{cases} \tag{14}$$

Where y_i and x_i are the i^{th} pixel values of the watermarked image and original image respectively.

7. The pixel values x_i of original image can be reconstructed by:

$$x_i = \begin{cases} y_i + 1,\ if\ |y_i - x_{i-1}| > P\ and\ y_i < x_{i-1} \\ y_i - 1,\ if\ |y_i - x_{i-1}| > P\ and\ y_i > x_{i-1} \\ y_i, \quad Otherwise \end{cases} \tag{15}$$

Thus by using these x_i values we can reconstruct the original image. The above steps use only one peak point for embedding the data. In order to increase the embedding capacity multiple peak points can be used.

2.2. Algorithm for Embedding and Extraction of Message Bits in Color Images

Consider, a color image H with N pixels. Each pixel in the color image will have three individual color components of red (R), green (G) and blue (B). The pixel values of all these color components will be in the range of [0, 255]. The message bits can be embedded in all the three planes and these planes can be recombined to form the original color image. So, the embedding capacity of the color image is three times greater when compared to the grayscale images. The pseudo algorithm for histogram modification based on pixel differences for color images are as follows:

1. Read the input color image.
2. Separate the whole image into three planes of individual components R, G and B.

Consider that we are embedding the message bits in R plane. The remaining steps of the algorithm are similar to the steps in section 2.1. Finally after the extraction of the message bits from the R, G, and B planes, these planes are combined to form an RGB image.

2.3. An Example for the Embedding and the Extraction Process

Let us consider a host image *H* as shown below.

230	229	230	231
228	229	232	230
229	232	231	230
233	235	234	231

The inverse s-order scanned vector is:

1	2	3	4	5	6	7	8
230	229	230	231	230	232	229	228

9	10	11	12	13	14	15	16
229	232	231	230	231	234	235	233

The pixel differences of the adjacent pixels are:

1	2	3	4	5	6	7	8
230	1	1	1	1	2	3	1

9	10	11	12	13	14	15	16
1	3	1	1	1	3	1	2

Here the peak point *P* is *1*, number of *1*'s in the image is 10, so the embedding capacity of the image is 10. Let the embedding message bits be 11001111.
The watermarked pixel values of the image is:

1	2	3	4	5	6	7	8
230	228	231	231	230	233	228	227

9	10	11	12	13	14	15	16
230	233	231	230	231	235	236	232

The watermarked image is:

230	228	231	231
227	228	233	230
230	233	231	230
231	235	236	232

The inverse s-order scanning of watermarked image is:

1	2	3	4	5	6	7	8
230	228	231	231	230	233	228	227

9	10	11	12	13	14	15	16
230	233	231	230	231	235	236	232

The extracted message bits are 1 1 0 0 1 1 1 1.
Reconstructed pixel values are:

1	2	3	4	5	6	7	8
230	229	230	231	230	232	229	228

9	10	11	12	13	14	15	16
229	232	231	230	231	234	235	233

Reconstructed image is:

230	229	230	231
228	229	232	230
229	232	231	230
233	235	234	231

The detailed explanation for the above example is given below. For example, consider a sample grayscale image with 4×4 pixels. First, scan the whole image in inverse s-order and calculate the pixel difference d_i between the pixels x_i and x_{i-1}. Now the peak point is 1. The number of 1s in the image is 10. So the embedding capacity of this image is 10. Assume that the message bit to be embedded in the image is 11001111. Always the first bit in the image is kept unaltered, so $y_1=x_1$ =230. Now, $|x_2-x_1|=|229-230|=1=P$, the first message symbol 1 is embedded in x_1 by altering x_1, since $x_1>x_2$, $y_2=x_2$ -1=228. The difference between x_2 and x_3 is |229-230|=1= *P*, then the second message symbol 1 is embedded in x_3 by setting $y_3=x_3+1$. Since $x_2<x_3$, y_3=231. Then $|x_4-x_3|=|231-230|=1=P$. Now the third message symbol to be added is 0. Here $y_4=x_4+0$=231 and so on. The embedding process continues until all of message symbols are embedded, and then the resulting watermarked image is obtained.

If the peak point *P* is known, then the image can be completely restored to its original state before the embedding occurred. Since $|y_2-x_2|=|228-230|=2=P+1$, the message symbol 1 is extracted. The difference between y_3 and x_2 is |229-231|=2=*P*+1, a message symbol 1 is extracted. The next message is extracted if $|y_4-x_3|=|231-230|=1=P$, the message bit extracted is '0' and the process continues until all the message bits are extracted. For the reconstruction of the original image $y_1=x_1$. x_2 value is restored by setting $x_2= y_2$ =1= 228+1=229, since $|y_2-x_1|=|228-230|=2>P$ and 228<230. For x_3, $|y_3-x_2|=|231-229|=2>P$ and 231>229, so $x_3=y_3$-1=231-1=230. Thus, the watermarked image is restored as an exact copy of the original host image.

3. Experimental Results

The performance of the proposed data hiding scheme on different images has been evaluated.

$$Embedding\ capacity = Total\ number\ of\ pixels\ at\ the\ peak\ point\ P\ (\text{bits}) \quad (16)$$

$$Bitrate = \frac{Embedding\ capacity}{Total\ number\ of\ pixels}(bits\ per\ pixel) \quad (17)$$

$$PSNR = 10\log_{10}\left(\frac{255^2}{MSE}\right)(dB) \quad (18)$$

Where, Mean Square Error (MSE) is:

$$\text{MSE}=\frac{1}{M\times N}\sum_{i=1}^{M}\sum_{j=1}^{N}(x(i,j)-w(i,j))^2 \quad (19)$$

Where, $x(i, j)$ Is the original image.
$w(i, j)$ Is the watermarked image.

3.1. For Grayscale Images

According to the proposed method, the images having highest correlation between the adjacent pixels will have the highest embedding capacity and the images with lower correlation between pixels have lower embedding capacity. Among all of the grayscale images shown in Figure 3, the "Barbara" image is having the highest embedding capacity and the "Baboon" image is found to have the lowest embedding capacity. Also, the bitrate is higher for the images having high embedding capacity. The results are tabulated in Table 1. The embedding capacity, bitrate and PSNR of the image embedded with message bits are calculated as in equations 16-19. All the experiments were done on the eight grayscale images - "Lena", "Baboon", "Barbara", "Goldhill", "Flower", "Tire", "Rice", and "Cameraman" of size 512×512. The host and the watermarked images are shown in Figures 3 and 4.

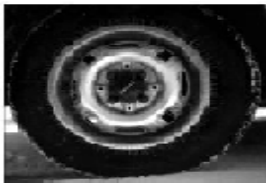
a) Lena.

b) Goldhill.

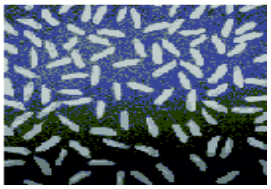
c) Baboon.

d) Barbara.

e) Flower.



f) Tire.



g) Rice.

h) Cameraman.

Figure 3. Host images of size 512×512.

Table 1. Hiding capacity and distortion for grayscale images of size 512×512.

Host Image	Embedding Capacity in Bits		PSNR in dB	Bitrate in bpp
	Proposed Method	Ni Method		
Lena	40740	2618	33.79	0.154
Goldhill	35022	2618	37.88	0.110
Baboon	16465	2759	36.24	0.063
Barbara	41327	2405	36.63	0.157
Flower	39654	3010	34.56	0.075
Tire	33125	2675	39.67	0.081
Rice	40100	3250	35.66	0.081
Cameraman	37885	2905	40.01	0.077

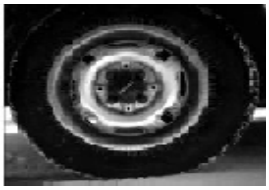
a) Lena (PSNR=33.79dB).

b) Goldhill (PSNR=37.88dB).

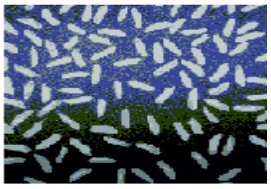
c) Baboon (PSNR=36.24dB).

d) Barbara (PSNR=36.63dB).

e) Flower (PSNR=34.56dB).



f) Tire (PSNR=39.67dB).



g) Rice (PSNR=35.66dB).

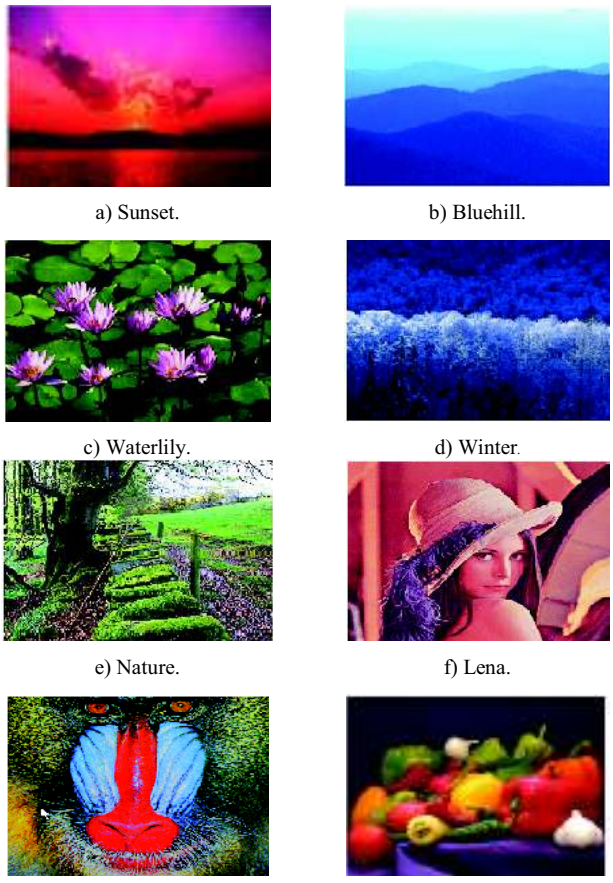
h) Cameraman (PSNR=40.01dB).

Figure 4. Watermarked images.

3.2. For RGB Images

For RGB images the embedding capacity depends upon the correlation of pixels in all the three individual R, G and B planes. The embedding capacity of an image may vary for the R, G and B planes. If the blue components in an image are highly correlated the embedding capacity in the blue plane of the image will be higher when compared to other two planes. Among

all images shown in Figure 5, the "Bluehill" image is found to have the highest embedding capacity because this image has higher correlation between adjacent pixels. The "Baboon" image is having the lowest embedding capacity because of the lower correlation of the adjacent pixels. The results are tabulated in Table 2. The embedding capacity and the PSNR have been calculated for the R, G and B planes of three different images. The embedding capacity of the RGB image is three times greater than the grayscale image. The experiments were done on eight different images- "Sunset", "Bluehill", "Waterlily", "Winter", "Nature", "Lena", "Baboon" and "Peppers" of size 800×600. The sample images used for the experiment are shown in Figure 5.



a) Sunset. b) Bluehill. c) Waterlily. d) Winter. e) Nature. f) Lena. g) Baboon. h) Peppers.

Figure 5. Sample RGB images.

Table 2. Hiding capacity and distortion for R, G and B planes for color images of size 800×600.

Host Image	Hiding Capacity in Bits				PSNR (dB)		
	R	G	B	Total	R	G	B
Sunset	100281	113199	105975	319455	44.79	44.91	44.54
Bluehill	211054	186406	224209	621669	47.85	47.53	48.51
Waterlily	95191	128804	91425	315420	44.61	44.42	44.58
Winter	95090	126754	91832	313676	43.41	45.45	44.18
Nature	89989	137847	92002	319838	45.21	43.95	43.25
Lena	245291	187352	110735	543381	46.98	44.25	45.63
Baboon	90980	98560	99420	288960	44.87	42.56	43.45
Peppers	126684	98456	173204	398344	44.52	44.01	44.86

4. Conclusions

In this paper, an efficient extension of the histogram modification technique is presented by considering the differences between adjacent pixels rather than simple pixel values. The pixel differences have a laplacian-like distribution. The adjacent pixels are highly correlated and this enables us to achieve large payload capacity. The payload capacity is very high in the grayscale images when compared with the older methods. In RGB images each color plane has its own embedding capacity, so the payload capacity of the RGB images and the gray scale images is approximately in the ratio of 3:1. In addition to the payload, sender and receiver must share the peak point information in order to retrieve the embedded message bits. All the experiments were simulated using MATLAB. In future, this algorithm can be implemented for video sequences by separating the video sequences into individual frames. Each frame can be considered as an image and the same data hiding algorithm can be implemented on each frame and finally, the individual frames can be combined to form the original video sequence.

References

[1] Awrangjeb M., "An Overview of the Reversible Data Hiding," *in Proceedings of 6th International Conference on Computer and Information Technology*, Bangladesh, pp. 75-79, 2003.

[2] Celik U., Sharm G., Tekalp A., and Saber E., "Lossless Generalized LSB Data Embedding," *Journal of IEEE Transactions on Image Processing*, vol. 14, no. 2, pp. 253-266, 2005.

[3] Chin-Chen C., Wei-Liang T., and Kuo-Nan C., "Lossless Data Hiding Based on Histogram Modification for Image Authentication," *in Proceedings of IEEE/IFIP International conference on Embedded and Ubiquitous Computing*, Shanghai, vol. 1, pp. 506-511, 2008.

[4] De-vleeschouwer C., Delaigle J., and Macq B., "Circular Interpretation of Bijective Transformation in lossless Watermarking for Media Asset Management," *IEEE Transactions on Multimedia*, vol. 5, no. 1, pp. 97-105, 2003.

[5] Ni Z., Shi Y., Ansari N., Su W., Sun Q., and Lin X., "Robust Lossless Image Data Hiding Designed for Semi-Fragile Image Authentication," *IEEE Transactions on Circuits Systems Video Technology*, vol. 18, no. 4, pp. 497-509, 2008.

[6] Norishige M., "Digital Watermarking Technology with Practical Applications," *Informing Science Special Issue on Multimedia Informing Technology-Part 1*, vol. 2, no. 4, pp. 107-111, 1999.

[7] Tian J., "Reversible Data Embedding Using Difference Expansion," *IEEE Transactions on Circuits Systems for Video Technology*, vol. 13, no. 8, pp. 890-896, 2003.

[8] Wu M. and Lin B., "Data Hiding in Image and Video: Part I-Fundamental Issues and Solutions," *IEEE Transactions on Image Processing*, vol. 12, no. 6, pp. 685-695, 2003.

[9] Wu M., Yu H., and Lin B., "Data Hiding in Image and Video: Part II-Designs and Applications," *IEEE Transactions on Image Processing*, vol. 12, no. 6, pp. 696-705, 2003.

Rajkumar Ramaswamy obtained his BE degree in electronics and communication engineering from Maharaja Engineering College in the year 2007 and his Master's degree in communication systems from Kumaraguru college of Technology in 2010. His interest is in the field of image processing, data mining and wireless communication.

Vasuki Arumugam obtained her BE degree in electronics and communication engineering from PSG College of Technology in the year 1989 and her Master's degree in applied electronics from Coimbatore Institute of Technology in 1991. She has completed her PhD in image compression from Anna University Chennai in 2010. She has published over 30 papers in journals and conferences and her research interest is in the field of data/ image compression. She is currently working as professor in the Department of ECE, Kumaraguru College of Technology, India.