Available online at http://www.journalcra.com

International Journal of Current Research
Vol. 9, Issue, 02, pp.46112-46117, February, 2017

INTERNATIONAL JOURNAL OF CURRENT RESEARCH

ISSN: 0975-833X

RESEARCH ARTICLE

ANALYSIS OF IMAGE COMPRESSION ALGORITHM USING BLOCK TRUNCATION CODING WITH MODIFIED QUANTIZATION FOR GRAY SCALE IMAGE

***Ruchika Sewaiwar and Prof. Bharti**

Department of Electronics and Communication Engineering, SIS Tech. Bhopal, India

ARTICLE INFO

Article History:
Received 09th November, 2016
Received in revised form
25th December, 2016
Accepted 18th January, 2017
Published online 28th February, 2017

Key words:

BTC, Compression, Quantization, Algorithm, Image retrieval, Multimedia.

ABSTRACT

In the present era of communication system, the requirement of image storage and transmission for image processing are increasing exponentially. This is why; the need for better compression technology is in extremely demands. In this paper, a gray image compression method using modified quantization scheme is proposed. This method having the advantages of BTC and quantization both. The BTC algorithm with quantization has some controlling parameters through which we can control the quality and compression of the image. The performance of the proposed method has been evaluated in terms of Entropy, PSNR, MSE and SSIM. The result of the proposed work is better than the previous work of literature.

Citation: Ruchika Sewaiwar and Prof. Bharti, 2017. "Analysis of Image compression algorithm using Block Truncation coding with modified quantization for gray scale image", *International Journal of Current Research*, 9, (02), 46112-46117.

INTRODUCTION

Block truncation coding is one of the lossy coding techniques applicable for Gray scale images. It reduces the file size but loses some extent of original information of the image (Wang *et al.,* 2002). The significant advantages of this coding approach are low computational complexity and high parallelism. The basic idea of BTC is to perform moment preserving quantization for blocks of pixels. The input image is divided into non-overlapping blocks of pixels of sizes 4×4, 8×8 and so on. Mean and standard deviation of the blocks are calculated. Mean is considered as the threshold and reconstruction values are determined using mean and standard deviation. Then a bitmap of the block is derived based on the value of the threshold which is the compressed or encoded image. Using the reconstruction values and the bitmap the reconstructed image is generated by the decoder. Thus in the encoding process, BTC produces a bitmap, mean and standard deviation for each block. It gives a compression ratio of 4 and bit rate of 2 bits per pixel when a 4×4 block is considered. This method provides a good compression without much degradation on the reconstructed image. But it shows some artifacts like staircase effects or raggedness near the edges. Due to its simplicity and easy implementation, BTC has gained big interest in its further improvement and application for image compression.

****Corresponding author: Ruchika Sewaiwar***
Department of Electronics and Communication Engineering, SIS Tech. Bhopal, India.

The algorithm for BTC is as follows:

Step 1: Input a gray scale image of size M×N pixels and the dimension of the square block k by which the image is to be divided into non-overlapping blocks.
Step 2: Divide the image into various blocks, each of size k×k, value of k can be 4, 8, 16, and so on. Each block, W is represented as;

$$\boldsymbol{W} = \begin{bmatrix} \boldsymbol{w_1} & \cdots & \boldsymbol{w_k} \\ \vdots & \ddots & \vdots \\ \boldsymbol{w_1}^2 & \cdots & \boldsymbol{w_k}^2 \end{bmatrix}$$

Step 3: calculate the mean and variance of gray level in block W

$$\mu = \frac{1}{p}\sum_{i=1}^{p} w_i \quad \text{..... (1)}$$

$$m_1 = \frac{1}{k}\sum_{i=1}^{k} f(x_i) \quad \text{..... (2)}$$

$$m_2 = \frac{1}{k}\sum_{i=1}^{k} f(x_i)^2 \quad \text{..... (3)}$$

m_1 is the sample mean and the sample variance σ^2 of image block is given by:

$$\sigma^2 = m_2 - m_1 \quad \text{..... (4)}$$

Step 3: Now the compressed bit map is obtained by

$$B = \begin{cases} 1, w_i > \mu \\ 0, w_i \leq \mu \end{cases} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (5)$$

Step 4: The bit map B, μ and ı are transmitted to the decoder. The algorithm for decoder is as follows:

Step 1: Calculate a and b

$$a = \mu + \sigma\sqrt{\frac{p}{q}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (6)$$

$$b = \mu - \sigma\sqrt{\frac{q}{p}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (7)$$

where p = No. of 0's in the bit map and q = No. of 1's in the bit map

Step 2: The reconstructed image Z can be obtained by replacing the element 1 in B with H and element 0 with L.

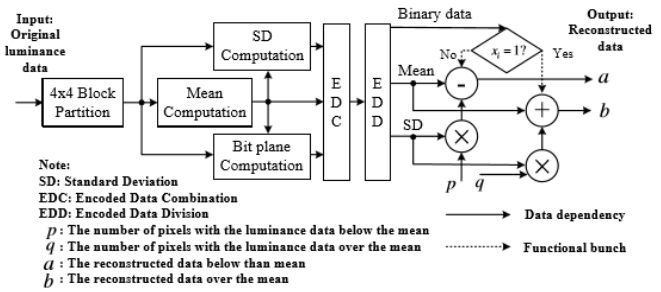

Figure 1. Block diagram of Block Truncation Coding

Proposed Work

In this proposed work lossy image compression has been implemented. For the implementation of Lossy image compression Block Truncation coding with region based segmentation has been applied. In the first stage image of size 256×256 has been segmented on the basis of region. Then a block of size n×n (where n= 4, 8 or 16) has been chosen. We obtain the minimum, maximum and mean value of pixel of that block. On the basis of threshold based on above parameter we evaluate a bit map for the particular block. The process is applied on every block of the image. Thus obtained bit pattern or logic matrix with parameters is send to the decoder end. On the decoder end the image is decompressed on the basis of transmitted bit map information.

Step 1: Input a gray scale or color image of size 256×256 pixels and the dimension of the square block k by which the image is to be divided into non-overlapping blocks
Step 2: image is segmented into different region using region based segmentation method.
Step 3: Divide the image into various blocks, each of size k×k, value of k can be 4, 8, 16, and so on. If image is overlapping corresponding to the block, there is zero padding.
Step 4: find minimum value of the pixel, maximum value of the pixel and mean value of the pixel. Evaluate the Threshold using below formula for each region

$$Threshold = \frac{max.\ pixel\ value + min.\ pixel\ value + mean}{3} \quad \ldots\ldots (8)$$

Step 5: Now the compressed bit map for each region is calculated by

$$\mathbf{B} = \begin{bmatrix} b_1 & \cdots & b_k \\ \vdots & \ddots & \vdots \\ {b_1}^2 & \cdots & {b_k}^2 \end{bmatrix}$$

Where, $b_j = \begin{cases} 1, w_i > T \\ 0, w_i \leq T \end{cases}$

Step 6: The bit map B, minimum value of the pixel, maximum value of the pixel and mean value is sent to the decoder.
Step 7: Pixels in the image block W are then classified into two ranges of values. The upper range is those Gray levels which are greater than T and lower range is those which are less than or equal to T. The mean of higher range (μ_H) and the lower range (μ_L) are calculated using (9) and (10) respectively. Then these two values are used for reconstruction of the image.

$$\mu_H = \frac{1}{p}\sum_{i=1}^{p} w_i, w_i > T \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (9)$$

$$\mu_L = \frac{1}{q}\sum_{i=1}^{q} w_i, w_i \leq T \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (10)$$

Where p is the number of Gray value greater than T and q is the number of Gray value less than T.

Step 8: Decode bitmap block B with the reconstruction values μ_H and the lower range μ_L in such a way that the elements assigned 0 are replaced with μ_L and elements assigned 1 are replaced with μ_H.

Then the decoded image block Z can be represented as,

$$\mathbf{Z} = \begin{bmatrix} z_1 & \cdots & z_k \\ \vdots & \ddots & \vdots \\ {z_1}^2 & \cdots & {z_k}^2 \end{bmatrix}$$

Where, $z_i = \begin{cases} \mu_L, b_j = 0 \\ \mu_L, b_j = 1 \end{cases}$

Simulation result

This paper has been implemented on the basis of above discussed algorithm using MATLAB simulator. First of all the work has been simulated on 4×4 block size then 8×8 and 16×16 block size for different input and reference images. Region based segmentation provides better quality in terms of MSE, PSNR and SSIM, which has been evaluated on the simulator. Finally the result has been compared with different existing method in the literature.

Simulation of Block truncation coding with modified quantization for 4×4 block size

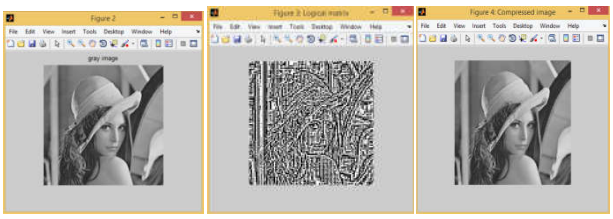

Figure 1. Lena Input image (a) gray scale image (b) logical matrix image (c) compressed image

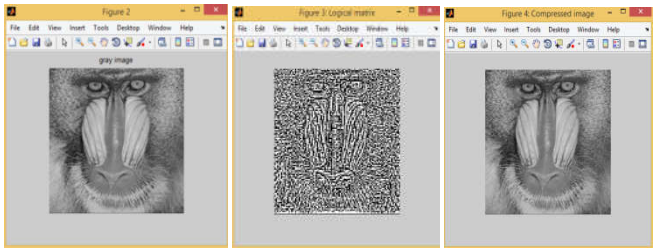

Figure 2. Baboon Image (a) Gray scale image (b) logical matrix image (c) compressed image

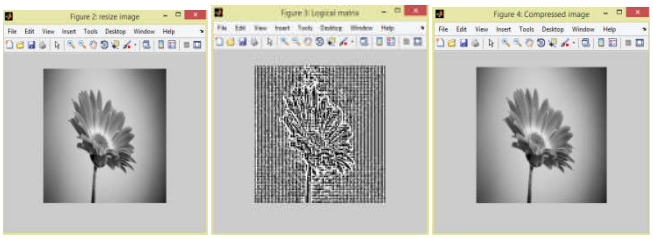

Figure 3. Flower Image (a) Gray scale image (b) logical matrix image (c) compressed image

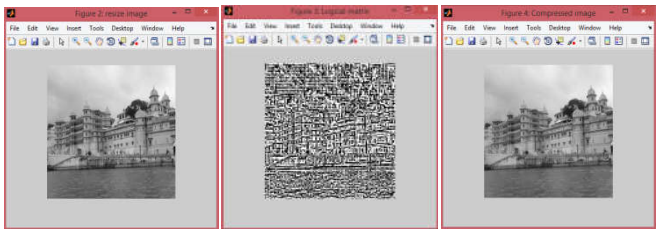

Figure 4. Fort Input image (a) gray scale image (b) logical matrix image (c) compressed image

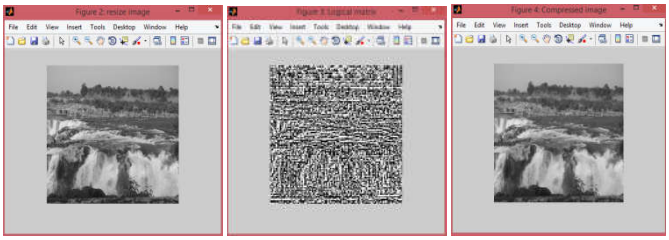

Figure 5. Fall Input image (a) gray scale image (b) logical matrix image (c) compressed image

Result analysis for 4*4 Block Size

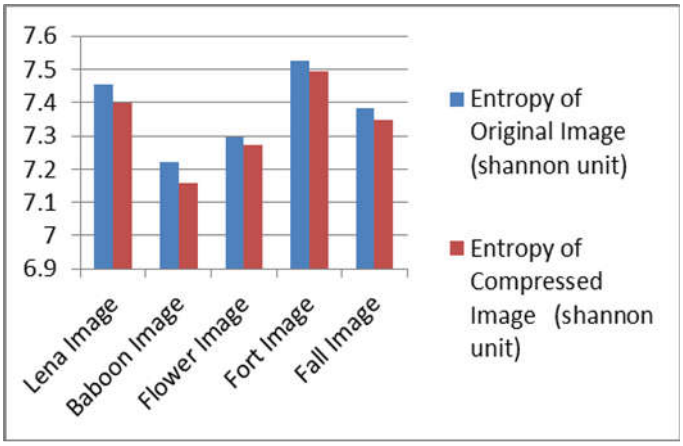

Figure 6. Graph depicting ENTROPY for different image of 4 ×4 block size

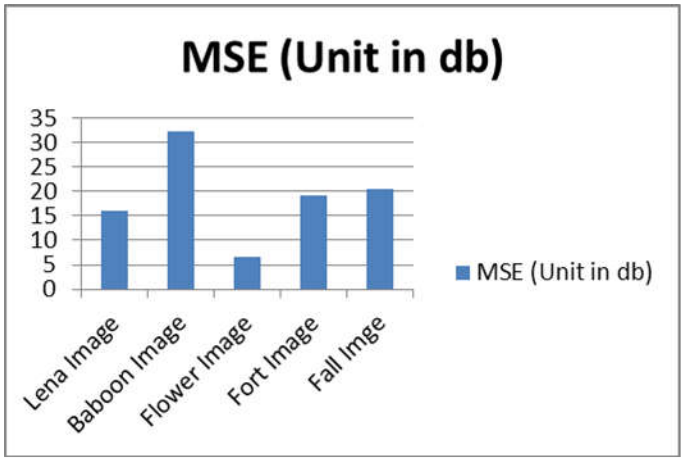

Figure 7. Graph depicting MSE for different image of 4 ×4 block size

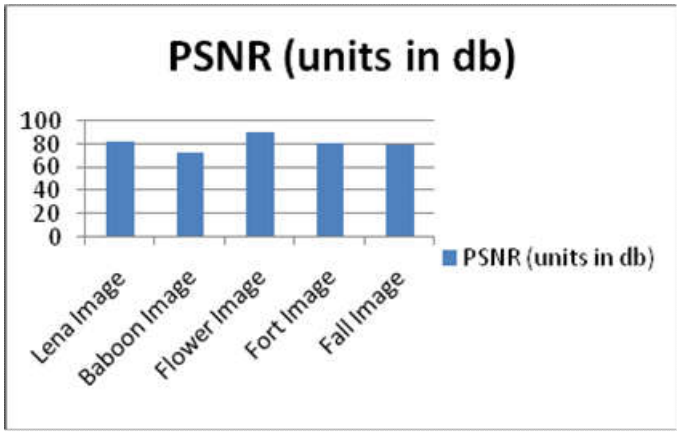

Figure 8. Graph depicting PSNR for different image of 4 ×4 block size

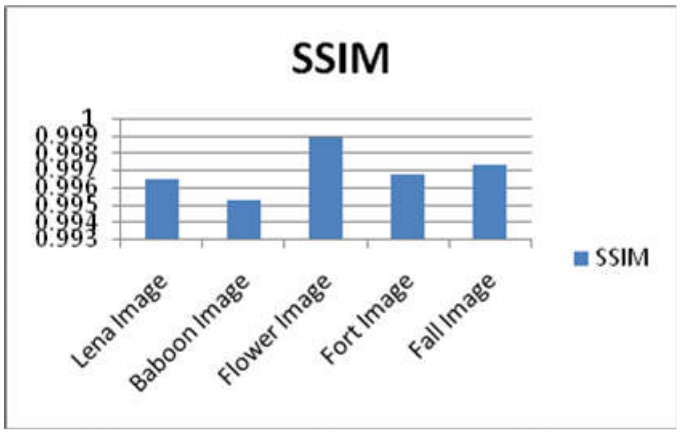

Figure 9. Graph depicting SSIM for different image of 4 ×4 block size

Simulation of Block truncation coding with modified quantization for 8×8 block size

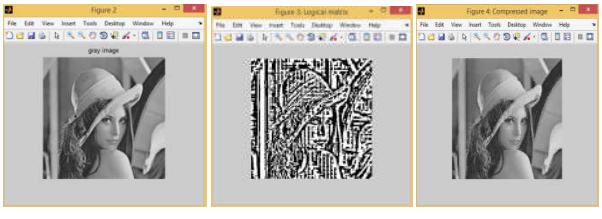

Figure 10. Lena Input image (a) gray scale image (b) logical matrix image (c) compressed image

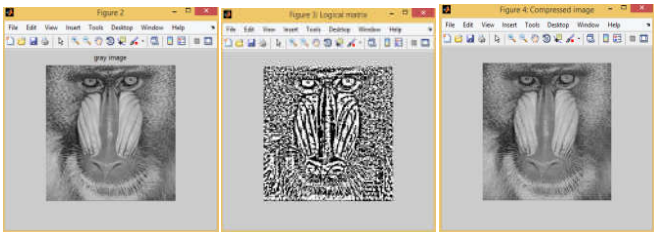

Figure 11. Baboon Input image (a) gray scale image (b) logical matrix image (c) compressed image

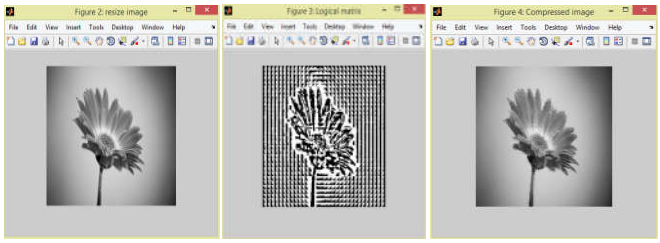

Figure 12. Flower Input image (a) gray scale image (b) logical matrix image (c) compressed image

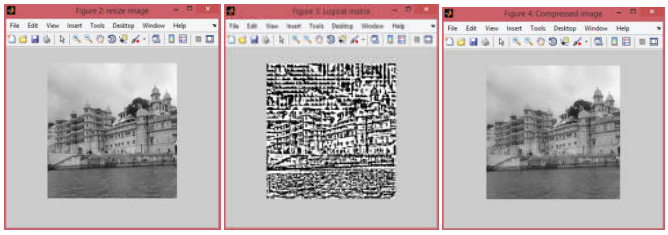

Figure 13. Fort Input image (a) gray scale image (b) logical matrix image (c) compressed image

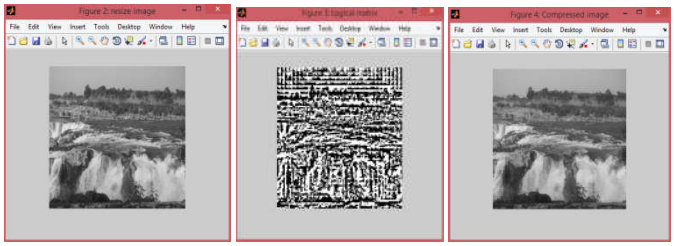

Figure 14. Fall Input image (a) gray scale image (b) logical matrix image (c) compressed image

Result analysis for 8×8 Block Size

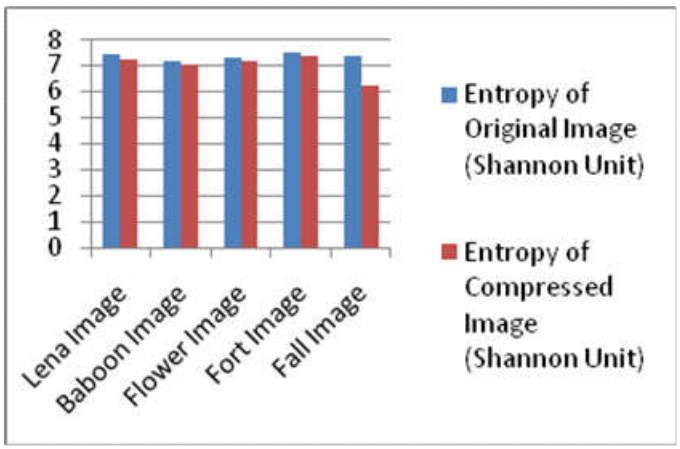

Figure 15. Graph depicting ENTROPY for different image of 8 ×8 block size

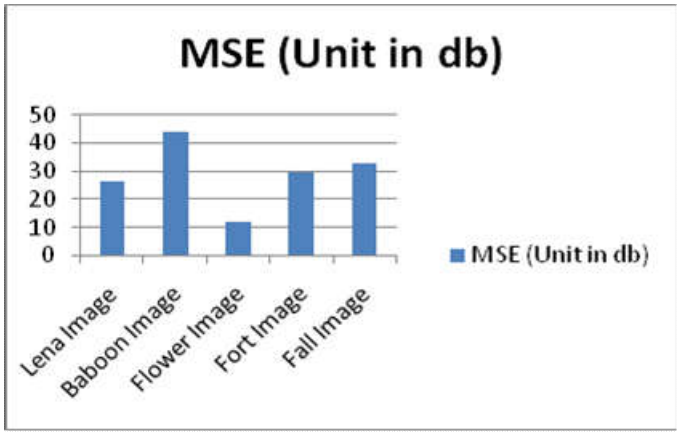

Figure 16. Graph depicting MSE for different image of 8 ×8 block size

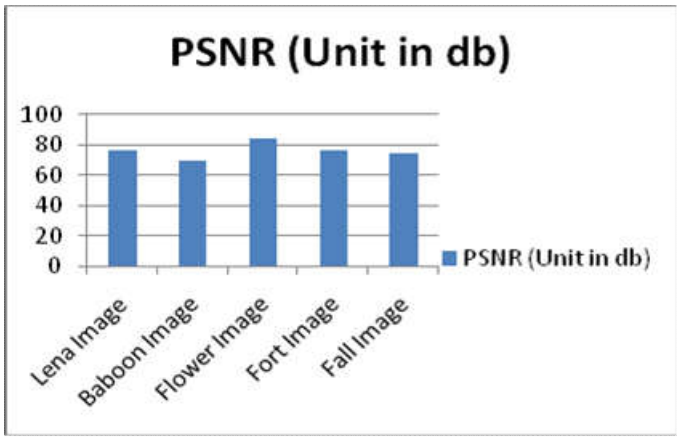

Figure 17. Graph depicting PSNR for different image of 8 ×8 block size

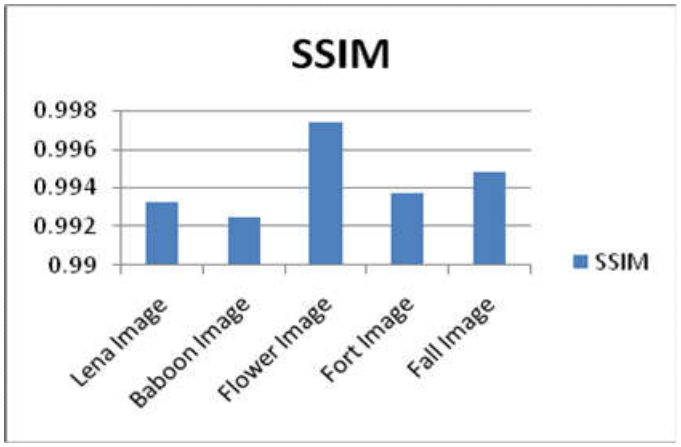

Figure 18. Graph depicting SSIM for different image of 8 ×8 block size

Simulation of Block truncation coding with modified quantization for 16×16 block size

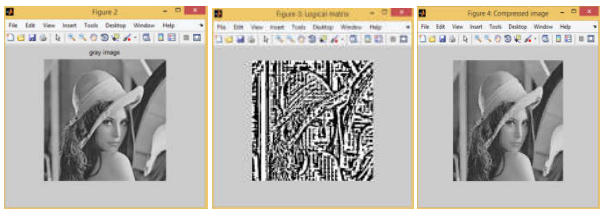

Figure 19. Lena Input image (a) gray scale image (b) logical matrix image (c) compressed image

Table 1. Performance in terms of MSE, PSNR, ENTROPY and SSIM for different image

Image	Entropy of Original Image	Entropy of Compressed Image	MSE	PSNR	SSIM
Lena Image	7.4543	7.3982	16.0643	81.8468	0.9965
Baboon Image	7.2193	7.1572	32.2887	72.9427	0.9952
Flower Image	7.2978	7.2722	6.5197	90.4435	0.9989
Fort Image	7.5246	7.4951	19.1468	80.7472	0.9967
Fall Image	7.3815	7.3488	20.4254	79.5282	0.9973

Table 2. Performance in terms of MSE, PSNR, ENTROPY and SSIM for different image

Image	Entropy of Original Image	Entropy of Compressed Image	MSE	PSNR	SSIM
Lena Image	7.4543	7.2804	26.5617	76.8181	0.9933
Baboon Image	7.2193	7.0802	44.0471	69.8373	0.9925
Flower Image	7.2978	7.2074	11.8553	84.4639	0.9974
Fort Image	7.5246	7.4155	29.3857	76.4635	0.9937
Fall Image	7.3815	7.2667	32.5785	74.8594	0.9948

Table 3. Performance in terms of MSE, PSNR, ENTROPY and SSIM for different image

Image	Entropy of Original Image	Entropy of Compressed Image	MSE	PSNR	SSIM
Lena Image	7.4543	7.0336	38.4451	73.1205	0.9897
Baboon Image	7.2193	6.8710	53.3275	67.9254	0.9901
Flower Image	7.2978	6.9816	19.4488	79.5139	0.9959
Fort Image	7.5246	7.1854	6.5177	74.2906	0.9912
Fall Image	7.3815	7.2667	32.5785	74.8594	0.9948

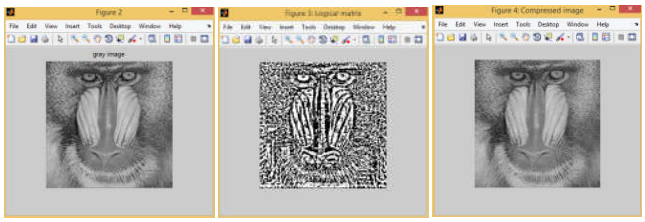

Figure 20. Baboon Input image (a) gray scale image (b) logical matrix image (c) compressed image

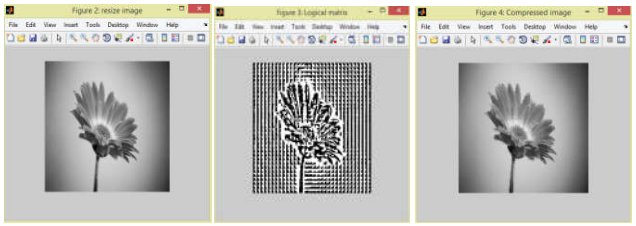

Figure 21. Flower Input image (a) gray scale image (b) logical matrix image (c) compressed image

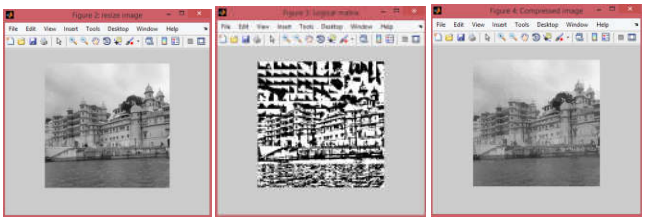

Figure 22. Fort Input image (a) gray scale image (b) logical matrix image (c) compressed image

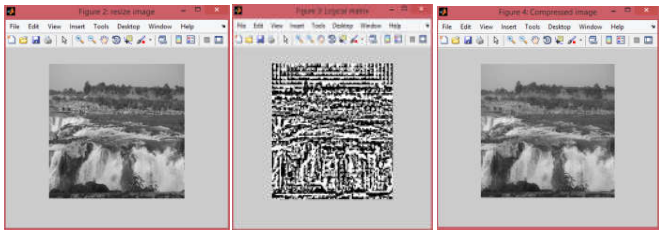

Figure 23. Fall Input image (a) gray scale image (b) logical matrix image (c) compressed image

Result analysis for 16×16 Block Size

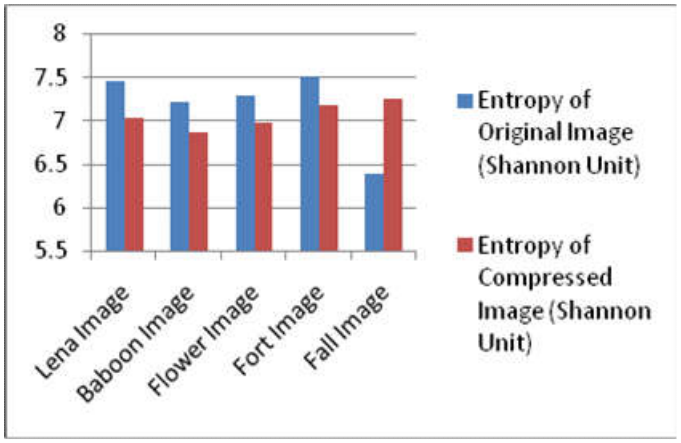

Figure 24. Graph depicting SSIM for different image of 16 ×16 block size

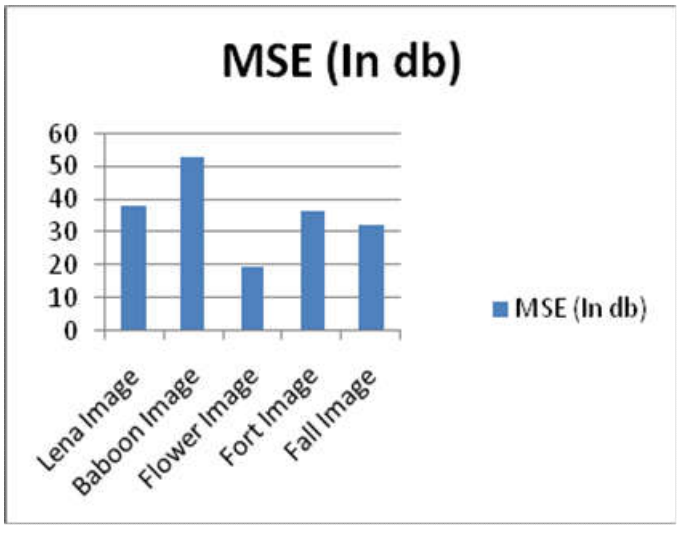

Figure 25. Graph depicting MSE for different image of 16 ×16 block size

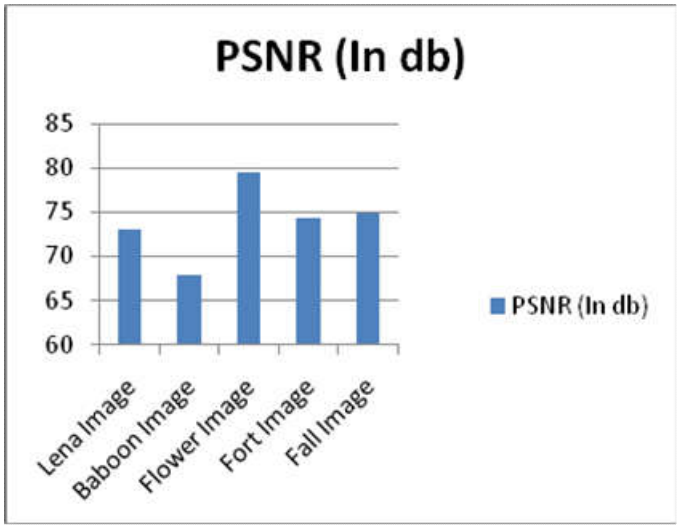

Figure 26. Graph depicting PSNR for different image of 16 ×16 block size

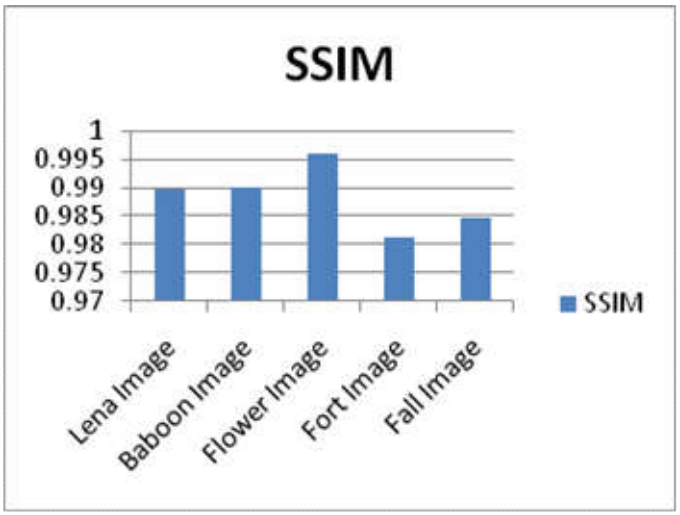

Figure 27. Graph depicting SSIM for different image of 16 ×16 block size

Conclusion

In this paper lossy image compression using Block Truncation Coding with modified quantization has been performed using MATLAB simulator. Different block size taken for the implementation like 4×4, 8×8 and 16×16 and the image size is 256 ×256. The performance analysis has been carried out using MSE, PSNR and SSIM for different real and reference images. Result shows better improvement over the existing methodology.

REFERENCES

Delp, E. J. and O. R. Mitchell, 1979. "Image compression using block truncation coding". IEEE Transactions on Communications, 27:1335 – 1342.

Gonzalez, R. C. and R. E. Woods, 2002. "Digital Image Processing", Prentice Hall.

Huffman, D. 1952. "A method for the construction of minimum redundancy codes". Proceeding of the IRE, 40:1098 – 1101.

Jacquin, A. E., 1992. "Image coding based on a fractal theory of iterated contractive image transformations". IEEE Transactions on Image Processing, 1:18 – 30.

Jayamol Mathews, Madhu S. Nair and Liza Jo, 2013. "Modified BTC Algorithm for Gray Scale Images using max-min Quantizer", IEEE.

Jing-Ming Guo and Yun-Fu Liu, 2014. "Improved Block Truncation Coding Using Optimized Dot Diffusion", IEEE Transactions on image processing, VOL. 23, NO. 3, March.

Jing-Ming Guo, Heri Prasetyo and Jen-Ho Chen 2015. "Content-Based Image Retrieval Using Error Diffusion Block Truncation Coding Features", IEEE Transactions on image processing, vol. 24, no.3, march.

Jing-Ming Guo, Heri Prasetyo, and Jen-Ho Chen, 2014. "Content-Based Image Retrieval Using Error Diffusion Block Truncation Coding Features", IEEE Transactions on Circuits and Systems for Video Technolo "Content-Based Image Retrieval Using Features Extracted From Halftoning-Based Block Truncation Coding".

Ki-Won Oh and Kang-Sun Choi, 2014. "Parallel Implementation of Hybrid Vector Quantizer based Block Truncation Coding for Mobile Display Stream Compression", IEEE ISCE.

Langdon, G. and J. Rissanen, 1981. Compression of black white images with arithmetic coding. IEEE Transactions on Communications, 29:858 – 867.

Moffat, A. 1991. Two-level context based compression of binary images. In Proc. Of the Data Compression, pages 382 – 391. IEEE Computer Society Press.

Oien, G. E. and S. Lepsoy, 1994. "Fractal-based image coding with fast decoder convergence". Signal Processing, 40:105 – 117.

Salomon, D. 2000. "Data Compression: The Complete Reference". Springer-Verlag, New York.

Seddeq E. Ghrare and Ahmed R. Khobaiz, 2014. "Digital Image Compression using Block Truncation Coding and Walsh Hadamard Transform Hybrid Technique", IEEE conference.

Storer, J. A. and H. Helfgott, 1997. Lossless image compression by block matching. *The Computer Journal*, 40:137 – 145.

Tabuman, D. and M. W. Marcelin, 2001. "JPEG 2000: image compression fundamentals, standards and practices", Kluwer Academic Publishers, 2001.

Thomas, L. and F. Deravi, 1995. "Region-based fractal image compression using heuristic search". IEEE Transactions on Image Processing, 4:832 – 838.

Wang, Z. and A. C. Bovik, 2002. "A universal image quality index". IEEE Signal Processing Letters, 9:81 – 84.

Wang, Z., A. C. Bovik, and L Lu, 2002. "Why is image quality assessment so difficult?" In Proc. of the IEEE international conference on Acoustic, Speech, and Signal Processing, pages 3313 – 3316.

Wang, Z., A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, 2004. "Image quality assessment: From error visibility to structural similarity". IEEE Transactions on Image Processing, 13:600 – 612, 2004.

****