International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

A Survey of Lossless Image Compression Techniques

Fouzia I Khandwani[1], P E Ajmire[2]
[1]PG Scholar, [2]HOD, Department of Computer Science, G S Science, Arts & Commerce College, Khamgaon, India
[1]ziakhandwani@gmail.com, [2]peajmire@rediffmail.com

Abstract: Memory is a vital resource which should be used efficiently. Compression helps us achieve this goal. Image compression techniques are mainly used to decrease the memory size requirement for the image. These compression techniques are of two types, lossy and lossless. In lossless, the original image is exactly reconstructed after the decompression whereas, lossy may loose some information from the image data. Lossless gives more accuracy as compared to the lossy compression.

Keywords: Image Compression, Lossy, Lossless, Run Length Encoding, Huffman Encoding, Compression Ratio.

I. INTRODUCTION

Images usually occupy more space on a hard disk or bandwidth in a transmission system than words. It goes against the saying, 'An image is worth a thousand words'. Hence, in the wide area of signal processing, efficient signal representations is a very high-activity research domain. Efficiency, in this perspective, means to include a representation from which we can recover some estimation of the original signal, which doesn't occupy larger space. Unluckily, these are conflicting requirements; to have better pictures, we usually require more memory [1].

This paper presents various techniques used for image compression. The major need for image compression is to decrease the size the image for storage purpose without degrading the quality of the image. The decrease in image size allows more images to be stored in a specified amount of disk or memory space. This reduces the time needed for images to be sent over the Internet or downloaded from Web pages and time required to process the image. It also reduces the redundancy & transforms the data in an efficient way [2].

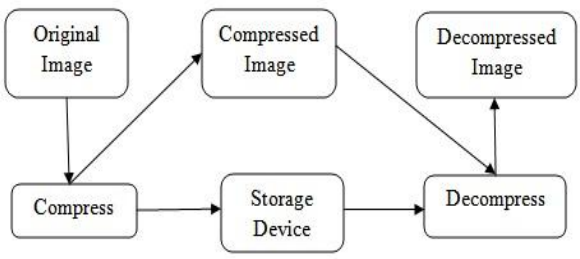

Fig. 1 Flow of Image Compression

Digital image processing is the use of computer algorithms to perform image processing on digital images. It allows various algorithms to be applied to the input data and can overcome the problems such as the build-up of noise and signal distortion during processing [3]. The flow of image compression can be easily understood from the figure above.

II. TYPES OF IMAGE COMPRESSION

Image compression is broadly divided into two types which are as follows:

2.1 Lossy

2.2 Lossless

2.1 Lossy Compression Techniques: In lossy compression, as the name states that there is the loss of data in some manner. The decompressed image is not same as the actual image [4]. Lossy compression is most frequently used to compress multimedia data. Lossless compression is typically required for text and data files, such as bank records and text articles. Mostly it is beneficial to make a master lossless file that can then be used to produce compressed files for different purposes [3].

a) Predictive Coding
b) Transform Coding
c) Fractal
d) Chroma Sampling

2.2 Lossless Compression Techniques: Lossless compression is a compression in which after decompression the image remains same as the original image. Lossless data compression most probably exploits statistical redundancy to express data more precisely without any loss in information [2]. It compresses the image by encoding all the information from the original file, so when the image is decompressed, it will be exactly matching the original image [5].

a) Run Length Encoding
b) Entropy Encoding
c) Huffman Encoding
d) Arithmetic Coding
e) Lempel–Ziv–Welch Coding
f) Deflation
g) Chain Codes

a) Run-length encoding (RLE) is one of the simplest forms of data compression methods. The principle of RLE is to exploit the repeating values in a source. This repeating string of characters is called a run. The algorithm counts the consecutive repeating amount of a

symbol and uses that value to represent the run. In RLE, runs of data are stored as a single data value and count, rather than as the original run. This is most useful on data that contains many such runs: for example, simple graphic images such as icons, line drawings, and animations. It is not useful with files that don't have many runs as it could greatly increase the file size. It is a method that allows data compression for information in which pixels are repeated constantly. It is based on the fact that the repeated pixel can be substituted by a number indicating how many times the pixel is repeated and the pixel itself [6]. This is most useful on data that contains many such runs, for example, simple graphic images such as icons, line drawings, and animations. It is not useful with files that don't have many runs as it could greatly increase the file size. Run-length encoding performs lossless image compression. Run-length encoding is used in fax machines [7].

b) Entropy encoding is a lossless data compression scheme that is independent of the specific characteristics of the medium. One of the main types of entropy coding creates and assigns a unique prefix-free code to each unique symbol that occurs in the input. These entropy encoders then compress data by replacing each fixed-length input symbol with the corresponding variable-length prefix-free output codeword. The length of each codeword is approximately proportional to the negative logarithm of the probability. Therefore, the most common symbols use the shortest codes.

c) Huffman coding was proposed by Dr David A. Huffman in 1952, 'a method for the construction of minimum redundancy code'. Huffman code is a technique for compressing data. Huffman's greedy algorithm looks at the occurrence of each character and it as a binary string in an optimal way. Huffman coding is a form of statistical coding which attempts to reduce the number of bits required to represent a string of symbols. The algorithm accomplishes its goals by allowing symbols to vary in length. Shorter codes are assigned to the most frequently used symbols, and longer codes to the symbols which appear less frequently in the string. Code word lengths are no longer fixed like ASCII. Code word lengths vary and will be shorter for the more frequently used characters. It is a more successful method used for text compression. Huffman's idea is to replace fixed-length codes by variable-length codes, assigning shorter code words to the more frequently occurring symbols and thus decreasing the overall length of the data. When using variable-length code words, it is desirable to create a prefix-code, avoiding the need for a separator to determine codeword boundaries. Huffman coding creates such a code [8]. The Huffman algorithm is simple and can be described in terms of creating a Huffman code tree. The procedure for building this tree is:

- Start with a list of free nodes, where each node corresponds to a symbol in the alphabet.
- Select two free nodes with the lowest weight from the list.
- Create a parent node for these two nodes selected and the weight is equal to the weight of the sum of two child nodes.
- Remove the two child nodes from the list and the parent node is added to the list of free nodes.
- Repeat the process starting from step-2 until only a single tree remains.

After building the Huffman tree, the algorithm creates a prefix code for each symbol from the alphabet simply by traversing the binary tree from the root to the node, which corresponds to the symbol. It assigns 0 to a left branch and 1 for a right branch. The algorithm presented above is called as a semi-adaptive or semi-static Huffman coding as it requires knowledge of frequencies for each pixel from the image. Along with the compressed output, the Huffman tree with the Huffman codes for symbols or just the frequencies of pixels which are used to create the Huffman tree must be stored. This information is needed during the decoding process and it is placed in the header of the compressed file [9].

d) Among all the techniques, arithmetic coding is a powerful technique for lossless statically encoding and gain much more attention in few years. In arithmetic coding instead of coding each image pixel (symbol) individually, the entire image sequence is assigned single arithmetic code word. A code word from interval 0 to 1 (0, 1) is defined. The output from an arithmetic coding process is a single number less than 1 and greater than or equal to 0.This single number can be uniquely decoded to create the exact stream of symbols that went into its construction. To construct the output number, the symbols are defined a set probabilities.

e) Lempel–Ziv–Welch (LZW) is a universal lossless data compression algorithm created by Abraham Lempel, Jacob Ziv, and Terry Welch. It was published by Welch in 1984 as an improved implementation of the LZ78 algorithm published by Lempel and Ziv in 1978. The algorithm is simple to implement and has the potential for very high throughput in hardware implementations. It is dictionary-based compression technique which allows mapping of a variable length of the image sequence to a fixed length of code [10]. LZW algorithm records the pattern in the dictionary. The first 255 entries contain the value of ASCII; therefore, the actual allocation of the index to the string starts from index 256. The main working principle of LZW algorithm is the multiple occurrences of bit sequence for a given image that needs to be encoded. LZW algorithm builds a dictionary by replacing the multiple occurrences of the pattern by an index code. As it is

International Journal of Electrical Electronics & Computer Science Engineering
Volume 5, Issue 1 (February, 2018) | E-ISSN : 2348-2273 | P-ISSN : 2454-1222
Available Online at www.ijeecse.com

adaptive technique so no need to transmit the dictionary. At the receiver side, the dictionary will be rebuilt during decoding process [2].

f) Deflate is a lossless data compression algorithm that uses a combination of the LZ77 algorithm and Huffman coding. It was formerly defined by Phil Katz for version 2 of his PKZIP archiving tool and was later specified in RFC 1951 [1]. The Deflate compressed data format consists of a series of blocks, corresponding to successive blocks of input data. Each block is compressed using a combination of the LZ77 algorithm and Huffman coding. The LZ77 algorithm finds repeated substrings and replaces them with backward references (relative distance offsets). The LZ77 algorithm can use a reference to a duplicated string occurring in the same or previous blocks, up to 32K input bytes back [11].

g) A chain code is a lossless compression algorithm for monochrome images. The basic principle of chain codes is to separately encode each connected component, or "blob", in the image. For each such region, a point on the boundary is selected and its coordinates are transmitted. The encoder then moves along the boundary of the image and, at each step, transmits a symbol representing the direction of this movement. This continues until the encoder returns to the starting position, at which point the blot has been completely described, and encoding continues with the next blob in the image. This encoding method is particularly effective for images consisting of a reasonably small number of large connected components. Some popular chain codes include the Freeman Chain Code of Eight Directions (FCCE), Vertex Chain Code (VCC), Three Orthogonal symbol chain code (3OT), Directional Freeman Chain Code of Eight Directions (DFCCE) and Unsigned Manhattan Chain Code (UMCC). In particular, FCCE, VCC, 3OT and DFCCE can be transformed from one to another. A related blob encoding method is crack code. Algorithms exist to convert between chain code, crack code, and run-length encoding. Recently, the combination of Move-to-front transform and adaptive Run-length encoding accomplished efficient compression of the popular chain codes. Chain codes also can be used to obtain high levels of compression for image documents, outperforming standards like DjVu and JBIG2 [12].

III. COMPARATIVE ANALYSIS

Compression ratio is the ratio between the original size of the image and the compressed size of the image.

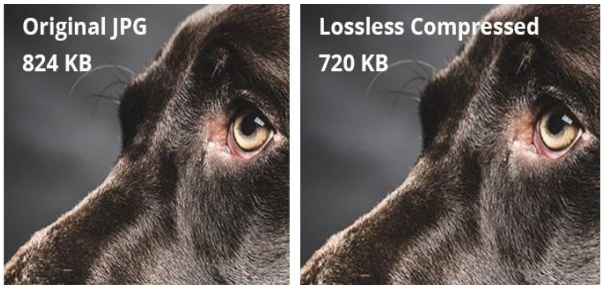

Image X

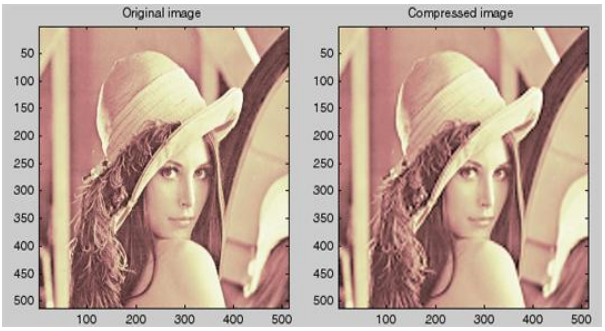

Image Y

Fig. 2 Original and Compressed Images for Algorithm Comparison

Algorithm	Image X	Image Y	Application Area
	Compression Ratio	Compression Ratio	
Run Length Encoding	1.03	1.02	Used mostly for frequently occurring sequences of pixels
Huffman coding	1.57	1.19	Used in JPEG
Arithmetic Coding	1.84	1.58	Used in TIFF and GIF files
LZW	1.28	1.36	Used mostly for TIFF, BMP

According to the comparative study between Deflate compression algorithms and Lempel-Ziv-Welch (LZW) data compression algorithm, the deflate algorithm is efficient in both compression rate as well as the speed at which the compression is done [11].

1. Run Length Encoding: In the worst case RLE generates the output data which is 2 times more than the size of input data. This is due to the fewer amount of runs in the source file. And the files that are compressed have very high values of compression ratio. This algorithm does not provide significant improvement over the original file.

2. Huffman Coding vs. Arithmetic Coding: Huffman Coding Algorithm uses a static table for the whole coding process, so it is faster. However it does not produce efficient compression ratio. On the contrary, Arithmetic algorithm can generate a high compression ratio, but its compression speed is slow [13].

Compression Method	Arithmetic	Huffman
Compression Ratio	Very Good	Poor
Compression Speed	Slow	Fast
Decompression Speed	Slow	Fast
Memory Space	Very Low	Low
Compressed Pattern Matching	No	Yes
Permits Random Access	No	Yes

IV. CONCLUSION

Image Compression is a significant area of research due to its wide range of applications. In this paper, a survey on various lossless compressing techniques is carried out. As studied in different papers related to lossy and lossless compression techniques, different compression techniques are used for better compression ratio for different types of data inputs. There are different types of entropy coding techniques that can be used for image compression. Comparative analysis is provided for the discussed techniques based on the compression ratio achieved by each technique. Lossless compression is an efficient method of compressed a given image without any failure of its original information in a resultant image. It will look same as the original. According to the studied papers, the arithmetic coding provides better compression ratio than Huffman coding. But the time taken by arithmetic coding is more than Huffman coding. So for more improvement in the arithmetic coding, one can apply the concept of hybridization of arithmetic coding.

V. REFERENCES

[1] S K Bandyopadhyay, T U Paul and A Raychoudhury, "Image Compression using Approximate Matching and Run Length", International Journal of Advanced Computer Science and Applications, Vol. 2, No. 6, 2011.

[2] D Chudasama, K Parmar, D Patel, K J Dangarwala, S Shah, "Survey of Image Compression Method Lossless Approach", International Journal of Engineering Research & Technology, Vol. 4 Issue 03, March-2015.

[3] M Kaur, N Kaur, "A Literature Survey on Lossless Image Compression", International Journal of Advanced Research in Computer and Communication Engineering, Vol. 4, Issue 3, March 2015.

[4] V K Padmaja, B Chandrasekhar, "Literature review of image compression algorithm", IJSER, Vol. 3, 2012.

[5] M K Mathur, S Loonker, D Saxena, "Lossless Huffman Coding Technique for Image Compression And Reconstruction Using Binary Trees", IJCTA, JAN-FEB 2012.

[6] F. Semiconductor, "Using the Run Length Encoding Features on the MPC5645S", pp. 1–8, 2011.

[7] S. Gaurav Vijayvargiya and R. P. Pandey, "A Survey: Various Techniques of Image Compression", Vol. 11, No. 10, 2013.

[8] M. Grossberg, I. Gladkova, S. Gottipati, M. Rabinowitz, P. Alabi, T. George, and A. Pacheco, "A comparative study of lossless compression algorithms on multi-spectral imager data", Proc. - 2009 Data Compression Conference.

[9] W Z Wahba, A Y A Maghari, "Lossless Image Compression Techniques Comparative Study", International Research Journal of Engineering and Technology, Vol. 3, No. 2, 2016.

[10] D Kaur and K Kaur, "Data Compression on Columnar- Database using Hybrid Approach", International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 5, pp. 889-894, May 2013.

[11] S Oswal, A Singh, K Kumari, "Deflate Compression Algorithm", International Journal of Engineering Research and General Science, Vol. 4, No. 1, January-February, 2016.

[12] Digital Image Processing By Rafael C. Gonzalez and Richard Eugene Woods.

[13] A J Maan, "Analysis and Comparison of Algorithms for Lossless Data Compression", International Journal of Information and Computation Technology, Vol. 3, No. 3, 2013.