ISSN:2231-0711

Two Out of Three Threshold Visual Cryptography Data Hiding Scheme for Block Truncation Coding

Ashwini N , Shreekanth T

Department of Electronics and Communication,
Sri Jayachamarajendra College of Engineering,
Mysore-India

Abstract— **For Image processing with low computational complexity Block truncation coding (BTC) is an efficient compression tool. However, it has some deficiencies like the blocking and false contour which may cause defective image reproduction. These defects may be improved with the use of error-diffused BTC but the security issues can limit its industrial application. To improve the security issue and to widen the scope of market application, a new method of embedding a secret binary data in the Block Truncation Coded image shares is presented in this paper. The proposed scheme implements two of three threshold visual encryption. Error reduction is achieved by Floyd-Steinberg Error diffusion. The data hiding and recovery involves simple XOR operation which is a low complexity operation. In this scheme, the size of the shares is same as that of the original image. There is no expansion in the share size. The technique can be used with or without Error Diffusion.**

Keywords—**Block Truncation Coding, Error Diffusion, Secret Sharing, Threshold Visual Cryptography.**

I. Introduction

Block truncation coding (BTC) is a quantization method for compressing digital images. Its advantages are simplicity, fault tolerance, the relatively high compression efficiency and good image quality of the decoded image. Apart from data compression, Block Truncation Coding (BTC) of images provides data hiding. Over the years several methods of data hiding using BTC have been published in the literature [1]- [6]. In 2012, Jing-Ming Guo and Yun-Fu Liu [7] have reported a method to Hide a secret watermark in two BTC encoded shares. In their method XNOR scheme is employed to embed the water marks .In their scheme a single water mark is embedded in two shares. Same logic is also extended for multiple water marks.

However, in this paper, we describe a technique of hiding a secret in three BTC encoded shares such that any two shares or all the three shares combined can recover the secret. This is the 'two of three' threshold visual cryptographic scheme.

II. Basic BTC

In Block Truncation Coding (BTC), the given gray scale image is divided into non overlapping equal sized blocks and these blocks are processed to generate their respective Binary Blocks (Bit Maps) as follows[7].

A. BTC Encoding (Binarization) of blocks

Let matrix X represent the block under consideration. Let the mean pixel value and the standard deviation of X be μ and σ respectively. Let the pixel elements of the block at row i and column j be X(i, j) for i =1 to M and j = 1 to N, where M is the number of rows and N is the number of columns of X. Then, the Binarized Block BB of X is obtained using the following Equation.

$$\mathrm{bb(i,j)} = \begin{cases} 1, & \text{if } x(i,j) > \mu \\ 0, & \text{otherwise} \end{cases} \qquad (1)$$

for i =1 to M and j = 1 to N.

Here, bb(i, j) is the element of BB at row i and column j. The encoding process also generates two quantization pixel levels called low-mean and high-mean as follows.

The low-mean L is calculated as,

$$\mathrm{L} = \mu - \sigma\sqrt{\frac{q}{M*N-q}} \qquad (2)$$

and the high-mean H is calculated as,

$$\mathrm{H} = \mu + \sigma\sqrt{\frac{M*N-q}{q}} \qquad (3)$$

Here, q is the number of 1's in the binarized block B and M*N is the total number of elements in B.

B. BTC Decoding (Reconstruction)

The Binarized Block BB is decoded by substituting for its 1's and 0's to get the reconstructed block RB, as follows.

$$\mathrm{rb(i,j)} = \begin{cases} \mathrm{H}, & \text{if } \mathrm{bb(i,j)} = 1 \\ \mathrm{L}, & \text{if } \mathrm{bb(i,j)} = 0 \end{cases} \qquad (4)$$

for i =1 to M and j = 1 to N.

The H and L values as given by Eqs. (2) and (3) are so chosen that the mean and standard deviation of RB are same as those of X. The BTC encoding (binarization) and decoding (reconstruction) process can be represented as shown in Fig.1.

Here, RB is a two level quantized approximation of X.

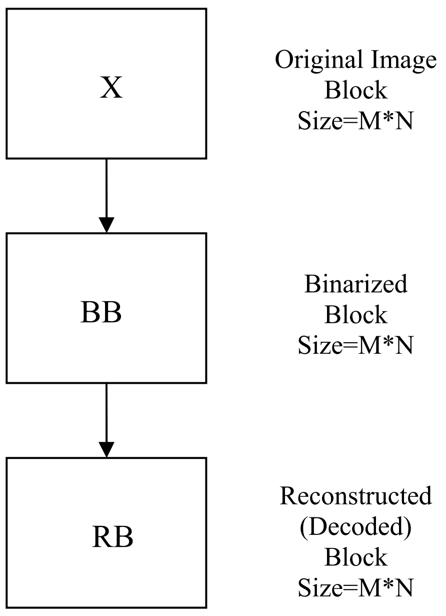

X
Original Image Block Size=M*N
BB
Binarized Block Size=M*N
RB
Reconstructed (Decoded) Block Size=M*N

Fig.1. Basic BTC operation

C. Special case when all the elements of X are equal

In this case u = x(i, j) for all i's and j's and σ = 0. Hence, from Eq. (1), bb(i, j) = 0 for all i's and j's. Therefore BB is an all zero block. In this case, from Eqs. (2) and (3), L = H =μ. Since bb(i, j)'s are all zeros, the reconstructed block C according to Eq. (4) would be all L's. Thus when all the elements of C are same, the corresponding Binary Block is an all zero matrix. In this case C and X are equal.

Example 1. Consider a 3*3 image block X with values as shown in Table Ia. The mean μ and standard deviation σ of X are calculated and found to be, μ = 34.77 and $\sigma = 10.98$. The Binarized Block BB, found using Eq. (1) is shown in Table Ib. The low-mean and high-mean are found to be 22.49 and 44.60. On rounding off, L = 22 and H =45. The decoded (reconstructed)block C is shown in Table Ic.

Table Ia. IMAGE BLOCK X.

21	53	36
39	24	20
48	40	32

Table Ib. BINARIZED BLOCK BB.

0	1	1
1	0	0
1	1	0

Table Ic. DECODED BLOCK C.

22	45	45
45	22	22
45	45	22

The BTC operation is applied to all the blocks of the given image to get the corresponding Binarized Blocks. From these Binarized Blocks, the BTC decoded image is reconstructed. The aggregation of the Binarized Blocks in the corresponding order is designated as the Binarized Image which is used in data hiding operations.

III. HIDING A SECRET IN TWO SHARES

Let the case of hiding a binary secret document image W in two shares of a BTC encoded image be considered. Let I be the original gray scale image of size m*n. Let the Binarized Image of I be designated as B. The size of B is same as that of I. One way to hide W in two shares is to split W into two parts and hide them respectively in two BTC shares as follows.

A. Alternate White Pixel grouping of W

The white pixels of W are split alternatively into two equal non over lapping shares and stored in matrix W1 and W2 respectively. W1 and W2 have the same size as that of W. The location of a white pixel of W occupies the same location (row-column position) when stored either in W1 or W2. When a white pixel from W goes into W1, its 4-connected neighbourhood white pixels in W go into W2 and vice-versa.

Alternative white pixels from W are extracted using a checkerboard mask which provides both horizontal and vertical alternative distribution of white pixels A checker board matrix has alternating 0's and 1's along both rows and columns. A 6*6 checker board matrix is shown in Table II. In our method, the checker board matrix C of the same size as that of W is used to extract W1 and W2 from W using logical *and* operation as follows.

$$W1 = \text{and}(W, C) \quad (5)$$

$$W2 = \text{and}(W, \sim C) \quad (6)$$

Here, ~C is the complement of C

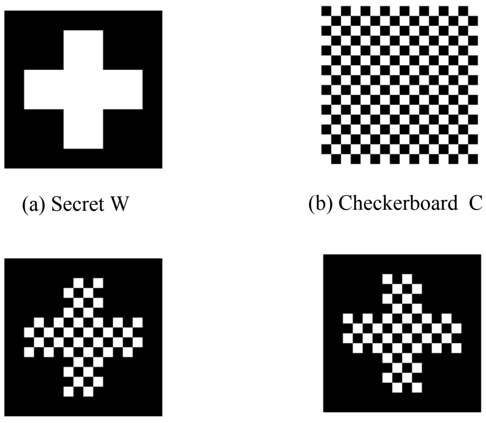
Table II. 6*6 CHECKERBOARD MATRIX

0	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0
0	1	0	1	0	1
1	0	1	0	1	0

The logical *and* operation of W and C in Eq. (5), extracts those white pixels (1's) of W which are common to both W and C. Since ~C is the complement of C, *and* operation in Eq. (6) extracts the remaining white pixels.

Example 2. A 16*16 sized W is shown in Fig. 2(a). Same sized checkerboard C is shown in Fig. 2(b).

In a given region of the checkerboard, the number of 1's is almost 50% of the total number of pixels. Therefore the number of white pixel in W1 will be approximately 50% of W. Similarly, for W2, the number will be about 50% of W. When alternate white pixels are extracted to W1 and W2, the shape of the secret image W is retained in W1 and W2 with super imposed checkerboard mask. The user can easily identify the object in W1 and W2. W1 and W2 are almost same in shape and size.


(a) Secret W (b) Checkerboard C

(c) Extract W1 (d) Extract W2

Fig.2. Alternate white pixel extraction from W

B. *Embedding W1 and W2 in two Shares*

Using the basic Binarized Image B of I, the two shares are initially generated as,

$$B1 = B \text{ XOR } W1 \quad (7)$$

and

$$B2 = B \text{ XOR } W2 \quad (8)$$

Here, it is assumed that the size of W is also m*n which is same as the size of B. Now consider the XOR result of B1 and B2. From Eqs. (5) and (6),

$$B1 \text{ XOR } B2 = (B \text{ XOR } W1) \text{ XOR } (B \text{ XOR } W2) \quad (9)$$

On simplification of the RHS, B gets cancelled and we get,

$$B1 \text{ XOR } B2 = W1 \text{ XOR } W2 \quad (10)$$

Substituting for W1 and W2 from Eqs. (5) and (6) and further Boolean simplification leads to,

$$B1 \text{ XOR } B2 = W$$

Or

$$W = B1 \text{ XOR } B2 \quad (11)$$

Thus W is fully recovered using Eq. (11).

In the data hiding scheme, B1 and B2 are decoded to get R1 and R2 based on Eq. (4). R1 and R2 are the BTC quantized approximations of the original image I and they constitute the 2 final shares.

C. *Recovery of B1, B2 and W from R1 and R2*

Each block of R1 or R2 is a two level block. Each block is binarized by replacing the H (higher) value by 1's and L (lower) values by 0's. In the special case where a single value is present in the block, its binary representation is an all zero matrix. The binarized matrices of R1 and R2 are B1 and B2 respectively. After getting B1 and B2, Eq. (11) is used to recover W.

IV. Two of Three Scheme

Here, the secret W is split into three equal non over lapping shares and stored in matrices W1, W2 and W3 such that the adjacent white pixels of W go into successive shares W1, W2, W3, W1, W2,… so on. This is done as follows.

A. *Trisection of W into W1, W2 and W3*

The locations of the white pixels of W are represented by the linear index vector[8] (single subscript index into a vector containing all the values of the matrix in column order) as,

$$L=\text{find}(W) \quad (12)$$

The find(W) function gives the linear index list. Here, L is the column vector whose elements give the locations of the white pixels (value =1) of W in the column order. That is, the matrix W is sequentially scanned for white pixels along column 1 from top to bottom (row 1 to row m), then along column 2, and so on. While scanning in this order, the location index of the first white pixel of W is stored in L(1), the second white pixel of W is stored in L(2) and so on. The white pixels are trisected as follows. The matrix size of W1 is set to m*n which is same size as that W.

Elements at L(1), L(4), L(7)….L(3*i+1) of W1 are set to 1. Remaining elements are set to zero (black pixels).

Elements at L(2), L(5), L(8)….L(3*i+2) of W2 are set to 1. Remaining elements are set to zero (black pixels).

Elements at L(3), L(6), L(9)….L(3*i+3) of W3 are set to 1. Remaining elements are set to zero (black pixels).

Thus W1, W2 and W3 hold 1/3 of white pixels of W in the order as described above. If the size of L is an exact multiple of 3, then the number of white pixels in W1, W2 and W3 are exactly 1/3 of W. Otherwise they differ by 1 among themselves.

Example 3. An 8*8 sized W matrix is shown in Table III. The white pixels of W are marked with white background while the black pixels are marked in gray. The linear indices of all the elements of W are numbered in Table III in the column order. Then, L = find(W) gives the indices of white pixels only in the column order as,

L=[12 13 19 20 21 22 26 27 28 29 30 . . 46 52 53]

There are 24 elements in L.

Here, L(1) =12, L(2) = 13, … L(24) = 53.

The white pixel indices of W1, W2 and W3 are,

W1(white) = [L(1) L(4) … L(22)]
= [12 20 26 29 34 37 43 46]

W2(white) = [L(2) L(5) … L(23)]
= [13 21 27 30 35 38 44 52]

W3(white) = [L(3) L(6) … L(24)]
= [19 22 28 31 36 39 45 53]

Table III. Matrix W and its linear indices.

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Table IV. Matrix W1.

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

Matrix W1 is shown in Table IV. Matrix W2 and W3 will be similar.

Matrices W1, W2 and W3 are non over lapping and are 1/3 of W. Therefore combining 2 out of these 3 will cover 2/3 of W and the combination approximately represents W. Matrices W2 and W3 of Example 3 have been combined and the result is shown in Table V.

Table V. Combination of W2 and W3.

1	9	17	25	33	41	49	57
2	10	18	26	34	42	50	58
3	11	19	27	35	43	51	59
4	12	20	28	36	44	52	60
5	13	21	29	37	45	53	61
6	14	22	30	38	46	54	62
7	15	23	31	39	47	55	63
8	16	24	32	40	48	56	64

B. *Embedding of W1, W2 and W3 in three BTC shares*

Using the basic Binarized Image B of I and W1, W2 and W3 the three shares are generated as follows.

B1 = B XOR W1 (12)

B2 = B XOR W2 (13)

B3 = B XOR W3 (14)

C. *Recovery of W from the shares B1,B2 and B3*

Now the recovery of W from the shares B1, B2 and B3 is done using XOR operations. Consider B1 XOR B2. Substituting for B1 and B2 from Eqs. (12) and (13),

B1 XOR B2 = (B XOR W1) XOR (B XOR W2)

According to the XOR logic, B gets cancelled on the RHS and

B1 XOR B2 = W1 XOR W2

Calling W1 XOR W2 as W12, the above relation can be expressed as,

W12 = W1 XOR W2 = B1 XOR B2 (15)

Since W1 and W2 are binary matrices of non overlapping white pixels, W1 XOR W2 = W1+W2. Therefore W12 represents 2/3 of W and the shape of W12 is approximately same as that of W. Similarly,

W13 = W1 XOR W3 = B1 XOR B3 (16)

W23 = W2 XOR W3 = B2 XOR B3 (17)

W13 and W23 also represent 2/3 of W with their shapes approximately same as that of W. Thus using any two binary shares the original secret W can be recovered with tolerable visual distortion.

In practical data hiding schemes, binary matrices B1, B2 and B3 are converted block by block into BTC quantized, reconstructed images R1, R2 and R3 as indicated by Eq. (4).

D. *Recovery of B1, B2, B3 and W from R1, R2 and R3*

Each block of R1 or R2 or R3 is a two level block. Each block is binarized by replacing the corresponding H (higher) value by 1's and L (lower) values by 0's. In the special case where a single value is present in the block, its binary representation is an all zero matrix. The binarized matrices of R1,R2 and R3 are B1, B2 and B3 respectively. After getting B1, B2 and B3, Eqs. (15), (16) and (17) are used to recover W12, W13 and W23.

E. *Algorithm for two of three data hiding and recovery*

Two of three threshold visual cryptographic share generation and secret recovery can be described as follows.

1) *Share Generation (Data Hiding):* **Algorithm 1.**

Input: Gray scale Image I and the binary secret document image W.

Output: Three gray scale shares R1, R2 and R3.

1. Get the BTC binarized image B of I as described in section II.
2. Decompose W into three non overlapping shares W1, W2 and W3 as described in section IV.A.
3. Get B1, B2 and B3 using Eqs. (12), (13) and (14) as,

   B1 = B XOR W1
   B2 = B XOR W2
   B3 = B XOR W3
4. From B1, B2 and B3, create block by block, R1, R2 and R3 respectively as indicated by Eq. (4).

2) *Secret recovery:* **Algorithm 2.**

Input: Any two shares out of R1, R2 and R3.

Output: Secret Images W12 or W13 or W23.

1. Convert block by block, the given shares Ri and Rj into their respective BTC binary equivalents Bi and Bj as described in section IV.D.
2. Recover the secret image as Wij = Bi XOR Bj. Wij will be either W12 or W13 or W23 depending on the shares Bi and Bj as dictated by Eqs. (15), (16) and (17). Thus,

   W12 = B1 XOR B2,
   W13 = B1 XOR B3,
   W23 = B2 XOR B3.

F. *Error reduction by diffusion*

When a secret is hidden in a binary share, the corresponding bit values are complemented. This in turn contributes additional errors in the BTC reconstructed shares because of the quantization process involved in BTC. To reduce the severity of the truncation error, Error Diffusion techniques can be adopted [7], [9], [10]. In this paper we have adopted Floyd-Steinberg Error Diffusion technique[9] to reduce the effect of quantization error. Since the technique is well known, it is not discussed here. The Error diffusion corrections are applied to those pixels affected by data hiding operation, because each white pixel of W1/W2/W3 complements the corresponding BTC bit in the share. This results in the interchange of L and H values in the BTC encoding process which contributes additional error. Because of error diffusion the edges and contours of the hidden data are smoothened. The results of data hiding using BTC with and without Error diffusion are given in the next section.

V. Test Results

The proposed secret sharing method has been implemented using MATLAB 7.0(R2010a). In the following Example 4, Error diffusion is not included. Only BTC encoding/reconstruction with data hiding is employed. Same image and secret data have been used with Error Diffusion technique.

A. *Results Without Error diffusion*

Example 4. A secret black and white image W containing the word "12345" is embedded in 3 shares of image 'Lena' of size 256*256. The results are shown in Fig. 3(a to i). Here a modestly large BTC block size is chosen at 16*16 to highlight the error due to larger block sizes.

Figs. 3(a), 3(b) and 3(c) show the BTC reconstructed shares R1, R2 and R3 respectively. Corresponding Binarized images B1, B2 and B3 are shown in Figs. 3(d), 3(e) and 3(f) respectively. Figures 3(g), 3(h) and 3(i) show the recovered secret images W12, W13 and W23 respectively. Each one of them represent 2/3 of W. All the three are almost same. The hidden object "12345" is very faintly visible in Figs. 3(a), 3(b) and 3(c) and little more visible in Figs. 3(d), 3(e) and 3(f).

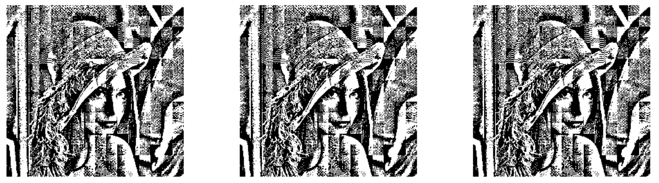
In Fig.4 recovered secret set is not shown, because it remains same irrespective of the block size. When the block size decreases, the magnitude of the quantization error also decreases.

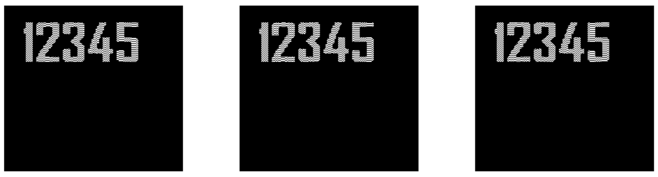
The error in the recovered secret WR can be expressed using the Correct Decode Rate (CDR)[7] defined as,

$$\mathrm{CDR} = \frac{\text{number of elements in (WR XNOR W)}}{\text{number of elements in W}} \qquad (24)$$

CDR calculated for different block sizes is given in Table VI. The size of W is 256*256.

(a) Share R1 (b) Share R2 (c) Share R3

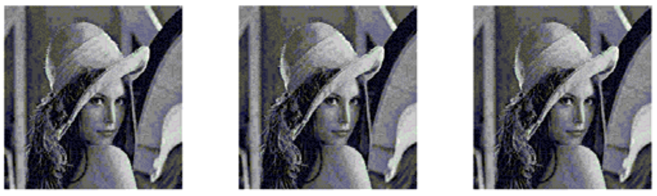

(d) Binary B1 (e) Binary B2 (f) Binary B3



(g) Secret W12 (h) Secret W13 (i) Secret W23

Fig. 3. BTC Shares, their Binary Formats and the Recovered Secret Images for Block Size **16*16**.

This visibility of hidden object in the shares can be almost eliminated by choosing a smaller block size for BTC. This is demonstrated in Fig. 4, where a BTC block size of 4*4 is used.


(a) Share R1 (b) Share R2 (c) Share R3

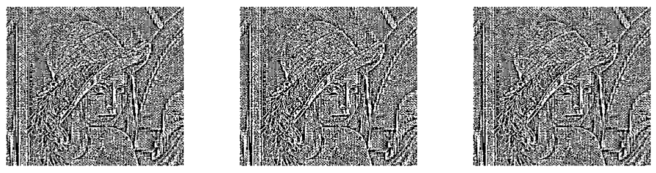

(d) Binary B1 (e) Binary B2 (f) Binary B3

Fig. 4. BTC Shares, their Binary Formats and the Recovered Secret Images. for Block Size **4*4.**

Table VI. CDR values without Error Diffusion

Block Size	CDR W12	CDR W13	CDR WR23
8*8	0.9740	0.9740	0.9739
2*2	0.9740	0.9740	0.9739

The CDR values remain same as the block size varies because the block size does not affect the secret recovery. The CDR value is very near to 1 indicating that the difference between W and W12/W13/W23 is very small.

Mean Square Error (MSE) values between the original image I and BTC reconstructed, secret hidden shares R1, R2 and R3 are given in Table VII.

TABLE VII. MSE VALUES FOR DIFFERENT BLOCK SIZES.

Block size	MSE R1	MSE R2	MSE R3
4*4	184.4	181.8	184.6
4*2	131.1	128.6	131.6
2*4	154.4	150.7	153.1
2*2	86.1	84.5	85.0

From Table VII, it can be seen that MSE decreases as the block size decreases. A lower block size increases the computation time

B. Results with Error Diffusion

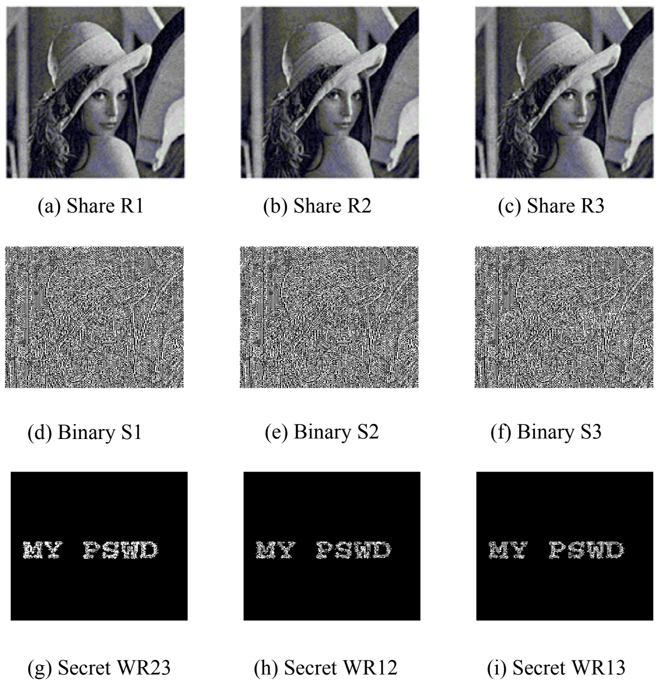


Fig. 5.BTC Shares, their Binary Formats and the Recovered Secret Images for Block Size **4*4**, with Error Diffusion

The image shares, binary images and recovered secret images with Error Diffusion are shown in Fig. 5 in which the binary images have lost their edge sharpness.

Because of Error Diffusion, The recovered secret also gets diffused. Then the error in the recovered secret expressed as CDR's are shown in Table VIII. Because of Error Diffusion, The recovered secret also gets diffused.

CDR's for different block sizes is given in Table VIII. The size of W is 256*256.

TABLE VIII. CDR VALUES WITHOUT ERROR DIFFUSION

Block Size	CDR W12	CDR W13	CDR WR23
4*4	0.5945	0.5992	0.5869
4*2	0.5933	0.5953	0.5877
2*4	0.5808	0.5937	0.5881
2*2	0.5800	0.6043	0.5955

When the block size is reduced, CDR's remains almost constant. The MSE between the original image I and the BTC data hidden shares for different block sizes are shown in Table IX.

TABLE IX. MSE VALUES FOR DIFFERENT BLOCK SIZES.

Block Size	MSE R1	MSE R2	MSE R3
8*8	314.1	309.4	313.4
8*4	233.4	227.7	233.7
4*8	268.2	271.6	270.9
4*4	185.4	185.0	189.6
4*2	130.4	127.5	135.5
2*4	155.2	148.9	151.7
2*2	79.0	76.7	76.8

From Table IX, it can be seen that MSE decreases as the block size decreases. MSE versus block size is shown in Fig.6.

As the block size decreases, the number of blocks to be processed increases correspondingly. Therefore the BTC encoding/data hiding/decoding process time increases correspondingly.

In all these cases the three shares exhibit almost same error values, because, the three shares are almost identical. When all the three shares are available, the full W can be recovered.

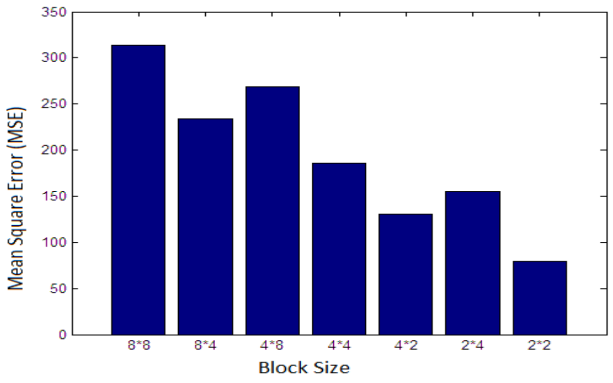


Fig.6. MSE versus Block size

C Comparison of results with and without ED

The results of the proposed data hiding scheme with and without error diffusion are given in tables VI to IX. For the data hiding scheme without error diffusion, CDR value is close to 100% but the mean square error (MSE) is high. For the data hiding scheme with error diffusion, CDR value is close to 60% but the mean square error (MSE) is low compared to the data hiding scheme without error diffusion. Depending on the requirements one can opt for data hiding scheme with or without error diffusion.

VI. CONCLUSION

A new technique of hiding data in three BTC shares to realize two of three threshold encryption is presented. The data hiding and recovery involves simple XOR operation which is a low complexity operation. The proposed scheme is simple and efficient. In this scheme, the size of the shares is same as that of the original image. There is no expansion in the share size. The present method is easy to operate and technique can be used with or without Error Diffusion which produces excellent results. The proposed technique can be extended for three out of four threshold visual cryptography and can also be extended for color images

REFERENCES

[1] Nimrod Peleg, "Block Truncation Coding" *cs.haifa.ac.il/~nimrod/Compression/Image/I5btc2005.pdf.*

[2] Pasi Fränti, Olli Nevalainen and Timo Kaukoranta, *"Compression of Digital Images by Block Truncation Coding: A Survey"* ,The Computer Journal, 37 (4), 308-332, 1994.

[3] Doaa Mohammed, Fatma Abou-Chadi, *"Image Compression Using Block Truncation Coding"*, Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT), February Edition, 2011.

[4] Edward J. Delp , O. Robert Mitchell, *"Image Compression using Block Truncation Coding"*, IEEE Transactions On Communications, Vol. Com-27, No. 9, September 1979

[5] Zou Xinguang ,Sun Shenghe "*Information Hiding Using Secret Sharing Scheme*", Innovative Computing, Information and Control, 2006.

[6] Cheonshik Kim, "*Data Hiding Based on Compressed Dithering Images", Studies in Computational Intelligence* , Volume 283, 2010, pp 89-98.

[7] Jing-Ming Guo and Yun-Fu Liu, *"High Capacity Data Hiding for Error Diffused Block Truncation Coding"*, IEEE transactions on image processing, vol. 21, no. 12, december pp.4808-4818, 2012.[8] www.mathworks.in

[9] R. W. Floyd and L. Steinberg, *"An adaptive algorithm for spatial gray scale,"* in *Proc. SID Dig. Soc. Inform. Display*, 1975, pp. 36–37.

[10]Halftoning by Error Diffusion www.ece .utexas.edu/~bevans/projects/.../talks/ErrorDiffusion.html