Redundant Data Elimination for Image Compression and Internet Transmission using MATLAB

R. Challoo, I.P. Thota, and L. Challoo
Texas A&M University-Kingsville
Kingsville, Texas 78363-8202, U.S.A.

ABSTRACT

Data collection, representation, and transmission are important topics in science and engineering education. Data is a mean to convey information. The main objective of this paper is to minimize the number of bits required to represent an image for internet transmission and improve the accuracy of discrete interpolation using the discrete cosine transform to map input data sequence that satisfies the boundary conditions. The critical boundary conditions are set and mapping for the input data is developed. A typical example is simulated to show the efficiency of the proposed algorithm. The data compression scheme was simulated using MATLAB. This paper can help the reader in understanding image compression using Discrete Cosine Transform. Redundant data elimination and high data compression rate can result in high Internet transmission rate.

Keywords: Image Compression, Data Transmission, MATLAB

1. INTRODUCTION

An image can be represented with many pixels along with some redundant data. Elimination of redundant data reduces the number of bits used to represent the data. In some cases repetition of data contains less important information which can be eliminated and the image can be represented with less number of bits with greater efficiency. Here, we take advantage of Discrete Cosine Transform (DCT) to get a better quality image rather than using Discrete Fourier Transform (DFT) for image compression and we remove redundant data to increase storage capacity and transmission rate.

2. IMPLEMENTATION OF DISCRETE COSINE TRANSFORM

A. The Process Steps

The following is a general overview of the JPEG process.

- The picture is represented as 8x8 blocks of pixels.
- From left to right, top to bottom, the DCT is applied to every block and the D matrix is calculated.
- Every block is compressed using method of quantization.
- The group of compressed blocks that comprise the image is saved in a significantly decreased space.
- The image is recovered through a process of decompression when required, a method that utilizes the Inverse Discrete Cosine Transform (IDCT).

B. Quantization

Our 8x8 block of DCT coefficients is now set for compression by the process of quantization. Extremely useful and remarkable characteristics of the process of JPEG are that in this process, different stages of compression of picture and quality are available by selection of specific quantization matrices. It allows the user to decide on quality levels in borders from 1 to 100, where 1 deals the poorest quality of picture and the highest compression, while 100 gives the best quality and the lowest compression [5]. Individual examinations concerning the human being visual scheme have resulted in the typical quantization matrix JPEG. With a quality level of 50, this matrix renders both high compression and also excels in quality of the decompressed picture.

$$Q50 = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Quantization can be performed by dividing each and every individual pixel value in matrix D which is also known as transformed image matrix by the corresponding pixel values of the quantization matrix and rounded off to their nearest whole number. For our example quantization level 50 matrix is used:

$C_{i,j}$ = (rounding off) $D_{i,j} / Q_{i,j}$

$$C_{i,j} = \begin{bmatrix} 10 & 4 & 2 & 5 & 1 & 0 & 0 & 0 \\ 3 & 9 & 1 & 2 & 1 & 0 & 0 & 0 \\ ?7 & ?5 & 1 & ?2 & ?1 & 0 & 0 & 0 \\ ?3 & ?5 & 0 & ?1 & 0 & 0 & 0 & 0 \\ ?2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The coefficients located near the upper-left corner represent the lower frequencies to which the human eye is most susceptible of the picture block.

The pixel value of zero represents the less important features in the image and the nonzero values represent the highest frequency and also more important features in that image. Only nonzero pixel value elements are used to recover the original image.

3. METHODOLOGY

In this paper we compress the images using DCT and reconstruct the original image from the compressed data using IDCT.

First we take an image which can be represented with 8x8 elements and the total of 64 pixels. This is a gray scale image and the pixel values for black and white picture range from -127 to 128. Subtract 128 from all the pixel values and perform DCT on that image.

To compress the image we use quantization. This quantization process is performed on the resultant matrix after the DCT is applied. We can choose the quality levels for this process. Quality level 50 is used for our compression which gives great quality image with high compression. In the final resultant matrix, zero pixel values indicate lower frequencies and they are of less importance, whereas, non-zero pixel values indicate higher frequencies and are more important in reconstructing the original image.

To implement DCT practically, we chose 10 random images. We first created the basis:

a) 3D plot with basic resolution (64 plots of 8x8 pixels) using "surf" function.

b) 3D plot with x20 resolution (64 plots of 160x160 pixels) using "mesh" function.

c) 2D plot with x10 resolution (64 plots of 80x80 pixels) using "mesh" function.

d) 2D plot with x10 resolution (64 plots of 80x80 pixels) using "imshow" function.

Then we performed DCT on these ten images. We have also shown the power of DCT coefficients which shows that the power of the coefficients decreases from top to bottom. Signal-to-Noise Ratio (SNR) graph is also plotted for image number 8. Also, a pseudo color graph was plotted for the penny image using three dimensional image which looks like the original penny image.

4. RESULTS AND DISCUSSIONS

A. Details of the algorithm:

A basic JPEG compression (gray level image):

1) Take an image (two dimensional matrix) and divide it to 8x8 matrices

2) For every matrix (8x8) utilize the DCT conversion (from the signal processing toolbox). An (8x8) matrix is obtained.

3) Build an 8x8 matrix, which is the total of all the matrices, such that sum_matrix = A + B + C +...

4) Sort rudiments of the 8x8 matrix from the maximum to the minimum and get the indices listing.

5) Sum the last matrix with part of the elements which have the higher coefficients, until you have a sufficient ratio (let's say 80%). [7]

For example:

```
idx = sort (sum_matrix (:));
part_of_energy=sum_matrix (idx(1:20));
all_energy = sum_matrix (:);
Ratio = part_of_energy/all_energy;
```

6) Save the partial list of index, number of matrices (rows, …) and from every matrix from 2nd step save only these coefficients

B. For reconstruction:

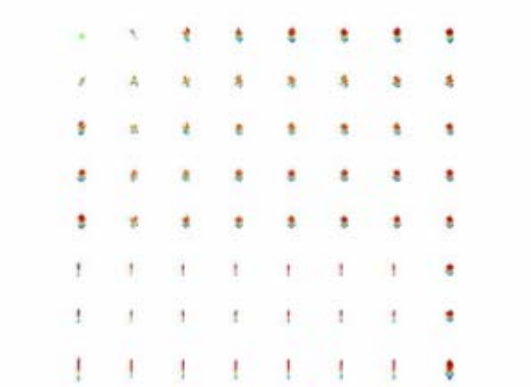
1) Build matrices of 2nd step by the zero command:
A = zero(8,8); B = zero(8,8); ...

2) In every matrix, save the coefficients in the correct places, so that the new matrix A is equivalent to the matrix A from 2nd step of the encoding.

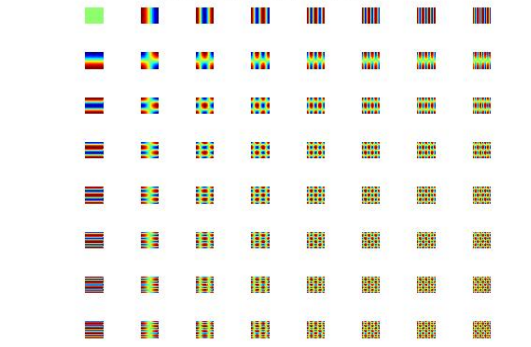
3) For each matrix perform IDCT.

4) Compose these inverse-transformed matrices back into the big matrix which is the reconstructed image. Usually, you will have most of the energy inside the upper-left coefficient (1, 1) which corresponds to the DC (or average value) of the whole picture.

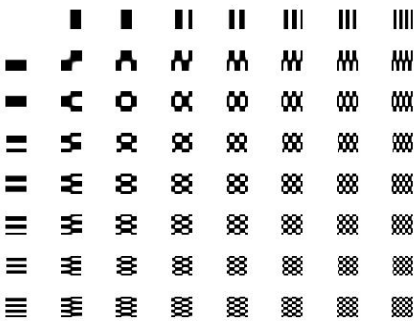
a) 3D plot with basic resolution (64 plots of 8x8 pixels) using "surf" function.



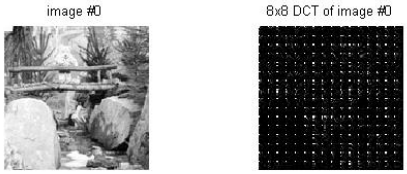
b) 3D plot with x20 resolution (64 plots of 160x160 pixels) using "mesh" function

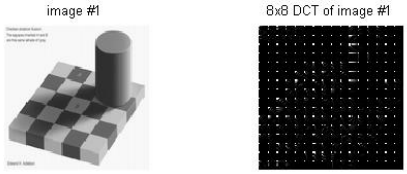


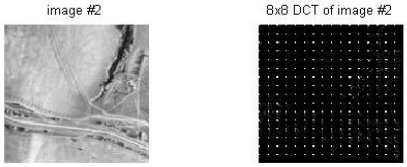
c) 2D plot with x10 resolution (64 plots of 80x80 pixels) using "mesh" function

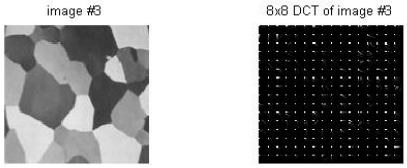


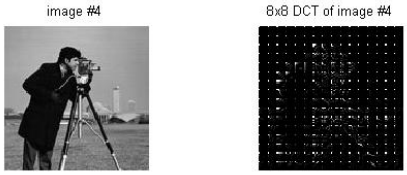
d) 2D plot with x10 resolution (64 plots of 80x80 pixels) using "imshow" function.

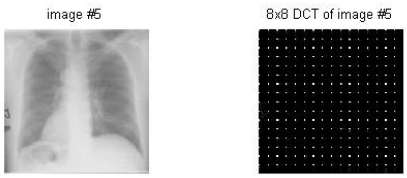


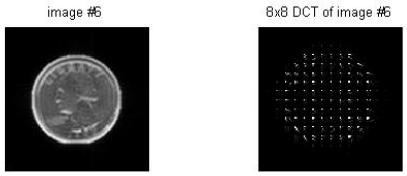


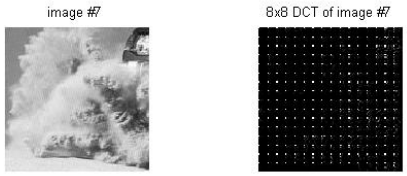


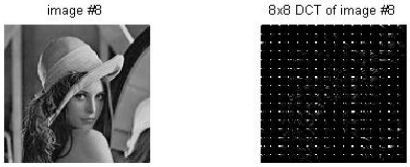


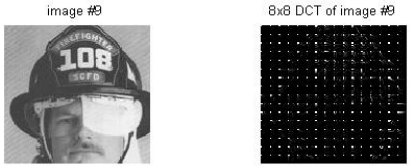












A 2 dimensional DCT on 8x8 block was done for each picture from 0 to 9 and the DCT was saved inside a cell of the size: 10 cells of 128x128 matrix. DCT 8x8 block transform for each picture is shown.

Power of DCT coefficients

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 26.2db	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

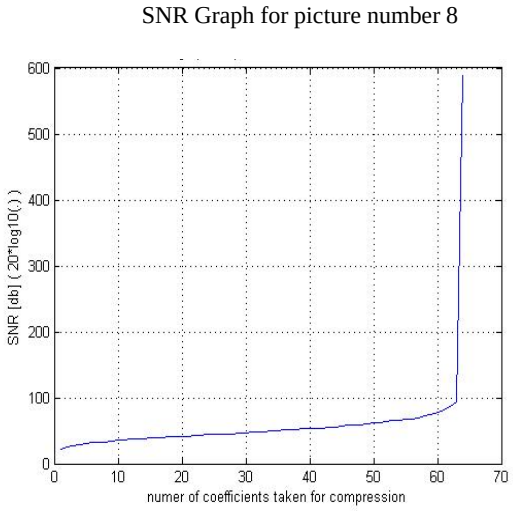
Statistics on the cell array of the DCT transforms was performed. A matrix of 8x8 which will describe the value of each "DCT-base" over the transform of the 10 given pictures was created. Since some of the values were negative, and we were interested in the energy of the coefficients, abs()^2 values were

added into the matrix. This is consistent with the definition of the "Perceval relation" in Fourier Coefficients.

We have used this matrix to choose only the wanted number of coefficients. The matrix is initialized to zeros.

coef_selection_matrix = zeros (8, 8); [7]

Compressed picture set (to show the degrading).
Compressed set = [1 3 5 10 15 20 30 40];



calc_snr - calculates the SNR of a figure being compressed.

Assumption: SNR calculation is done in the following manner:

The deviation from the original image is considered to be the noise therefore: Noise= original image - compressed image.

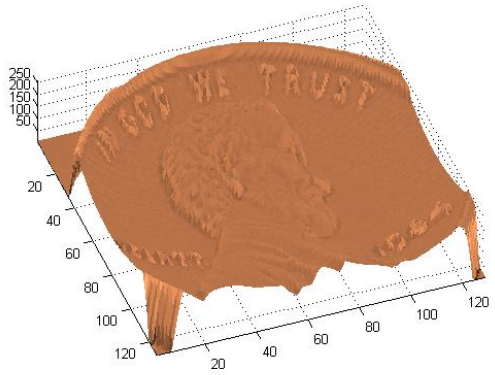
The SNR is defined as:

SNR=energy_of_image/energy_of_noise

Which yields:

SNR=energy_of_image/((original_image-compressed_image) ^2)

C. Dimensional Plot



The drawings of a conspiracy in pseudo color with the shine proportional to Laplacian of the height. A cell is bright if its height is bigger than the average of his four neighbors and sinks if its height is less than the average of his four neighbors. It is a “model of unaccustomed illumination ", but it produces a picture which resembles a photograph of a cent. Finally, a three dimensional copper colored, surface plot with the Laplacian lighting model was produced.

5. CONCLUSIONS

Image compression is one of the necessities of modern Digital world. For instance, with the explosive growth of the Internet there is a growing need for audio compression or data compression in general. Goals of such compressions are to minimize the storage memory, communication channel bandwidth, and transmission time.

Data compression using transformations such as the DCT and the DFT are the basis for many coding standards such as JPEG, MP3 and AC-3. In this paper DCT was used for the compression of an Image/sound signal and the data compression scheme was simulated using MATLAB.

6. REFERENCES

[1]http://en.wikipedia.org/wiki/Discrete_Fourier_transform

[2]http://en.wikipedia.org/wiki/Discrete_cosine_transform

[3] Gurjar, M. and Jagannathan, P., "Image Compression: Review and Comparison of Haar Wavelet Transform and Vector Quantization. URL: WWW.ee.bgu.ac.il/~greg/graphics/compress.html

[4]http://web.uct.ac.za/depts/physics/laser/hanbury/intro_ip.html

[5] Smith, S. W., "The Scientist and Engineer's Guide to Digital Signal Processing," California Publishing Company, 1997. URL: http://www.dspguide.com/ch27/6.htm

[6] Watson, A. B., "Image Compression Using the Discrete Cosine Transform," NASA AMES Research Center. URL: http://www.mathematica-journal.com/issue/v4i1/article/81-88_Watson.mj.pdf

[7] Gonzalez R.C., Woods R.E., and Eddins S.C., " Digital Image Processing using MATLAB," 2003.