Comparative Analysis of Lossless Image Compression Algorithms

[1] Susmitha Alamuru,
, [2] Ishani Mishra, [3]Divya Sharma
[1]Sr. Assistant Professor, [2] Sr. Assistant Professor, [3] Sr. Assistant Professor
[1]Department of Electronics & Communication Engineering,
[1]New Horizon College Of Engineering, Bangalore

Abstract : Lossless compression techniques are developed in order to store data with less memory.. This paper provides the performance analysis of lossless compression techniques on the basis of compression ratio. It describes the basic lossless techniques as Huffman encoding, run length encoding, arithmetic encoding and Lempel-Ziv-Welch encoding, Lossless predictive coding , Predictive coding followed by Integer Wavelet Transform and Bit plane coding briefly with their effectiveness under varying parameters using MATLAB .

***IndexTerms* - Lossless Compression, Huffman encoding, run length encoding, arithmetic encoding and Lempel-Ziv-Welch encoding, Lossless predictive coding , Integer Wavelet Transform and Bitplane coding.**

I. INTRODUCTION

Images generally occupy more space in a hard disk and to transmit the images more bandwidth would be required. So, we need to compress the image without losing the original information in it . Efficiency, in this context, means to have a representation of the original image , but which doesn’t occupy a lot of space. The aim of image compression is to represent an image with the least amount of bits possible. There are “lossless” and “lossy” methods in image compression . In lossy compression, the image quality may be degraded in order to meet a given target data rate for storage and transmission. The applications for lossy coding are the transmission of the image and video through a band limited network and efficient storage. In lossy compression ,the reduction of degradation of the image quality depends on the given the data rate. In lossy compression, most algorithms transform pixels into the transform domain using the DCT (Discrete Cosine Transform) or the DWT (Discrete Wavelet Transform). The source of the loss is either quantization of the transform coefficients or the termination of the encoding at a given data rate. In order to meet a data rate budget, the transform coefficients are explicitly quantized with a given step size as in ZTE , JPEG , H.263 and MPEG-2 . Implicit quantization is used in algorithms such as EZW and SPIHT , which can be truncated at any point in the bitstream during encoding.

In the lossless compression, the image after decompression is identical to that of the original. The issue in lossless coding is how much we can reduce the data rate. The main approach for lossless image compression is predictive coding or entropy encoding. For predictive coding, DPCM (Differential Pulse Code Modulation) is often used. Linear predictors are used where the prediction is obtained by the linear combination of previously decoded neighbours. For entropy coding, Run-length coding , Huffman coding , or arithmetic coding is used. Context modelling can be included in the entropy encoding, which is to estimate the probability distribution function of the symbols conditioned on the context, so as to increase the compression performance. The context consists of a combination of neighbouring pixels already encountered. The structure of predictive coding is determined by the number of neighbouring pixels used for the prediction, weighting for the linear combination of the neighbouring pixels and the method of context modelling. The JPEG lossless mode uses DPCM and Huffman coding or arithmetic coding. FELICS (Fast and Efficient Lossless Image Compression) developed by P. Howard is a simple and fast lossless encoder. CALIC (Context-based, Adaptive, Lossless Image Coding) by X. Wu and JPEG-LS are other examples of lossless compression algorithms using predictive coding. Another method for lossless encoding is to use the reversible integer-to-integer wavelet transform. The results of the integer wavelet transform are integers, so as to recover the originals completely with the inverse integer wavelet transform. The lifting scheme (LS) is used to implement the integer wavelet transform.

II LOSSLESS COMPRESSION CODING TECHNIQUES

Lossless compression allows extracting the original image to be perfectly reconstructed from the compressed image. Compression is a two step process, the first step creates a statistical model and the second step uses this model. The mapping of input data into bit sequences is done in such a way that the frequently encountered data will produce shorter output than less frequent data. There are two ways in constructing a statistical model one is called the static model and the other is the adaptive model. The simplest one among the two is the static model where in the data is analyzed and a model is constructed, then this model is stored with the compressed data. Even though this model is simple and modular it has some disadvantages. It is an expensive model for storage and forces using the same model for all the data compressed. It also performs poorly on files containing heterogeneous data. Adaptive models are the most popular type of models in practice. These models are dynamically updated as the data is compressed. Both the encoder and decoder begin with a trivial model yielding poor compression initially, but as they learn more about the data, performance is improved. Lossless compression methods may be categorized according to the type of data they are designed to compress. While, in principle any general purpose lossless compression algorithm (general-purpose meaning that they can compress any bit string) can be used on any type of data, but many are unable to achieve significant compression on data that are not of the form for which they were designed to compress.

Lossless compression is preferred for artificial images such as technical drawings, icons or comics. This is because lossy compression methods when used especially at low bit rates, introduce compression artifacts. It can also be used for high value content, such as medical imagery or image scans in health industry where there is archiving of large number of images. Lossless compression increases the efficiency of sharing and viewing personal images, uses less storage space and is quicker in transmission and reception of images. It is also used in the chain of retail stores where the introduction of new products or the removal of discontinued items can be much more easily completed when all employees receive, view and process images in the same way. In federal government agency where viewing, storing and transmitting processes can reduce large amounts of time spent in explaining and solving the problem or issue, lossless compression is used. It is widely used in the security industry where it can greatly increase the efficiency of recording, processing and storing images. In Museums, where accurate representation of museum images or gallery items is required, especially when uploading them to a website lossless image compression can be effectively useful.

A. Entropy Coding

Entropy measures the amount of information present in the data. After the data has been quantized into a finite set of values it can be encoded using an entropy coder to achieve additional compression using probabilities of occurrence of data. This technique reduces the statistical redundancy. The entropy coder encodes the given set of symbols with the minimum number of bits required to represent them. It is a variable length coding which means that it assigns different number of bits to different gray levels. If the probability of occurrence is more, then fewer bits/sample will be assigned.

Entropy (H): Suppose we have M input levels or symbols (S1, S2…SM) with their probabilities (P1, P2…., PM)

$$H = - \sum_{k=1}^{M} P_k \log_p P_k = \sum_{k=1}^{M} P_k \log_2 (\frac{1}{P_k})$$

The average number of bits per pixel is given by

$$R = \sum_{k=1}^{M} P_k N_k$$

where pk represents the probabilities of the symbols and Nk represent the number of bits per the code generated. Coding efficiency (ŋ) can also be calculated using H and R generated earlier

$$\eta = \frac{H}{R} * 100$$

B. Run-Length Coding

The neighbouring pixels in a typical image are highly correlated to each other. Often it is observed that the consecutive pixels in a smooth region of an image are identical or the variation among the neighbouring pixels is very small. Appearance of runs of identical values is particularly true for binary images where usually the image consists of runs of 0‟s or 1‟s. Even if the consecutive pixels in grayscale or colour images are not exactly identical but slowly varying, it can often be pre-processed and the consecutive processed pixel values become identical. If there is a long run of identical pixels, it is more economical to transmit the length of the run associated with the particular pixel value instead of encoding individual pixel values.

Run-length coding is a simple approach to source coding when there exists a long run of the same data, in a consecutive manner, in a data set. As an example, the data d = 5 5 5 5 5 5 5 19 19 19 19 19 19 19 19 19 19 19 19 0 0 0 0 0 0 0 0 23 23 23 23 23 23 contains long runs of 5‟s, 19‟s, 0‟s, 23‟s etc. Rather than coding each sample in the run individually, the data can be represented compactly by simply indicating the value of the sample and the length of its run when it appears. In this manner the data d can be run-length encoded as (5 7) (19 12) (0 8) (23 6). Here the first value represents the pixel, while the second indicates the length of its run.

On the other hand, binary (black and white) images, such as facsimile, usually consist of 0‟s and 1‟s. As an example, if a segment of a binary image is represented as d = 00000000011111111111000000000000000111000000000000010011111111111, it can be compactly represented as c(d) = (9, 11, 15, 3, 13, 1, 2, 10) by simply listing the lengths of alternate runs of 0‟s and 1‟s. While the original binary data d requires 65 bits for storage, its compact representation c(d) requires 32 bits only under the assumption that each length of run is being represented by 4 bits.

C. Huffman Coding

Huffman coding is based on optimum prefix codes. The more frequently occurring symbols can be allocated with shorter code words than the less frequently occurring symbols.The two least frequently occurring symbols will have code words of the same length, and they differ only in the least significant bit.Average length of these codes is close to entropy of the source. The steps involved in the algorithm are given below

i) Produce a set N = {N1, N2,..., Nm} of m nodes as leaves of a binary tree. Assign a node Ni with the source symbol si, i = 1, 2,..., m and label the node with the associated probability pi.

ii) Find the two nodes with the two lowest probability symbols from the current node set, and produce a new node as a parent of these two nodes.

iii) Label the probability of this new parent node as the sum of the probabilities of its two child nodes.

iv) Label the branch of one child node of the new parent node as 1 and the branch of the other child node as 0.

v) Update the node set by replacing the two child nodes with smallest probabilities by the newly generated parent node. If the number of nodes remaining in the node set is greater than 1, go to Step (ii)

vi) Transverse the generated binary tree from the root node to each leaf node Ni, i = 1, 2, ..., m, to produce the codewordof the corresponding symbol si, which is a concatenation of the binary labels (0 or 1) of the branches from the root to the leaf node.

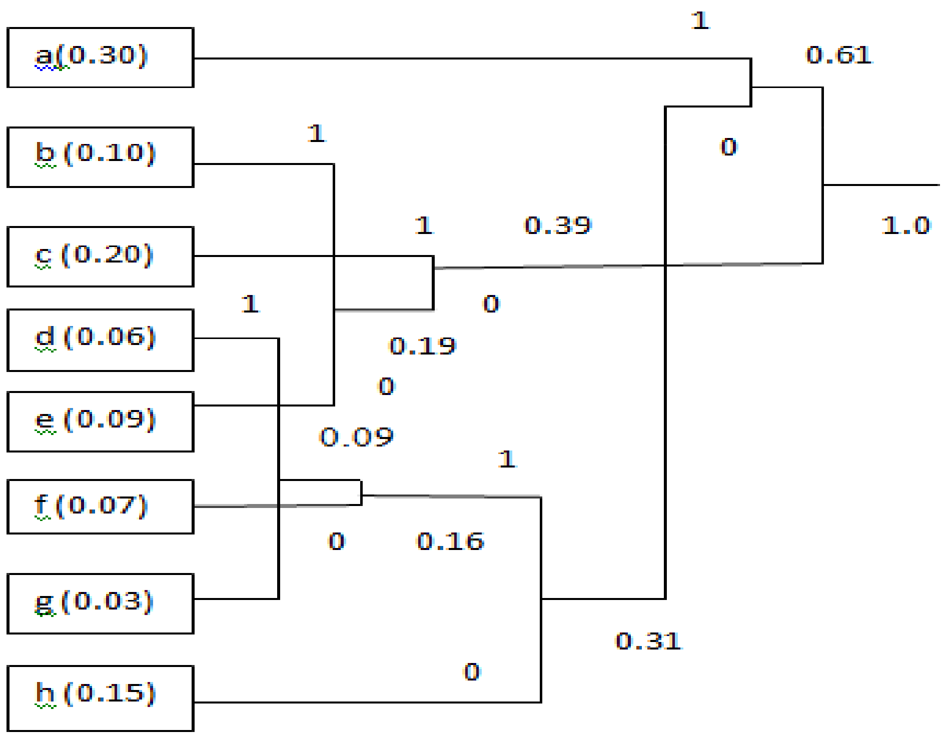


Fig: 2.1 Huffman tree Construction

Symbols	*Probability*	*Huffman code*
A	*0.30*	*1 0*
B	*0.10*	*0 0 1*
C	*0.20*	*0 1*
D	*0.06*	*1 1 1 1 1*
E	*0.09*	*0 0 0*
F	*0.07*	*1 1 1 0*
G	*0.03*	*1 1 1 1 0*
H	*0.15*	*1 1 0*

Table 2.1. Huffman Code Table

D. Arithmetic coding

Arithmetic coding is a variable-length source encoding technique. In traditional entropy encoding techniques such as Huffman coding, each input symbol in a message is substituted by a specific code specified by an integer number of bits. Arithmetic coding deviates from this paradigm. In arithmetic coding a sequence of input symbols is represented by an interval of real numbers between 0.0 and 1.0. The longer the message, the smaller the interval to represent the message becomes. More probable symbols reduce the interval less than the less probable symbols and hence add fewer bits in the encoded message. As a result, the coding result can reach to Shannon's entropy limit for a sufficiently large sequence of input symbols as long as the statistics are accurate. Arithmetic coding offers superior efficiency and more flexibility compared to the popular Huffman coding. It is particularly useful when dealing with sources with small alphabets such as binary alphabets and alphabets with highly skewed probabilities. Huffman coding cannot achieve any compression for a source of binary alphabets. As a result arithmetic coding is highly efficient for coding bi-level images. However, arithmetic coding is more complicated and is intrinsically less error resilient compared to the Huffman coding. The arithmetic coding requires significantly higher computation because of the requirement of multiplication to compute the intervals. However several multiplication-free arithmetic coding technique have been developed for binary compression.

E. Lempel ziv welch coding

Lempel-Ziv-Welch (LZW) is a category of a dictionary–based compression method . It maps a variable number of symbols to a fixed length code. LZW places longer and longer repeated entries into a dictionary. It emits the code for an element, rather than the string itself, if the element has already been placed in the dictionary. In a dictionary-based data compression technique, a symbol or a string of symbols generated from a source alphabet is represented by an index to a dictionary constructed from the source alphabet. In a dictionary based coding to build a dictionary that has frequently occurring symbols and string of symbols. When a new symbol or string is found and that is contained in the dictionary, it is encoded with an index to the dictionary or else, the new symbol is not in the dictionary, the symbol or strings of symbols are encoded in a less efficient manner.
It has two phases:

- Building an indexed dictionary
- Compressing a string of symbols

F. Lossless Predictive Coding Technique

The principle behind predictive coding seems obvious when it is explained, but nothing has appeared in current literature providing a statement as to which filtering techniques can be used for lossless data compression.
There is no particular condition in specific for invertible filters but broad statements can be made about sufficient conditions for lossless filtering. The condition as given in [21] says "The digital implementation of a filter is lossless if the output is the result of a digital one-to-one operation on the current sample and the recovery processor is able to construct the inverse operator". It is important to note that the operations must be one-to-one not only on paper but also on the computer. Integer addition of the current sample and a constant is one-to-one under some amplitude constraints on all computer architectures. Integer addition can be expressed as

$e(n) = f o f(x(n) + s(n))$

Where f*of* is the overflow operator, x (n) is the current sample, s(n) an integer value which defines the transformation at time n, and e (n) is the current filter output.

The reverse operation is given by the equation
x(n) = f*of* (e(n) - s(n))
This process always leads to an increase in number of bits required. To overcome this, rounding operation on the predictor output is performed making the predictor lossless. The lossless predictive encoder and decoder are shown

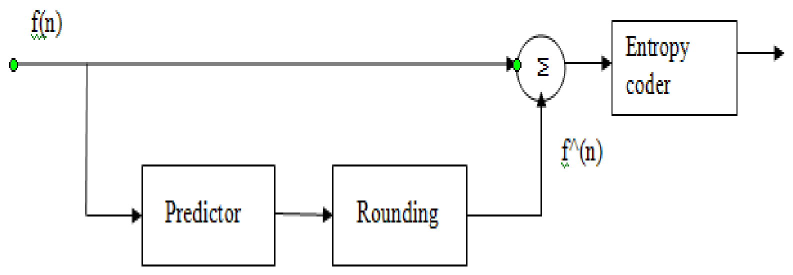


Figure 2.2 Predictive Encoder

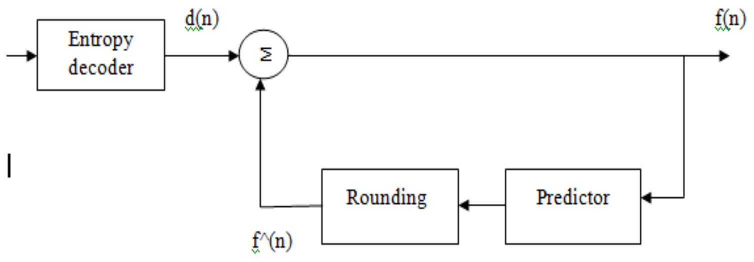


Figure 2.3 Predictive Decoder

In the process of predictive coding input image is passed through a predictor where it is predicted with its two previous values.
f^ (n) = α * f (n-1) + β * f(n-2)
f^(n) is the rounded output of the predictor, f(n-1) and f(n-2) are the previous values, α and β are the coefficients of the second order predictor ranging from 0 to 1. The output of the predictor is rounded and is subtracted from the original input. This difference is given by
d (n) =f (n)-f^ (n)
Now this difference is given as an input to the decoder part of the predictive coding technique. In the decoding part the difference is added with the f^ (n) to give the original data.
f (n)= d(n) + f^(n)

G. Bit plane coding

Bit-plane coding has been widely used in lossless compression of gray-scale images or color palette images. Progressive transmission can be used in bit-plane coding strategy. A gray-scale image can be converted into bit-plane images. We first note that any integer value s can be represented by a polynomial of base-2, as follows:
s= $a_{n-1}2^{n-1} + a_{n-2}2^{n-2}$+............$a_1 2^1 a_0 2^0$
where $a_{n-1}, a_{n-2} \ldots . a_0$ are the n binary coefficients associated with the corresponding powers of 2. This basically converts the integer s into the binary number representation $a_{n-1}, a_{n-2} \ldots . a_0$with a_{n-1} as the most significant bit and a0 as the least significant bits. Following this, all the pixel values having intensities in the range [0, 255] of an image array can be converted into its 8-bit binary representation. If we now consider an array containing only the i th bit of every pixel(i = 0,1,.....,7), we get the i th bit plane i

III. EXPERIMENTAL RESULTS

A. Result of huffman coding

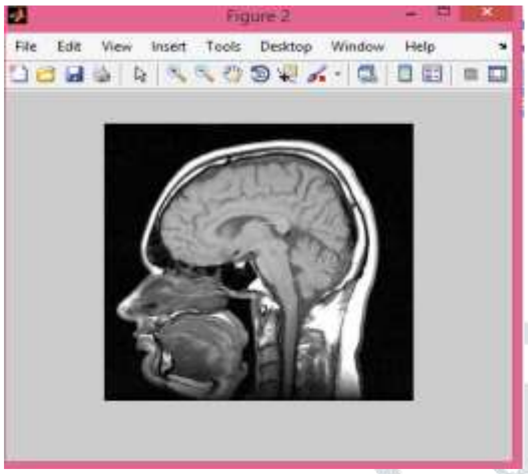


Figure 3.1 Original Image

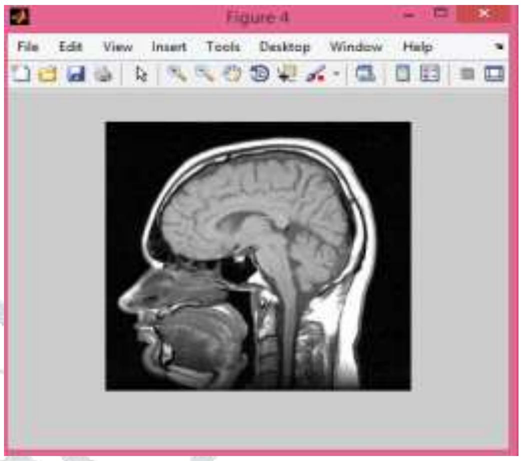
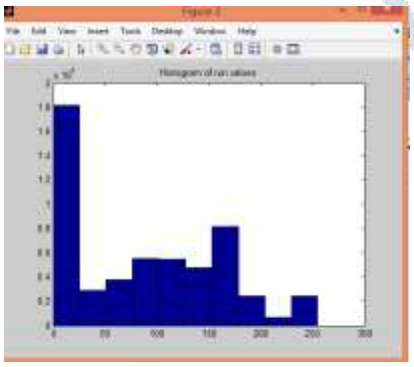
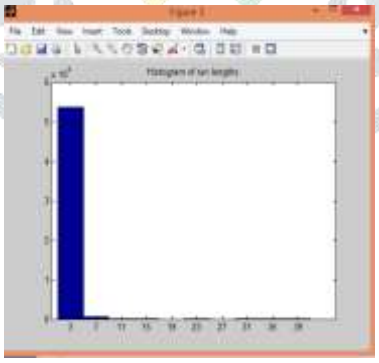
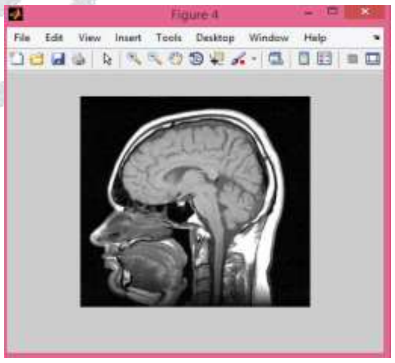


Figure 3.2 Reconstructed Image

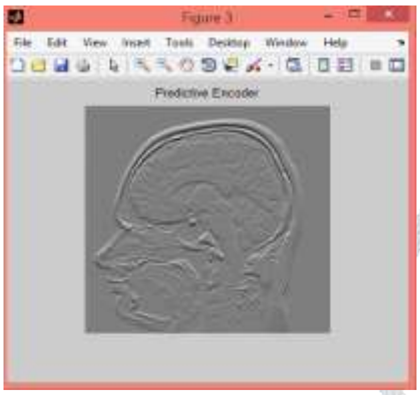
B. Result of Run Length Coding


3.3Histogram of Run values

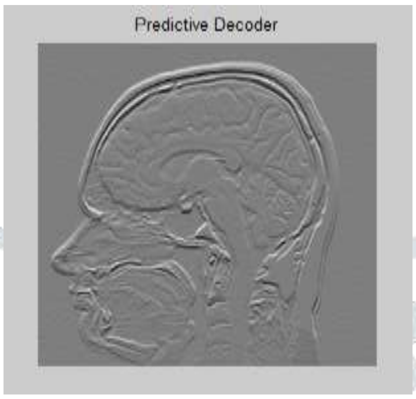

3.4Histogram of Run lengths


3.5Reconstructed Image

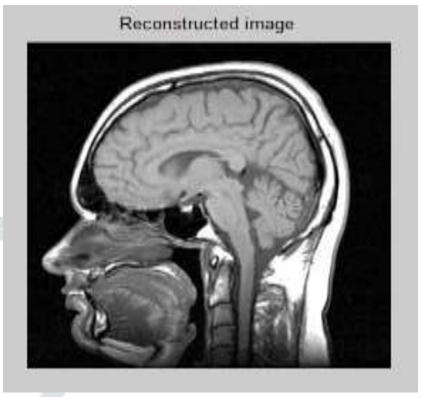
C. Result of Predictive coding



3.6 PREDICTIVE ENCODER

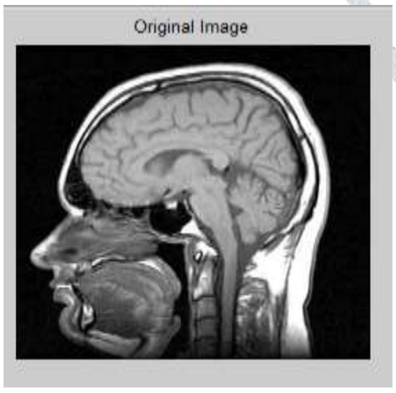


3.7 PREDICTIVE DECODER

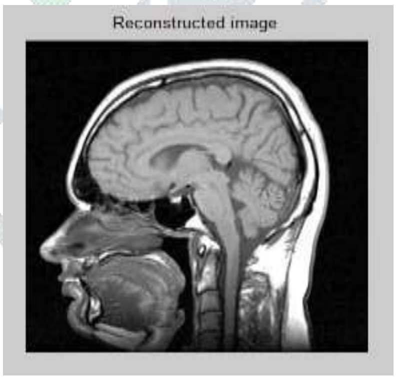


3.8 RECONSTRUCTED IMAGE

D. ***Result of lzw coding***



3.9 ORIGINAL IMAGE



3.10 RECONSTRUCTED IMAGE

E. RESULT OF BITPLANE CODING USING BINARY CODE

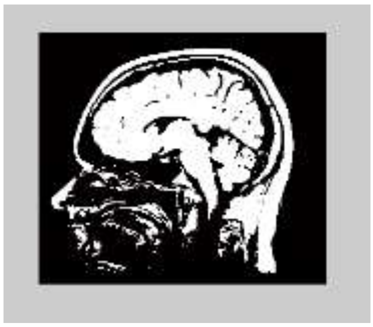


Fig3.11 Bitplane 1

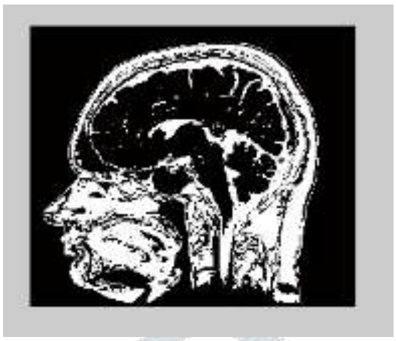


Fig3.12 Bitplane 2

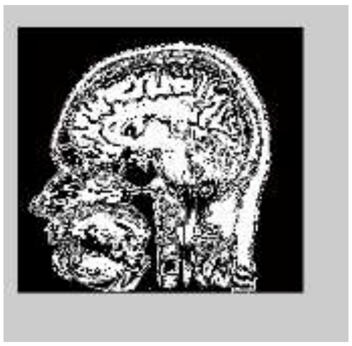


Fig3.13 Bitplane 3

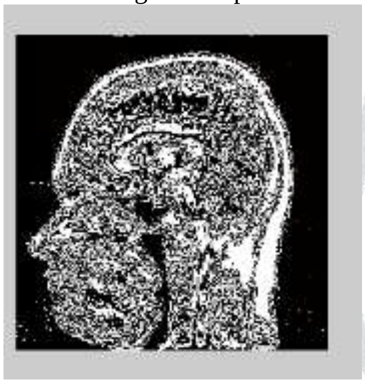


Fig3.14 Bitplane 4

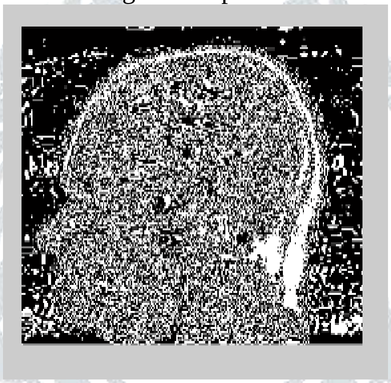


Fig3.15 Bitplane 5

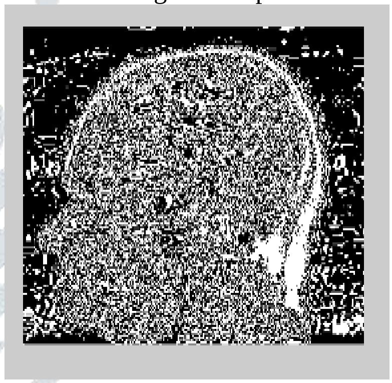


Fig3.16 Bitplane 6

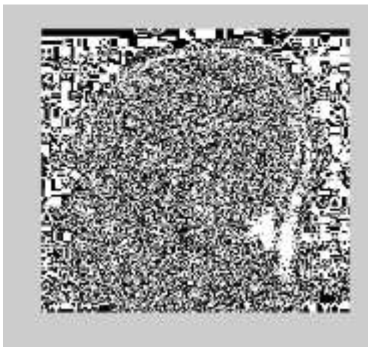


Fig3.17 Bitplane 7

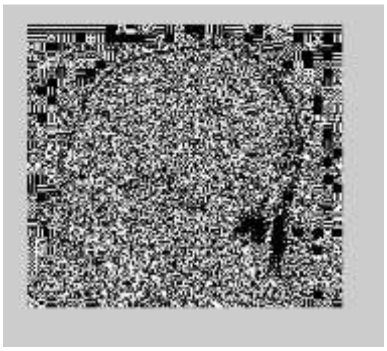


Fig3.18 Bitplane 8

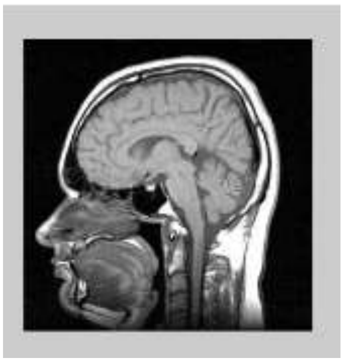


Fig3.19 Reconstructed Image

F. Comparison of compression ratio of all loss less compression methods

$$\text{Compression Ratio} = \frac{\text{Size of Original image}}{\text{Size of Reconstructed image}}$$

Image	Huffman Coding	Runlength Coding	Predictive Coding	LZW Coding	Bitplane Coding using Binary code
Head	1.1458	0.6901	1.3899	1.1215	1.7949

Table 3.1. ***Comparision of compression ratiosof compression mtehods***

IV CONCLUSION

We have taken a standard image and applied the lossless compression algorithms on it. It is found that the compression ratio of run length coding is very low because run length coding is only effective for repetitive data , and the compression ratio of LZW is higher than run length coding because LZW is the best technique for reducing the size of files containing more repetitive data. The compression ratio of huffman coding is higher than run length and LZW coding which is in the range 2:1 and the compression ratio of predictive coding is higher than Huffman coding because this method encodes the predicted image rather than the original image that contains lower entropy . Amongst all the lossless compression algorithms discussed we found that the better compression ratio is given by bit plane coding.

References

[1] Rafael C. Gonzalez, Richard E. Woods, Digital Image Processing, Pearson Education, 2002.

[2] Dr. T. Bhaskara Reddy, Miss. Hema Suresh Yaragunti , Dr. S. Kiran, Mrs. T. Anuradha , “A Novel Approach of Lossless Image Compression using Hashing and Huffman Coding”, International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181, Vol. 2 Issue 3, March – 2013.

[3] Mohammed Al-laham1 & Ibrahiem M. M. El Emary2, “Comparative Study BetweenVarious Algorithms of Data Compression Techniques”, Proceedings of the World Congress on Engineering and Computer Science 2007 WCE CS 2007, October 24-26, 2007, San Francisco, USA.

[4] Jagadish H. Pujar, Lohit M. Kadlaskar , A New Lossless Method of Image Compression and Decompression Using Huffman Coding Techniques, Journal of Theoretical and Applied Information Technology, 2005 – 2010, p-18 to 23.

[5] Streams, S.D., L. Tan and N. Magotra, 1993. “Lossless compression of waveform data for efficient storage and transmission”. IEEE Trans. Geosci. Remote Sens., 3I: 645-654.