UNDERSTANDING AND USING

# Lossy Image Compression



## Introduction

- As explained in the previous lecture there are mainly two types of compression:
  - Lossless Compression
    - **RLE r**un-**l**ength **e**ncoding method
    - Huffman coding method
    - Arithmetic Coding method
    - **LZW** algorithm
  - **Lossy Compression**
    - JPEG compression with Fourier transformation method

# Introduction continued

The ultimate aim in lossy compression is to:

- transform the data, which is not easy to be compressed using any of the lossless compression methods, to a domain that can **generate relatively high redundancy** in the data values,
- Mostly the used transform will effect the original values and cause a minor changes to the original values when reversed back, but this will be not very noticeable in the user domain.
- Used in junction with lossless compression
- The media that are used in this kind of compression can be audios, images, and videos. We'll use JPEG image compression as the reference use case.

3

# JPEG compression

- The most popular image compression is the JPEG compression. Four steps: **Colour conversion, DCT transformation, Quantization, and Encoding.**
- **First** step convert the 24-bit RGB (3 layers) image to YUV image as following. Separate luminance (brightness) elements from chrominance (colour) elements.
  - Y=0.299R+0.587G+0.114B
  - U=0.493(B-Y)
  - V=0.877(R-Y)

4

# JPEG compression continued

**Second** step, using DCT (*discrete cosine transformation*) the one of the most common data transformations used in video compression (mostly used with MPEG1 and MPEG2).

Its appeal is based on how it processes data such a way that it can be more efficiently compressed using RLE and Huffman techniques.

5

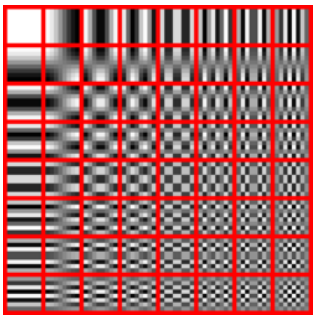
# JPEG compression continued

- The equation takes a group of pixels that have been converted into a YUV format and process them into the frequency format.

$$S_{UV} = \frac{1}{4} C_U C_V \sum_{x=0}^{7} \sum_{y=0}^{7} S_{xy} \; COS \frac{((2x+1)U\pi)}{16} \; COS \frac{((2x+1)V\pi)}{16}$$

$$Where \; C_U C_V = \frac{1}{\sqrt{2}} \; for \; U, V = 0; \quad C_U C_V = 1 \; otherwise$$

8x8 basis used in DCT



Use the following links for Fourier series tutorial and use the Fourier series calculator:
http://www.facstaff.bucknell.edu/mastascu/eLessonsHTML/Freq/Freq4.html

6

# Discrete Cosine Transform (DCT)

- The DCT is in a class of mathematical operations that includes the well known Fast Fourier Transform (FFT), as well as many others.
- The basic purpose of these operations is to take a signal and **transform it from one type of representation to another**.
  - For example, an image is a two-dimensional signal that is perceived by the human visual system. The DCT can be used to convert the signal (spatial information) into numeric data ("frequency" or "spectral" information) so that the image's information exists in a quantitative form that can be manipulated for compression.

http://cs.stanford.edu/people/eroberts/courses/soco/projects/data-compression/lossy/jpeg/dct.htm

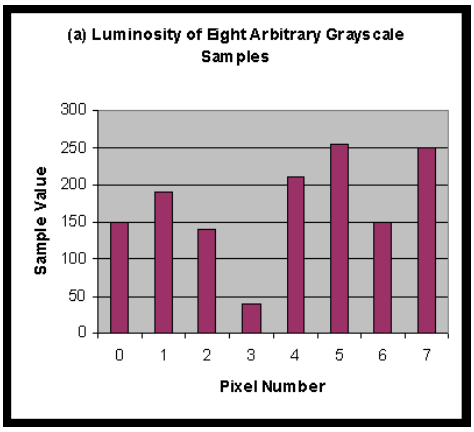
7

# Discrete Cosine Transform (DCT)

- The signal for a graphical image can be thought of as a three-dimensional signal. The X and Y-axes of the image's signal are the two dimensions of the screen, while the amplitude of the signal, the Z-axis, is the value of the pixel at (X, Y).
- This can be represented visually by a two-dimensional array where each cell contains the numerical value of the pixel at that location.
- As the specifics of a two-dimensional DCT matrix are rather complex, we will simplify the problem by first considering the derivation and intentions of a **one-dimensional DCT matrix**.

8

# Discrete Cosine Transform (DCT)




We start with a set of eight arbitrary grayscale samples as charted below, where each bar represents the luminosity of a single pixel.

These values contain all the information necessary to define the eight pixels. Simple **entropy or statistical encoding of this data will not be extremely effective** because in continuous tone images, the levels of luminosity have equal probabilities of occurring.

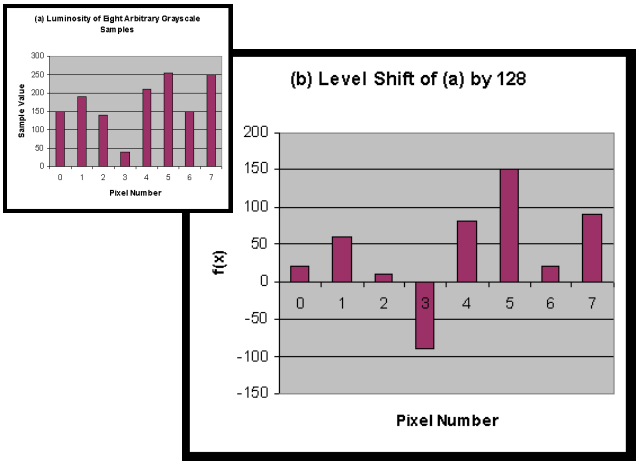
As a more effective alternative, the **DCT can manipulate this data**, separating information crucial to the definition of the image from information that's presence is not perceivable by the human eye.

The insignificant information can then be "discarded" through the quantization phase of JPEG coding, thus achieving large-scale compression.

No information is lost, nor is any compression achieved, in the DCT stage. This initial phase is merely a preparatory step that allows for and leads to the lossy coefficient quantization stage that follows.

9

# Discrete Cosine Transform (DCT)



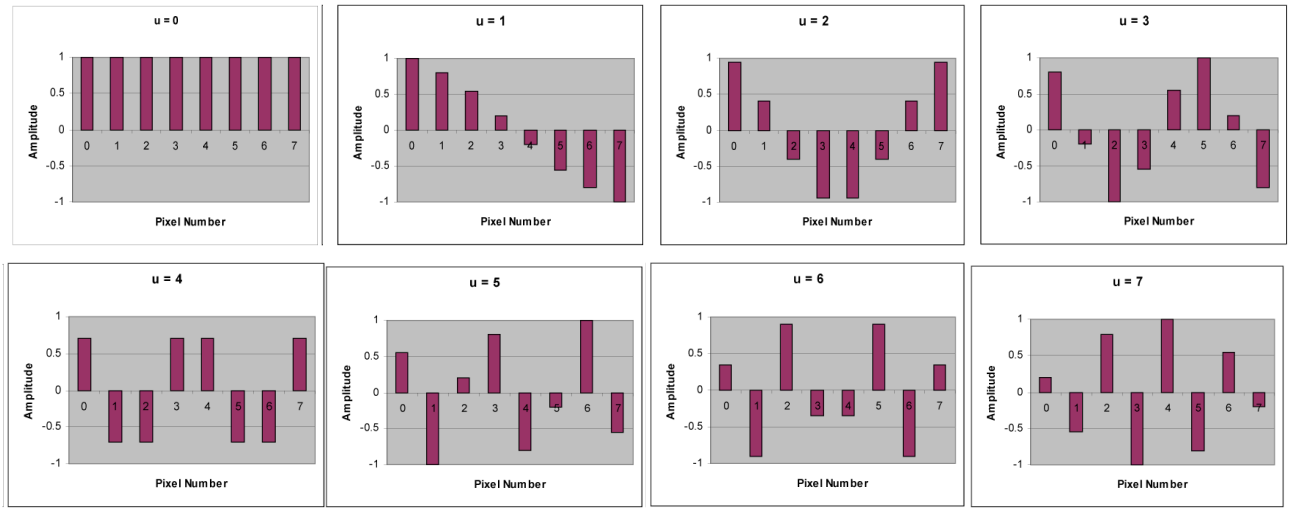

The first step involved in the "rearrangement" of the data displayed in figure (a) is to perform a level shift by 128, the result of which is shown in figure.

The samples have values in the range of 0 to 255. By shifting the level of their graph by 128, half their range, we obtain the values of f(x) in figure. Using f(x), we can decompose the eight sample values into **a set of waveforms** of different spatial frequencies. This decomposition is where the separation of the more significant low-frequency components from the less significant high-frequency components takes place.

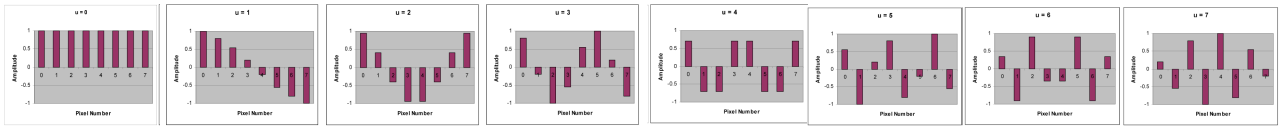
10

# Discrete Cosine Transform (DCT)

Below is a set of waveforms of eight different spatial frequencies, all of uniform amplitude and sampled at eight points.




# Discrete Cosine Transform (DCT)

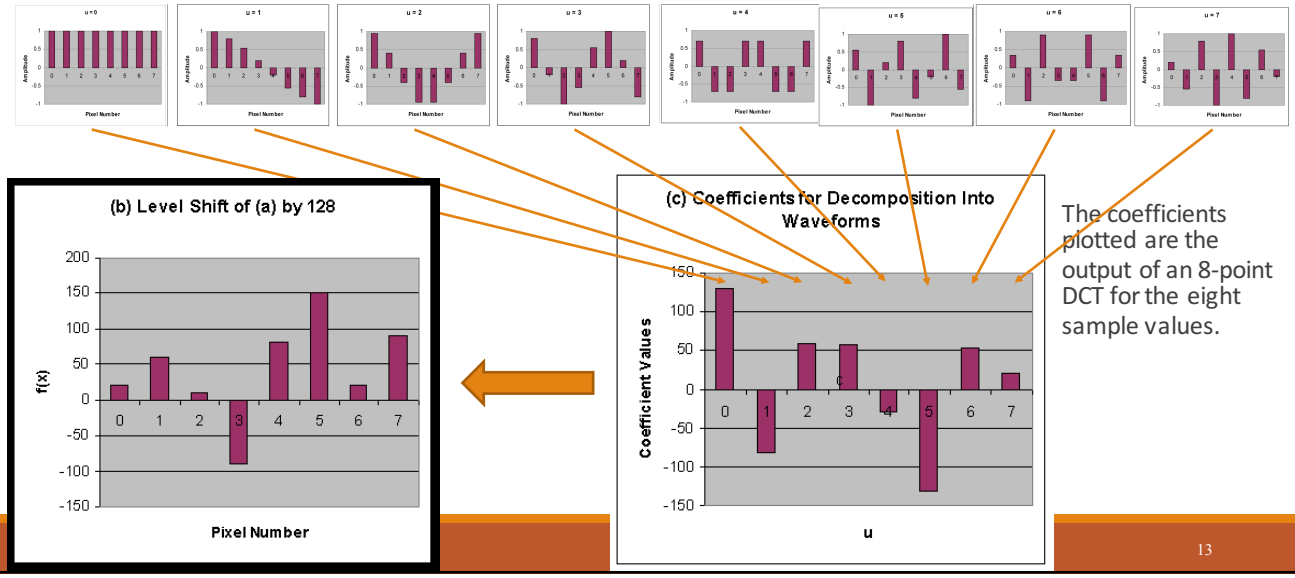


The far left waveform (u = 0) is simply a constant, whereas the other seven waveforms (u = 1, …, 7) show an alternating behavior at progressively higher frequencies.

These waveforms, which are called cosine basis functions, are independent, meaning that there is no way that a given waveform can be represented by any combination of the other waveforms.

However, the complete set of eight waveforms, when scaled by numbers called coefficients and added together, can be used to represent any eight sample values such as our example. The intention is to use the Discrete Cosine Transform to determine the values of the coefficients.

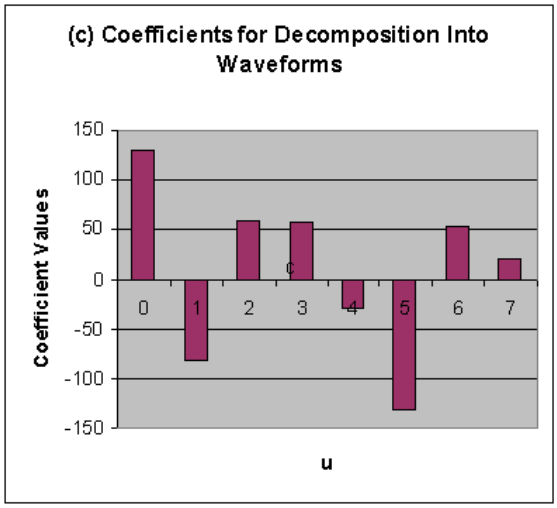
# Discrete Cosine Transform (DCT)




The coefficients plotted are the output of an 8-point DCT for the eight sample values.



# Discrete Cosine Transform (DCT)




- There is a **direct correlation** between the magnitude of the coefficient for a given waveform and the impact of that particular waveform on the quality of the picture.
- The coefficient that scales the constant basis function (u = 0) is called the **DC coefficient**, while the other coefficients are called **AC coefficients**.
- Note that the **DC term gives the average over the set of samples**.
- Furthermore, the **DC term is usually a great deal larger** in magnitude than the AC terms.
- **As the elements move farther away from the DC term (**greater spatial frequencies**), they tend to become lower and lower in value**.
- This suggests that **higher-frequency image components play a relatively small role in the determining picture quality,** while the majority of image definition comes from lower-frequency image components.
- This idea becomes extremely important when applied two-dimensionally to an image, for JPEG exploits this exact concept when deciding what information can be eliminated to achieve compression.

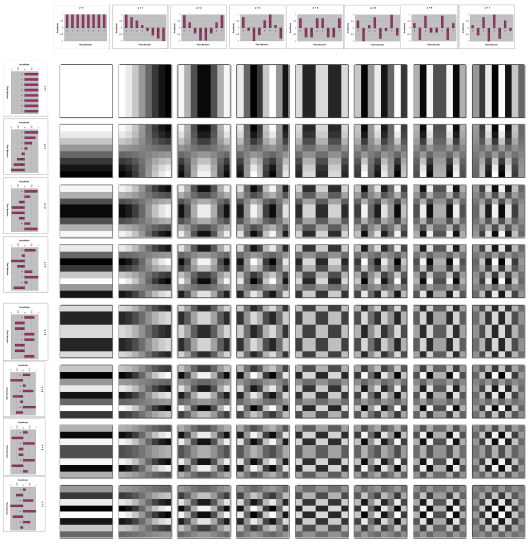
# The two-dimensional DCT

The one-dimensional DCT can be extended to apply to two-dimensional image arrays.

The two-dimensional cosine basis functions from which sample waveforms are composed by multiplying a horizontally oriented set of one-dimensional 8-point basis functions by a vertically oriented set of the same functions.

By convention, the DC term of the horizontal basis functions is to the left, and the DC term for the vertical basis functions is at the top. Consequently, the top, left element of a two-dimensional DCT matrix contains a value that is almost always of a very great magnitude.

Furthermore, mirroring the trend found in a one-dimensional DCT matrix, the farther away an AC term is from the DC term, the higher the frequency its corresponding waveform will have and the smaller its magnitude will be.



15

# The two-dimensional DCT

The actual formula for the two-dimensional DCT is shown below.

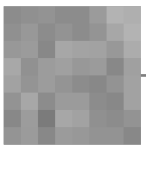
The DCT is performed on an N x N square matrix of pixel values, and it yields an N x N square matrix of frequency coefficients. (In practice, N most often equals 8 because a larger block, though would probably give better compression, often takes a great deal of time to perform DCT calculations, creating an unreasonable tradeoff.

As a result, DCT implementations typically break the image down into more manageable 8 x 8 blocks.)

$$DCT(i,j) = \frac{1}{\sqrt{2N}} C(i)\, C(j) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} pixel(x,y) \cos\left[\frac{(2x+1)i\pi}{2N}\right] \cos\left[\frac{(2y+1)j\pi}{2N}\right]$$

$$C(x) = \frac{1}{\sqrt{2}} \text{ if } x \text{ is } 0, \text{ else } 1 \text{ if } x > 0$$

16



# The two-dimensional DCT

**Input Pixel Matrix**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 140 | 144 | 147 | 140 | 140 | 155 | 179 | 175 |
| 144 | 152 | 140 | 147 | 140 | 148 | 167 | 179 |
| 152 | 155 | 136 | 167 | 163 | 162 | 152 | 172 |
| 168 | 145 | 156 | 160 | 152 | 155 | 136 | 160 |
| 162 | 148 | 156 | 148 | 140 | 136 | 147 | 162 |
| 147 | 167 | 140 | 155 | 155 | 140 | 136 | 162 |
| 136 | 156 | 123 | 167 | 162 | 144 | 140 | 147 |
| 148 | 155 | 136 | 155 | 152 | 147 | 147 | 136 |

**Output DCT Matrix**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 186 | -18 | 15 | -9 | 23 | -9 | -14 | 19 |
| 21 | -34 | 26 | -9 | -11 | 11 | 14 | 7 |
| -10 | -24 | -2 | 6 | -18 | 3 | -20 | -1 |
| -8 | -5 | 14 | -15 | -8 | -3 | -3 | 8 |
| -3 | 10 | 8 | 1 | -11 | 18 | 18 | 15 |
| 4 | -2 | -18 | 8 | 8 | -4 | 1 | -7 |
| 9 | 1 | -3 | 4 | -1 | -7 | -1 | -2 |
| 0 | -8 | -2 | 2 | 1 | 4 | -6 | 0 |

- These are two matrices representing the DCT input and DCT output blocks from a gray-scale image.
- Each element of the output matrix is a coefficient by which the waveform of the corresponding spatial frequency is multiplied in the decomposition of the image sample.
- As shown in the output matrix, the DC coefficient, 186, is relatively large in magnitude, and the AC terms become lower in magnitude as they move farther from the DC coefficient.
- This means that by performing the DCT on the input data, we have concentrated the representation of the image in the upper left coefficients of the output matrix, with the lower right coefficients of the DCT matrix containing less useful information.
- The next step, quantization of the coefficients in the output matrix, "discards" the less useful data and in turn compresses the image data.

17

# Third step: Quantization – what's new?

- Quantization is the process of reducing the number of bits to store an integer value by reducing its precision.
- Given a matrix of DCT coefficients, we can generally reduce the precision of the coefficients more and more as we move away from the DC coefficient.

**DCT Matrix Before Quantization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 92 | 3 | -9 | -7 | 3 | -1 | 0 | 2 |
| -39 | -58 | 12 | 17 | -2 | 2 | 4 | 2 |
| -84 | 62 | 1 | -18 | 3 | 4 | -5 | 5 |
| -52 | -36 | -10 | 14 | -10 | 4 | -2 | 0 |
| -86 | -40 | 49 | -7 | 17 | -6 | -2 | 5 |
| -62 | 65 | -12 | -2 | 3 | -8 | -2 | 0 |
| -17 | 14 | -36 | 17 | -11 | 3 | 3 | -1 |
| -54 | 32 | -9 | -9 | 22 | 0 | 1 | 3 |

**A Sample Quantization Matrix**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 3 | 5 | 7 | 9 | 11 | 13 | 15 | 17 |
| 5 | 7 | 9 | 11 | 13 | 15 | 17 | 19 |
| 7 | 9 | 11 | 13 | 15 | 17 | 19 | 21 |
| 9 | 11 | 13 | 15 | 17 | 19 | 21 | 23 |
| 11 | 13 | 15 | 17 | 19 | 21 | 23 | 25 |
| 13 | 15 | 17 | 19 | 21 | 23 | 25 | 27 |
| 15 | 17 | 19 | 21 | 23 | 25 | 27 | 29 |
| 17 | 19 | 21 | 23 | 25 | 27 | 29 | 31 |

18

# Quantization – what's new?

**DCT Matrix After Dequantization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 90 | 0 | -7 | 0 | 0 | 0 | 0 | 0 |
| -35 | -56 | 9 | 11 | 0 | 0 | 0 | 0 |
| -84 | 54 | 0 | -13 | 0 | 0 | 0 | 0 |
| -45 | -33 | 0 | 0 | 0 | 0 | 0 | 0 |
| -77 | -39 | 45 | 0 | 0 | 0 | 0 | 0 |
| -52 | 60 | 0 | 0 | 0 | 0 | 0 | 0 |
| -15 | 0 | -19 | 0 | 0 | 0 | 0 | 0 |
| -51 | 19 | 0 | 0 | 0 | 0 | 0 | 0 |

- The low-frequency elements have been modified, but only by small amounts.
- The high-frequency areas of the matrix have, for the most part, been reduced to zero, eliminating their effect on the decompressed image. In this sense, insignificant data has been discarded and the image information has been compressed.
- How are the values in the quantization matrix are selected? Two most common experimental approaches for testing the effectiveness:
  - Measure the mathematical error found between an input image and its output image after it has been decompressed, trying to determine an acceptable level of error.
  - Simply "eyeball it". Although judge the effect of decompression on the human eye is purely subjective.
- Although JPEG allows for the use of any quantization matrix, ISO has done extensive testing and developed a standard set of quantization values that cause impressive degrees of compression.

19

# JPEG compression continued

- **What's next?**

**DCT Matrix After Dequantization**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 90 | 0 | -7 | 0 | 0 | 0 | 0 | 0 |
| -35 | -56 | 9 | 11 | 0 | 0 | 0 | 0 |
| -84 | 54 | 0 | -13 | 0 | 0 | 0 | 0 |
| -45 | -33 | 0 | 0 | 0 | 0 | 0 | 0 |
| -77 | -39 | 45 | 0 | 0 | 0 | 0 | 0 |
| -52 | 60 | 0 | 0 | 0 | 0 | 0 | 0 |
| -15 | 0 | -19 | 0 | 0 | 0 | 0 | 0 |
| -51 | 19 | 0 | 0 | 0 | 0 | 0 | 0 |

20

# JPEG compression continued

- **Fourth** step, this makes RLE technique or/and Huffman encoding compression far more efficient.
- The result now is an encoded image with small size comparing with the source image.

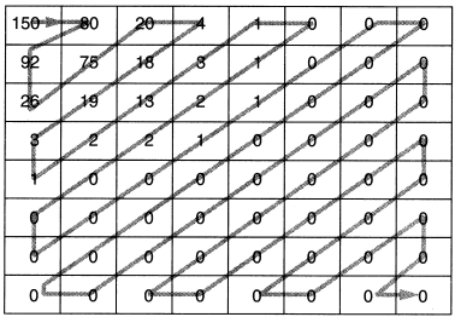
21

# JPEG compression continued

The final step of the JPEG image compression process is to compress the quantized DCT values. This is done through a three-part procedure detailed below.

**1. Convert the DC coefficient to a relative value –** First, the DC coefficient is changed from an absolute value to a relative value – relative to the DC coefficient in the previous 8 x 8 block, that is. Since **adjacent blocks** in an image exhibit a high degree of correlation, coding the DC term of a given block as the difference from the previous DC term typically produces a very small number that can be stored in a fewer number of bits.

**2. Reorder the DCT block in a zig-zag sequence** – So many coefficients in the DCT image are truncated to zero values during the coefficient quantization stage. They are coded using a Run-Length Encoding (RLE) algorithm. One way to increase the length of runs is to reorder the coefficients in the zig-zag sequence. This way, the JPEG algorithm moves through the block selecting the highest value elements first and eventually working its way to the lowest value elements, thus optimizing the effect of RLE.




**3. Entropy Encoding** – Finally, the JPEG algorithm outputs the DCT block's elements using an entropy encoding mechanism that combines the principles of RLE and Huffman encoding.

22

# Original and DCT coded block

An $8 \times 8$ block from the Y image of 'Lena'

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 200 | 202 | 189 | 188 | 189 | 175 | 175 | 175 |
| 200 | 203 | 198 | 188 | 189 | 182 | 178 | 175 |
| 203 | 200 | 200 | 195 | 200 | 187 | 185 | 175 |
| 200 | 200 | 200 | 200 | 197 | 187 | 187 | 187 |
| 200 | 205 | 200 | 200 | 195 | 188 | 187 | 175 |
| 200 | 200 | 200 | 200 | 200 | 190 | 187 | 175 |
| 205 | 200 | 199 | 200 | 191 | 187 | 187 | 175 |
| 210 | 200 | 200 | 200 | 188 | 185 | 187 | 186 |

$f(i,j)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 515 | 65 | -12 | 4 | 1 | 2 | -8 | 5 |
| -16 | 3 | 2 | 0 | 0 | -11 | -2 | 3 |
| -12 | 6 | 11 | -1 | 3 | 0 | 1 | -2 |
| -8 | 3 | -4 | 2 | -2 | -3 | -5 | -2 |
| 0 | -2 | 7 | -5 | 4 | 0 | -1 | -4 |
| 0 | -3 | -1 | 0 | 4 | 1 | -1 | 0 |
| 3 | -2 | -3 | 3 | 3 | -1 | -1 | 3 |
| -2 | 5 | -2 | 4 | -2 | 2 | -3 | 0 |

$F(u,v)$

Quantized and Reconstructed Blocks

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 32 | 6 | -1 | 0 | 0 | 0 | 0 | 0 |
| -1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| -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 |

$\hat{F}(u,v)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 512 | 66 | -10 | 0 | 0 | 0 | 0 | 0 |
| -12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| -14 | 0 | 16 | 0 | 0 | 0 | 0 | 0 |
| -14 | 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 |

$\tilde{F}(u,v)$

After IDCT and Difference from Original

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 199 | 196 | 191 | 186 | 182 | 178 | 177 | 176 |
| 201 | 199 | 196 | 192 | 188 | 183 | 180 | 178 |
| 203 | 203 | 202 | 200 | 195 | 189 | 183 | 180 |
| 202 | 203 | 204 | 203 | 198 | 191 | 183 | 179 |
| 200 | 201 | 202 | 201 | 196 | 189 | 182 | 177 |
| 200 | 200 | 199 | 197 | 192 | 186 | 181 | 177 |
| 204 | 202 | 199 | 195 | 190 | 186 | 183 | 181 |
| 207 | 204 | 200 | 194 | 190 | 187 | 185 | 184 |

$\tilde{f}(i,j)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1 | 6 | -2 | 2 | 7 | -3 | -2 | -1 |
| -1 | 4 | 2 | -4 | 1 | -1 | -2 | -3 |
| 0 | -3 | -2 | -5 | 5 | -2 | 2 | -5 |
| -2 | -3 | -4 | -3 | -1 | -4 | 4 | 8 |
| 0 | 4 | -2 | -1 | -1 | -1 | 5 | -2 |
| 0 | 0 | 1 | 3 | 8 | 4 | 6 | -2 |
| 1 | -2 | 0 | 5 | 1 | 1 | 4 | -6 |
| 3 | -4 | 0 | 6 | -2 | -2 | 2 | 2 |

$\epsilon(i,j) = f(i,j) - \tilde{f}(i,j)$

# Same steps on a less homogeneous block

Another $8 \times 8$ block from the Y image of 'Lena'

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 70 | 70 | 100 | 70 | 87 | 87 | 150 | 187 |
| 85 | 100 | 96 | 79 | 87 | 154 | 87 | 113 |
| 100 | 85 | 116 | 79 | 70 | 87 | 86 | 196 |
| 136 | 69 | 87 | 200 | 79 | 71 | 117 | 96 |
| 161 | 70 | 87 | 200 | 103 | 71 | 96 | 113 |
| 161 | 123 | 147 | 133 | 113 | 113 | 85 | 161 |
| 146 | 147 | 175 | 100 | 103 | 103 | 163 | 187 |
| 156 | 146 | 189 | 70 | 113 | 161 | 163 | 197 |

$f(i,j)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -80 | -40 | 89 | -73 | 44 | 32 | 53 | -3 |
| -135 | -59 | -26 | 6 | 14 | -3 | -13 | -28 |
| 47 | -76 | 66 | -3 | -108 | -78 | 33 | 59 |
| -2 | 10 | -18 | 0 | 33 | 11 | -21 | 1 |
| -1 | -9 | -22 | 8 | 32 | 65 | -36 | -1 |
| 5 | -20 | 28 | -46 | 3 | 24 | -30 | 24 |
| 6 | -20 | 37 | -28 | 12 | -35 | 33 | 17 |
| -5 | -23 | 33 | -30 | 17 | -5 | -4 | 20 |

$F(u,v)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -5 | -4 | 9 | -5 | 2 | 1 | 1 | 0 |
| -11 | -5 | -2 | 0 | 1 | 0 | 0 | -1 |
| 3 | -6 | 4 | 0 | -3 | -1 | 0 | 1 |
| 0 | 1 | -1 | 0 | 1 | 0 | 0 | 0 |
| 0 | 0 | -1 | 0 | 0 | 1 | 0 | 0 |
| 0 | -1 | 1 | -1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$\hat{F}(u,v)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| -80 | -44 | 90 | -80 | 48 | 40 | 51 | 0 |
| -132 | -60 | -28 | 0 | 26 | 0 | 0 | -55 |
| 42 | -78 | 64 | 0 | -120 | -57 | 0 | 56 |
| 0 | 17 | -22 | 0 | 51 | 0 | 0 | 0 |
| 0 | 0 | -37 | 0 | 0 | 109 | 0 | 0 |
| 0 | -35 | 55 | -64 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

$\tilde{F}(u,v)$

After IDCT and Difference from Original

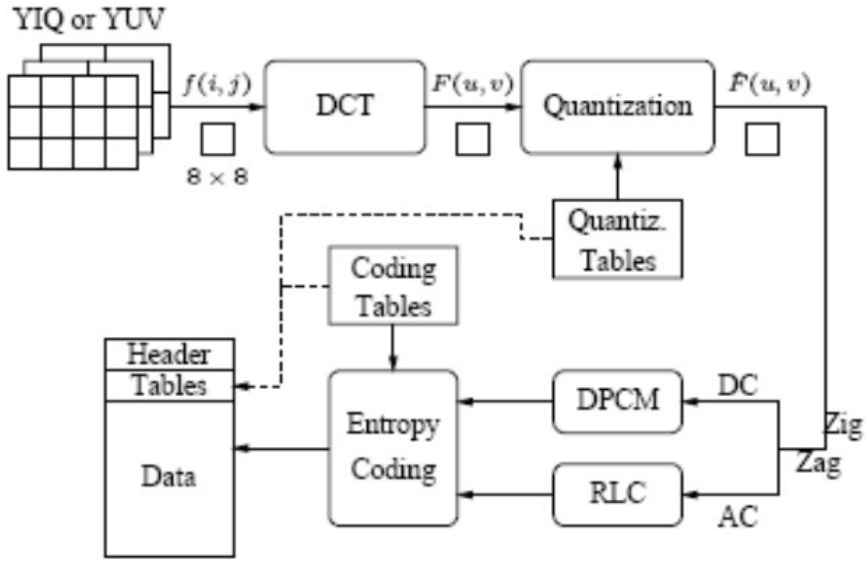
| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 70 | 60 | 106 | 94 | 62 | 103 | 146 | 176 |
| 85 | 101 | 85 | 75 | 102 | 127 | 93 | 144 |
| 98 | 99 | 92 | 102 | 74 | 98 | 89 | 167 |
| 132 | 53 | 111 | 180 | 55 | 70 | 106 | 145 |
| 173 | 57 | 114 | 207 | 111 | 89 | 84 | 90 |
| 164 | 123 | 131 | 135 | 133 | 92 | 85 | 162 |
| 141 | 159 | 169 | 73 | 106 | 101 | 149 | 224 |
| 150 | 141 | 195 | 79 | 107 | 147 | 210 | 153 |

$\tilde{f}(i,j)$

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 0 | 10 | -6 | -24 | 25 | -16 | 4 | 11 |
| 0 | -1 | 11 | 4 | -15 | 27 | -6 | -31 |
| 2 | -14 | 24 | -23 | -4 | -11 | -3 | 29 |
| 4 | 16 | -24 | 20 | 24 | 1 | 11 | -49 |
| -12 | 13 | -27 | -7 | -8 | -18 | 12 | 23 |
| -3 | 0 | 16 | -2 | -20 | 21 | 0 | -1 |
| 5 | -12 | 6 | 27 | -3 | 2 | 14 | -37 |
| 6 | 5 | -6 | -9 | 6 | 14 | -47 | 44 |

$\epsilon(i,j) = f(i,j) - \tilde{f}(i,j)$

# JPEG encoder




25

# JPEG encoding example code

```
%Modified based on
http://uk.mathworks.com/matlabcentral/fileexchange/38518-jpeg-
compression/content/jpegimplementation.m
clc;
clear all;
I = rgb2gray(imread('image4.jpg'));
[row coln]= size(I);
row=floor(row/8)*8;
coln=floor(coln/8)*8;
I1=I;
I= double(I);
%---------------------------------------------------
% Subtracting each image pixel value by 128
%---------------------------------------------------
I = I - 128;
quality = input('What quality of compression you require - ');

%---------------------------------------------------
% Quality Matrix Formulation
%---------------------------------------------------
Q50 = [ 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];

 if quality > 50
     QX = round(Q50.*(ones(8)*(100-quality)/50));
     QX = uint8(QX);
 elseif quality < 50
     QX = round(Q50.*(ones(8)*(50/quality)));
     QX = uint8(QX);
 elseif quality == 50
     QX = Q50;
 end
```

```
%---------------------------------------------------
% Formulation of forward DCT Matrix and inverse DCT matrix
%---------------------------------------------------
DCT_matrix8 = dct(eye(8));
iDCT_matrix8 = DCT_matrix8';      %inv(DCT_matrix8);

%---------------------------------------------------
% Jpeg Compression
%---------------------------------------------------
dct_restored = zeros(row,coln);
QX = double(QX);
%---------------------------------------------------
% Jpeg Encoding
%---------------------------------------------------
%---------------------------------------------------
% Forward Discret Cosine Transform
%---------------------------------------------------

for i1=[1:8:row]
    for i2=[1:8:coln]
        zBLOCK=I(i1:i1+7,i2:i2+7);
        win1=DCT_matrix8*zBLOCK*iDCT_matrix8;
        dct_domain(i1:i1+7,i2:i2+7)=win1;
    end
end
%---------------------------------------------------
% Quantization of the DCT coefficients
%---------------------------------------------------
for i1=[1:8:row]
    for i2=[1:8:coln]
        win1 = dct_domain(i1:i1+7,i2:i2+7);
        win2=round(win1./QX);
        dct_quantized(i1:i1+7,i2:i2+7)=win2;
    end
end
```

```
%---------------------------------------------------
% Jpeg Decoding
%---------------------------------------------------
% Dequantization of DCT Coefficients
%---------------------------------------------------
for i1=[1:8:row]
    for i2=[1:8:coln]
        win2 = dct_quantized(i1:i1+7,i2:i2+7);
        win3 = win2.*QX;
        dct_dequantized(i1:i1+7,i2:i2+7) = win3;
    end
end
%---------------------------------------------------
% Inverse DISCRETE COSINE TRANSFORM
%---------------------------------------------------
for i1=[1:8:row]
    for i2=[1:8:coln]
        win3 = dct_dequantized(i1:i1+7,i2:i2+7);
        win4=iDCT_matrix8*win3*DCT_matrix8;
        dct_restored(i1:i1+7,i2:i2+7)=win4;
    end
end
I2=dct_restored;

% ---------------------------------------------------
% Conversion of Image Matrix to Intensity image
%---------------------------------------------------

K=mat2gray(I2);

%---------------------------------------------------
%Display of Results
%---------------------------------------------------
figure(1),imshow(I1);title('original image');
figure(2),imshow(K);title('restored image from dct');
```

26

# JPEG encoding example code


27

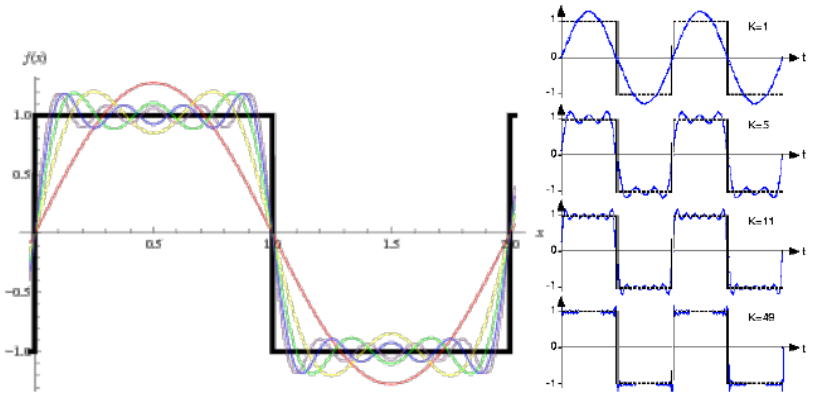
# JPEG encoding example code

- This image is compressed increasingly more from left to right.
- Note **ringing artefacts** and **blocking artefacts**.

http://www.cs.cf.ac.uk/Dave/Multimedia/PDF/11_JPEG.pdf

28

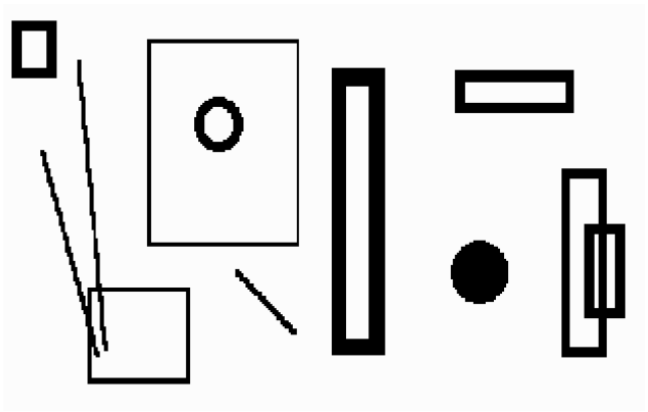
# JPEG encoding example code

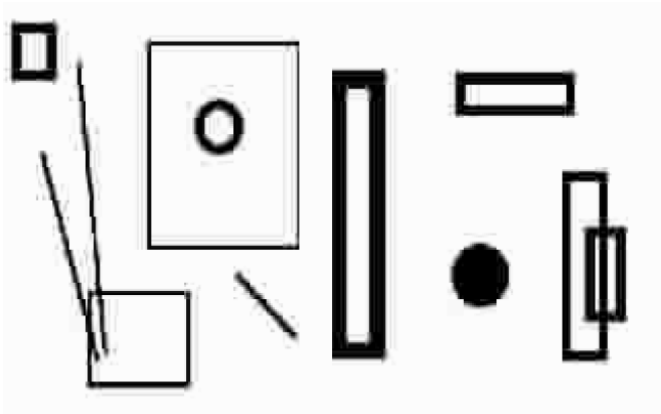



- Artefacts around sharp boundaries are due to Gibb's phenomenon.
- Basically: inability of a finite combination of cosines to describe jump discontinuities.

http://www.cs.cf.ac.uk/Dave/Multimedia/PDF/11_JPEG.pdf 29

# Gibb's phenomenon





http://www.cs.cf.ac.uk/Dave/Multimedia/PDF/11_JPEG.pdf 30