JPEG IMAGE COMPRESSION & EDITING IMPLEMENTED IN MATLAB

Kumar Gaurav[1] ,Sujeet Kumar Gupta[2] ,Om Prakash Sinha[3], Mr. Sukanta Kumar Tulo[4]

Department of Applied Electronic and Instrumentation Engineering
Gandhi Institute of Engineering & Technology Gunupur ,Orissa /BPUT ,Rourkela ,Orissa (India)

Abstract : **In this project I implement basic MATLAB code for image compression and some editing of images of different format mainly tiff and jpeg . I will show that I have implemented the DCT algorithm and some basic algorithm for compression of images throughout my project. Although I obtained the compressed image up to various coefficients, and increasing the quality of images using histogram equalization, I came very close to it**

Keywords - image compression, histogram, editing of image, Restoring

I. INTRODUCTION

Multimedia images play vital role for store the memories and moments of life everyday. The amount of information encoded in an image is quite large in size. Even with the advances in bandwidth and Storage capabilities, if images were not compressed many applications would be too costly . This Research project attempts to answer the following questions: What are the basic principles of image compression? How do we measure how efficient a compression algorithm is? How to increase the vquility of images ? What are the alternatives to JPEG? Do they have any advantages or disadvantages?

II. BANDWIDTH AND TRANSMISSION

In our high stress, high productivity society, efficiency is key. Most people do not have the time or patience to wait for extended periods of time while an image is downloaded or retrieved. In fact, it has been shown that the average person will only wait 20 seconds for an image to appear on a web page. Given the fact that the average Internet user still has a 28k or 56k modem, it is essential to keep image sizes under control. Without some type of compression images would be too cumbersome and impractical for use. The following table is used to show the correlation between modem speeds and download time. Note that even high speed Internet users require over one second to download the image.

III. FIGURES AND TABLES

Modem Speed	**Throughput – How Much Data Per Second**	**Download Time For a 40k Image**
14.4k	1kB	40 seconds
28.8k	2kB	20 seconds
33.6k	3kB	13.5 seconds
56k	5kB	8 seconds
256k DSL	32kB	1.25 seconds
1.5M T1	197kB	0.2 seconds

Table1: Download Time Comparison

Results:

I will present the steps of obtaining a finalized images file and how I applied each steps in matlab. I will do many changes in matlab codes and find lastly compressed image and we can also save this image for transmitting purpose.

Step1: Converting the original image to 8x8 matrices, DCT transform, quantizing algorithm

These steps are relatively easy in matlab, which is specifically set up to work with matrices. The 2-D discrete cosine transform is done simply with the dct2() command. After the matrix spread into 8x8 matrices and performing the DCT, a simple piecewise division by the quantization matrix obtains the quantized matrices needed for the next step.

Step 2: Zig-Zag Encoding of Quantized Matrices:- I feel no matlab implementation for this function, so I wrote code by myself. I took advantage of the fact that each diagonal row has addresses that add up to the same number. Depending on whether the number is even or odd determined the direction of the iteration through the row. The code I wrote is able to zig-zag through any matrix of equal height and width. This could be useful if one wanted to experiment with deviding images into matrices larger than 8x8.

Step 3: Conversion of quantized vectors into the JPEG defined bitstream

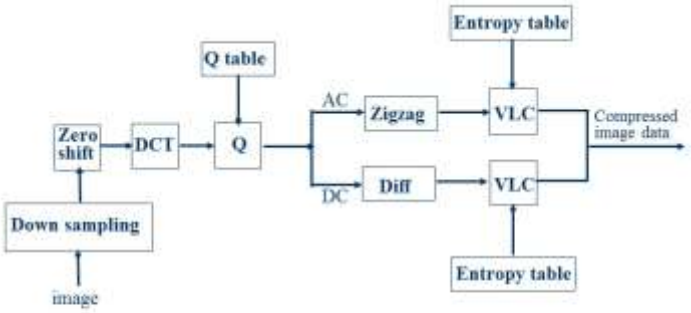
For this step, I started with an old implementation of the default AC code written by Yu Hen Hu. After updating the code to work with Matlab 14 I modified the code to encode the first number in the incoming vector with the default DC code, the table for which I added to the file. The function returns a completed bitstream to correspond to the input of the quantized vector.I am sorry to say I cannot guarantee that the code contained in vecenc.m is at all reliable. I tested single vectors with standard examples and obtained the correct result. However, having never obtained a final result, I cannot guarantee that this function complies with the JPEG standard.

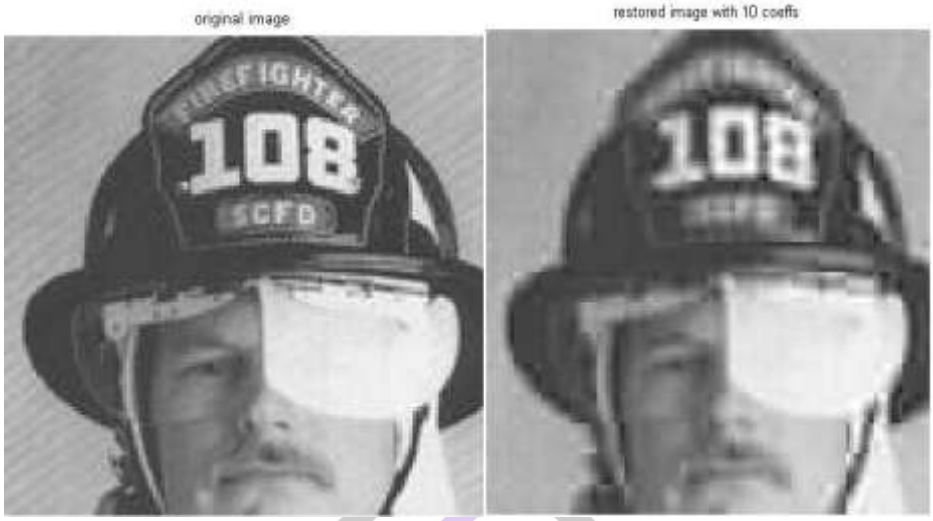
Step 4: Construction of the JPEG File header, Writing the FileBeing relatively inexperienced with coding in general, this step presented me with the most trouble overall. It took several hours to determine how to encode a binary vector into a file. It took even longer to realize that each byte encoded into the file was being represented with the least significant bits on the left side. After overcoming that obstacle, I was faced with the task of constructing a file header for my bit stream.

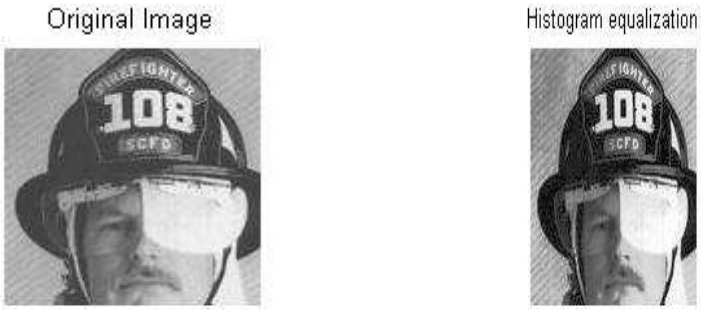
The JPEG standard only goes as far as conversion to the binary bit stream. While that process is well defined in scientific papers, the construction of a JPEG file header is not. In the matlab file head.m, I tried to express what I learned about the process in the most expressive way possible.

It seems that the JPEG file is broken into many blocks. Each block begins with two bytes, the first being FF in hexadecimal and the second being 'XX' where different 'XX's denoting different blocks. The second part of each block is the length, in bytes, of the block including the two length bytes. The rest of the block contains the data as defined by the block type.

As of this writing, I am in the middle of coding head.m, but will be unable to finish due to time constraints. I am confident, though, that I have a very good understanding of how the rest of the header construction would proceed.







FUTURE SCOPE

1. Analysis and compression of medical imagery is an important area of Biomedical Engineering.
2. The techniques can be used along with modified SPIHT for improved image compression.
3. The techniques can be extended for video compression.
4. The techniques can be extended for any other image processing applications for better results.
5. It can be used for restoring the image by compression method and fast speed transmission

CONCLUSION

While not completing the goal I set out to achieve, I have demonstrated that conversion from a grayscale image to the JPEG encoded binary bit stream is a fairly simple and straightforward process. It comes as no surprise to me that the file I/O was the most challenging part of the process.

As for where to go from here, I hope to complete the project in my free time and publish the matlab files on the internet. I believe that while there are far more powerful and efficient implementations of the JPEG algorithm out there, other students like myself would benefit from a simple and straightforward implementation that emphasizes step-by step explanations of what is going on and why.

REFERENCES

[1]. Wallace, G., The JPEG still picture compression standard, Communications of the ACM, 34 (1999) 31-44.

[2]. http://in.mathworks.com/matlabcentral/answers/152071-jpeg-compression-algorithm• implementation-in-matlab.

[3]. http://in.mathworks.com/matlabcentral/fileexchange/4328-jpeg-compression. [4]. An Introduction to Matlab by Krister Ahlersten .

[5]. Digital Signal and Image Processing using MATLAB, Volume 2: Advances and

Applications: The Deterministic Case, 2e by Blanchet.