A Simple Tutorial to JPEG compression using MATLAB

Seife Kassahun

May 9, 2011

1 Introduction

This tutorial describes the popular JPEG image coding format.The aim is to compress images while maintaining acceptable image quality.This is achieved by dividing the image in blocks of 8x8 pixels and applying discrete cosine transform(DCT) on the partitioned image . The resulting coefficents are quantized , less significant coefficents are set to zero.After quantisation we can use any off-the -shelf compressing software to compress and decompress the resulting values.

2 The DCT

To simplify matters we will assume that the width and height of the image are a multiple of 8x8 and that image is either gray scale or RGB (red ,green,blue). Each pixel is assigned a value between 0 and 255, or triplet of RGB values.For our case we will take only the gray scale version of the image.Matlab has a feature that support the conversion of an RGB image to grayscale.After conversion we will pass each 8x8 block to the Matalb dct2 built in sub routine .

3 Quantisationand Zigzag scan

So far there has been no compression .We will now proceed by introducing zeros in the arrays by means of quantisation .Every coefficents(c),c_{ij} is divided by an integer q_{ij} and rounded to the nearest integer.The importance of the coefficents is dependent on the human visual system. The eye is much more sensitive to low frequencies.Measurements led to standard quantisation table listed below.

$$Q_{LUM} = \begin{pmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 626 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{pmatrix}$$

On the result matrix we apply the zigzag scan in odd order . For the Zigzag scan i have developed a matlab code.

Next I will try to illustrate the above processes by using an example of an 8x8 matrix values obtained from a 2x2 check board image .

$$A = \begin{pmatrix} 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \end{pmatrix}$$

Level shift by subtracting 128 from each entry (values now in range -128 to +127)

$$AS = \begin{pmatrix} -78 & -78 & -78 & -78 & 72 & 72 & 72 & 72 \\ -78 & -78 & -78 & -78 & 72 & 72 & 72 & 72 \\ -78 & -78 & -78 & -78 & 72 & 72 & 72 & 72 \\ -78 & -78 & -78 & -78 & 72 & 72 & 72 & 72 \\ 72 & 72 & 72 & 72 & -78 & -78 & -78 & -78 \\ 72 & 72 & 72 & 72 & -78 & -78 & -78 & -78 \\ 72 & 72 & 72 & 72 & -78 & -78 & -78 & -78 \end{pmatrix}$$

Perform the DCT by doing B=2D-DCT(AS):we can use the matlab built in dct2 routine to apply the 2D-DCT

$$B = \begin{pmatrix} -24.0000 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -492.6402 & 0 & 172.9922 & 0 & -115.5897 & 0 & 97.9922 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 172.9922 & 0 & -60.7468 & 0 & 40.5897 & 0 & -34.4103 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -115.5897 & 0 & 40.5897 & 0 & -27.1212 & 0 & 22.9922 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 97.9922 & 0 & -34.4103 & 0 & 22.9922 & 0 & -19.4919 \end{pmatrix}$$

Now we will divide each B_{ij} value by q_{ij} of the quantisation matrix from the table shown previously.

$$Bq = \begin{pmatrix} -1 & 07 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -41 & 0 & 9 & 0 & -2 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 10 & 0 & -2 & 0 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -3 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The following are entries of the quantized matrix in zigzag order:
columns 1 through 38
-1 0 0 0 -41 0 0 0 0 0 0 10 0 9 0 0 0 0 0 0 0 0 -3 0 -2 0 -2 0 0 0 0 0 0 0 0 0 1 0
columns 39 through 64
1 0 0 0 2 0 0 0 0 0 0 0 0 0 0 -1 0 0 0 0 0 0 0 0 0 0

These are the values we compress and decompress using any of the off-the shelf or develop our own . For this tutorial i use gzip .
Next we go in reverse order to get the compressed version of the original image.

Restore B by multiplying by Q which is the quantise matrix table.

$$Bnew = \begin{pmatrix} -16 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -492 & 0 & 171 & 0 & -116 & 0 & 110 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 170 & 0 & -58 & 0 & 0 & 0 & -62 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -105 & 0 & 64 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 92 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Apply a 2 Dimensional Inverse DCT. Matlab has a built in idct2 for this process.

$$Asnew = \begin{pmatrix} -76 & -69 & -87 & -80 & 76 & 83 & 65 & 72 \\ -87 & -78 & -62 & -79 & 75 & 58 & 74 & 83 \\ -65 & -91 & -73 & -87 & 83 & 69 & 87 & 61 \\ -68 & -86 & -72 & -55 & 51 & 68 & 82 & 64 \\ 64 & 82 & 68 & 51 & -55 & -72 & -86 & -68 \\ 61 & 87 & 69 & 83 & -87 & -73 & -91 & -65 \\ 83 & 74 & 58 & 75 & -79 & -62 & -78 & -87 \\ 72 & 65 & 83 & 76 & -80 & -87 & -69 & -76 \end{pmatrix}$$

Undo the level shift by adding 128 to each entry:

$$Anew = \begin{pmatrix} 52 & 59 & 41 & 48 & 204 & 211 & 193 & 200 \\ 41 & 50 & 66 & 49 & 203 & 186 & 202 & 211 \\ 63 & 37 & 55 & 41 & 211 & 197 & 21 & 5189 \\ 60 & 42 & 56 & 73 & 179 & 196 & 210 & 192 \\ 192 & 210 & 196 & 179 & 73 & 56 & 42 & 60 \\ 189 & 215 & 197 & 211 & 41 & 55 & 37 & 63 \\ 211 & 202 & 186 & 203 & 49 & 66 & 50 & 41 \\ 200 & 193 & 211 & 204 & 748 & 41 & 59 & 52 \end{pmatrix}$$

Finally you can see the two matrix for the original on the left and the compressed version on the right.

$$A = \begin{pmatrix} 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 50 & 50 & 50 & 50 & 200 & 200 & 200 & 200 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \\ 200 & 200 & 200 & 200 & 50 & 50 & 50 & 50 \end{pmatrix} Anew = \begin{pmatrix} 52 & 59 & 41 & 48 & 204 & 211 & 193 & 200 \\ 41 & 50 & 66 & 49 & 203 & 186 & 202 & 211 \\ 63 & 37 & 55 & 41 & 211 & 197 & 21 & 5189 \\ 60 & 42 & 56 & 73 & 179 & 196 & 210 & 192 \\ 192 & 210 & 196 & 179 & 73 & 56 & 42 & 60 \\ 189 & 215 & 197 & 211 & 41 & 55 & 37 & 63 \\ 211 & 202 & 186 & 203 & 49 & 66 & 50 & 41 \\ 200 & 193 & 211 & 204 & 748 & 41 & 59 & 52 \end{pmatrix}$$

References

[1] Arno Swart , *JPEG compression using matlab*,2003.

[2] James Storer , *JPEG compression example slides*,2011.