AN INTRODUCTION TO JPEG IMAGE COMPRESSION ALGORITHM

ANMOL JYOT MAAN

University College of Engineering, Punjabi University, Patiala
Email:-Anmolmaan12@gmail.com

Abstract—JPEG is a standard image compression technique. This paper introduces the JPEG image compression algorithm and examines each step in the compression and decompression.

Keywords— DCT, Quantization, zigzag reordering, Huffman coding

I. INTRODUCTION

JPEG is an image compression standard developed by the Joint Photography Experts Group". The main purpose of JPEG image compression is to store the data necessary to reconstruct an image using as little space as possible. Most commonly JPEG uses a lossy compression algorithm and as the name suggests, some information is removed from the image when compressing. This algorithm achieves much of its compression by exploiting the limitations of the human eye. Because human eyes are less sensitive to higher frequency, compression can be achieved by suppressing the high frequency components. JPEG is used to compress either full-color or gray scale images. In the case of color images, RGB is transformed into a luminance/chrominance color space (YCbCr, YUV, etc.). This is done to allow different processing on the luminance and chrominance components. The pixel values for each component are divided into 8x8 blocks and the DCT is applied to each block. The resulting coefficients are then quantized. The larger the quantization coefficients, the more data are lost. After quantization, zigzag scan is performed on these quantized 8x8 blocks. The purpose behind this is to retain DC and low frequency components and discard high frequency components. Zigzag coefficients are encoded by Run Length and Huffman coding. This step is lossless. When desired, the image can be reconstructed through decompression using Inverse Discrete Cosine Transform (IDCT). The coding system architecture of the JPEG is described in figures below.

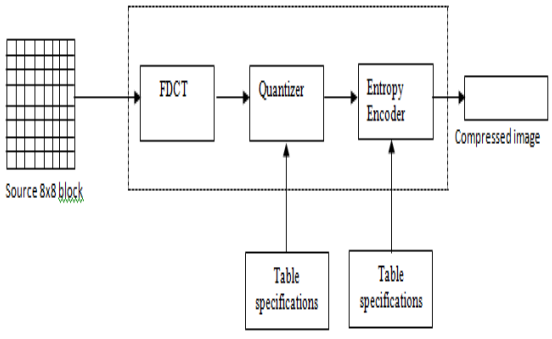

Figure 1.*JPEG Encoder*

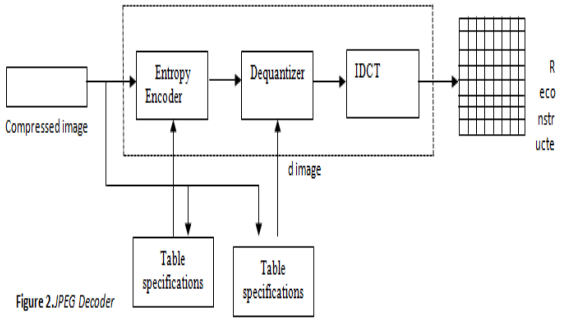

Figure 2.*JPEG Decoder*

II. DISCRETE COSINE TRANSFORM

The DCT is closely related to DFT (Discrete Fourier Transform). Both can transform an image from spatial domain to frequency domain. However, DCT is more appropriate than DFT for image compression. The Forward DCT and Inverse DCT are defined as:

$$F(u,v) = \frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7} p(x,y)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$

$$p(x,y) = \frac{1}{4}\sum_{u=0}^{7}\sum_{v=0}^{7} C(u)C(v)\,F(u,v)\cos\left[\frac{(2x+1)u\pi}{16}\right]\cos\left[\frac{(2y+1)v\pi}{16}\right]$$

Where $C(u,v) = \begin{cases}\frac{1}{\sqrt{2}} \text{ if } u,v = 0\\ 1 \text{ if } u,v > 0\end{cases}$

And p(x, y) is the matrix of values from the original image. Let's take an 8x8 block of pixel values from an image.

Original =

$$\begin{bmatrix}139 & 144 & 149 & 153 & 155 & 155 & 155 & 155\\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156\\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156\\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159\\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 159\\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157\\ 162 & 162 & 161 & 163 & 162 & 157 & 157 & 157\\ 162 & 162 & 161 & 161 & 163 & 158 & 158 & 158\end{bmatrix}$$

Next step is to do the level shift. The DCT is designed to work on pixels ranging from -128 to 127. Therefore

the above block is leveled off by subtracting 128 from each entry. The resulted matrix is:

M =

$$\begin{bmatrix} 11 & 16 & 21 & 25 & 27 & 27 & 27 & 27 \\ 16 & 23 & 25 & 28 & 31 & 28 & 28 & 28 \\ 22 & 27 & 32 & 35 & 30 & 28 & 28 & 28 \\ 31 & 33 & 34 & 32 & 32 & 31 & 31 & 31 \\ 31 & 32 & 33 & 34 & 34 & 27 & 27 & 31 \\ 33 & 33 & 33 & 33 & 32 & 29 & 29 & 29 \\ 34 & 34 & 33 & 35 & 34 & 29 & 29 & 29 \\ 34 & 34 & 33 & 33 & 35 & 30 & 30 & 30 \end{bmatrix}$$

The DCT can now be applied on original matrix which results in following matrix.

D =

$$\begin{bmatrix} 235.6 & -1 & -12.1 & -5.2 & 2.1 & -1.7 & -2.7 & 1.3 \\ -22.6 & -17.5 & -6.2 & -3.2 & -2.9 & -0.1 & 0.4 & 1.3 \\ -10.9 & -9.3 & -1.6 & 1.5 & 0.2 & -0.9 & -0.6 & -.1 \\ -7.1 & -1.9 & 0.2 & 1.5 & 0.9 & -0.1 & 0.0 & 0.3 \\ -0.6 & -0.8 & 1.5 & 1.6 & -0.1 & -0.7 & 0.6 & 1.3 \\ 1.8 & -0.2 & 1.6 & -0.3 & -0.8 & 1.5 & 1.0 & -1 \\ -1.3 & -0.4 & -0.3 & -1.5 & -0.5 & 1.7 & 1.1 & -.8 \\ -2.6 & 1.6 & -3.8 & -1.8 & 1.9 & 1.2 & -0.6 & -.4 \end{bmatrix}$$

This matrix now contains 64 DCT coefficients. The upper left coefficient correlates to the low frequencies. The lower right values represent higher frequencies. As human eye is more sensitive to low frequencies, the next step of quantization will reflect this fact.

III. QUANTIZATION

Quantization is where compression really happens. It is the main source for loss in JPEG compression. In this step each of the 64 frequency components are divided by a separate quantization coefficients and results are rounded to the nearest integer value. This results in many zero coefficients in the quantized matrix. Higher frequencies are quantized less accurately as they are less visible to eyes. In this step, varying levels of image compression and quality can be obtained through the selection of specific quantization matrices. This step allows the user to decide on quality levels ranging from 1 to 100, where 1 provides the highest compression and the poorest image quality and 100 provides the best quality. For a single image, more than one quantization table can be used (up to four). Each quantization table that is used must be stored in compressed file. The quantization step can be defined as:

$$C_{i,j} = \text{round}\left(\frac{D_{i,j}}{Q_{i,j}}\right)$$

For the following step, the quantization matrix with quality level of 50 is chosen. Note that the values in the upper left corner which correspond to low frequency components are fairly low. On the other hand, the values in the lower right region which are used to quantize the high frequency components are higher.

Q =

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

The resulted matrix after division is:

$$C = \begin{bmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Here, zeros represent the higher frequencies that have been discarded. This results in lossy compression. Only the remaining non zero coefficients will be used to reconstruct the image. If the quantization matrix with quality level 10 was used, it would result in significantly many zeros. On the contrary, the quantization matrix with quality level 90 would result in very few zeros.

IV. ZIGZAG REORDERING

Now the 8x8 matrix has been quantized and more than half of the DC coefficients are equal to zero. JPEG incorporates run-length coding to take advantage of this by encoding quantized coefficients in the zigzag sequence as shown in figure below.

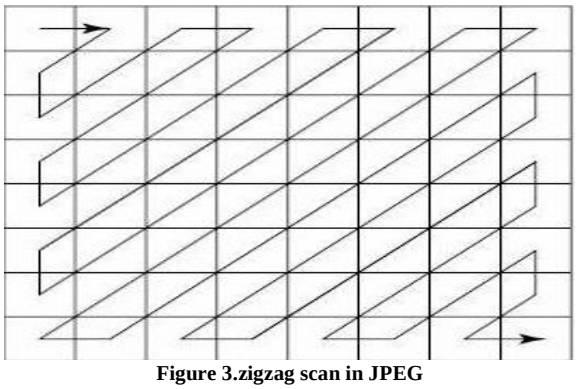

Figure 3.zigzag scan in JPEG

The quantized 8x8 matrix can be reordered into a 64x1 vector that will place all the non-zero entries from the matrix in the first few entries of the new vector. The string of zeros following the last non-zero entry can be

eliminated as it contains unnecessary information. The zigzag reordering pattern turns the matrix into a long string of matrix coefficients, which will be shown as a row vector.

$$V = [15\,0\,-2\,-1\,-1\,-1\,0\,0\,1\,0\,0\,0\,0\,0\,0\,\ldots]$$

All the zeros that follow the last "1" will be eliminated and "end of block (EOB)" will be added which signifies that the string is from one 8x8 matrix.Further,run-length encoding replaces values in 64 –vector by a pair (skip, value), where skip is the number of zeros before a value, and value is the value. The special pair (0, 0) specifies the end of block.

V. HUFFMAN CODING

In depth coverage of coding is beyond the scope of this paper. The zigzag coefficients are encoded using run-length encoding and Huffman coding. Huffman coding is based on the frequency of the symbols being encoded and not on the symbols themselves, so it allows redundancies in the symbols to be compressed. The length of each codeword is a function of its relative frequency of occurrence. Smaller codeword represents the most common symbol. Huffman coding is also uniquely decodable, which means that the code generated from a string cannot represent something other than the original input when it is decoded.

VI. DECOMPRESSION

Decoding works exactly in the opposite direction. Reconstruction begins by decoding the bit stream representing the quantized matrix C. Each element of C is then multiplied by the corresponding element of quantization matrix which results into following matrix:

R =

$$\begin{bmatrix} 240 & 0 & -10 & 0 & 0 & 0 & 0 & 0 \\ -24 & -12 & 0 & 0 & 0 & 0 & 0 & 0 \\ -14 & -13 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Then the IDCT is applied to above matrix. And 128 is added to each element of the resulted matrix, which gives us the decompressed version of the original 8x8 image block.

The decompressed matrix is:

$$\begin{bmatrix} 144 & 146 & 149 & 152 & 154 & 156 & 156 & 156 \\ 148 & 150 & 152 & 154 & 156 & 156 & 156 & 156 \\ 155 & 156 & 157 & 158 & 158 & 157 & 156 & 155 \\ 160 & 161 & 161 & 162 & 161 & 159 & 157 & 155 \\ 163 & 163 & 164 & 163 & 162 & 160 & 158 & 156 \\ 163 & 164 & 164 & 164 & 162 & 160 & 158 & 157 \\ 160 & 161 & 162 & 162 & 162 & 161 & 159 & 158 \\ 158 & 159 & 161 & 161 & 161 & 161 & 159 & 158 \end{bmatrix}$$

A reconstructed image that has been compressed using the JPEG compression technique will be an approximation of the original because of the loss of information in the compression procedure.

VII. IMAGE EXAMPLE

The JPEG image compression is done on the image shown in figure 4 with the quality level 15.

Figure 4.original,

Figure 5.quality 15 – 90% zeros

CONCLUSION

Image compression is an extremely important part of modern computing. By having the ability to compress images to a fraction of their original size, valuable (and expensive) disk space can be saved. Further, Image compression has played an important part in the development of the Internet as the transportation of images from one computer to another becomes easier and less time consuming. The JPEG image compression algorithm provides a very effective way to compress images with minimal loss in quality. It is very useful for storing pictures as visually unimportant information can be discarded. It has become the most popular image format. In this paper, the ability of the JPEG image compression to retain image quality while drastically reducing the file size is introduced through the implementation of DCT, quantization, reordering and Huffman coding.

REFERENCES

[1]. Gregory K. Wallace," The JPEG Still Picture Compression Standard", Submitted in December 1991 for publication in IEEE Transactions on Consumer Electronics.

[2]. John W. O'Brien ," The JPEG Image Compression Algorithm", APPM-3310 FINAL PROJECT,DECEMBER 2, 2005.

[3]. Pao-Yen Lin "Basic Image Compression Algorithm and Introduction to JPEG Standard", National Taiwan University, Taipei, Taiwan, ROC 2009.

[4]. Ken Cabeen, Peter Gent "Image Compression and the Discrete Cosine Transform", College of the Redwoods.

[5]. V.S Arun "Image Compression Using JPEG Algorithm", Digital Signal & Image Processing Centre, MSRSAS, Banglore.

[6]. Ian Synder "JPEG Image Compression", December 2009.

[7].William B. pennebaker ,Joan L. Mitchell , "JPEG still image data compression standard " , Library of congress cataloging-in -publication, Eight printing 2004 by kluwerAcadimic publishers.