Volume 3, Issue 11, November 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

A Review on Lossless Image Compression Techniques

Aarti
Department of CSE, ACET
India

Abstract— ***Compression is a technique to reduce the quantity of data without excessively reducing the quality of the data. Image compression is the application of data compression on digital images. Image compression is used to minimize the amount of memory needed to represent an image, for storage or transmission of information. When hearing that image data are reduced, one could expect that automatically also the image quality will be reduced. A loss of information is, however, totally avoided in lossless compression, where image data are reduced while image information is totally preserved.TV and fax machines are both examples of image transmission and digital video player is the example of image storage. In this paper we review and discuss about the Image, different types of Image formats, Lossless image compression, need of lossless compression, its principles, benefits, performance measures and various techniques of lossless image compression.***

Keywords— ***Benefits of Compression, Compression Techniques, Image, Image formats Lossless Image compression, Need, Principles, Performance Measures.***

I. INTRODUCTION

An image can be thought of as a matrix of pixel (or intensity) values. Images are defined simply as a set of two-dimensional arrays of integer data (the samples), represented with a given precision (number of bits per component). Each array is termed a component, and color images have multiple components, which usually result from a representation in some color space (e.g., RGB, YUV, CMYK) [9]. A common characteristic of most images is that the neighboring pixels are correlated and therefore contain redundant information. As digital images generally contain a significant amount of redundant information, these large storage requirements can be reduced by proper data compression. Image data compression is concerned with minimizing the number of bits required to represent an image. Compressing an image is significantly different than compressing raw binary data. Generally, images carry three main type of information: redundant, irrelevant, and useful[12]. Different types of images are used in remote sensing, bio medical and video processing techniques which require compression for transmission and storage. [2].Compression is achieved by removing redundant or extra bits from the image.

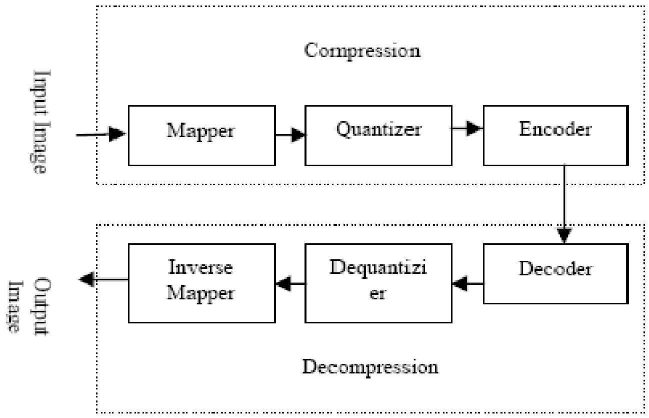

Fig 1.1 General Compression Decompression

As shown in Fig.1.1, First of all the image is taken from the image dataset. The mapper converts the input image into inter pixel coefficients.
Transformation for the mapper may be DCT, wavelet or Curvelet transform. Each has its own advantages and disadvantages. Second stage is the quantizer which simply reduces the number of bits needed to store the transformed coefficients. It is many to one mapping in which larger values are quantized into smaller value. It is a lossy process and it is the main source of compression in an encoder. Quantization reduced the number of bits so it results some kind of information loss. Quantizer can be scalar or vector quantization. In scalar Quantizer quantization is performed on each coefficient while in vector quantization it can be performed on groups.

An entropy encoder compressed the quantized values and improves the compression. The reverse Process Decoder, De quantizer and inverse mapper is obtained to reconstruct the image and it is called decompression [25].

1.1. Need of Lossless Compression

The need for lossless compression arises from the fact that many applications, such as the compression of digitized medical data, require that no loss be introduced from the compression method.

1.2. Principles behind Compression

The principle of image compression algorithms are

(i) reducing the redundancy in the image data and (or)

(ii) producing a reconstructed image from the original image with the introduction of error that is insignificant to the intended applications. The aim here is to obtain an acceptable representation of digital image while preserving the essential information contained in that particular data set [3].

1.3 Benefits of Lossless Image Compression

- Lossless compression algorithms usually exploit statistical redundancy in such a way as to represent the sender's data more concisely, but nevertheless perfectly. Lossless compression is possible because most real-world data has statistical redundancy[32]
- Lossless compression schemes are reversible so that the original data can be reconstructed. Lossless compression results in a closer representation of the original media, and thus a higher quality end product

II. RELATED WORK

Various Image File Formats

Images are an important part of today's digital world [9]. Some of these compression methods are designed for specific kinds of images, so they will not be so good for other kinds of images. Some algorithms even let you change parameters they use to adjust the compression better to the image. File formats are essential when it comes to compatibility and storing images .Different image formats use different types of data compression. An image file format is a standard way to organize and store image data. It defines how the data is arranged and the type of compression (if any) to be used.There are several dozens of different image file formats. Some of them use compression techniques while others do not. Some formats support only lossy while others also allow lossless compression. Some compression techniques are more useful for images of natural scenes rather than computer generated or artificial images [5]. Many facsimile and document imaging file formats support a form of lossless data compression often described as CCITT encoding. The CCITT (International Telegraph and Telephone Consultative Committee) is a standards organization that has developed a series of communications protocols for the facsimile transmission of black-and-white images over telephone lines and data networks. These protocols are known officially as the CCITT T.4 and T.6 standards but are more commonly referred to as CCITT Group 3 and Group 4 compression, respectively. Sometimes CCITT encoding is referred to, not entirely accurately, as Huffman encoding. CCITT 1-dimensional encoding. The other types of CCITT encodings are [6].

Table 1 Image Compression standards and formats

ISO/IEC/ITU-T	JPEG · JPEG 2000 · lossless JPEG · JBIG · JBIG2 · PNG · WBMP
Others	BMP · GIF · ICER · ILBM · PCX · PGF · TGA · TIFF · JPEG XR / HD Photo

2.1 BMP

BMP (bitmap) is a bitmapped graphics format used internally by the Microsoft Windows graphics subsystem (GDI),and used commonly as a simple graphics file format on that platform. It is an uncompressed format. The Bitmap (BMP) file format deal with graphic file related to Microsoft windows OS. Normally these files are uncompressed so they are large. These files are used in basic windows programming [19]. BMP images are binary files.BMP file does not support true colors.

2.2 PNG

PNG (Portable Network Graphics) (1996) is a bitmap image format that employs lossless data compression. PNG was created to both improve upon and replace the GIF format with an image file format that does not require a patent license to use. It uses the DEFLATE compression algorithm, that uses a combination of the LZ77 algorithm and Huffman coding. PNG was designed for distribution of images on the internet not for professional graphics and as such other color spaces. Lossless PNG format is best compare to lossy JPEG. Typically, an image in a PNG file can be 10% to 30% more compressed than in a GIF format [18].PNG format have smaller size and more colors compare to others.

2.3 TIFF

TIFF (Tagged Image File Format) is a file format for mainly storing images, including photographs and line art. It is one of the most popular and flexible flexible format which can be used for lossless or lossy Compression Originally created by the company Aldus, jointly with Microsoft, for use with PostScript printing, TIFF is a popular format for high color depth images, along with JPEG and PNG. TIFF format is widely supported by image-manipulation applications, and by scanning, faxing, word processing, optical character recognition, and other applications. In practice, TIFF is used as a lossless image storage format in which image compression is not used. For web transmission TIFF files are not used because TIFF files require large size.

2.4 GIF

Graphics Interchange Format (GIF) is useful for images that have less than 256 colors, grayscale.GIF is limited to an 8 bit or 256 colors. so that it can be used to store simple graphics ,logos and cartoon style images. It uses loss less compression.

2.5 RAW

RAW file format includes images directly taken from Digital cameras. These formats normally use loss less or lossy compression method and produce smaller size Images like TIFF. The Disadvantage of RAW Image is that they are not standardized image and it will be different for different manufactures. So these images require manufacture's software to view the images.

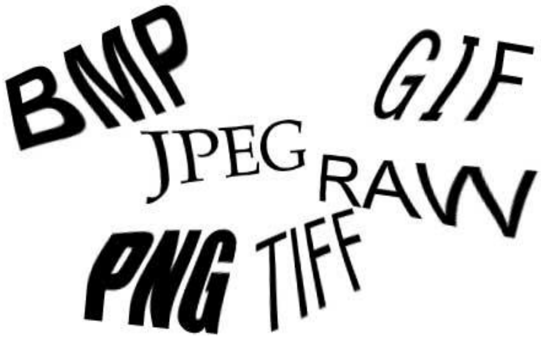

Fig. 2 Compressed image file formats

2.6 JPEG

Joint Photographic Expert Group (JPEG) is a lossy compression technique to store 24 bit photographic images. It is widely accepted in multimedia and imaging industries. JPEG is 24 bit color format so it have millions of colors and more superior compare to others[21].it is used for VGA(video graphics Array) display. JPEG have lossy compression and it support 8 bit gray scale image and 24 bit color images.

2.7 JPEG2000

JPEG 2000 is a compression standard for lossless and lossy storage.JPEG2000 improves the JPEG format. it is nearly same as JPEG.

2.8 Exif

The Exif (Exchangeable Image File Format) is similar to JFIF format with TIFF extension. it is used to record and exchange of images with image metadata between the digital camera and editing and viewing software.

2.9 WEBP

WEBP is a new image format that use lossy image compression. It was designed by Google to reduce image file size to increase the speed when web page is loading. It is based on VP8s in a frame coding[25].

2.10 NETPBM

NetPbm format contain three family formats: the PPM (portable Pixel Map), the PGM (portable Gray Map) and the Portable bit map. [20] These files are pure ASCII files or raw binary files. The portable pix map format (PPM), the portable gray map format (PGM) and the portable bitmap format (PBM) are image file formats designed to be easily exchanged between platforms. They are also sometimes referred to collectively as the portable any map format (PNM)[43].

3 Lossless Image Compression

Lossless image compression is the process of compressing and subsequently decompressing images without the loss of data. Their properties generate an exact duplicate of the original image upon decompression. In lossless compression schemes, the reconstructed image, after compression, is numerically identical to the original image [8]. The goal of lossless image compression is to represent an image signal with the smallest possible number of bits without loss of *any* information, thereby speeding up transmission and minimizing storage requirements [9].Lossless compression gives good quality of compressed images, but yields only less compression[16]. The overall compression scheme may be a combination of a lossy compression process followed by a lossless compression process. Various image, video, and audio compression standards follow this model, and several of the lossless compression schemes used in these standards.The general model of a lossless compression scheme is as depicted in the following figure.

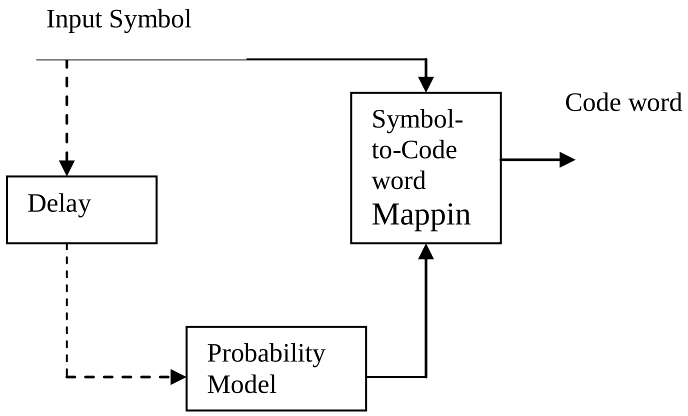

Fig 3.1: A generic model for lossless compression

Given an input set of symbols, a modeler generates an estimate of the probability distribution of the input symbols. This probability model is then used to map symbols into code words. The combination of the probability modeling and the symbol-to-code word mapping functions is usually referred to as entropy coding. The key idea of entropy coding is to use short code words for symbols that occur with high probability and long code words for symbols that occur with low probability. The probability model can be derived either from the input data or from a priori assumptions about the data[10]. Lossless compression is sometimes preferred for artificial images such as technical drawings, icons or comics. Lossless image compression is required (or desired) in applications where the pictures are subject to further processing (e.g., for the purpose of extraction of special information), intensive editing, or repeated compression/decompression. It is generally the choice also for images obtained at great cost, or in applications where the desired quality of the rendered image is yet unknown. Thus, medical imaging, pre-press industry, image archival systems, precious art works to be preserved, and remotely sensed images, are all candidates for lossless compression. Lossless data compression algorithms usually exploit statistical redundancy to represent data more concisely without losing information. Lossless compression is possible because most real-world data has statistical redundancy [1] . Lossless compression methods may also be preferred for high value content, such as medical imagery or image scans made for archival purposes.

3.1 Lossless Compression Techniques [11]

The different techniques for lossless image compression include Run – length encoding, Shannon Fano Coding, Golomb Coding, Symbol-based Coding, Huffman Coding, Arithmetic Coding , Dictionary algorithm such as LZW (Lempel-Ziv Welch) compression method, Area Coding , Bit-plane coding, Lossless predictive coding, Byte Pair Encoding, Predictive Partial Matching etc.

3.1.1 Repetitive Sequence Suppression or Run Length Encoding

RLE is the simplest image compression technique in which sequence of identical symbols are replaced by a pair containing the symbol and the length at which the number is repeated [20]. It is widely accepted compression technique in the fax standard. RLE is particularly effective when compressing binary images since there are only two possible intensities (black and white). Additionally, a variable-length coding can be applied to the run lengths themselves.
The approximate run-length entropy is

$$H_{RL} = H_0 + H_1 / L_0 + L_1 \quad (1)$$

where $H0$ and $H1$ are entropies of the black and white runs and $L0$ and $L1$ are the average values of black and white run lengths. Images with repeating intensities along their rows or columns can often be compressed by representing runs of identical intensities as ***run-length pairs***, where each run-length pair specifies the start of a new intensity and the number of consecutive pixels having that intensity. This technique is used for FAX data compression, in BMP file format, etc. Here is an example of RLE[29]:
Input: AAABBCCCCDEEEEEEAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAA AAAA AAAA
Output: 3A2B4C1D6E38A

The first step in this technique is read file then it scans the file and find the repeating string of characters .when repeating characters found it will store those characters with the help of escape character followed by that character and count the binary number of items it is repeated. This method is useful for image having solid black pixels. This algorithm is also effective for repeating of characters. But it is not effective if data file has less repeating of characters. We can compress the run-length symbols using Huffman coding, arithmetic coding, or dictionary based methods.Two oldest image compression standards developed for FAX are CCIT Group 3 and 4 standards for binary image compression.

1D CCITT Compression

Each line in the image is encoded as a series of variable-length Huffman code words representing the run lengths of alternating white and black runs in a left-to-right scan of the line. The compression method is often called a modified Huffman (MH) coding. The code words are of two types: terminating codes and makeup codes.

2D CCITT compression

The 2D compression method used by CCITT Group 3 and 4 standards is a line-by-line method where the position of each black-to-white or white-to-black run transition is coded with respect to the position of a reference element a0 that is situated on the current coding line. The previously coded line is called the reference line; the reference line for the first line of each image is an imaginary white line. The 2D coding technique is called Relative Element Address Designate (READ) coding. The coding is 2D in a sense that information from the previous line is used to encode the current line. Run-length coding is implemented in CCITT, JBID2, JPEG, M-JPEG, MPEG-1,2,4, BMP standards

TABLE 2

Result of Performing Shannon Fano on HELLO

Symbol	Count	$\log_2 \frac{1}{p_i}$	Code	# of bits used
L	2	1.32	0	2
H	1	2.32	10	2
E	1	2.32	110	3
O	1	2.32	111	3
TOTAL number of bits:				10

3.1.2 Shannon Fano Coding Technique (Variable-Length Coding (VLC))

This is another variant of Static Huffman Coding algorithm [36]. It is one of the earliest compression techniques, invented in 1949 by Claude Shannon and Robert Fano. This technique involves generating a binary tree to represent the probabilities of each symbol occurring. The symbols are ordered such that the most frequent symbols appear at the top of the tree and the least likely symbols appear at the bottom. The code for a given symbol is obtained by searching for it in the Shannon-Fano tree, and appending to the code a value of 0 or 1 for each left or right branch taken, respectively. For example, if "A" is two branches to the left and one to the right its code would be "0012". Shannon-Fano coding does not always produce optimal codes due to the way it builds the binary tree from the bottom up. For this reason, Huffman coding is used instead as it generates an optimal code for any given input [29]. It is a top-down approach. It is used to encode messages depending upon their probabilities [1]. The method is defined as given below:- 1. For a given list of symbol create a probability table. 2. Sorting the table based on the probability and places the most frequent symbol at the top of a list. 3. The table is divided into equally two halves upper and lower which having a same probability as much as possible. 4. The upper half of the list defined with „0‟ digit and the lower half with a „1‟. 5. Repeat the steps 3 and 4 for each of the two halves then further divide the groups and adding bits to the codes and stop the process when each symbol has a corresponding leaf on the tree.

An Example: coding of "HELLO"

Symbol H E L O

Count 1 1 2 1

Frequency count of the symbols in "HELLO".

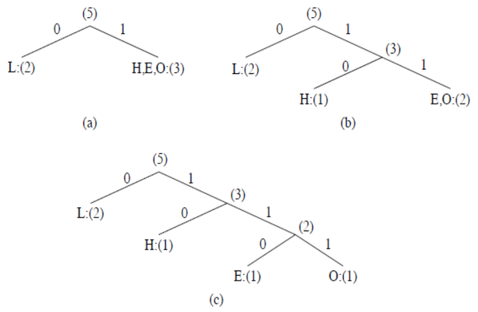

Fig 3.2 Coding Tree for HELLO by Shannon Fano

3.1.3 Golomb coding

Golomb coding is a lossless data compression method using a family of data compression codes invented by Solomon W. Golomb in the 1960s. Alphabets following a geometric distribution will have a Golomb code as an optimal prefix code, making Golomb coding highly suitable for situations in which the occurrence of small values in the input stream is significantly more likely than large values. Rice coding (invented by Robert F. Rice) denotes using a subset of the family of Golomb codes to produce a simpler (but possibly suboptimal) prefix code; Rice used this in an adaptive coding scheme, although "Rice coding" can refer to either that scheme or merely using that subset of Golomb codes. Whereas a Golomb code has a tunable parameter that can be any positive value, Rice codes are those in which the tunable parameter is a power of two. This makes Rice codes convenient for use on a computer, since multiplication and division by 2 can be implemented more efficiently in binary arithmetic[33]. Golomb codes can only be used for representing nonnegative integers. However it is easily extended to accept sequences containing negative numbers using an overlap and interleave scheme, in which all values are re-assigned to some positive number in a unique and reversible way. The sequence begins: 0, -1, 1, -2, 2, -3, 3, -4, 4 ... The nth negative value (i.e., -n) is mapped to the nth odd number (2n-1), and the m^{th} positive value is mapped to the m^{th} even number (2m)[33].There are many possible Golomb codes. It is important to properly select a divisor m for the effective application. When the integers are geometrically distributed with probability mass function in the form:

$$P(n) = (1-\rho)\rho^n \qquad (2)$$

for some $0 < \rho < 1$, Golomb codes are optimal (give the shortest average code length), when

$$m=[\log_2 (1+\rho)/ \log_2 (1/\rho)] \qquad (3)$$

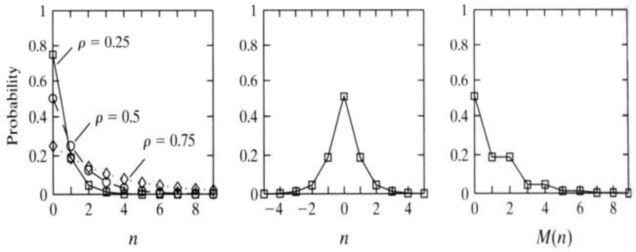

Fig 3.3 Symbol probabilities Two-sided exponentially and its mapped For different ρ decaying function version

Probabilities of image intensities are quite unlikely to match the geometrical distribution, therefore, Golomb codes are NOT often used to code intensities. However, the intensity differences usually can be approximated as geometrically distributed. The only problem with intensity differences is that some of them will be negative. To handle negative differences mappings are used. Golomb codes are used in JPEG-LS and AVS compression standards[26]. The Golomb–Rice coder is used in the entropy coding stage of Rice Algorithm based lossless image codecs.

3.1.4 Symbol-based coding

In symbol- or token-based coding, an image is represented as a collection of frequently occurring sub-images, called symbols. Each symbol is stored in a symbol dictionary and the image is coded as a set of triplets {(x1,y1,t1), (x2,y2,t2),…}, where each (xi,yi) pair specifies the location of a symbol in the image and a token ti is the address of the symbol (sub-image) in the dictionary. Therefore, each triplet represents an instance of a dictionary symbol in the image. Storing repeated symbols only once can compress image significantly (especially, for text document storage) if symbols are repeating.

Consider a simple bilevel image

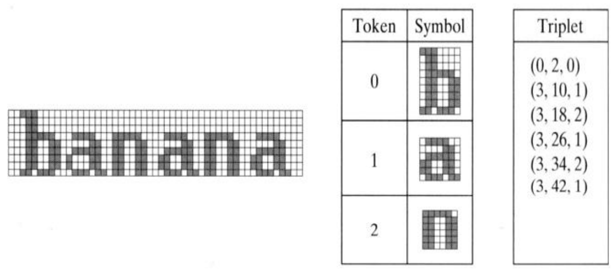

Fig 3.4 Symbol Based Coding

consisting of a word “banana” that is composed of three symbols: “b” (1), “a” (3), and “n” (2). Since “b” is the first symbol identified in the coding process, its 9x7 bitmap is stored in location 0 of the symbol dictionary. The token identifying the bitmap “b” is 0. Therefore, the first triplet in the image is 0,2,0) indicating that the upper left corner of the bitmap specified by the token 0 should be placed at location (0,2) in the image. After the bitmaps for “a” and “n” images are identified and placed to the dictionary, the image can be encoded with 5 triplets. If the six triplets required to locate the symbols in the image together with the three bitmaps needed to define them are smaller than the original image, compression occurs. In our case, the original is 9x51x1 = 459 bits; assuming that each triplet requires 3 bytes, the compressed representation is 6x3x8 + 9x7+6x7+6x6 = 285 bits. C = 1.61. Like many other compression methods, symbol-based decoding is significantly faster than encoding. The symbol bitmaps and the triplets can be further compressed. Finally, if only exact symbol matches are allowed, the compression will be lossless. Symbol-based coding is implemented in JBIG2 compression.

JBIG2 Compression

JBIG2 is a standard for bilevel image compression that segments an image into overlapping and (or) non-overlapping regions of text, halftone, and generic content.

Text regions are composed of characters; each of them is represented by a character bitmap (subimage).

Halftone regions are similar to text regions in a sense that they are composed of patterns arranged in a regular grid.

Generic regions contain no-text, non-halftone information. They are compressed using either arithmetic or run-length coding.

3.1.5. Statistical Coding

It includes the following techniques.

1. Huffman Coding
2. Arithmetic Coding
3. LZW Coding. [22][23]
4. Area coding

1. Huffman Coding Technique

Huffman coding is one of the most popular technique for removing coding redundancy. It has been used in various compression applications, including image compression. It is a simple, yet elegant, compression technique that can supplement other compression algorithms [13]. Huffman coding can reduce the file size by 10% to 50% by removing the irrelevant information. In this technique, smaller bit code is given to the pixel values which occur frequently and the higher bit code for repeated pixel value. In order to encode images.

- First of all image is divided in to 8X8 blocks
- Then each block is coded with particular symbols
- Huffman code is applied to the each block
- Encoding all the blocks

It is an efficient source coding algorithm for source symbols that are not equally probable. It yields the smallest possible number of code symbols per source symbol [14]. Huffman coding is the optimal lossless scheme for compressing a bit stream. It works by first calculating probabilities. Define permutations $\{0, 1\}^n$ by assigning symbols, say A;B;C;D. The bit stream might look like AADAC for example. Now the symbols are assigned new codes, the higher the probability the smaller the number of bits in the code [13]. The codes are the output of the Huffman coder in the form of a bit stream. Huffman Coding Algorithm is a bottom-up approach because a set of symbols and their probabilities are ordered from top to bottom in terms of decreasing probability values. To form the first source prediction, the bottom two probabilities (0.04 and 0.06) are combined to form a “compound symbol” with probability of 0.1. This compound symbol and its associated probability are placed in the first source prediction column so that probabilities of the reduced source also are ordered from the most to the least probable. This process is repeated until a reduced source with two symbols is reached[26].The final code appears as shown. The average code length is:

$L_{avg} = 0.4 \cdot 1 + 0.3 \cdot 2 + 0.1 \cdot 3 + 0.1 \cdot 4 + 0.06 \cdot 5 + 0.04 \cdot 5 = 2.2$ *bits* / *pixel*

The entropy of the source is 2.14 bits/symbol.

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

Fig. 3.5 Huffman Source Reductions

2. Arithmetic Coding Technique

The arithmetic codes generate non-block codes; therefore, a one-to one correspondence between source symbols and code words does not exist. Instead, an entire sequence of source symbols is assigned to a single code word that defines an interval of real numbers between 0 and 1. As the number of symbols in the message increases, the interval used to

represent it becomes smaller and the number of bits needed to represent the interval becomes larger. Each symbol in the message reduces the size of the interval according to its probability of occurrence. Since the symbols are not coded one at a time, this technique can (theoretically) achieve the highest possible coding efficiency. Arithmetic coding is change the method of replacing each bit with a code word. So it replaces a string of input data with a single floating point number as a output. The main purpose of this technique is to given an interval to each potential bit data.In Arithmetic coding, the last symbol is encoded and decoded first[24].
1. Arithmetic coding is a more modern coding method that usually than Huffman coding.
2. Huffman coding assigns a code word to each symbol which has an integral bit length.
3. Arithmetic coding can treat the whole string data as one unit.
4. A message is represented by a half-open interval [a, b) where a and b are real numbers between 0 and 1. Initially, the interval is [0, 1). When the message becomes longer, the length of the interval shorts and the number of bits needed to represent the interval increases.
It is based on following principle.

- The symbol alphabet should not infinite.
- All possible symbol sequence of give length should not infinite.
- The number of real number in the interval [0,1] can assign a unique subinterval for any given input sequence of symbols.

Arithmetic coding is often used when for coding binary symbols. As each symbol (bit) begins the coding process, its context is formed in the context determination block. Arithmetic coding is implemented in JBIG1, JBIG2, JPEG-2000, H.264, MPEG-4 AVC[26].

3. LZW (Lempel-Ziv Welch) Coding
LZW is the Lempel-Ziv-Welch algorithm created in 1984 by Terry Welch. It is the most commonly used derivative of the LZ78 family, despite being heavily patent-encumbered [29]. LZW is the most popular method. This technique has been applied for data compression. It is dictionary based coding, in the static dictionary coding the dictionary is fixed during the encoding and decoding while in dynamic dictionary coding the dictionary is updated when new word is introduced. The Lempel-Ziv-Welch (LZW) coding assigns fixed-length code words to variable-length sequences of coding symbols without a priori knowledge of the symbols probabilities. At the onset of the coding process, a dictionary (codebook) containing the source symbols to be coded is constructed. For 8-bit monochrome images, the first 256 words of the dictionary are assigned to intensities 0,1,…255. As the encoder sequentially examines image pixels, intensity sequences that are not in the dictionary are placed in algorithmically determined locations. If a 9-bit, 512-word dictionary is used for coding, the original 8+8 bits that were used to represent the two pixels will be replaced by a single 9-bit code word. The size of the dictionary is an important system parameter: if it is too small, the detection of matching intensity-level sequences will be less likely; if it is too large, the size of the code words will decrease compression performance. LZW improves on LZ78 in a similar way to LZSS; it removes redundant characters in the output and makes the output entirely out of pointers. It also includes every character in the dictionary before starting compression, and employs other tricks to improve compression such as encoding the last character of every new phrase as the first character of the next phrase. LZW is commonly found in the Graphics Interchange Format, as well as in the early specificiations of the ZIP format and other specialized applications. LZW is very fast, but achieves poor compression compared to most newer algorithms and some algorithms are both faster and achieve better compression. The main steps for this technique are given below:- Firstly it will read the file and given a code to each character. If the same characters are found in a file then it will not assign the new code and then use the existing code from a dictionary. The process is continuous until the characters in a file are null.
Algorithm :

```
BEGIN
s = next input character;
while not EOF
{ c = next input character;
if s + c exists in the dictionary
s = s + c;
else
{ output the code for s;
add string s + c to the dictionary with a new code;
s = c;
}
}
output the code for s;
END
```

LZW compression for string “ABABBABCABABBA”
Let’s start with a very simple dictionary (also referred to as a “string table”), initially containing only 3 characters, with codes as follows:
code string

----
1 A
2 B
1 C

Now if the input string is “ABABBABCABABBA”, the LZW compression algorithm works as follows:

s	c	output	code	string
			1	A
			2	B
			3	C
A	B	1	4	AB
B	A	2	5	BA
A	B			
AB	B	4	6	ABB
B	A			
BA	B	5	7	BAB
B	C	2	8	BC
C	A	3	9	CA
A	B			
AB	A	4	10	ABA
A	B			
AB	B			
ABB	A	6	11	ABBA
A	EOF	1		

The output codes are: 1 2 4 5 2 3 4 6 1. Instead of sending 14 characters, only 9 codes need to be sent (compression ratio = 14/9 = 1.56).

Fig 3.6 Working of LZW compression algorithm

A unique feature of the LZW coding is that the code dictionary is created while the data is encoded. The LZW coding is used in GIF, TIFF and PDF formats[26].

a. LZC

LZC, or Lempel-Ziv Compress is a slight modification to the LZW algorithm used in the UNIX compress utility. The main difference between LZC and LZW is that LZC monitors the compression ratio of the output. Once the ratio crosses a certain threshold, the dictionary is discarded and rebuilt [29]

b. LZT

Lempel-Ziv Tischer is a modification of LZC that, when the dictionary is full, deletes the least recently used phrase and replaces it with a new entry. There are some other incremental improvements, but neither LZC nor LZT is commonly used today.

c. LZMW

Invented in 1984 by Victor Miller and Mark Wegman, the LZMW algorithm is quite similar to LZT in that it employs the least recently used phrase substitution strategy. However, rather than joining together similar entries in the dictionary, LZMW joins together the last two phrases encoded and stores the result as a new entry. As a result, the size of the dictionary can expand quite rapidly and LRUs must be discarded more frequently. LZMW generally achieves better compression than LZT, however it is yet another algorithm that does not see much modern use.

d. LZAP

LZAP was created in 1988 by James Storer as a modification to the LZMW algorithm. The AP stands for "all prefixes" in that rather than storing a single phrase in the dictionary each iteration, the dictionary stores every permutation. For example, if the last phrase was "last" and the current phrase is "next" the dictionary would store "lastn", "lastne", "lastnex", and "lastnext".

e. LZWL

LZWL is a modification to the LZW algorithm created in 2006 that works with syllables rather than than single characters. LZWL is designed to work better with certain datasets with many commonly occuring syllables such as XML data. This type of algorithm is usually used with a preprocessor that decomposes the input data into syllables.

f. LZJ

Matti Jakobsson published the LZJ algorithm in 1985 and it is one of the only LZ78 algorithms that deviates from LZW. The algorithm works by storing every unique string in the already processed input up to an arbitrary maximum length in the dictionary and assigning codes to each. When the dictionary is full, all entries that occurred only once are removed[29].

4. Area coding

Area coding is an enhanced form of run length coding, reflecting the two dimensional character of images. This is a significant advance over the other lossless methods. For coding an image it does not make too much sense to interpret it as a sequential stream, as it is in fact an array of sequence.Area coding is highly effective and it can give better compression ratio but it has certain limitation that it can be applied to non - linear transformation[29].

3.1.6 Bit-plane coding

The bit-plane coding is based on decomposing a multilevel (monochrome or color) image into a series of binary images and compressing each binary image by one of several well known binary compression methods. The intensities of an m-bit monochrome image can be represented in the form of base-2 polynomial:

$a_{m-1}2^{m-1} + a_{m-2}2^{m-2}+..+ a_1 2^1+ a_0 2^0$

However, this approach leads to the situation, when small changes of intensity can have significant impact on bit-planes[26]. The lowest order bit-plane (corresponds to the least significant bit) is generated by a0 bits of each pixel, while the highest order bit-plane contains the am-1 bits. However, this approach leads to the situation, when small changes of intensity can have significant impact on bit-planes. Bit-plane coding is implemented in JBIG1 and JPEG-2000 standards.

3.1.7 Lossless Predictive Coding

Predictive lossless coding provides effective lossless image compression of both photographic and graphics content in image and video media [35]. Eliminating the interpixel redundancies of closely spaced pixels by extracting and coding only the new information (the difference between the actual and predicted value) in each pixel.

System architecture

- The same predictor in the encoder and decoder sides
- Rounding the predicted value
- RLC symbol encoder

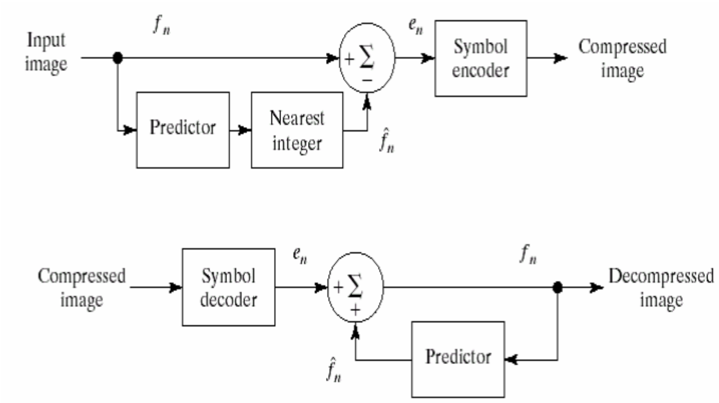

Fig 3.7 Predictive Encoder and Decoder

3.1.8 BPE

BPE (Byte Pair Encoding), which is developed by Philip Gage[41] , is a multi-pass digram coding algorithm. The algorithm repeats the encoding process until no further compression is possible, either because there are no more frequently occurring pairs or there are no more unused bytes to represent pairs[39]. In this , the most common pair of consecutive bytes of data is replaced with a byte that does not occur within that data. A table of the replacements is required to rebuild the original data.

Suppose we wanted to encode the data

Aaabdaaabac

The byte pair "aa" occurs most often, so it will be replaced by a byte that is not used in the data, "Z".

Now we have the following data and replacement table:

ZabdZabac

Z=aa

Then we repeat the process with byte pair "ab", replacing it with Y:

ZYdZYac

Y=ab

Z=aa

We could stop here, as the only literal byte pair left occurs only once. Or we could continue the process and use recursive byte pair encoding, replacing "ZY" with "X":

XdXac

X=ZY

Y=ab

Z=aa

This data cannot be compressed further by byte pair encoding because there are no pairs of bytes that occur more than once.
To decompress the data, simply perform the replacements in the reverse order[38].

3.1.9 PPM
PPM is an adaptive statistical data compression technique based on context modeling and prediction. The name stands for Prediction by Partial Matching. PPM models use a set of previous symbols in the uncompressed symbol stream to predict the next symbol in the stream. PPMd is an open-source data compression algorithm developed by Dmitry Shkarin [40]. WinZip 10.0 and 11.0 uses Version I, revision 1 of the algorithm. We also use the same version in our comparison [39]. PPM compression implementations vary greatly in other details. The actual symbol selection is usually recorded using arithmetic coding, though it is also possible to use Huffman encoding or even some type of dictionary coding technique. The underlying model used in most PPM algorithms can also be extended to predict multiple symbols. It is also possible to use non-Markov modeling to either replace or supplement Markov modeling. The symbol size is usually static, typically a single byte, which makes generic handling of any file format easy[42].

Performance Measures
The performance depends on the type and the structure of the input source. Various metrics are used to evaluate the performance of Lossless Image compression algorithms. The effectiveness of lossless compression schemes can be described using a relative measure, "compression ratio" or by describing an absolute measure, the "bit rate" of an image. The true effectiveness of the compression scheme may be better indicated by comparing the encoded bit rate with a measure of how much information is "really" encoded in the image. One such measure is the "entropy" of the image, a term from information theory [28] which is essentially the average amount of information per pixel value in the image. The "zero order" entropy does not take into account any information contained in surrounding pixels[27]. The performance of an image compression technique must be evaluated by considering different aspects.

- *Compression efficiency (Compression Ratio/Factor, bit per pixel bpp or bit rate)*
  Compression Ratio is the ratio between the size of the source file and the size of the compressed file [36].
  Compression Ratio C_R = uncompressed file size/ compressed file size or,

$$C_R = \frac{n_1}{n_2}$$

where n1 denotes the number of information carrying units in the original image and n2 denotes the number of information carrying units in the Compressed image. Compression ratio and bit distortion always contradict each other, so the techniques pursuing for higher compression ratio with less distortion even without information loss has been one of the popular issue in image compression [34].

$$\text{Compression ratio} = \frac{\text{old file size}}{\text{new file size}} : 1$$

$$\text{No. bits per pixel (bpp)} = \left(\frac{\text{new file size}}{\text{old file size}}\right) \times \text{old no. bpp}$$

Compression Factor is the inverse of the compression ratio. That is the ratio between the size of the source file and the size of the compressed file.

Compression factor = size before compression/ size after compression

There are some other methods to evaluate the performance of compression algorithms. Compression time, computational complexity and probability distribution are also used to measure the effectiveness.

Compression Time
Time taken for the compression and decompression should be considered separately. Some applications like transferring compressed video data, the decompression time is more important, while some other applications both compression and decompression time are equally important. If the compression and decompression times of an algorithm are less or in an acceptable level it implies that the algorithm is acceptable with respective to the time factor. With the development of high speed computer accessories this factor may give very small values and those may depend on the performance of computers[36]. Others are PSNR (peak signal to noise ratio) and Mean square error (MSE). Peak signal-to-noise ratio(PSNR) is one of the quantitative measure for image quality evaluation which is based on the mean square error(MSE) of the reconstructed image[37]. PSNR is the measurement of the peak error between the compressed image and original image. The higher the PSNR contains better quality of image. To compute the PSNR first of all MSE (mean square error) is computed.

Mean Square Error (MSE) is the cumulative difference between the compressed image and original image. Small amount of MSE reduce the error and improves image quality. The MSE for N x M size image is given by:

$$MSE = \frac{\sum_{M,N} [I_1(m,n) - I_2(m,n)]^2}{M * N}$$

In the above equation, *M* and *N* are the number of rows and columns in the input images. The PSNR is computed from following equation

$$PSNR = 10\log_{10}\left(\frac{R^2}{MSE}\right)$$

III CONCLUSIONS

We have reviewed and summarized various techniques of lossless image compression, different Image File formats, need of compression, principles behind compression, and performance measures. Hence all the lossless image compression techniques have been discussed. To conclude, all the lossless image compression techniques are useful in their related areas and every day new compression technique is developing which gives better compression ratio. This review paper gives clear idea about lossless compression techniques and image formats . Based on review of different types of images and their Compression algorithms we conclude that the compression algorithm depends on the following factors: quality of image (image format), amount of compression and time of compression.

REFERENCES

[1] S.Gavaskar, Dr.E.Ramaraj, R.Surendiran, "A Compressed Anti IP Spoofing Mechanism Using Cryptography ," *IJCSNS International Journal of Computer Science and Network Security*, VOL.12 No.11, November 2012 .

[2] Woods, R. C. 2008. Digital Image processing. New Delhi: Pearson Pentice Hall, Third Edition,Low price edition, Pages 1-904.

[3] David Salomon, Data Compression, The Complete Reference, 2nd Edition Springer-Verlag 1998.

[4] *Keshab K. Parhi, Takao Nishitan*; "Digital Signal processing for multimedia systems", ISBN 0-8247-1924

[5] http://www.prepressure.com/library/file-formats

[6] www. FileFormt.info

[7] http://www.owlnet.rice.edu/~elec539/Projects99/JAMK/proj1/

[8] http://gradworks.umi.com/14/39/1439199.html

[9] www.hpl.hp.com/research/papers/seroussiIEEE.pdf

[10] Rupinder Singh Brar, Bikramjeet singh, "A Survey on Different Compression Techniques and Bit Reduction Algorithm for Compression of Text/Lossless Data," *International Journal of Advanced Research in Computer Science and Software Engineering(IJARCSSE),*Volume 3, Issue 3, March 2013.

[11] Rajinder Kaur ,Mrs. Monica Goyal ,"A Survey on the different text data compression techniques ," *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET),* Volume 2, Issue 2, February 2013 .

[12] www.ijcaonline.org/journal/number20/pxc387621.pdf

[13] http://www.cprogramming.com/tutorial/computersciencetheory/huffman.h tml

[14] http://ee.lamar.edu/gleb/dip/index.html

[15] B. Venkatraman,V. Vaithiyanathan and B.Karthikeyan," Review on Lossless Image Compression Techniques for Welding Radiographic Images," *Research Journal of Applied Sciences, Engineering and Technology 5(1): 167-170, 2013.*

[16] Kaouri, A. H. . Fractal coding of still images. Queen's university of Belfast, UK. 2002.

[17] Sachin Dhawan," A Review of Image Compression and Comparison of its Algorithms*, " IJECT Vol. 2,Issue 1, March 2011*

[18] "Understanding Image Types"http://www.contentdm.com/USC/ tutorial/image-filetypes.pdf.1997-2005,DiMeMa,Inc,Unpublished.

[19] *John Miano*; "Compressed image file formats: JPEG,PNG, GIL, XBM, BMP", Edition-2, January-2000, page 23.

[20] *Majid Rabbani, Paul W.Jones;* "Digital Image Compression Techniques". Edition-4, 1991.page 51.

[21] http://www.find-images.com

[22] *Ronald G. Driggers*; "Encylopedia of optical engineering",Volume 2, Edition 1,2003.

[23] *Ioannis Pitas*; "Digital image processing algorithms and applications." ISBN 0-471-37739-2

[24] A.S. Ragab, Abdalla S.A. Mohmed, M.S. Hamid,"Efficiency of Analytical Transforms for Image Compression" 15th National Radio Science Conference, Feb.24-26, 1998, Cairo- Egypt.

[25] MR.CHANDRESH K PARMAR, PROF.KRUTI PANCHOLI, "A REVIEW ON IMAGE COMPRESSION TECHNIQUES," *JOURNAL OF INFORMATION, KNOWLEDGE AND RESEARCH IN ELECTRICAL ENGINEERIN* , VOLUME – 02, ISSUE – 02,2013.

[26] ee.lamar.edu/gleb/dip/08-2%20%20 Image%20 compression.pdf

[27] D. Clunie, Lossless Compression of Grayscale Medical Images, “Effectiveness of Traditional and State of the Art Approaches," in Proc. SPIE (Medical Imaging),vol.3980, Feb. 2000.

[28] Shannon CE, Weaver W, The mathematical theory of communication. University of Illinois Press, Urbana IL, 1949.

[29] http://www.ieeeghn.org/wiki/index.php/History_of_Lossless_Data Compression_Algorithms

[30] http://www.dsp.ee.ccu.edu.tw/lie/File/96-IP/Lie-Gonzalez_ch8.pdf

[31] http://rimtengg.com/coit2007/proceedings/pdfs/43.pdf

[32] http://www.maximumcompression.com/lossless_vs_lossy.php

[33] http://www.princeton.edu/~achaney/tmve/wiki100k/docs/Golomb_coding.html

[34] http://wavesharp.com/wujiaji/paper/wu_icip08.pdf

[35] http://www.google.com/patents/US7689051

[36] S.R. KODITUWAKKU, U. S.AMARASINGHE, “COMPARISON OF LOSSLESS DATA COMPRESSION ALGORITHMS FOR TEXT DATA”, *Indian Journal of Computer Science and Engineering*, Vol 1 No 4 416-425

[37] www.rimtengg.com/iscet/proceedings/pdfs/image%20p roc/101.pdf

[38] https://en.wikipedia.org/wiki/Byte_pair_encoding

[39] Aydın CARUS, Altan MESUT,” COMPARISON OF THE PERFOR MANCE OF COMPRESSION METHODS DEPENDING ON NATURAL LANGUAGES ,”*INTERNATIONAL SCIENTIFIC CONFERENCE 23-24November 2007,GABROVO*

[40] Shkarin, D., PPM: One Step to Practicality, *Proceedings of IEEE Data Compression Conference*, Snowbird:202-211, 2002.

[41] Gage, P., A New Algorithm for Data Compression, *The C Users Journal* 12:23-38,1994.

[42] http://www.maximumcompression.com/algoritms.php

[43] http://en.wikipedia.org/wiki/Netpbm_format

