RESEARCH PAPER

RESEARCH HUB International Multidisciplinary Research Journal
e-ISSN: 2349-7637 | Vol.09 | Iss.01 | January 2022 | pp. 1-9
Double-Blind Peer Reviewed/Refereed Journal

Enhanced Encoding Technique for Lossless Image Compression

*Dr. Jigar Patel

Director, Kalol Institute of Management, Kalol, Gujarat (India)

Abstract

This paper proposes new encoding technique for lossless image compression. Here the pixel color information is stored in the file using efficient and sophisticated encoding technique which will be further compressed by existing compression algorithms and coding techniques. The final file with proposed extension generated by this scheme gives the better compression ratios than the existing image formats. The file is further decoded and image is recovered without any loss of information. The paper also discusses results and comparative analysis using popular image set compression ratio comparison of proposed scheme with existing image formats. That shows proposed encoding scheme will helps to save storage space where the lossy compression techniques are not recommended.

Keywords: *Compression, Decoding, Encoding Technique, Image Compression, Lossless*

Article Publication

Published Online: 12-Jan-2022

***Author's Correspondence**

Dr. Jigar Patel

Director, Kalol Institute of Management, Kalol, Gujarat (India)

drjigarvpatel[at]gmail.com

10.53573/rhimrj.2022.v09i01.001

Introduction

Nowadays image capturing and storing is quite common due to large availability of digital camera which can capture quality color images with high resolution. On the other hand, storage requirement to store captured images is also increasing every day. Image compression is playing important role to save storage space of our devices. Image compression is the process of reducing the amount of storage required to represent a digital image. It is a technique to build a compact representation of an image in order to reduce the image storage or transmission requirements. Image compression can be achieved normally by the one or more from Coding Redundancy, Interpixel Redundancy and Psychovisual Redundancy. In the Coding redundancy optimal code words are used. Interpixel redundancy results from correlations between the pixels of an image. Psychovisual redundancy is due to data that is ignored by the human visual system.

The techniques used in this paper are first two types of redundancies known as interpixel redundancy and coding redundancy. In the proposed technique each pixel color value is compared with the previous pixel color value and the color difference is stored in the proposed file format instead of pixel color value. In most of the cases the color difference is smaller between neighborhood pixels. To make optimal code new coding technique use variable length code by which pixel information is stored using one, two, three and four byte coding format as per requirement. Here coding is defined in such a manner that for the small color difference only one byte code is required instead of three bytes to store the image pixel color information. The intermediate file generated by the proposed encoding scheme has the sequence of bytes with large redundancy since all bytes value is between 0-255. The intermediate file of byte sequence is further compressed by existing compression techniques like 7z which internally uses

the Lempel-Ziv Markov-chain Algorithm. This algorithm is doing a very effective and relatively fast compression to intermediate file and finally generate highly compressed file in proposed format as shown in Fig. 1.

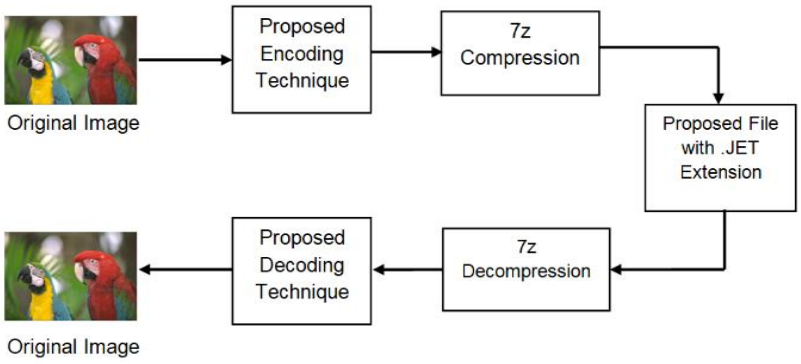

Fig. 1. Image Compression Process

The paper is divided in five sections. After section 1 of introduction section 2 discuss the related work on the digital image processing and common image formats. Proposed encoding technique is explained in the section 3 including algorithms for encoding and decoding of proposed scheme. Section 4 discusses the result and comparison of proposed format with the existing well-known formats of lossless image compression. Section 5 contains the conclusions of the paper.

Related Works

In most of the natural images, the values of the neighboring pixels are strongly correlated, in the other word the value of any given pixel can be reasonably predicted from the values of its neighbors [2][3][13]. Image compression techniques reduce the number of bits required to represent an image by taking advantage of these redundancies [1]. There are many image formats to store the images in lossless manner but among all GIF, PNG and TIFF are quite popular.

GIF Format

The Graphics Interchange Format (GIF) is a lossless 8 bit-per-pixel bitmap image format that was introduced by CompuServe in 1987 [7]. GIF images is using the Lempel-Ziv-Welch (LZW) coding [10], [11] for the compression, a dictionary-based technique to exploit redundancy. The initial size of dictionary is 29, while this fills up, the dictionary size is doubled, until the maximum dictionary size of 4096 is reached. Afterwards the compression algorithm behaves like a static dictionary algorithm [8]. GIF is widely used for lossless compression of both natural and synthetic images.

PNG Format

PNG is a file format for lossless image compression and storage. Portable Network Graphics (PNG) is a W3C recommendation for coding of still images which has been patent free replacement for GIF. It is based on a predictive scheme and entropy coding. The prediction is done on the three nearest causal neighbors and there are five predictors that can be selected on a line-by-line basis. The entropy coding uses the Deflate [12] algorithm of the popular Zip file compression utility, which is based on LZ77 [9] coupled with Huffman coding. PNG is capable of lossless compression only and supports paletted color and true color, gray scale an optional alpha plane, interlacing and other features. For most images, PNG can achieve greater compression than GIF.

TIFF format

TIFF or TIF is Tagged Image File Format use for storing raster graphics images. It is one of the popular image formats among the graphics artist, publishing industry and the photographers. In the latest release of the TIFF is support for the palette color images and LZW compression. TIFF is flexible, adaptable file format for handling image and data within a single file. The ability to store image data in a lossless format makes a TIFF file a useful

image archive, because, unlike standard JPEG files, a TIFF file using lossless compression may be edited and re-saved without losing image quality [4].

The algorithm like CALIC [14], [15], JPEG-LS [5], and JPEG2000 [6] are very popular lossless image compression but it takes more processing for prediction of next pixel. CALIC provides best compression ratios over typical images, whereas, JPEG-LS is a low complexity alternative with competitive efficiency. The JPEG2000 standard, on the other hand, is a wavelet-based method, which provides a unified approach for lossy-to-lossless compression.

Proposed Encoding and Decoding Technique

In this proposed scheme, I have taken the true color image with 24-bit color depth. The types of images are popular lossless image formats like PNG, TIFF or BMP. Afterward an input image is converted in proposed format. The input image is scanned pixel by pixel as shown in the Fig. 2.

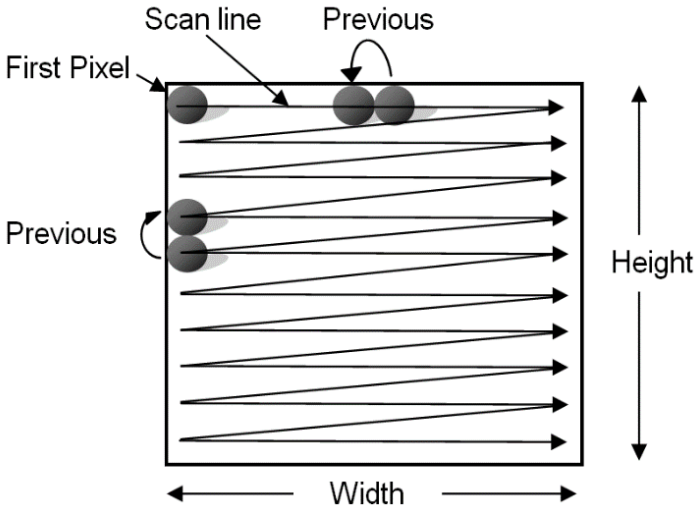

Fig. 2. Scanning Pattern of image

Each pixel Red, Green and Blue color values have been extracted and converted into the binary form. The proposed binary form is in multiple of 8-bit so we can easily convert each 8-bit chunk into byte which is store in temporary encoded file. The encoded file is containing the bytes which values are ranging from 0 to 255. Later I have applied 7z compression on this temporary encoded file and generating final file with proposed JET (Java based Encoding Technique) format to store image file which is in better compressed form than existing formats like PNG or TIFF.

Mathematically we can see that we have two function f and g as follows.

$$f: N \rightarrow b$$

Where N = numbers representing the color difference value between -255 to 255 of Red, Green or Blue component.

b =bit code contains 0 and 1.

and

$$g: b \rightarrow B$$

Where b = bit code contains 0 or 1.

B = Equivalent byte with value between 0-255 which store in the encoded file.

Here to convert RGB component value into byte forms via applying composite function gof which is reversible with function f-1og-1. Here in the proposed encoding algorithm, we are fetching the image width and height first and it is converted into the binary form with length of 2 byte each. So, first four bytes in the encoded file are reserved

to store image dimension. The first two bytes store the width of image and successive two bytes store the height of the image. After that we start to fetch the color information from the first pixel with coordinate (0, 0) and get the difference of each Red, Green, and Blue component. By adding difference with the previous pixel, we get pixel original value of Red, Green and Blue component. The previous pixel color is according to following formula.
For pixel (0, 0) the previous pixel Red=Green=Blue=0,
For any pixel (0, j) the previous pixel is (0, j-1),
For any other pixel (i, j) the previous pixel is (i-1, j).

Here in the encoding process every current pixel (CURRENT) Red, Green and Blue values are compared with the previous pixel (PREVIOUS) Red, Green and Blue values and final RED_Difference, GREEN_Difference and BLUE_Difference will be calculated by following equations.

$$\text{RED\_Difference} = \text{CURRENT.Red} - \text{PREVIOUS.Red}$$
$$\text{GREEN\_Difference} = \text{CURRENT.Green} - \text{PREVIOUS.Green}$$
$$\text{BLUE\_Difference} = \text{CURRENT.Blue} - \text{PREVIOUS.Blue}$$

This difference values are converted into binary form and store different bit pattern showing in the format 1,2,3,4 as shown in Fig. 3.

In the format 1, I am taking one byte to store the RGB differences in which first two bits set 00. Remaining six bits are divided to store three color differences that each color difference takes two bits to store the value. As I am taking two bits to store the value I can store four values [10, 11, 00, 01] which maps the value [-2, -1, 0, 1] respectively. The format is shown in the Fig. 3(a).

In the format 2, I am taking two bytes to store the RGB differences in which first bit is set 1. Remaining 15 bits are divided in the three chunks of 5 bits each to store the RGB color differences. In the chunk of 5 bits first bit is reserved for sign and remaining four bits are used to store the value. Therefore, in this format I can store the RGB color difference value between -15 to 15. The format is shown in Fig. 3(b).

In the format 3, I am taking three bytes to store the RGB differences in which first three bits are set 010. Remaining 21 bits are divided in the three chunks of 7 bits each to store the RGB color differences. In the chunk of 7 bits first bit is reserved for sign and remaining six bits are used to store the value. Therefore, in this format I can store the RGB color difference value between -63 to 63. The format is shown in Fig. 3(c).

In the format 4, I am taking four bytes to store the RGB differences in which first five bits are set 01100. Remaining 27 bits are divided in the three chunks of 9 bits each to store the RGB color differences. In the chunk of 9 bits first bit is reserved for sign and remaining eight bits are used to store the value. Therefore, in this format we can store the RGB color difference value between -255 to 255. The format is shown in Fig. 3(d).

After binary coding using four different types of formats every group of 8-bit called byte is stored in binary file in sequence. This encoded file containing byte values ranging from 0-255. Since images contain huge amount of data to store color information our encoded file contains large number of redundant bytes which is further compressed by 7z compression. In result of that we get highly compressed file with proposed .JET extension.

Here in Table I, few samples of current and previous pixel RGB value differences are taken in the comma separated list as triplets. The binary generated by the encoding techniques is also shown in the column where its fix value, RED Difference, GREEN Difference and BLUE Difference are separated by the space. Since the binary coding is in multiple of 8-bit, final value of one, two, three and four bytes equivalent decimal value is shown in the last column.

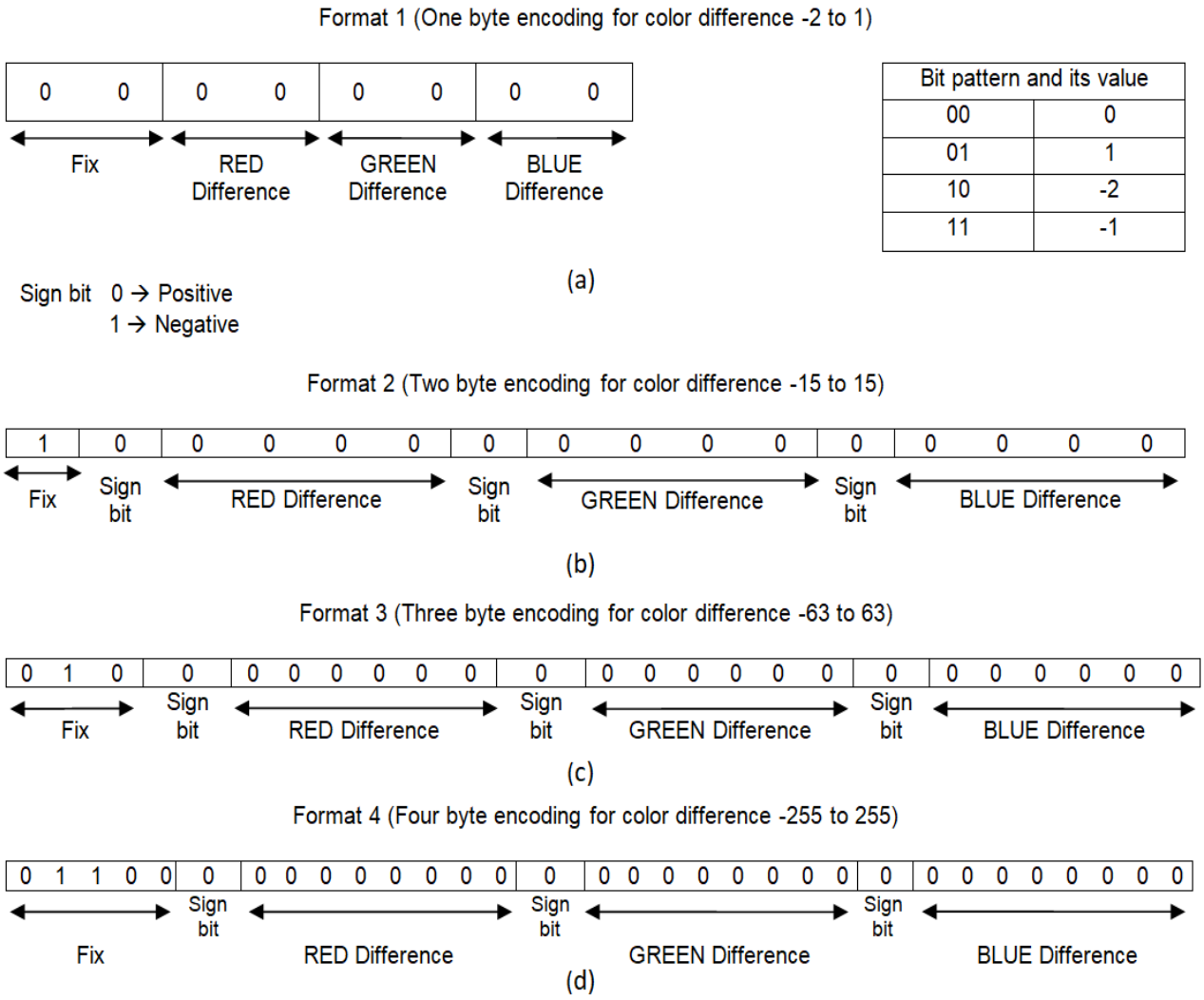

Fig. 3. Image Encoding Formats

TABLE I
Examples Of Encoded Values

RED, GREEN and BLUE Difference	Applicable Format	Binary Value according to Format 1, 2, 3, 4	Binary Values in Group of 8-bits	Decimal Value of Equivalent Byte(s)
(0,0,0)	Format 1	00 00 00 00	00000000	0
(-2,0,1)	Format 1	00 10 00 01	00100001	33
(-1,-1,-1)	Format 1	00 11 11 11	00111111	63
(1,-1,0)	Format 1	00 01 11 00	00011100	28
(-10,6,9)	Format 2	1 11010 00110 01001	11101000 11001001	232,201
(8,0,-15)	Format 2	1 01000 00000 11111	10100000 00011111	160,31
(-15,15,15)	Format 2	1 11111 01111 01111	11111101 11101111	253,239
(7,-7,-1)	Format 2	1 00111 10111 10001	10011110 11110001	158,241
(32,34,40)	Format 3	010 0100000 0100010 0101000	01001000 00010001 00101000	72,17,40
(-63,15,63)	Format 3	010 1111111 0001111 0111111	01011111 11000111 10111111	95,199,191
(-50,-50,-50)	Format 3	010 1110010 1110010 1110010	01011100 10111001 01110010	92,185,114
(50,50,50)	Format 3	010 0110010 0110010 0110010	01001100 10011001 00110010	76,153,50
(64,200,-100)	Format 4	01100 001000000 011001000 101100100	01100001 00000001 10010001 01100100	97,1,145,100
(-255,-255,-255)	Format 4	01100 111111111 111111111 111111111	01100111 11111111 11111111 11111111	103,255,255,255
(200,-10,192)	Format 4	01100 011001000 100001010 011000000	01100011 00100010 00010100 11000000	99,34,20,192
(200,-10,-192)	Format 4	01100 011001000 100001010 111000000	01100011 00100010 00010101 11000000	99,34,21,192

Encoding Algorithm:

1. Take any PNG image for compression.
2. Create the temporary intermediate file to store the sequence of bytes.
3. Store the image width in the file in first two bytes.
4. Store the image height in the file in next two bytes.
5. Scan the image pixels line by line and compare the Red, Green and Blue value with previous pixel.
   5.1. If all color difference values are between -2 to 1
      5.1.1. Write the value in one byte form as per the format 1.
   5.2. Else if any color difference values are in range -15 to 15
      5.2.1. Write the value in two byte form as per the format 2.
   5.3. Else if any color difference values are in range -63 to 63
      5.3.1. Write the value in three byte form as per the format 3.
   5.4. Else
      5.4.1. Write the value in four byte form as per the format 4.
6. Apply the 7z compression on the file.
7. Store resultant compressed file with .JET extension.

Decoding Algorithm:

1. Get the compressed .JET file as input.
2. Apply the 7z decompression and get the intermediate temporary file as sequence of bytes.
3. Get the first two bytes which gives the image width.
4. Get the next two bytes which gives the image height.
5. Read all the bytes from the file until end of file.
   5.1. If byte values is less than 64 then
      5.1.1. Get the color differences as per format 1.
   5.2. Else if byte value is greater 127 then
      5.2.1. Read one more byte from the file and combine two bytes.
      5.2.2. Get the color differences as per format 2.
   5.3. Else if byte value is greater than 63 and less than 96 then
      5.3.1. Read two more bytes from the file and combine three bytes.
      5.3.2. Get the color differences as per format 3.
   5.4. Else
      5.4.1. Read three more bytes from the file and combine four bytes.
      5.4.2. Get the color differences as per format 4.
   5.5. Get the Red, Green and Blue values pixel by pixel by adding color differences with previous pixel color values for each scan line in image buffer.
6. Generate original PNG image back from the image buffer.

Result and Discussion

Here, I have discussed the result of proposed scheme with existing formats like PNG and TIFF. The popular image data set is selected for my experiment as shown in Fig. 4. All images are containing depth of 24 bits with 8 bit per channel. The proposed encoding algorithm is implemented in JAVA programming language and proposed file has been generated for all the sample images to evaluate the result.

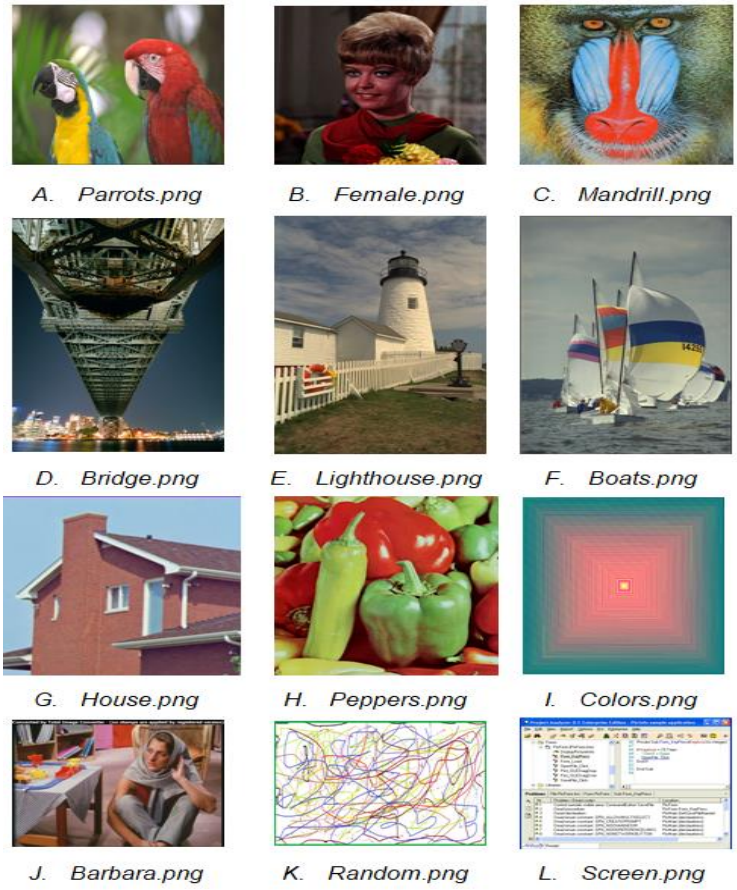

Fig. 4. Sample image set for the experiment.

After generating all the .JET files for the images the Compression Ratio (CR) is calculated by following formula.

CR = Original Image Size / Compressed Image Size

Table II shows the details of image like resolution, original size and Compression Ratios of TIFF, PNG and Proposed JET formats respectively. Here I have observed the proposed algorithm work well not only natural images and photographs but also shows better result in images like screen shots of computer screen or any random line drawing generated by any painting software. This algorithm shows high compression ratio for input images like Random.png or Screen.png because the redundancy of white pixels in the images. Finally average compression ratios are calculated for the sample images and it is 6.15 for proposed JET format which is higher in comparison of 2.57 of TIFF and 5.51 of PNG images. Therefore, the JET file compression ratio is more than double as compare to TIFF and more than ten percent higher as compare to PNG format. Fig. 5 gives the line graph of first ten images from the image set and it shows proposed JET format is at the top in most cases.

TABLE II
Comparison of Proposed encoding

Image Ref from Fig.	Resolution	Size of Original Image (in KB)	Compression Ratio		
			TIFF	PNG	JET (Proposed)
A. Parrots	768x512	1152	1.75	2.12	2.31
B. Female	256x256	192	1.27	1.28	1.55
C. Mandrill	512x512	768	0.92	1.24	1.18
D. Bridge	2749x4049	32609	1.24	1.15	1.54
E. Lighthouse	512x768	1152	1.52	1.76	2.00
F. Boats	512x768	1152	1.76	2.02	2.40
G. House	256x256	192	1.39	1.29	1.71

H. Peppers	512x512	768	1.58	1.77	3.25
I. Colors	4096x4096	49152	1.19	2.44	3.06
J. Barbara	720x576	1215	1.36	1.18	1.83
K. Random	1000x500	1465	3.96	8.23	8.72
L. Screen	540x448	709	12.8	41.6	44.3
	Average		2.57	5.51	6.15

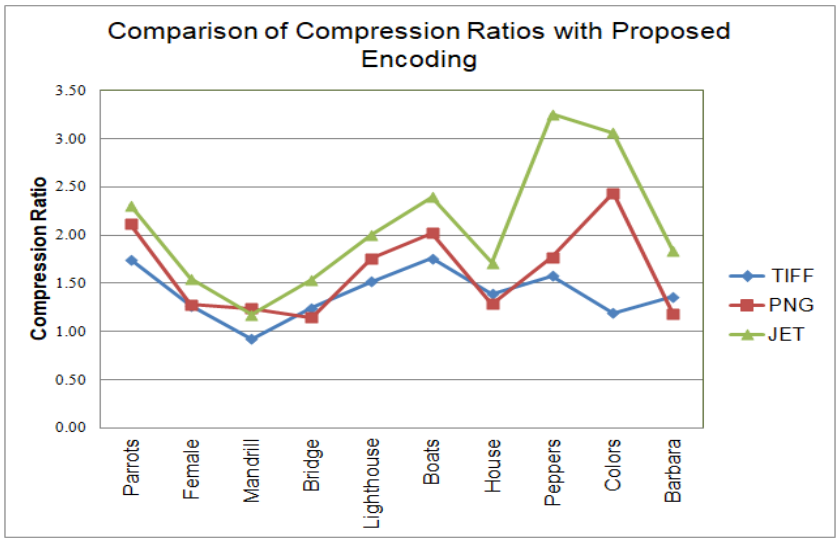

Fig. 5. Comparison of Compression Ratios with .JET

Conclusion

The proposed encoding scheme compress slightly better than existing image format like PNG and TIFF. The pixel information stored here is in highly optimized manner so that it consumes a smaller number of bytes. Further the intermediate file has lots of redundant bytes that are better compressed with 7z compression. After this two-step process proposed file with .JET extension gives the higher compression ratios than the PNG and TIFF. The proposed encoding scheme has shown even better compression ratios in the computer screenshots and random line drawing where normally numbers of white pixels are repeated.

References

A. Subramanya, "Image Compression Technique," Potentials IEEE, Vol. 20, Issue 1, Feb-March 2001, pp 19-23.

A. Vitali, A. Borneo, M. Fumagalli and R. Rinaldo, "Video over IP using standard-compatible multiple description coding," Journal of Zhejiang University- Science A, vol. 7, no. 5 ,2006, pp. 668676.

C. Ratael, gonzales, e. Richard, and woods, "Digital image processing," 2nd edition, Prentice Hall, 2002

https://en.wikipedia.org/wiki/TIFF#cite_note-tiff6-7, Accessed on 10 Feb. 2020.

ISO/IEC 14495-1, Lossless and near-lossless compression of continuous-tone still images—baseline, 2000.

ISO/IEC 15444-1, Information technology-JPEG 2000 image coding system-part 1: Core coding system, 2000.

J.M. Gilbert and R.W. Brodersen, "A lossless 2-d image compression technique for synthetic discrete-tone images," Data Compression Conference, 1998. DCC '98. Proceedings, 1998.

J. Miano, "Compressed image file formats: JPEG, PNG, GIF, XBM, BMP," ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 1999.

J. Ziv and A. Lempel. "A universal algorithm for sequential data compression." IEEE Transactions on Information Theory, Vol.23, Issue 3, 1977, pp. 337–343.

J. Ziv and A. Lempel. "Compression of individual sequences via variable-rate coding". IEEE Transactions on Information Theory, IT24(5):530–536, September 1978.

J. Ziv and A. Lempel. "Compression of two-dimensional data". IEEE Trans. on Information Theory, Vol. 32, Issue 1, 1986.

RFC. Deflate compressed data format specification version 1.3. http://tools.ietf.org/html/rfc1951.

S. P. Nana'vati., P. K. panigrahi. "Wavelets: applications to image compression- I,". joined of the scientific and engineering computing, vol. 9, Issue 3, 2004, pp. 4- 10.

X. Wu, "Lossless compression of continuous-tone images via context selection, quantization, and modelling," IEEE Transaction of Image Processing. Vol.6, Issue 5, 1997, pp. 656–664.

X. Wu, N. Memon, "Context-based, adaptive, lossless image codec," IEEE Trans. Comm., Vol. 45, Issue 4, 1997, pp. 437–444.