A SURVEY ON VARIOUS IMAGE COMPRESSION TECHNIQUES

Bhishm Tripathi

(Roll No.- 109CS0599)

Department of Computer Science & Engineering

National Institute of Technology

Rourkela-769008, Odisha, India

May, 2014

A SURVEY ON VARIOUS IMAGE COMPRESSION TECHNIQUES

Thesis submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

In

Computer Science & Engineering

By

Bhishm Tripathi

(Roll No.- 109CS0599)

Under the guidance of

Prof. Ramesh Kumar Mohapatra

Department of Computer Science & Engineering

National Institute of Technology

Rourkela-769008, Odisha, India

May, 2014

Dedicated to my parents and teachers

Contents

List of Figures

Department of Computer Science & Engineering

National Institute of Technology

Rourkela-769008, Odisha, India

Certificate

This is to certify that the work in the thesis entitled ***"A Survey On Various Image Compression Techniques"*** submitted by ***Bhishm Tripathi*** is a record of original research work carried out by him under our supervision and guidance in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering, National Institute of Technology, Rourkela. Neither this thesis nor any part of it has been submitted for any degree or academic award elsewhere.

Place: NIT Rourkela **Prof.Ramesh Kumar Mohapatra**

Assistant Professor

Department of CSE

Acknowledgement

First of all, I would like to express my deep sense of respect and gratitude towards my supervisor Prof. Ramesh Kumar Mohapatra, who has been the guiding force behind this work. I want to thank him for introducing me to the field of image compression and giving me the opportunity to work under him. His undivided faith in this topic and ability to bring out the best of analytical and practical skills in people has been invaluable in tough periods. Without his invaluable advice and assistance it would not have been possible for me to complete this thesis. I am greatly indebted to him for his constant encouragement and constant advice in every aspect of my academic life. I consider it my good fortune to have got an opportunity to work with such a wonderful person.

I also thank Prof. Ashok Kumar Turuk, Prof. Ratnakar Dash for serving on my scrutiny committee.

During my studies at NIT Rourkela, I made many friends. I would like to thank them all, for all the great moments I had with them.

When I look back at my accomplishments in life, I can see a clear trace of my family's concerns and devotion everywhere. My dearest mother, whom I owe everything I have achieved and whatever I have become, my beloved father, for always believing in me and inspiring me to dream big even at the toughest moments of my life, and my brother; who was always my silent support during all the hardships of this endeavor and beyond.

Bhishm Tripathi

Abstract

Image compression address the problem of reducing the amount of data required to represent a digital image with no significant loss of information. Storage problems, plus the desire to exchange images over the Internet, have lead to a large interest in image compression algorithms.

There are basically two types of compression techniques. One is Lossless Compression and other is Lossy Compression Technique. Comparing the performance of compression technique is difficult unless identical data sets and performance measures are used. Some of these techniques are obtained good for certain applications like security technologies. Some techniques perform well for certain classes of data and poorly from others thus techniques could be applied for certain data sets.

Introduction

What is Image Compression?

Data compression as a discipline has its origin in information theory. The work done by Bell Labs in the late 1940's addressed the problem of redundancy in data as it applied to message communication. Since then, a number of techniques have been developed in an effort to achieve maximum information content with minimum storage requirements for various data. Many techniques deal with data compression as a two stage process in that an efficient coding machine can be realized only through accurate modeling of the input data. Early information theory approaches to these problems resulted in the use of variable length codes based on a symbol probability table such as in Huffman and Shannon-Fano coding techniques. This type of statistical modeling formed the basis for future coding and modeling strategies and made way for future algorithms such the LZ family of compression schemes.
These types of compression algorithms operate under the constraint that data is conserved and are thus lossless.
Another alternative to lossless data compression involves accepting information loss as a trade-off to achieving greater compression ratios. These lossy techniques find their greatest utility in the signal processing areas of digitized image and audio. The Joint Photographic Experts Group (PE standardized the concept of transforming image the frequency domain in order to filter out noisy components in an effort to achieve greater compression.

Another approach involves using the unique properties of fractal generation to compress images. Fractal compression involves using an iterative technique & exploit redundancy in an image. Both PEG and fractal compression hold the potential for much progress in compression ratio gains over more traditional lossless techniques.

Although computational speeds have been a bottleneck in data compression, current computing technology has shifted the focus away

from speed and toward storage capacity limitations. The memory cost involved with the archiving of photographs and documents has become a motive behind the development of efficient information reduction of such data. As the technology in image processing systems advances, such as greater resolutions and color definitions, the information stored in image files increases dramatically.
In my work an analysis of image data storage is presented which includes examining the Graphics Interchange Format (GIF), the JPEG format, and a fractal technique for image compression. Several standard images are compressed in the respective file formats, and the results are compared with particular attention given to information loss and file length. *LZ Compression*

In 1984 Terry Welch adapted the LZ78 algorithm for use with high speed hard drive controllers. This algorithm uses a 4k dictionary as before but the initial 256 entries of the dictionary are initialized with individual bytes, and the remaining area is used for storing strings. A variation of the LZW algorithm is the core of the UNIX *compress* program. This type of compression was also incorporated in the V.42bis standard by the International Consultative Committee on Telephony and Telegraphy (CCITT) for use with modem communication. It can be implemented in either software or hardware.

Run-length encoding is a data compression algorithm that is supported by most bitmap file formats, such as TIFF, BMP, and PCX. RLE is suited for compressing any type of data regardless of its information content, but the content of the data will affect the compression ratio achieved by RLE. Although most RLE algorithms cannot achieve the high compression ratios of the more advanced compression methods, RLE is both easy to implement and quick to execute, making it a good alternative to either using a complex compression algorithm or leaving your image data uncompressed.

RLE works by reducing the physical size of a repeating string of characters. This repeating string, called a run, is typically encoded into

two bytes. The first byte represents the number of characters in the run and is called the run count. In practice, an encoded run may contain 1 to 128 or 256 characters; the run count usually contains as the number of characters minus one (a value in the range of 0 to 127 or 255). The second byte is the value of the character in the run, which is in the range of 0 to 255, and is called the run value.

Uncompressed, a character run of 15 A characters would normally require 15 bytes to store:

```
AAAAAAAAAAAAAAA
```

The same string after RLE encoding would require only two bytes:

```
15A
```

The 15A code generated to represent the character string is called an RLE packet. Here, the first byte, 15, is the run count and contains the number of repetitions. The second byte, A, is the run value and contains the actual repeated value in the run.

A new packet is generated each time the run character changes, or each time the number of characters in the run exceeds the maximum count. Assume that our 15-character string now contains four different character runs:

```
AAAAAAbbbXXXXXt
```

Using run-length encoding this could be compressed into four 2-byte packets:

```
6A3b5X1t
```

Thus, after run-length encoding, the 15-byte string would require only eight bytes of data to represent the string, as opposed to the original 15 bytes. In this case, run-length encoding yielded a compression ratio of almost 2 to 1.

Long runs are rare in certain types of data. For example, ASCII plaintext seldom contains long runs. In the previous example, the last run (containing the character t) was only a single character in length; a 1-

character run is still a run. Both a run count and a run value must be written for every 2-character run. To encode a run in RLE requires a minimum of two characters worth of information; therefore, a run of single characters actually takes more space. For the same reasons, data consisting entirely of 2-character runs remains the same size after RLE encoding.

In my example, encoding the single character at the end as two bytes did not noticeably hurt our compression ratio because there were so many long character runs in the rest of the data. But observe how RLE encoding doubles the size of the following 14-character string:

```
Xtmprsqzntwlfb
```

After RLE encoding, this string becomes:

```
1X1t1m1p1r1s1q1z1n1t1w1l1f1b
```

RLE schemes are simple and fast, but their compression efficiency depends on the type of image data being encoded. A black-and-white image that is mostly white, such as the page of a book, will encode very well, due to the large amount of contiguous data that is all the same color. An image with many colors that is very busy in appearance, however, such as a photograph, will not encode very well. This is because the complexity of the image is expressed as a large number of different colors. And because of this complexity there will be relatively few runs of the same color.

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 amount 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 (that's where the statistical part comes in). Code word lengths are no

longer fixed like ASCII. Code word lengths vary and will be shorter for the more frequently used characters.

The procedure meets expectations by making a twofold tree of hubs. These can be put away in a standard exhibit, the span of which relies on upon the quantity of images, . A hub can be either a leaf hub or an inward hub.

At first, all hubs are leaf hubs, which contain the image itself, the weight (recurrence of appearance) of the image and alternatively, a connection to a guardian hub which makes it simple to peruse the code (in opposite) beginning from a leaf hub. Inward hubs contain image weight, connections to two tyke hubs and the discretionary connection to a guardian hub. As a typical tradition, bit "0" speaks to emulating the left tyke and bit "1" speaks to taking after the right kid. A completed tree has up to leaf hubs and interior hubs. A Huffman tree that excludes unused images produces ideal code lengths.

The methodology basically starts with the leaf hubs containing the probabilities of the image they speak to, then another hub whose kids are the 2 hubs with littlest likelihood is made, such that the new hub's likelihood is equivalent to the aggregate of the youngsters' likelihood. With the past 2 hubs consolidated into one hub (subsequently not thinking of them as any longer), and with the new hub being currently viewed as, the system is rehashed until stand out hub remains, the Huffman tree. The easiest development calculation utilizes a need line where the hub with most reduced likelihood is given most noteworthy need:

1. Make a leaf hub for every image and add it to the need line.

2. While there is more than one hub in the line:

1. Uproot the two hubs of most noteworthy need (least likelihood) from the line

2. Make another inward hub with these two hubs as youngsters and with likelihood equivalent to the entirety of the two hubs' probabilities.

3. Add the new hub to the line.

3. The remaining hub is the root hub and the tree is finished.

Since proficient need line information structures oblige O(log n) time every insertion, and a tree with n leaves has 2n−1 hubs, this calculation works in O(nlog n) time, where n is the quantity of symbols.if the images are sorted by likelihood, there is a straight time (O(n)) strategy to make a Huffman tree utilizing two lines, the first containing the starting weights (alongside pointers to the related leaves), and consolidated weights (alongside pointers to the trees) being placed in the once again of the second line. This guarantees that the most reduced weight is constantly kept at the front of one of the two lines:

1. Begin with the same number of leaves as there are images.

2. Enqueue all leaf hubs into the first line (by likelihood in expanding request so that the most outlandish thing is in the leader of the line).

3. While there is more than one hub in the lines:

1. Dequeue the two hubs with the least weight by inspecting the fronts of both lines.

2. Make another inner hub, with the two recently evacuated hubs as youngsters (either hub can be either tyke) and the aggregate of their weights as the new weight.

3. Enqueue the new hub into the back of the second line.

4. The remaining hub is the root hub; the tree has now been produce.

Although this algorithm may appear "faster" complexity-wise than the previous algorithm using a priority queue, this is not actually the case because the symbols need to be sorted by probability before-hand, a process that takes O(n log n) time in itself.

As a rule, time many-sided quality is not imperative in the decision of calculation here, since n here is the quantity of images in the letters in order, which is normally a little number (contrasted with the length of

the message to be encoded); though multifaceted nature investigation concerns the conduct when n develops to be extensive.

It is by and large valuable to minimize the fluctuation of codeword length. For instance, a correspondence cushion accepting Huffman-encoded information may need to be bigger to manage particularly long images if the tree is particularly lopsided. To minimize fluctuation, just break ties between lines by picking the thing in the first line. This alteration will hold the numerical optimality of the Huffman coding while both minimizing difference and minimizing the length of the longest character code.

In science, a wavelet arrangement is a representation of a square-integrable (genuine or complex-esteemed) work by a certain orthonormal arrangement created by a wavelet. These days, wavelet change is a standout amongst the most mainstream applicants of the time-recurrence changes. My work gives a formal, numerical meaning of an orthonormal wavelet and of the necessary wavelet change.

Wavelet packing is a manifestation of information pressure appropriate for picture layering (now and again likewise feature clamping and sound squeezing). Eminent executions are JPEG 2000, Djvu and ECW for still pictures, REDCODE, Cineform, the BBC's Dirac, and Ogg Tarkin for feature. The objective is to store picture information in as meager space as could reasonably be expected in a document. Wavelet squeezing can be either lossless or lossy.[1]

Utilizing a wavelet change, the wavelet packing systems are satisfactory for speaking to homeless people, for example, percussion sounds in sound, or high-recurrence parts in two-dimensional pictures, for instance a picture of stars on a night sky. This implies that the transient components of an information sign can be spoken to by a littler measure of data than would be the situation if some other change, for example, the more far reaching discrete cosine change, had been utilized.

Wavelet packing is bad for different varieties of information: transient sign qualities mean great wavelet clamping, while smooth, intermittent signs are better packed by different systems, especially customary symphonious layering (recurrence space, as by Fourier changes and related).

Vector quantization (VQ) is a traditional quantization method from sign preparing which permits the displaying of likelihood thickness works by the circulation of model vectors. It was initially utilized for information clamping. It meets expectations by partitioning a vast set of focuses (vectors) into gatherings having more or less the same number of focuses closest to them. Each one gathering is spoken to by its centroid point, as in k-means and some other grouping calculations.

The thickness matching property of vector quantization is influential, particularly for distinguishing the thickness of expansive and high-dimensioned information. Since information focuses are spoken to by the file of their closest centroid, usually happening information have low mistake, and uncommon information high slip. This is the reason VQ is suitable for lossy information packing. It can likewise be utilized for lossy information redress and thickness estimation.

Vector quantization is focused around the aggressive learning ideal model, so it is nearly identified with the organizing toward oneself guide model.

Vector quantization, additionally called "piece quantization" or "example matching quantization" is frequently utilized as a part of lossy information squeezing. It lives up to expectations by encoding qualities from a multidimensional vector space into a limited set of qualities from a discrete subspace of lower measurement. A lower-space vector requires less storage room, so the information is packed. Because of the thickness matching property of vector quantization, the packed information has lapses that are contrarily relative to thickness.

The change is normally done by projection or by utilizing a codebook. Sometimes, a codebook can be additionally used to entropy code the discrete esteem in the same venture, by producing a prefix coded variable-length encoded esteem as its yield.

The set of discrete adequacy levels is quantized mutually as opposed to each one example being quantized independently. Consider a k-dimensional vector of plentifulness levels. It is compacted by picking the closest matching vector from a set of n-dimensional vectors , with n < k.

All conceivable mixes of the n-dimensional vector structure the vector space to which all the quantized vectors have a place.

Just the list of the codeword in the codebook is sent rather than the quantized qualities. This moderates space and attains more packing.

Twin vector quantization (VQF) is a piece of the MPEG-4 standard managing time space weighted interleaved vector quantization.

In the last few years, subband coding (SBC) has become one of the major techniques for image [l] and video compression *[a]*. There are basically two approaches for quantization of a subband decomposed image: either the subbands are quantized independently, or the dependencies between the subbands are taken into account. I have developed a SBC technique based on a block-zero tree coding (BZTC) that exploits the inter-band information. The proposed coding scheme is a generalization of the Zero Tree Coding (ZTC) algorithm [3]. The ZTC technique was developed in recognition of the difficulty in achieving efficient bit rate reductions for representing the significant coefficients via significance prediction. The basic idea of the BZTC algorithm is that sub bands are divided into blocks and their significance is represented by a tree coding technique. The BZTC algorithm is a very efficient coding scheme since ever lock at a given sub band level may have dependency in a set of blocks at the lower level sub bands of similar orientation. A

set of state transition rules are utilized for presenting the significance block map. Vector quantization is applied to the significant blocks using a multiband codebook. The encoder can terminate the coding process at any time whenever the desired target rate or distortion is achieved. This is possible since the bit stream is generated in an embedded fashion.

Emerging Applications of Image Compression

Uncompressed images require considerable storage capacity and very high bandwidth in transfer. In order to manage large image data objects efficiently, these objects need to be compressed to reduce the file size. Compression tries to eliminate redundancies in the pattern of data [1] [6]. Many studies have shown that users at a computer have a patience factor ranging from two to four seconds. Even though network speeds have been increasing consistently, very large image objects can take as much as a couple seconds to transmit. Given just a few seconds to retrieve, transmit, and display an image, improving the efficiency of storage and transmission of data objects is of paramount importance to image systems. Most solutions for this problem focus either on increasing network bandwidth by using a better network, using a better compression algorithm or employing other techniques to get higher performance [2,4,7,8].

The many benefits of image compression include less required storage space, quicker sending and receiving of images, and less time lost on image viewing and loading. But where and how is image compression used today?

Just as image compression has increased the efficiency of sharing and viewing personal images, it offers the same benefits to just about every

industry in existence. Early evidence of image compression suggests that this technique was, in the beginning, most commonly used in the printing, data storage, and telecommunications industries. Today however, the digital form of image compression is also being put to work in industries such as fax transmission, satellite remote sensing, and high definition television, to name but a few.

In certain industries, the archiving of large numbers of images is required. A good example is the health industry, where the constant scanning and/or storage of medical images and documents take place. Image compression offers many benefits here, as information can be stored without placing large loads on system servers. Depending on the type of compression applied, images can be compressed to save storage space, or to send to multiple physicians for examination. And conveniently, these images can uncompressed when they are ready to be viewed, retaining the original high quality and detail that medical imagery demands.

Image compression is also useful to any organization that requires the viewing and storing of images to be standardized, such as a chain of retail stores or a federal government agency. In the retail store example, the introduction and placement 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. Federal government agencies that standardize their image viewing, storage and transmitting processes can eliminate large amounts of time spent in explanation and problem solving. The time they save can then be applied to issues within the organization, such as the improvement of government and employee programs.

In the security industry, image compression can greatly increase the efficiency of recording, processing and storage. However, in this application it is imperative to determine whether one compression standard will benefit all areas. For example, in a video networking or closed-circuit television application, several images at different frame

rates may be required. Time is also a consideration, as different areas may need to be recorded for various lengths of time. Image resolution and quality also become considerations, as does network bandwidth, and the overall security of the system.

Museums and galleries consider the quality of reproductions to be of the utmost importance. Image compression, therefore, can be very effectively applied in cases where accurate representations of museum or gallery items are required, such as on a Web site. Detailed images that offer short download times and easy viewing benefit all types of visitors, from the student to the discriminating collector. Compressed images can also be used in museum or gallery kiosks for the education of that establishment's visitors. In a library scenario, students and enthusiasts from around the world can view and enjoy a multitude of documents and texts without having to incur traveling or lodging costs to do so.

Regardless of industry, image compression has virtually endless benefits wherever improved storage, viewing and transmission of images are required. And with the many image compression programs available today, there is sure to be more than one that fits any requirements best.

Issues in Image Compression

Size Reduction

- File size lessening remains the absolute most critical profit of picture layering. Contingent upon what document sort you're working with, you can keep on compacting the picture until its at your craved size. This implies the picture consumes up less room on the hard drive and holds the same physical size, unless we alter the picture's physical size in a picture proofreader. This record size decrease meets expectations gloriously for the Internet, permitting website admins to make picture rich destinations without utilizing much transmission capacity or storage too.

Slow Devices

- Some electronic gadgets, for example, machines or cams, may stack huge, uncompressed pictures gradually. Album drives, for instance, can just read information at a particular rate and can't show huge pictures progressively. Likewise, for a few webhosts that exchange information gradually, packed pictures stay fundamental for a completely practical site. Different types of capacity mediums, for example, hard drives, will likewise experience issues stacking uncompressed records rapidly. Picture squeezing considers the quicker stacking of information on slower gadget.

Degradation

- When you clamp a picture, in some cases you will get picture debasement, importance the nature of the picture has declined. In the event that sparing a GIF or PNG document, the information stays despite the fact that the nature of the picture has declined. On the off chance that we have to demonstrate a high-determination picture to somebody, expansive or little, we will discover picture squeezing as a drawback.

Data Loss

- With some regular document sorts, for example, JPEG, when a picture recoils in size the clamping system will toss a portion of the photograph's information for all time. To pack these pictures, we need to guarantee you had an uncompressed reinforcement before beginning. Else, we will lose the high caliber of the first uncompressed picture for all time.

Simulations and Results

Run Length Encoding:

This is a very simple compression method used for sequential data. It is very useful in case of repetitive data. This technique replaces sequences of identical symbols (pixels), called runs by shorter symbols. The run length code for a gray scale image is represented by a sequence { Vi , Ri } where Vi is the intensity of pixel and Ri refers to the number of consecutive pixels with the intensity Vi.

A "run: of consecutive pixels whose gray levels are identical is replaced with two values: the length of the run and the gray level of all pixels in the run. Exampe (50, 50,50,50) becomes (4,50)

Especially suited for synthetic images containing large homogeneous regions . The encoding process is effective only if there are sequences of 4 or more repeating characters

Applications – compression of binary images to be faxed.

Example:

89 89 89 76 76 76 can be written as (89,3),(76,3).

HUFFMAN ENCODING

This is a general technique for coding symbols based on their statistical occurrence frequencies (probabilities). The pixels in the image are treated as symbols. The symbols that occur more frequently are assigned a smaller number of bits, while the symbols that occur less frequently are assigned a relatively larger number of bits. Huffman code is a prefix code. This means that the (binary) code of any symbol is not the prefix of the code of any other symbol. Most image coding standards use lossy techniques in the earlier stages of compression and use Huffman coding as the final step.

Properties of Huffman encoding are:

1. Ranking pixel values in decreasing order of their probability
2. Pair the two values with the lowest probabilities, labeling one of them with 0 and other with 1.
3. Link two symbols with lowest probabilities .
4. Go to step 2 until you generate a single symbol which probability is 1.
5. Trace the coding tree from a root.

LZW CODING

LZW (Lempel- Ziv – Welch) is a dictionary based coding. Dictionary based coding can be static or dynamic. In static dictionary coding, dictionary is fixed during the encoding and decoding processes. In dynamic dictionary coding, the dictionary is updated on fly. LZW is widely used in computer industry and is implemented as compress command on UNIX.

The methods of the first group try to find if the character sequence currently being compressed has already occurred earlier in the input data and then, instead of repeating it, output only a pointer to the earlier occurrence.

The algorithms of the second group create a dictionary of the phrases that occur in the input data. When they encounter a phrase already present in the dictionary, they just output the index number of the phrase in the dictionary.

TRANSFORMATION CODING

In this coding scheme, transforms such as DFT (Discrete Fourier Transform) and DCT (Discrete Cosine Transform) are used to change the pixels in the original image into frequency domain coefficients (called transform coefficients). These coefficients have several desirable properties. One is the energy compaction property that results in most of the energy of the original data being concentrated in only a few of the significant transform coefficients. This is the basis of achieving the compression. Only those few significant coefficients are selected and the remaining are discarded. The selected coefficients are considered for further quantization and entropy encoding. DCT coding has been the most common approach to transform coding. It is also adopted in the JPEG image compression standard.

Output of Transformation coding are:

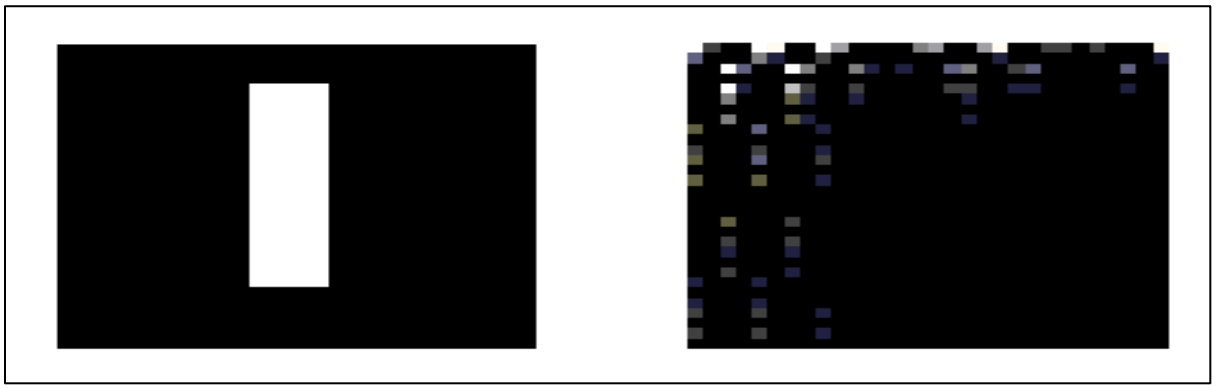

Fig.1

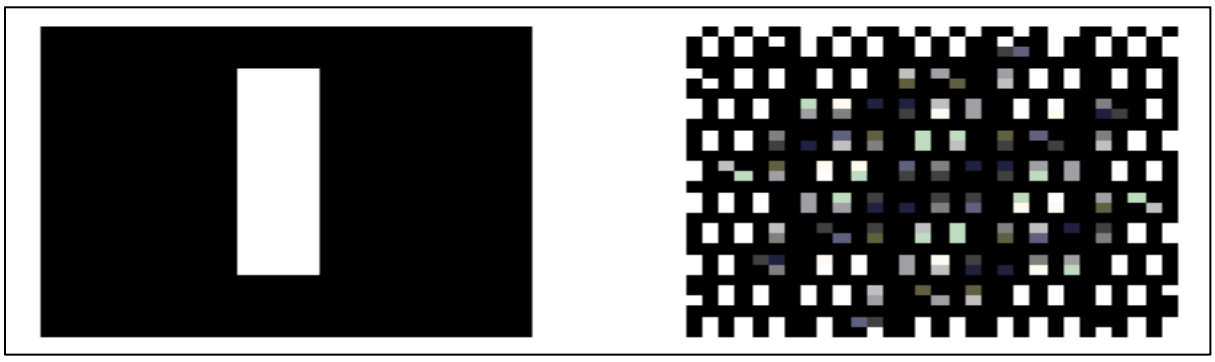

Fig.2

VECTOR QUANTIZATION

The basic idea in this technique is to develop a dictionary of fixed-size vectors, called code vectors. A vector is usually a block of pixel values. A given image is then partitioned into non-overlapping blocks (vectors) called image vectors. Then for each in the dictionary is determined and its index in the dictionary is used as the encoding of the original image vector. Thus, each image is represented by a sequence of indices that can be further entropy coded.

Quantization is the step where we actually throw away data. The objective of quantization is to reduce the precision and to achieve higher compression ratio.

Output of Vector Quantization coding is:

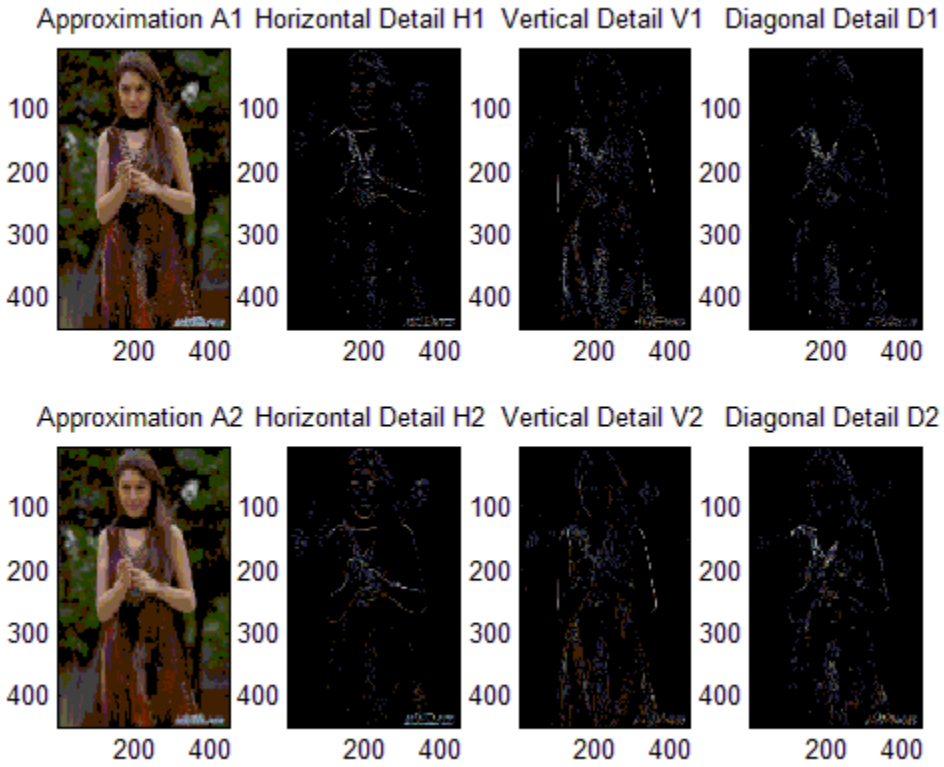

Fig.3

BLOCK TRUNCATION CODING

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

Output of Block Truncation coding are:

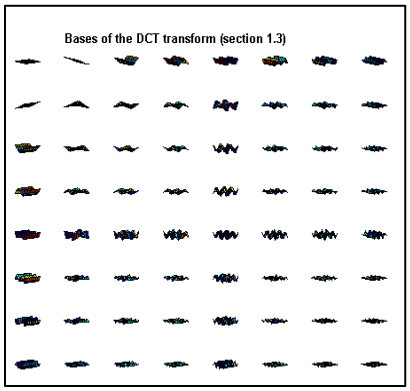

Fig.4

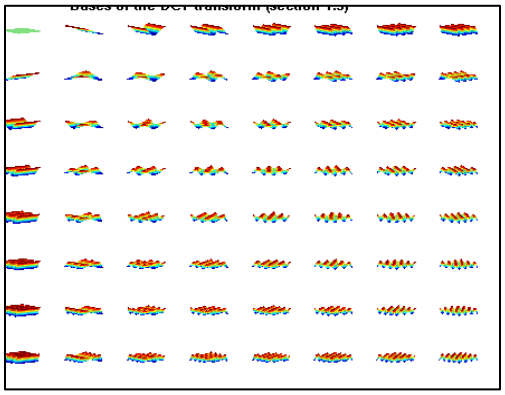

Fig.5

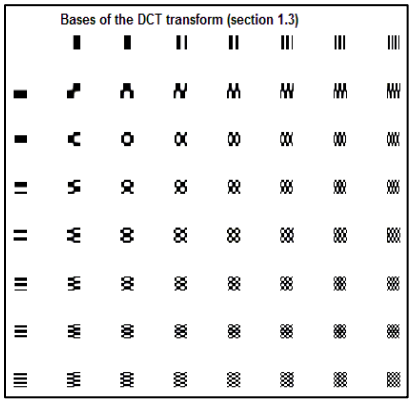

Fig.6

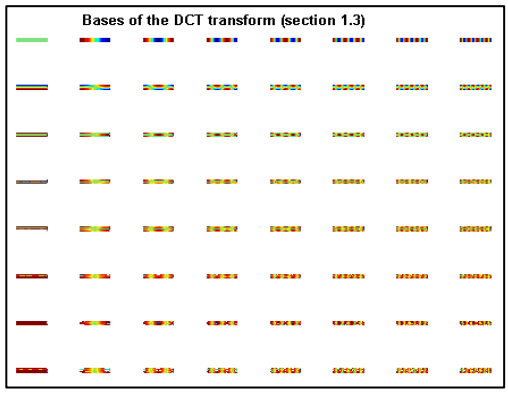

Fig.7

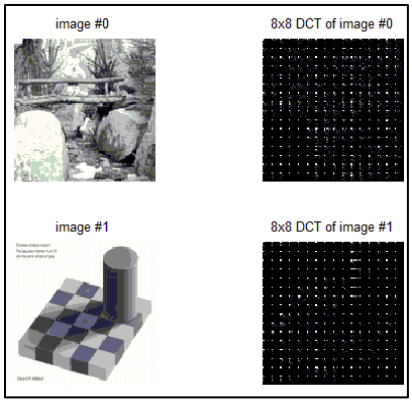

Fig.8

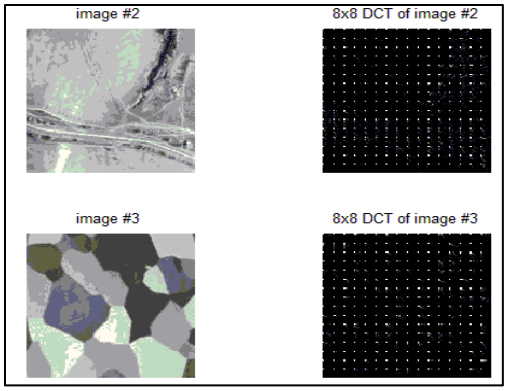

Fig.9

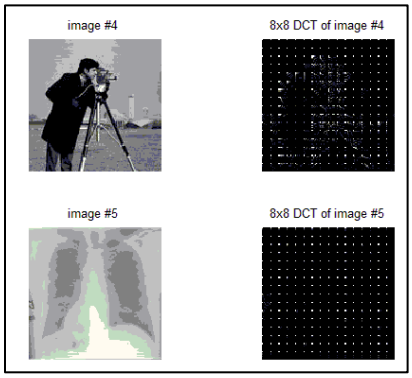

Fig.10

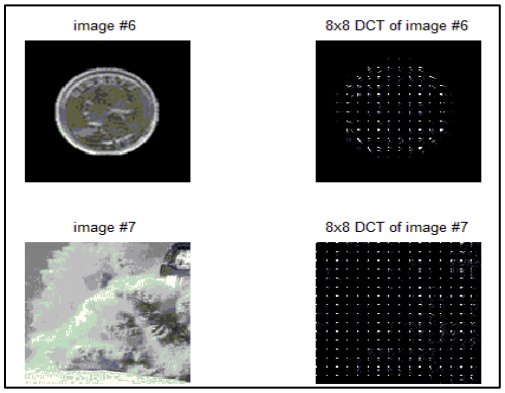

Fig.11

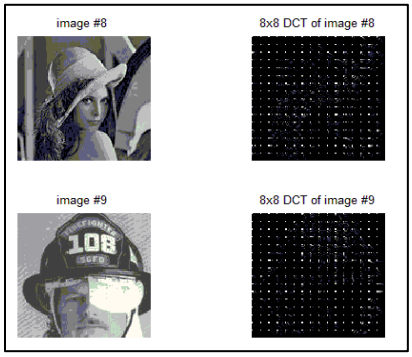

Fig.12

Power of DCT coefficients (section 1.6)

1 95.5db	2 52.6db	4 42.3db	9 33.9db	12 29.0db	19 22.5db	31 15.8db	42 9.6db
3 51.2db	5 40.8db	7 34.7db	13 28.9db	18 24.3db	24 20.2db	35 13.4db	39 10.6db
6 40.2db	10 33.3db	11 29.6db	16 25.0db	22 21.3db	30 16.3db	37 11.3db	45 9.3db
8 34.2db	15 27.4db	17 24.8db	23 20.9db	27 17.3db	33 14.3db	46 8.8db	51 4.3db
14 28.2db	21 22.3db	25 19.7db	29 16.7db	34 14.2db	44 9.3db	50 5.0db	56 -0.8db
20 22.4db	28 17.1db	32 15.6db	38 10.9db	41 10.1db	49 6.0db	55 0.5db	58 -3.3db
26 17.7db	36 12.4db	43 9.6db	48 7.1db	52 2.9db	57 -1.6db	59 -4.0db	61 -7.4db
40 10.5db	47 7.8db	53 1.9db	54 1.3db	60 -4.2db	62 -8.5db	63 -12.1db	64 -15.3db

Fig.13

Fig.14

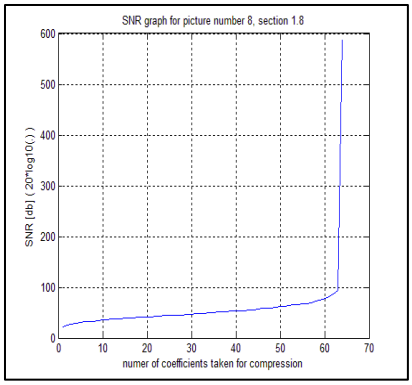

Fig.15

SUB BAND CODING

In this scheme, the image is analyzed to produce the components containing frequencies in well-defined bands, the sub bands. Subsequently, quantization and coding is applied to each of the bands. The advantage of this scheme is that the quantization and coding well suited for each of the sub bands can be designed separately.

Output of Sub Band coding are:

Fig.16

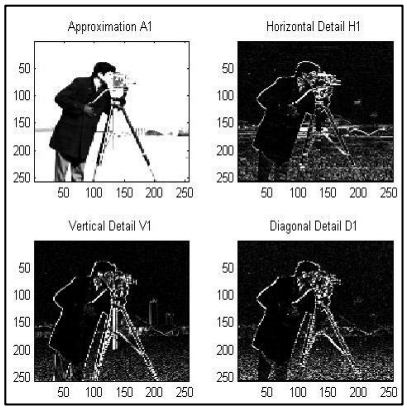

Fig.17

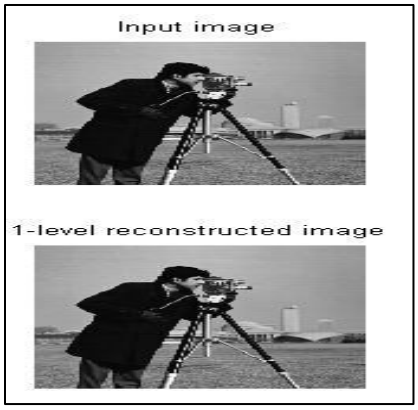

Fig.18

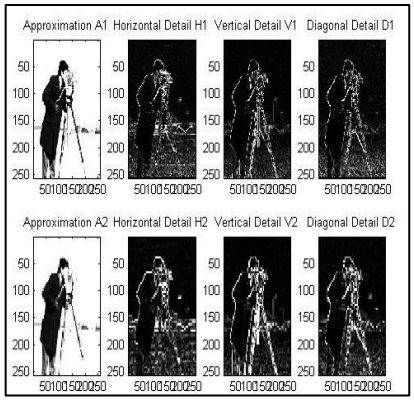

Fig.19

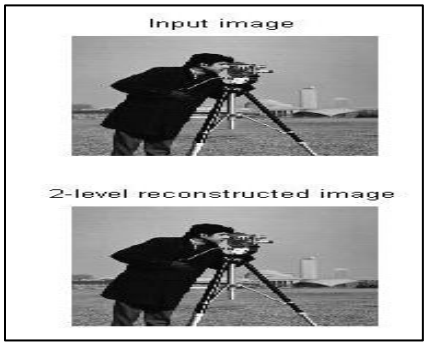

Fig.20

Conclusion

My work presents various types of image compression techniques. There are basically two types of compression techniques. One is Lossless Compression and other is Lossy Compression Technique. Comparing the performance of compression technique is difficult unless identical data sets and performance measures are used. Some of these techniques are obtained good for certain applications like security technologies. Some techniques perform well for certain classes of data and poorly for others. PCA (Principal Component Analysis) also found its applications as image compression. PCA can be implemented in two forms i.e. either statistical approach or neural network approach. The PCA Neural Network provides new way of generating codebook based

on statistical feature of PCA transformational coefficients. It leads to less storage of memory and reduction of calculation.

References

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

[2] H.Rheingold. "*The Virtual Community: Homesteading on the Electronic Frontier*". Addison-Wesley, MA, 1993.

[3] Hong Zhang, Xiaofei Zhang & Shun Cao, *" Analysis & Evaluation of Some Image Compression Techniques*," High Performance Computing in AsiaPacific Region, 2000 Proceedings, 4th Int. Conference, vol. 2, pp 799-803,14-17 May, 2000

[4] R. Nakatsu. "*Toward the Creation of a New Medium for the Multimedia era*". In Proceedings of IEEE, pages 825–836, May1998.

[5] Milos Klima, Karel Fliegel, "*Image Compression Techniques in the field of security Technology: Examples and Discussion",* 38th Annual Intn. Carnahan Conference, pp 278 – 284, 11 – 14 Oct. , 2004.

[6] Ismail Avcibas, Nasir Memon, Bulent Sankur, Khalid Sayood, " A Progressive Lossless / Near Lossless Image Compression Algorithm," IEEE Signal Processing Letters, vol. 9, No. 10, pp 312-314, October 2002.*

[7] R.V.Cox, B.G.Haskell, Y.leCun, B.Shahraray, and L.Rabiner. On the Applications of Multimedia Processing to Communications. In Proceedings of IEEE, pages 755–824, May 1998.

[8] Nadia Magnenat Thalmann, Prem kalra, and Marc Escher. Face to Virtual Face. In Proceedings of IEEE, pages 870 – 883, May 1998.

[9] David H. Kil and Fances Bongjoo Shin, " Reduced Dimension Image Compression And its Applications," Image Processing,

1995, International Conference, Vol. 3, pp 500 – 503, 23-26 Oct, 1995.

[10] C.K. Li and H.Yuen, "A High Performance Image Compression Technique For Multimedia Applications," IEEE Transactions on Consumer Electronics, Vol. 42, no. 2, pp 239-243, 2 May 1996.