Lossy Image Compression Using Block Truncation Coding With Region Based Segmentation Algorithm

[1]Arjun Bariya, [2]Dr. Anubhuti Khare
[1,2]Dept. of Electronics & communication Engineering, UIT, Bhopal, MP, India

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 & color image compression method using region based segmentation 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 PSNR, MSE, Entropy & SSIM. The result of the proposed work is evaluated by comparing the performance with that of the existing methods.

Keywords
BTC, Compression, Quantization, Algorithm, Image Retrieval, Multimedia

I. Block Truncation Coding

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 [9]. 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.

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;

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

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

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

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

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

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

$$\sigma^2 = m_2 - m_1 \quad (4)$$

Step 4: Now the compressed bit map is obtained by

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

Step 5: The bit map B, μ and 1 are transmitted to the decoder.

The algorithm for decoder is as follows:
Step 1: Calculate a & b

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

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

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.

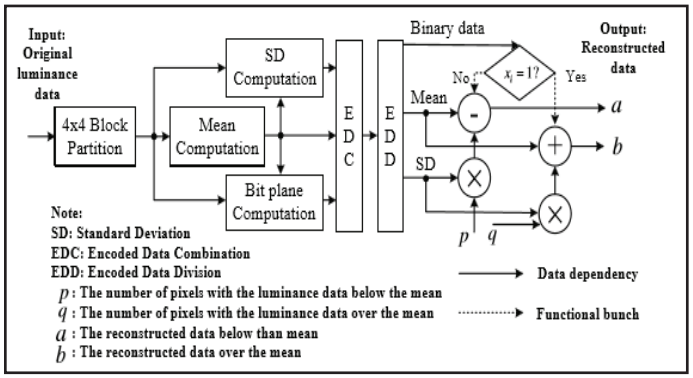


Fig. 1: Block Diagram of Block Truncation Coding

II. 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 & mean value of the pixel. Evaluate the Threshold using below formula for each region

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

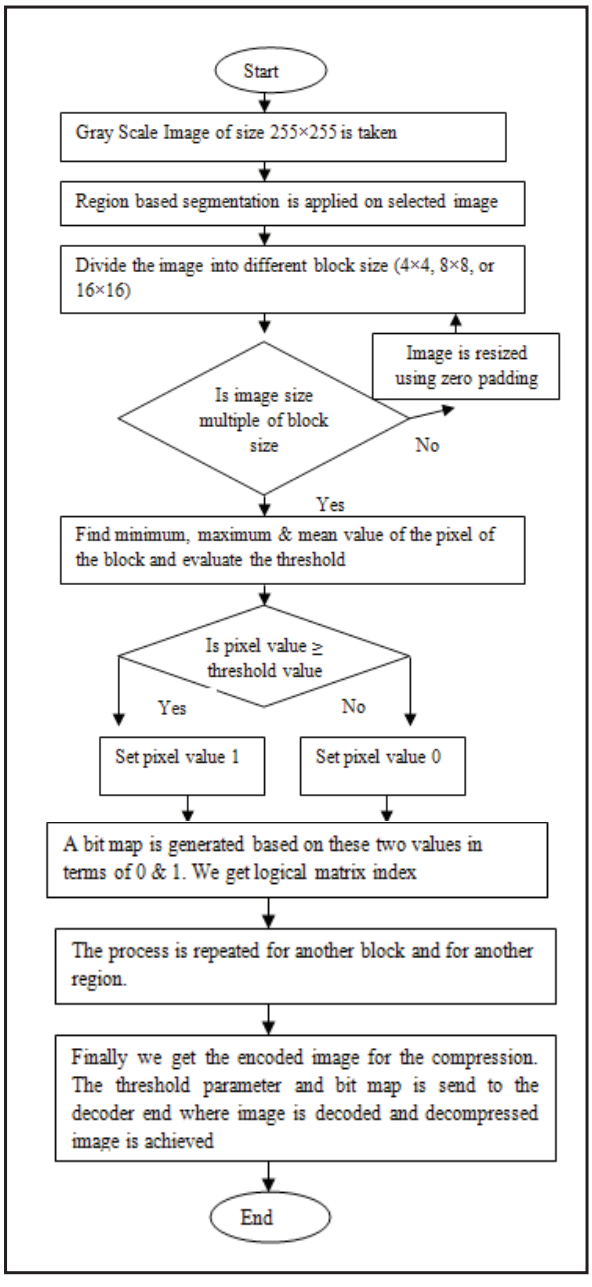


Fig. 2: Flow Chart of the Implemented Work

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 & 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.

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

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

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

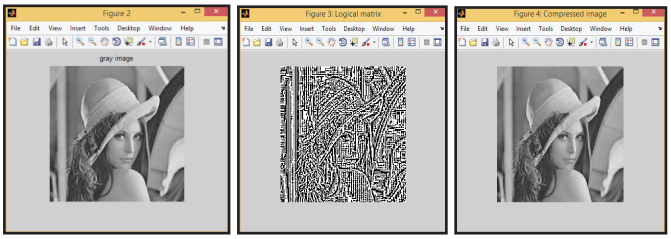
III. Simulation Result

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 Region based segmentation for 4×4 block size**

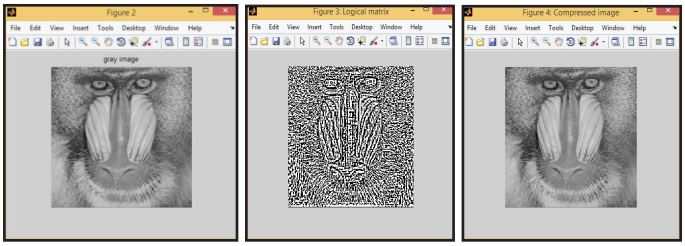
For the block size of 4×4, first the image (m×n) is divided in a block of size 4×4 then if there is any residue it is adjusted to fit in this size. The algorithm is applied on this image and the image is compressed. The obtained image having comparable resolution than the original image but the image size is reduced significantly. This algorithm is applied for different images and result has been analyzed. Five different Image of size 255×255 has been taken

and converted to the Gray Scale Image, by applying the proposed algorithm image has been compressed.



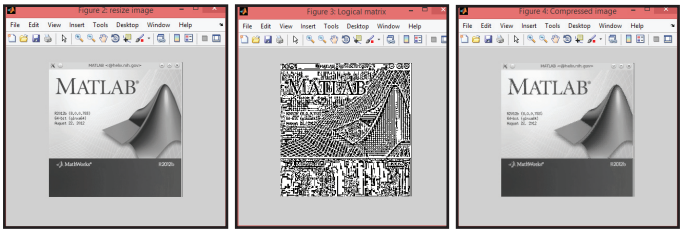
(a) (b) (c)

Fig. 3: Lena Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



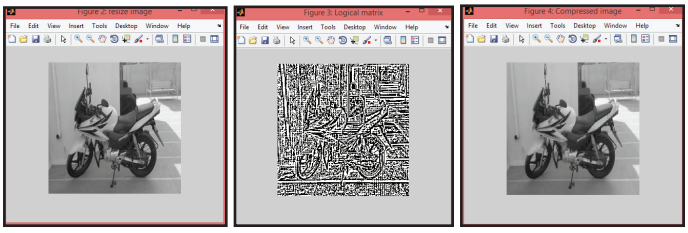
(a) (b) (c)

Fig. 4: Baboon Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



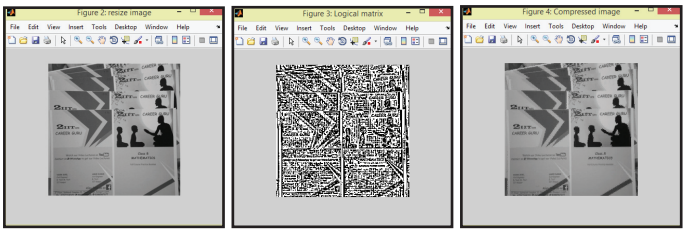
(a) (b) (c)

Fig. 5: Matlab Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



(a) (b) (c)

Fig. 6: Bike Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



(a) (b) (c)

Fig. 7: Books Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image

Result Analysis for 4×4 Block Size

Table 1: Performance in Terms of MSE, PSNR, Entropy & 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
Matlab Image	7.2925	7.2752	10.0519	87.7476	0.9983
Bike Image	7.3514	7.3210	12.5983	85.4110	0.9962
Books Image	6.9831	6.9563	13.7426	79.4591	0.9956

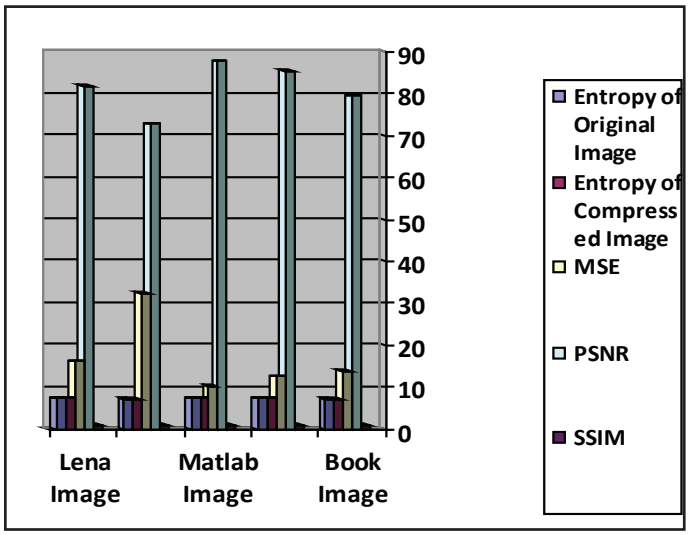
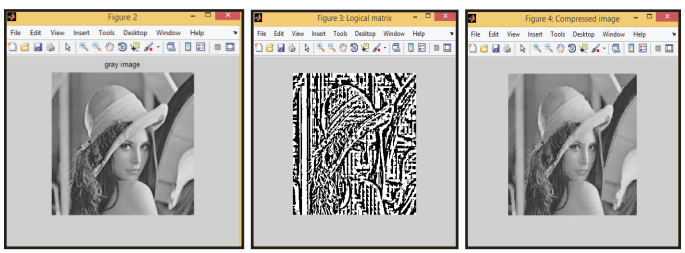


Fig. 8: Graph for MSE, PSNR, ENTROPY & SSIM for different image of 4 ×4 block size

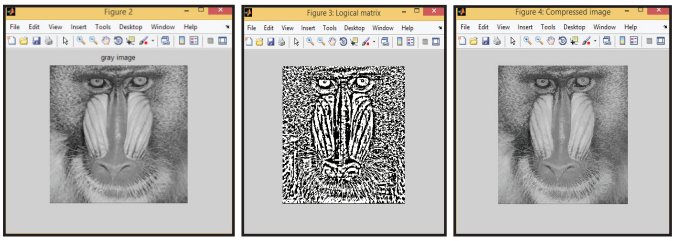
- **Simulation of Block Truncation Coding With Region based Segmentation for 8×8 block size**

Same approach as discussed for Block size 4×4, is adopted for block size 8×8. It is worth noting aspect to check the image dimension is fitting the block size or not. The PSNR, MSE, SSIM & Entropy is calculated based on the result. The result obtained in block size 8×8 is poor performance in terms of resolution but far better in terms of size. It is suggested that when image quality is comprisable this compression technique provides good performance in terms of storage capability. It is suggested to keep the block size of image less because increasing the number of order of block size will result poorer resolution of the image but compression ratio will be far better if we are increasing the number of block size.



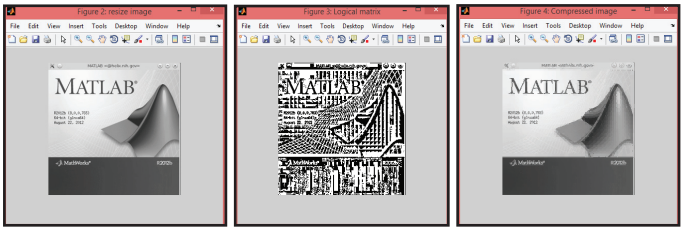
(a) (b) (c)

Fig. 9: Lena Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



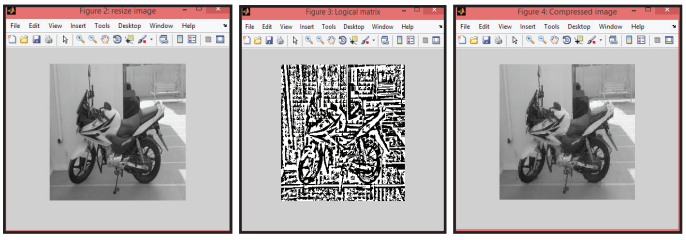
(a) (b) (c)

Fig. 10: Baboon Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



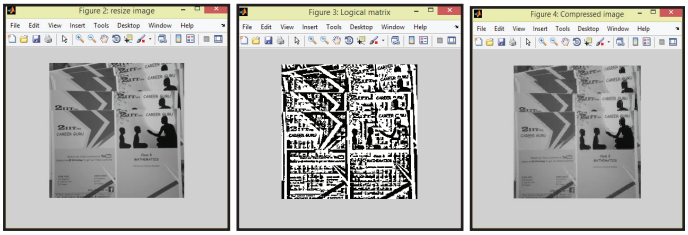
(a) (b) (c)

Fig. 11: Matlab Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



(a) (b) (c)

Fig. 12: Bike Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



(a) (b) (c)

Fig. 13: Books Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image

Result Analysis For 8×8 Block Size

Table 2: Performance in Terms of MSE, PSNR, ENTROPY & 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
Matlab Image	7.2925	7.2322	15.6778	83.3028	0.9959
Bike Image	7.3514	7.2581	19.8806	80.8493	0.9929
Books Image	6.9831	6.9096	20.0289	75.6923	0.9923

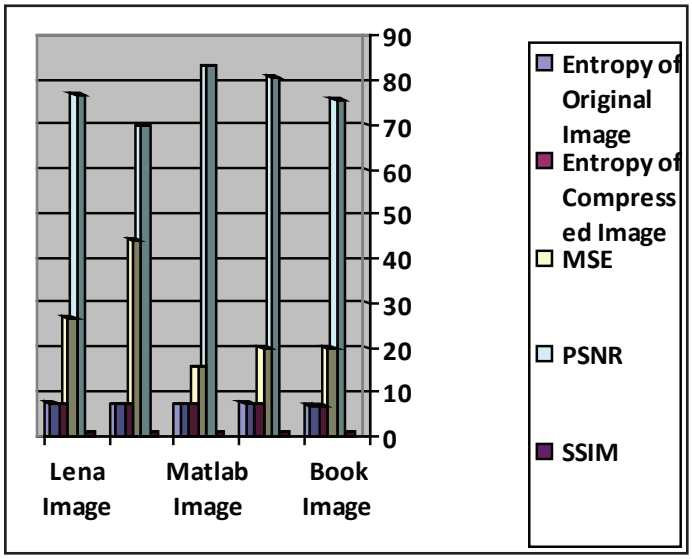
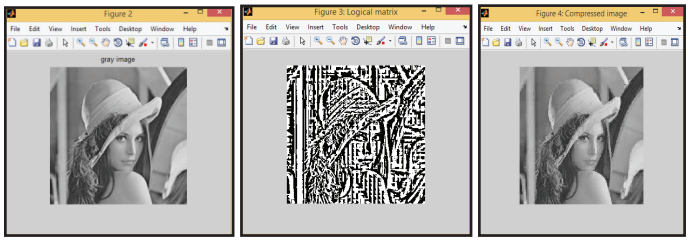


Fig. 14: Graph for MSE, PSNR, ENTROPY & SSIM for Different Image of 8 ×8 Block Size

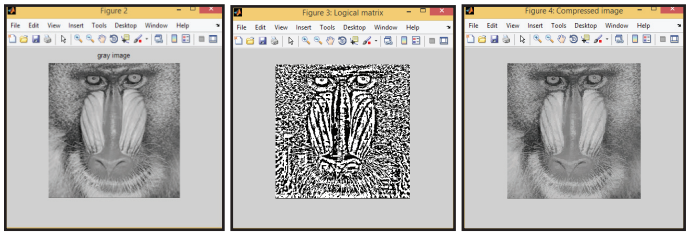
- **Simulation of Block truncation coding with Region based segmentation for 16×16 block size**

Methodology used to implement the BTC-PF Algorithm for block size 16×16 will be same as above discussed. This block size division criterion gives poor result than 4×4 o4 8×8 block size in terms of resolution but in terms of compression, compressed image size is smaller than 4×4 o4 8×8 block size.



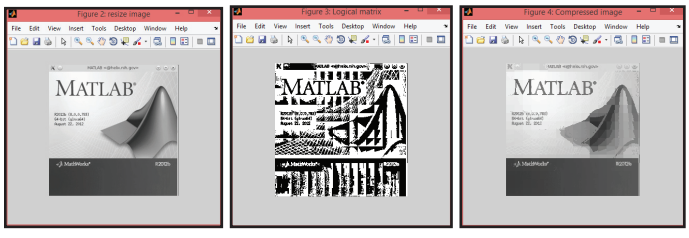
(a) (b) (c)

Fig. 15: Lena Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



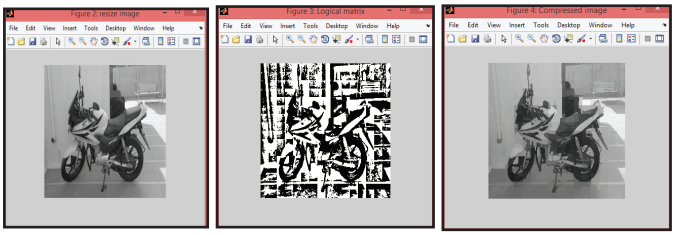
(a) (b) (c)

Fig. 16: Baboon Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



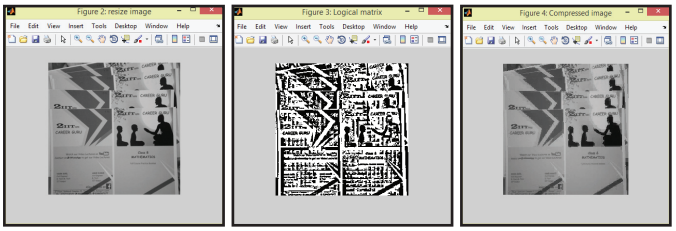
(a) (b) (c)

Fig. 17: Matlab Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



(a) (b) (c)

Fig. 18: Bike Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image



(a) (b) (c)

Fig. 19: Books Input Image (a) Gray Scale Image (b) Logical Matrix Image (c) Compressed Image

Result Analysis For 16×16 Block Size

Table 3: Performance in terms of MSE, PSNR, ENTROPY & SSIM for different image

Image	Entropy of Original Image	Entropy of Compressed Image	MSE	PSNR	SSIM
Lena Image	7.4543	7.0336	26.5617	76.8181	0.9933
Baboon Image	7.2193	6.8710	44.0471	69.8373	0.9925
Matlab Image	7.2925	7.0496	20.2815	80.7282	0.9947
Bike Image	7.3514	6.9859	29.3751	76.9452	0.9883
Books Image	6.9831	6.7030	20.0289	75.6923	0.9923

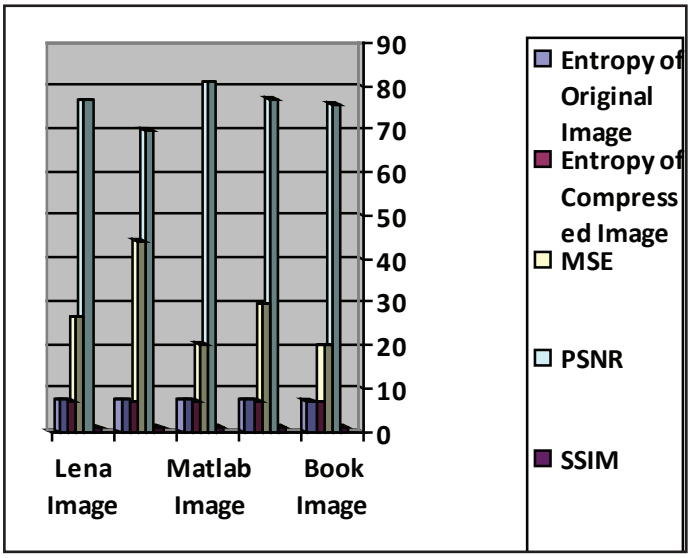


Fig. 20: Graph for MSE, PSNR, ENTROPY & SSIM for different image of 16×16 block size

- **Comparative analysis of Proposed method with previous method from Literature Survey**

Proposed methodology is implemented using MATLAB simulator and their results are compared with the work in literature and following worth noting points are deduced:

Jayamol Mathews et al [10] in their work found PSNR for 512×512 image size as 33.62. For block size 8×8 and 16×16, Dr. Ghadah AI-Khafaji [11] found PSNR as 33.44, 32.21 & SSIM as 0.9491 & 0.9147. For block size 4×4, 8×8 and 16×16 Jing-Ming Guo et al. [12] found PSNR as 26.18, 23.26, 21.24. For block size 4×4 and8×8 Ki-Won Oh et al. [13] found MSE as 344 & 294.

Where as the proposed methodology shows better results than the previous work which can be seen from the above simulation result

IV. Conclusion

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

References

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

[2] W. Pratt,"Digital Image Processing", John Wiley and Sons, New York, 1978.

[3] E. J. Delp, O. R. Mitchell,"Image compression using block truncation coding," IEEE Trans. Commun., Vol. 27, No. 9, pp. 1335–1342, Sep. 1979.

[4] I. H. Witten, M. N. Radford, J. G. Cleary,"Arithmetic coding for data compression", Communications of ACM, 60, pp. 520–540, 1987.

[5] A. Said, W. A. Pearlman,"A new, fast, and efficient image codec based on set partitioning in hierarchical trees", IEEE Transactions on Circuits and System for Video Techniques, 6, pp. 243– 250, 1996.

[6] Y. Q. Shi, H. Sun,"Image and video compression for multimedia engineering: Fundamentals, algorithms, and standards", CRC Press, USA, 1999.

[7] T. Ebrahimi, C. A. Christopoulos, D. T. Lee,"Special issue introduction, jpeg 2000", Signal Processing: Image Communication, 17:1–144, 2002.

[8] P. Li, J. P. Allebach,"Tone-dependent error diffusion," IEEE Trans. Image Process., Vol. 13, No. 2, pp. 201–215, Feb. 2004.

[9] R. Leonardi, J. R. Ohm,"Wavelet video coding - An overview (iso/iec jtc1/sc29/wg11 w7824). 2006.

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

[11] Dr. Ghadah AI-Khafaji,"The First International Conference of Electrical, Communication, Computer, Power and Control Engineering ICECCPCE'13", December 17-18, 2013

[12] Jing-Ming Guo, Yun-Fu Liu,"Improved Block Truncation Coding Using Optimized Dot Diffusion", IEEE Transactions

on image processing, Vol. 23, No. 3, March 2014.
[13] Ki-Won Oh, Kang-Sun Choi,"Parallel Implementation of Hybrid Vector Quantizer based Block Truncation Coding for Mobile Display Stream Compression, IEEE ISCE 2014.
[14] Seddeq E. Ghrare, Ahmed R. Khobaiz,"Digital Image Compression using Block Truncation Coding and Walsh Hadamard Transform Hybrid Technique", IEEE, 2014.