Volume 3, Issue 6, June 2013 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Image Compression and its Various Techniques

Dr. Ajit Singh*, Meenakshi Gahlawat
CSE, SES,
BPSMV, India

Abstract— In today's digital world information exchange is been held electronically. So there arises a need for secure transmission of the data. Besides security there are several other factors such as transfer speed, cost, errors transmission etc. that plays a vital role in the transmission process. Image compression is a technique that is very tremendously used for the intact and efficient transfer of data. It not only reduces the size of graphic file to be transferred but at the same time reduces the storage space requirements, cost of the data transferred, and the time required for the transfer. It makes the transmission process faster, provides larger bandwidth and security against illicit use of data. This paper provides a basic introduction about image compression, image coder, various types of image compression techniques and lastly the benefits related with the image compression techniques.

Keywords— Coding redundancy, Interpixel redundancy, Quantizer, Huffman coding, Subband coding

I. INTRODUCTION

Image compression is a technique that deals with reducing the amount of data required to represent a digital image by removing the redundant data. It is been used to minimize the size in bytes of a graphics file without degrading the quality of the image to an unacceptable level. The reduction in file size allows more images to be stored in a given amount of disk or memory space. It also reduces the time required for images to be sent over the Internet or downloaded from Web pages [1]. Image compression and coding techniques tends to eliminate three types of redundancies: coding redundancy, interpixel (spatial) redundancy, and psychovisual redundancy. The way each of them is explored is briefly described below.

- **Coding redundancy:** It is present when less than optimal codewords are used. It is a type of coding that is always reversible and is usually implemented using look-up tables (LUTs). Examples of image coding schemes that explore coding redundancy are the Huffman codes and the arithmetic coding technique.
- **Interpixel redundancy:** This type of redundancy is sometimes called spatial redundancy, interframe redundancy, or geometric redundancy. It is based on the fact that an image very often contains strongly correlated pixels, in other words, large regions whose pixel values are the same or almost the same. This type of redundancy can be expressed in several ways, one is by predicting a pixel value based on the values of its neighboring pixels. In order to do so, the original 2-D array of pixels is usually mapped into a different format, e.g., an array of differences between adjacent pixels. If the original image pixels can be reconstructed from the transformed data set the mapping is said to be reversible. Examples of compression techniques that use interpixel redundancy include: Constant Area Coding (CAC), (1-D or 2-D) Run-Length Encoding (RLE) techniques, and many predictive coding algorithms such as Differential Pulse Code Modulation (DPCM).
- **Psychovisual redundancy:** A bulk of experiments conducted on the psychophysical aspects of human vision have proven that the human eye is unable to respond with equal sensitivity to all incoming visual information; some pieces of information are more important than others. The knowledge of which particular types of information are more or less relevant to the final human user have led to image and video compression techniques that aim at eliminating or reducing any amount of data that is psychovisually redundant. The end result of applying these techniques is a compressed image file, whose size and quality are smaller than the original information, but whose resulting quality is still acceptable for the application at hand [2].

II. A TYPICAL IMAGE CODER

A typical lossy image compression system is shown in Figure given below. It consists of three closely connected components namely (a) Source Encoder (b) Quantizer, and (c) Entropy Encoder.

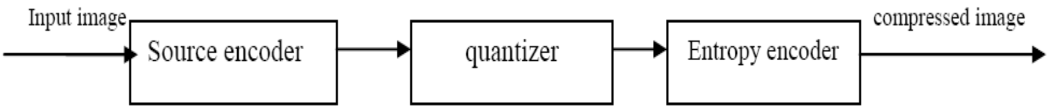

Fig.1 A Typical Image Coder

A. Source Encoder (or Linear Transformer)

Over the years, a variety of linear transforms have been developed which includes Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT) , Discrete Wavelet Transform (DWT) and many more, each with their own advantages and disadvantages.

B. Quantizer

A quantizer simply reduces the number of bits required to store the transformed coefficients by reducing the precision of those values. Since this is a many-to-one mapping, it is a lossy process and is the main source of compression in an encoder. Quantization can be performed on each individual coefficient, which is known as Scalar Quantization (SQ). It can also be performed on a group of coefficients together, and this is known as Vector Quantization (VQ). Both uniform and non-uniform quantizers can be used depending on the problem type.

C. Entropy Encoder

An entropy encoder further compresses the quantized values losslessly to give better overall compression. It uses a model to accurately determine the probabilities for each quantized value and produces an appropriate code based on these probabilities so that the resultant output code stream should be smaller than the input stream. The most commonly used entropy encoders are the Huffman encoder and the arithmetic encoder, although for applications requiring fast execution, simple run-length encoding (RLE) proved to be very effective [2].

III. IMAGE COMPRESSION TECHNIQUES

The image compression techniques are broadly classified into two categories depending on whether an exact replica of the original image could be reconstructed or not using the compressed image.
These are:

1. Lossless technique
2. Lossy technique

A. Lossless compression

It is a class of data compression algorithms that allows the reconstruction of the exact original from the compressed data. Lossless compression reduces a file's size with no loss of quality. This is one of the appropriate methods for reducing file sizes that can be applied to both image and audio files. Lossless compression basically tends to rewrite the data of the original file in a more efficient way. However, because no quality is lost, the resulting files are typically much larger than image and audio files compressed with lossy compression. For example, a file compressed using lossy compression may be one tenth the size of the original, while lossless compression is unlikely to produce a file smaller than half of the original size**.** It is also known as entropy coding since it uses statistics and decomposition techniques to eliminate or minimize redundancy [3].

Following techniques are included in lossless compression:

1. Run length encoding
2. Huffman encoding
3. LZW coding
4. Area coding

B. Lossy compression

It is a compression technique that does not decompress data back to 100% of the original. Lossy methods provide high degrees of compression and result in very small compressed files, but there is a certain amount of loss when they are restored to uncompressed form. Audio, video and some imaging applications can tolerate loss, and in many cases, it may not be noticeable to the human ear or eye. In other cases, it may be noticeable, but not that critical to the application. More the tolerance for loss, the smaller the file can be compressed, and faster the transmission for a file occurs over a network. Examples are MP3, JPEG, and MPEG. Lossy compression is never used for business data and text, which demand a perfect "lossless" restoration [8].

Lossy compression techniques includes following schemes:

1. Transformation coding
2. Vector quantization
3. Fractal coding
4. Block Truncation Coding
5. Subband coding

C. Lossless Compression Techniques

1) Run length encoding

Run Length Encoding (RLE) is a simple and popular data compression algorithm. It is based on the idea to replace a long sequence of the same symbol by a shorter sequence. Run-length encoding (RLE) is a very simple form of data compression in which runs of data (that is, sequences in which the same data value occurs in many consecutive data elements) are stored as a single data value and count, rather than as the original run. This is most useful for 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.

For example, consider a screen containing plain black text on a solid white background. There will be many long runs of white pixels in the blank space, and many short runs of black pixels within the text. Let us take a hypothetical single scan line, with B representing a black pixel and W representing white:

WWWWWWWWWWWWBWWWWWWWWWWWWBBBWWWWWWWWWWWWWWWWWWWWWWWWB
WWWWWWWWWWWWWW

If we apply the run-length encoding (RLE) data compression algorithm to the above hypothetical scan line, we get the following:
12W1B12W3B24W1B14W
This is to be interpreted as twelve Ws, one B, twelve Ws, three Bs, etc. The run-length code represents the original 67 characters in only 18 [4].

2) Huffman encoding

Huffman coding is an entropy encoding algorithm used for lossless data compression. The Huffman code refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been constructed in a particular way based on the estimated probability of occurrence for each possible value of the source symbol. Huffman coding is based on the frequency of occurrence of a data item (pixel in images). It is based on the principle to use a lower number of bits to encode the data that occurs more frequently. Codes are stored in a Code Book which are constructed for each image or a set of images. In all cases the code book plus encoded data must be transmitted to enable decoding [5].

Huffman algorithms have two ranges static as well as adaptive. Static Huffman algorithm is a technique that encodes the data in two passes. In first pass it requires to calculate the frequency of each symbol and in the second pass it constructs the Huffman tree.

Adaptive Huffman algorithm is expanded on Huffman algorithm that constructs the Huffman tree in one pass but take more space than Static Huffman algorithm.

3) LZW coding

If any data file on a computer is viewed, character by character, one would notice that there are many recurring patterns. LZW is a data compression method that takes advantage of this repetition. The original version of the method was created by Lempel and Ziv in 1978 (LZ78) and was further refined by Welch in 1984. Like any adaptive/dynamic compression method, the idea is to first start with an initial model, secondly read data piece by piece and lastly update the model and encode the data as one go along. LZW is a "dictionary"-based compression algorithm, this means that instead of tabularizing character counts and building trees, as done in case of Huffman encoding, LZW encodes data by referencing a dictionary. Thus, to encode a substring, only a single code number, corresponding to that substring's index in the dictionary, needs to be written to the output file. It generally performs best for files with repeated substrings, such as text files. LZW is widely used in computer industry and is implemented as compress command on UNIX [6].

4) Area coding

In area coding technique special codeword's are used to identify large areas of contiguous 1's or 0's. In this method the whole image is divided into blocks of size m*n pixels, which are classified as block having only white pixels, block having only black pixels or block with mixed intensity. The most frequent occurring category is then assigned the 1-bit codeword 0, and the remaining other two categories are assigned with 2-bit codes 10 and 11. The code assigned to the mixed intensity category is used as a prefix, which is followed by the mn-bit pattern of the block. Compression is achieved because the mn bits that are normally used to represent each constant area are replaced by a 1-bit or 2-bit codeword. When predominantly white text documents are being compressed, a slightly simpler approach called white block skipping is used to code the solid white areas as 0 and all other blocks including the solid black blocks are coded as 1 followed by the bit pattern of the block. This approach takes advantage of the anticipated structural patterns of the image to be compressed. As few solid black areas are expected, they are grouped with the mixed regions, allowing a 1-bit codeword to be used for the highly probable white blocks [7].

D. Lossy Compression Technique

1) Transformation coding

Transform domain coding is been used to change the pixels in the original image into frequency domain coefficients called transform coefficients. These coefficients have several desirable properties. Transform coding techniques uses a reversible, linear mathematical transform to map the pixel values onto a set of coefficients, which are then quantized and encoded. The key factor behind the success of transform-based coding schemes depends on many of the resulting coefficients for most natural images which have small magnitudes and can be quantized (or discarded altogether) without causing significant distortion in the decoded image. Different mathematical transforms, such as Fourier (DFT), Walsh-Hadamard (WHT), and Karhunen-Loeve (KLT), have been considered for this task. For compression purposes, the higher the capability of compressing information in fewer coefficients, the better is the transform obtained and for that reason, the Discrete Cosine Transform (DCT) has become the most widely used transform coding technique [9].

2) Vector quantization

A vector is usually defined as a block of pixel values. The basic idea behind the technique is to develop a dictionary of fixed-size vectors, called code vectors. Vector quantization, also called "block quantization" or "pattern matching quantization" is a technique commonly used in lossy data compression. It works by encoding values from a multidimensional vector space into a finite set of values from a discrete subspace of lower dimension. A lower-space vector requires less storage space, so the data is compressed. Due to the density matching property of vector quantization, the compressed data contains errors that are inversely proportional to density. The transformation is usually carried out by projection or by using a codebook. In some cases, a codebook can also be used to entropy code the discrete value in the same step, by generating a prefix coded variable-length encoded value as its output [10], [11].

3) Fractal Coding

Fractal encoding is a mathematical process that is been used to encode bitmaps containing a real-world image as a set of mathematical data that describes the fractal properties of the image. Fractal encoding relies on the fact that all natural, and most artificial, objects contain redundant information in the form of similar, repeating patterns called fractals. Fractal encoding is largely used for converting bitmap images to fractal codes. Fractal decoding is just the reverse, where a set of fractal codes are converted into a bitmap. The encoding process is extremely computationally intensive. Millions of iterations are required to find the fractal patterns in an image. Depending upon the resolution and contents of the input bitmap data, and output quality, compression time, and file size parameters are selected, compressing a single image could take time from a few seconds to a few hours (or more) on even a very fast computer. Decoding a fractal image is a much simpler process. The hard work was performed finding all the fractals during the encoding process. All the decoding process needs to do is to interpret the fractal codes and translate them into a bitmap image.

Currently, the most popular method of fractal encoding is a process called the Fractal Transform created in 1988 by Michael F. Barnsley of Iterated Systems. Barnsley's transform was the first practical algorithm that mathematically describe a real-world bitmap image in terms of its fractal properties.

Two tremendous benefits are immediately realized by converting conventional bitmap images to fractal data. The first is the ability for scaling any fractal image up or down in size without the introduction of image artefacts or a loss in detail that occurs in bitmap images. This process of "fractal zooming" is independent of the resolution of the original bitmap image, and the zooming is limited only by the amount of available memory in the computer. The second benefit is the fact that the size of the physical data used to store fractal codes is much smaller than the original bitmap data size. In fact, it is not uncommon for fractal images to be more than 100 times smaller than their bitmap sources. It is this aspect of fractal technology, called fractal compression that has promoted the greatest interest within the computer imaging industry.

Fractal compression is lossy. The process of matching fractals does not involve looking for exact matches, but instead looking for "best fit" matches based on the compression parameters such as encoding time, image quality, and size of output. But the encoding process is controllable to the point where the image is "visually lossless". That is, one is unable to notice where the data was lost. Fractal compression differs from other lossy compression methods, such as JPEG, in a number of ways. JPEG achieves compression by discarding image data that is not required for the human eye to perceive the image. The resulting data is then further compressed using a lossless method of compression. In order to achieve greater compression ratios, more image data must be discarded, resulting in a poorer quality image with a blocky appearance. Fractal images are not based on a map of pixels, neither the encoding is weighted to the visual characteristics of the human eye. Instead, there arises the need of discarding the bitmap when it is required to create a best-fit fractal pattern. Greater compression ratios are achieved using greater computationally intensive transforms that may degrade the image, but the distortion appears much more natural due to the fractal components. Most other lossy methods are also symmetrical in nature. That is, a particular sequence of steps is used to compress an image, and the reverse of those steps is used to decompress it. Compression and decompression will take about the same amount of time as well. Fractal compression is an asymmetrical process, that takes large amount of time to compress an image than to decompress it. This characteristic limits the usefulness of fractally compressed data to applications where image data is constantly decompressed but never recompressed. Fractal compression is therefore highly suited for use in image databases and CD-ROM applications [12].

4) Block truncation coding

Block truncation coding (BTC) is a simple and fast lossy compression technique designed for digitized gray scale images. It was originally introduced by Delp and Mitchell. The key idea behind BTC is to perform moment preserving (MP) quantization for blocks of pixels so that the quality of the image will remain acceptable and at the same time the storage space demand also remains decreased. Even if the compression gain of the algorithm is inferior to the standard JPEG compression algorithm, BTC has gained popularity due to its practical usefulness [13].

In this scheme, the image is divided into non overlapping blocks of pixels. For each block, threshold and reconstruction values needs to be determined. The threshold value is usually defined as the mean of pixel values in the block. Then a bitmap of the block is derived by replacing all pixels whose values are greater than or equal or less than to the threshold by a 1(0). Then for each of the segment containing group of 1s and 0s in the bitmap, the reconstruction value is been determined. This is expressed as the average of the values of the corresponding pixels in the original block [11].

5) Subband coding

Subband coding is a technique that decomposes the input signal into different frequency bands. After the input is decomposed to its constituents, the coding technique that best suits to each constituent for improving the compression performance is been employed [14]. To enable higher quality compression, one may use subband coding. First, a digital filter bank divides the input signal spectrum into some number (e.g., 32) of subbands. The psychoacoustic model then looks at the energy in each of these subbands, as well as in the original signal, and then it computes masking thresholds using psychoacoustic information. Each of the subband samples is quantized and encoded so as to keep the quantization noise below the dynamically computed masking threshold. The final step is to format all these quantized samples into groups of data called frames, to facilitate eventual playback by a decoder. Decoding is much easier than encoding, since no psychoacoustic model is involved. The frames are unpacked, subband samples are decoded, and a frequency-time mapping is used that is used to reconstructs an output signal [15].

IV. BENEFITS OF COMPRESSION

A. Storage Space

Compressing data files allows one to store more files in the storage space that is available. Lossless compression, used in zip file technology, will typically reduce a file to 50 percent of its original size. However, difference in the file size is not seen if the zip files are already in a compressed format, such as MP3 audio files or PDF (Portable Document Format) text-only files [16].

B. Bandwidth and Transfer Speed

The download process uses network bandwidth whenever we download a file, such as an MP3 audio file, from a server on the Internet. Bandwidth is defined as the speed at which the network transfers data and is measured in Mbps (megabits per second). Compressed files contain fewer "bits" of data than uncompressed files, and, as a consequence, use less bandwidth when we download them. This means that the transfer speed, that is the time it takes to download a file, is quicker. It will take 10 seconds to download a file if bandwidth of 1Mbps is available, and for downloading a file that is 10Mb (megabits) in size, it will only take 5 seconds to download the file if the file is compressed to 5Mb [16].

C. Cost

The costs of storing the data are reduced by compressing the files for storage because more files can be stored in available storage space when they are compressed. We will need to buy a second 250MB drive if we have 500MB (megabytes) of uncompressed data and a 250MB hard drive on which to store it. You will not need to buy the extra hard drive if you compress the data files to 50 percent of their uncompressed size [16].

D. Accuracy

It also reduces the chance of transmission errors since fewer bits are transferred [11].

E. Security

It also provides a level of security against illicit monitoring [11].

V. CONCLUSIONS

Firstly an introduction to image compression has been presented. After it typical image coder is been discussed, its various components and the basic diagram. Thirdly, image compression techniques namely lossless image compression and lossy image compression with its various subtypes are presented. Lastly various benefits of image compression is been discussed.

ACKNOWLEDGMENT

I would like to give my sincere gratitude to my guide Dr. Ajit Singh who guided me throughout, to complete this topic.

REFERENCES

[1] http://searchcio-midmarket.techtarget.com/definition/image-compression
[2] http://encyclopedia.jrank.org/articles/pages/6760/Image-Compression-and-Coding.html
[3] http://www.techterms.com/definition/lossless
[4] http://en.wikipedia.org/wiki/Run-length_encoding
[5] (*www.cs.cf.ac.uk/Dave/Multimedia/node210.html*).
[6] http://www.cs.duke.edu/csed/curious/compression/lzw.html**)**
[7] http://cis.cs.technion.ac.il/Done_Projects/Projects_done/VisionClasses/DIP_1998/Lossless_Compression/node26.html
[8] http://www.pcmag.com/encyclopedia/term/46335/lossy-compression
[9] http://encyclopedia.jrank.org/articles/pages/6760/Image-Compression-and-Coding.htm
[10] http://en.wikipedia.org/wiki/Vector_quantization**)**
[11] *rimtengg.com/coit2007/proceedings/pdfs/43.pdf*
[12] http://www.fileformat.info/mirror/egff/ch09_09.htm**)**
[13] ftp://cs.uku.fi/franti/papers/SURVEY.pdf
[14] www.ece.mcmaster.ca/~shirani/multi12/subband.pdf***)***
[15] http://en.wikipedia.org/wiki/Sub-band_coding
[16] *www.ehow.com ›* Internet

