CS 414 – Multimedia Systems Design

Lecture 8 – JPEG Compression (Part 3)

Klara Nahrstedt

Spring 2012

Administrative

- MP1 is posted

Today Covered Topics

- Hybrid Coding:
    - JPEG Coding

- Reading: Section 7.5 out of “Media Coding and Content Processing”, Steinmetz & Nahrstedt, Prentice Hall 2002

Ubiquitous use of digital images

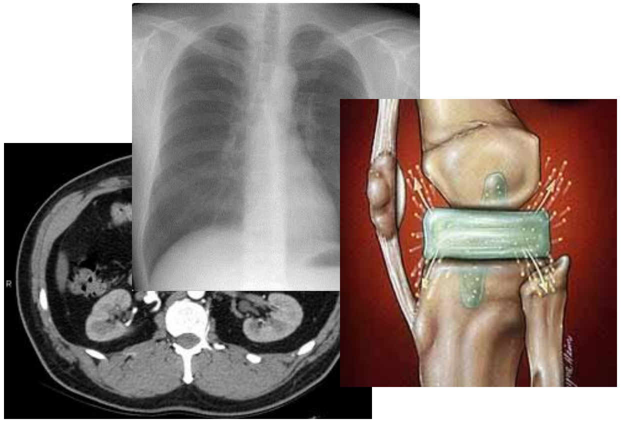
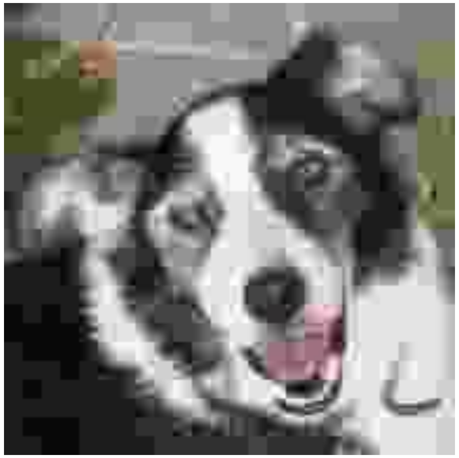


Image Compression and Formats

- RLE, Huffman, LZW (Lossless Coding)
- GIF
- **JPEG / JPEG-2000** (Hybrid Coding)

Original 60KB

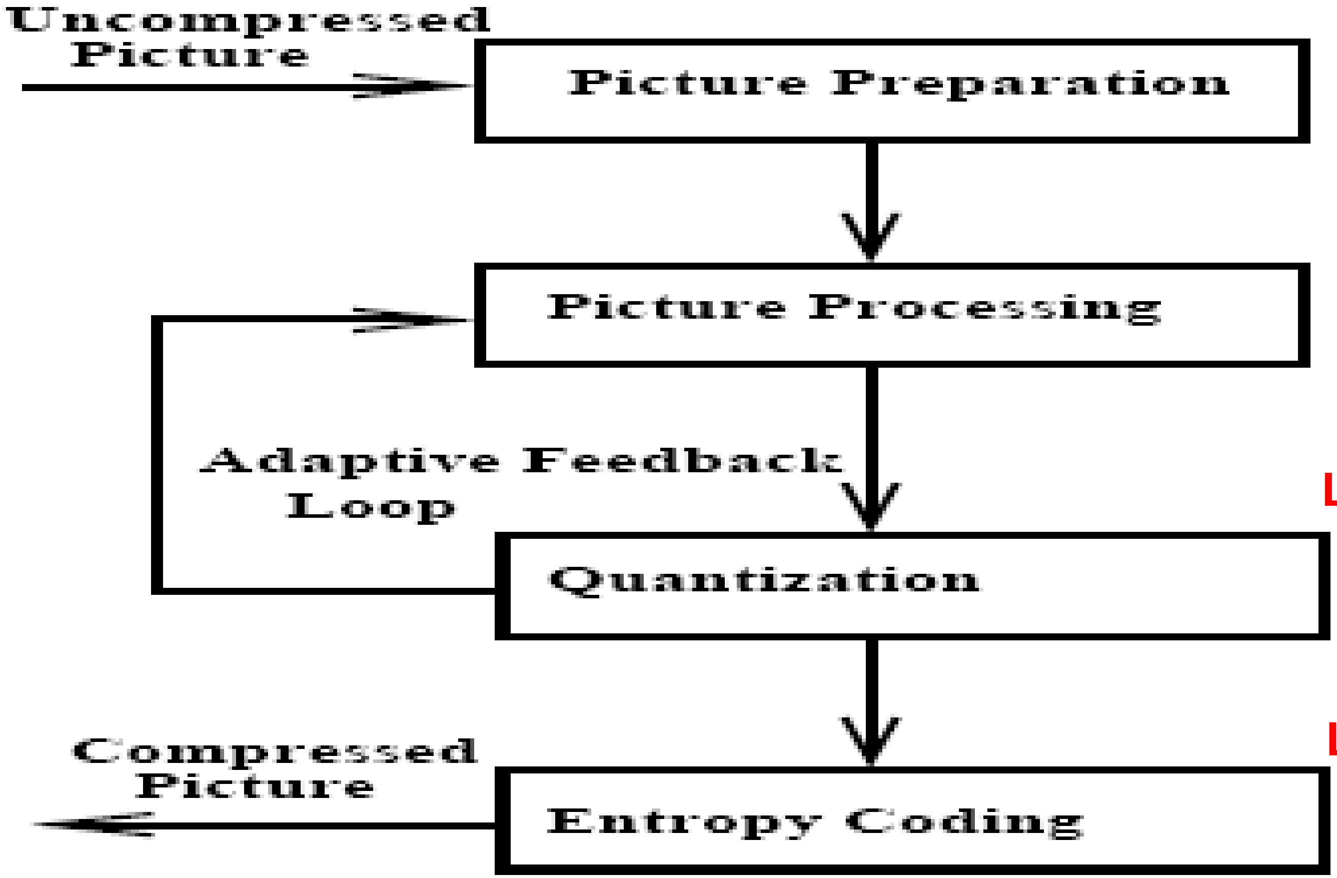


98% less 1KB

JPEG (Joint Photographic Experts Group)

- Requirements:
  - Very good compression ratio and good quality image
  - Independent of image size
  - Applicable to any image and pixel aspect ratio
  - Applicable to any complexity (with any statistical characteristics)

JPEG Belong to Hybrid Coding Schemes



Lossy Coding

Lossless Coding

RLE, Huffman or Arithmetic Coding

JPEG Compression (Baseline)

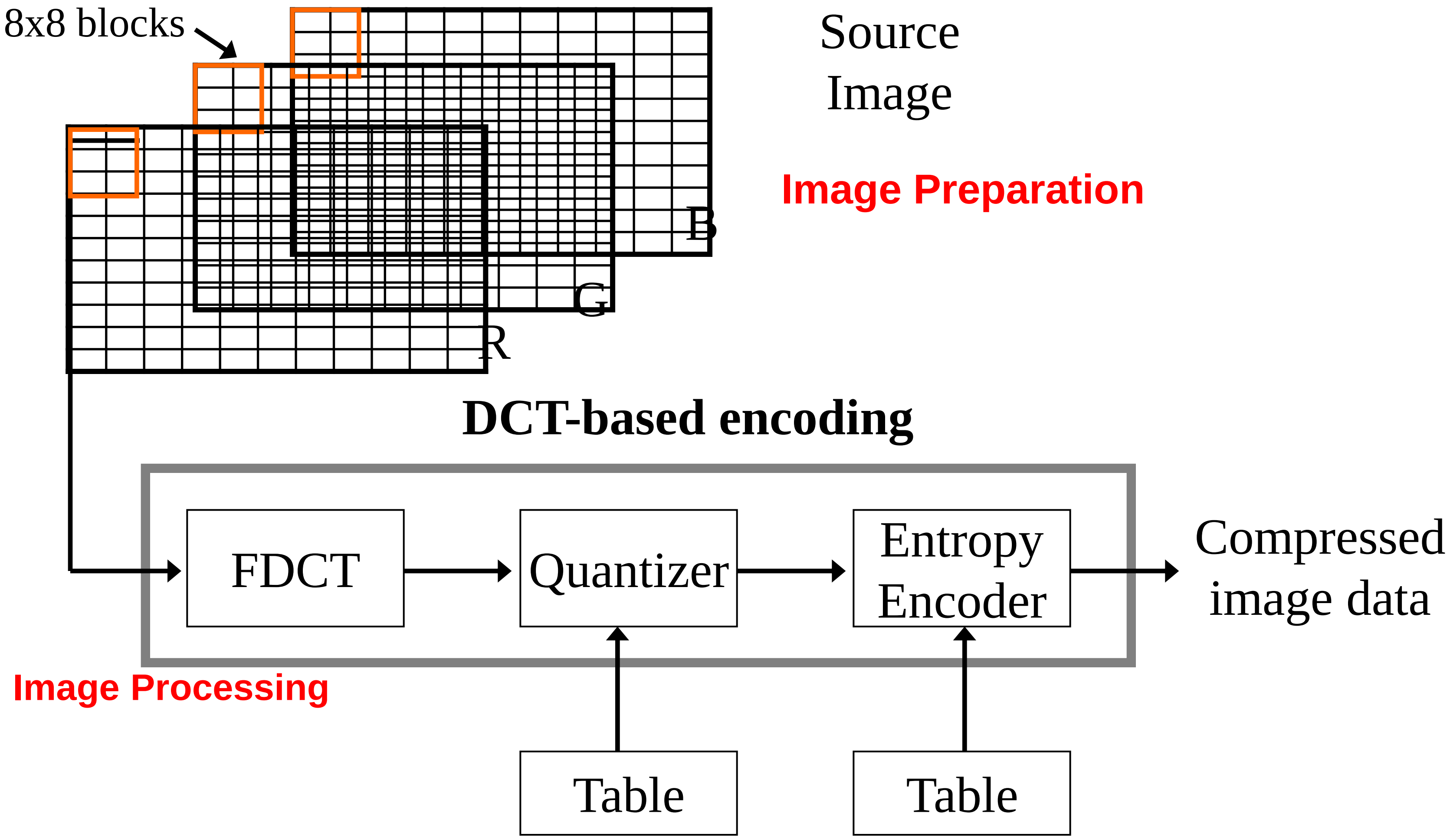


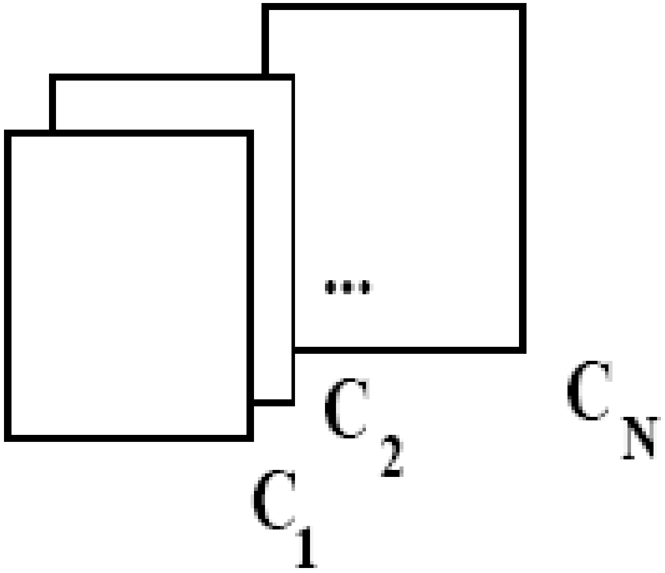
Image Preparation

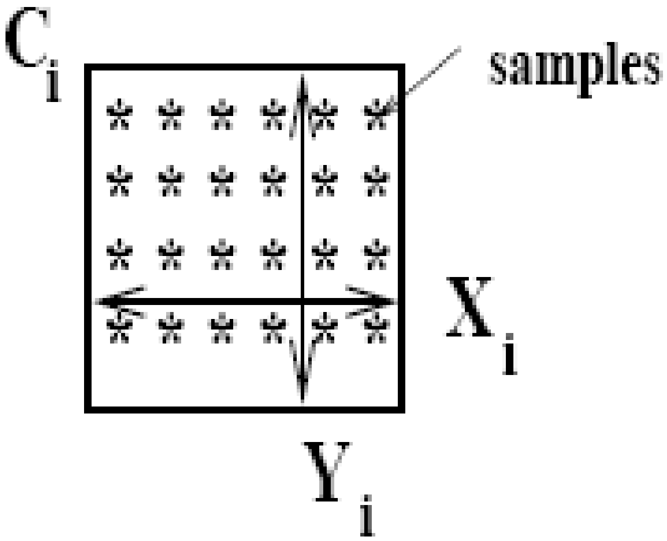
Image Processing

1. Image Preparation

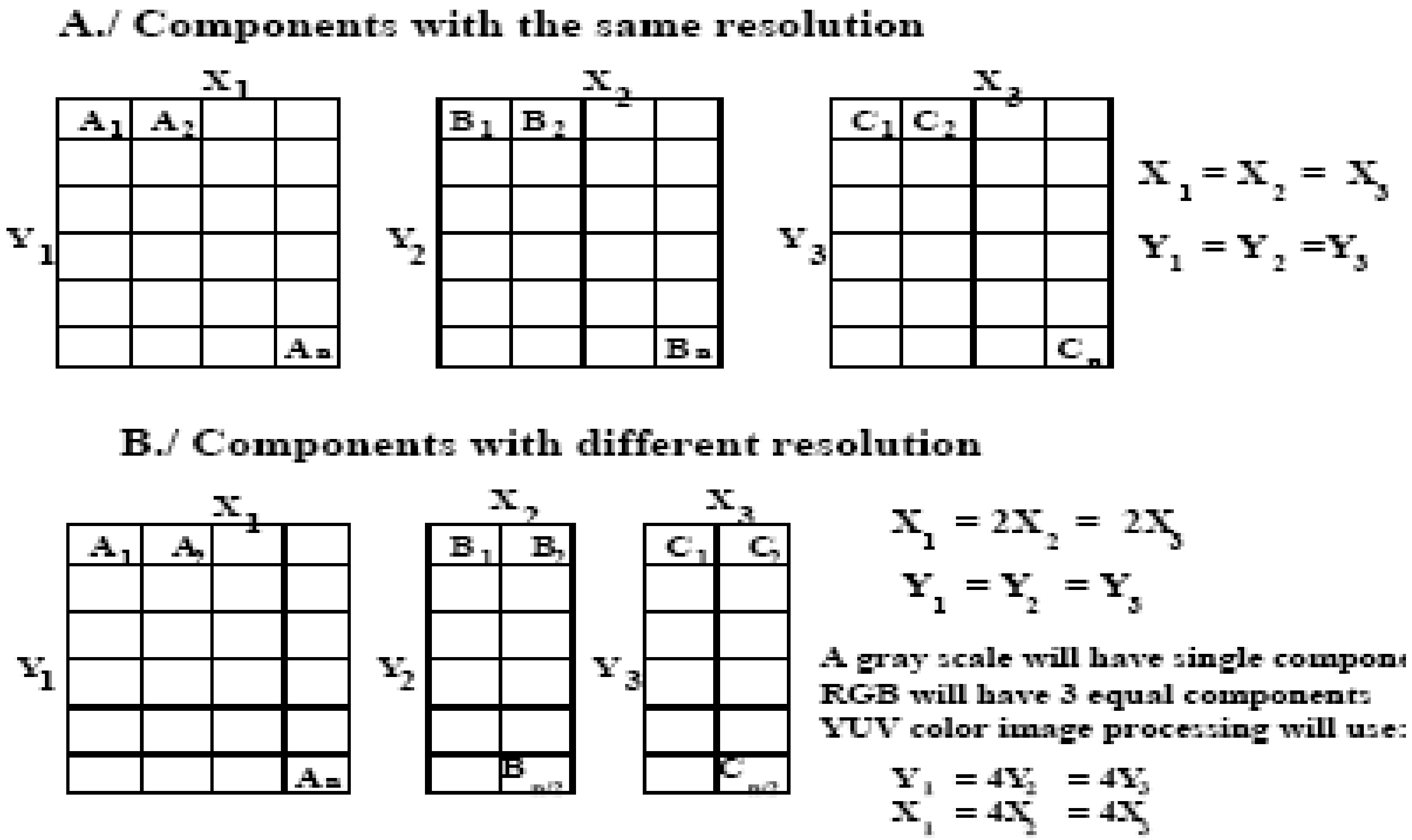
- The image preparation is **NOT BASED** on
  - 9-bit YUV encoding
  - Fixed number of lines and columns
  - Mapping of encoded chrominance
- Source image consists of components (C_i) and to each component we assign YUV, RGB or TIQ signals.

Division of Source Image into Planes





Components and their Resolutions



A./ Components with the same resolution

$X_1 = X_2 = X_3$

$Y_1 = Y_2 = Y_3$

B./ Components with different resolution

$X_1 = 2X_2 = 2X_3$

$Y_1 = Y_2 = Y_3$

A gray scale will have single compon
RGB will have 3 equal components
YUV color image processing will use:

$Y_1 = 4Y_2 = 4Y_3$
$X_1 = 4X_2 = 4X_3$

Color Transformation (optional)

- Down-sample chrominance components
  - compress without loss of quality (color space)
  - e.g., YUV 4:2:2 or 4:1:1

- Example: 640 x 480 RGB to YUV 4:1:1
  - Y is 640x480
  - U is 160x120
  - V is 160x120

Image Preparation (Pixel Allocation)

- Each pixel is presented by ‘p’ bits, value is in range of $(0,2^p-1)$
- All pixels of all components within the same image are coded with the same number of bits
- Lossy modes use precision 8 or 12 bits per pixel
- Lossless mode uses precision 2 up to 12 bits per pixel

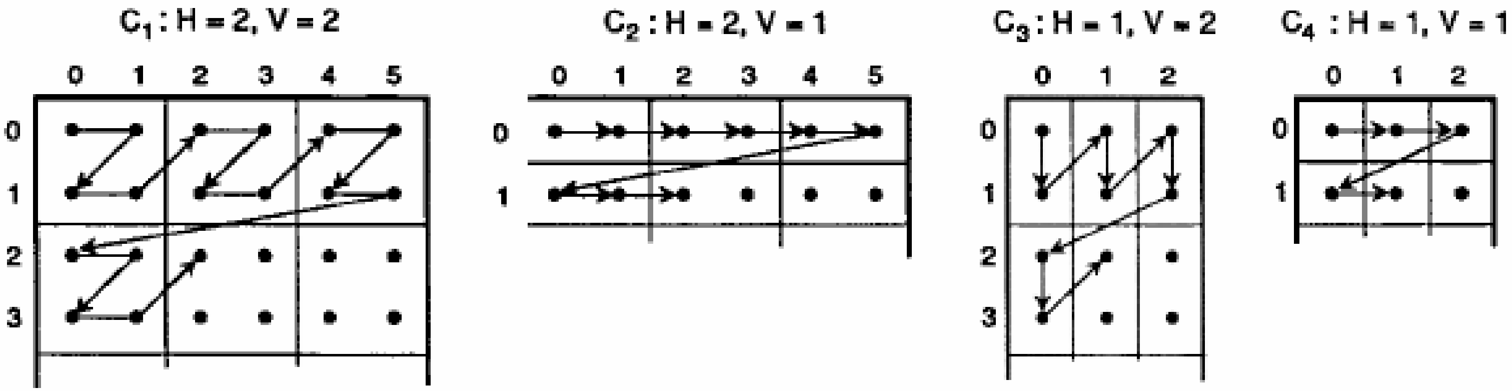
Image Preparation - Blocks

- Images are divided into data units, called blocks – definition comes from DCT transformation since DCT operates on blocks
- Lossy mode – blocks of 8x8 pixels; lossless mode – data unit 1 pixel

Data Unit Ordering

- *Non-interleaved*: scan from left to right, top to bottom for each color component
- *Interleaved*: compute one “unit” from each color component, then repeat
  - full color pixels after each step of decoding
  - but components may have different resolution

Example of Interleaved Ordering



$$\begin{array}{llll}
MCU_1 = d^1_{00}, d^1_{01}, d^1_{10}, d^1_{11}, & d^2_{00}, d^2_{01}, & d^3_{00}, d^3_{10}, & d^4_{00}, \\
MCU_2 = d^1_{02}, d^1_{03}, d^1_{12}, d^1_{13}, & d^2_{02}, d^2_{03}, & d^3_{01}, d^3_{11}, & d^4_{01}, \\
MCU_3 = d^1_{04}, d^1_{05}, d^1_{14}, d^1_{15}, & d^2_{04}, d^2_{05}, & d^3_{02}, d^3_{12}, & d^4_{02}, \\
MCU_4 = \underbrace{d^1_{20}, d^1_{21}, d^1_{30}, d^1_{31},}_{C_1 \text{ data units}} & \underbrace{d^2_{10}, d^2_{11},}_{C_2} & \underbrace{d^3_{20}, d^3_{30},}_{C_3} & \underbrace{d^4_{10},}_{C_4}
\end{array}$$

MCU: Minimum Coding Unit

[Wallace, 1991]

2. Image Processing

- Shift values [0, 2^P - 1] to [-2^{P-1}, 2^{P-1} - 1]
  - e.g. if (P=8), shift [0, 255] to [-127, 127]
  - DCT requires range be centered around 0
- Values in 8x8 pixel blocks are spatial values and there are 64 samples values in each block

Forward DCT

- Convert from spatial to frequency domain
  - convert **intensity function** into **weighted sum of periodic basis (cosine) functions**
  - identify **bands of spectral information** that can be thrown away without loss of quality
- Intensity values in each color plane often change slowly

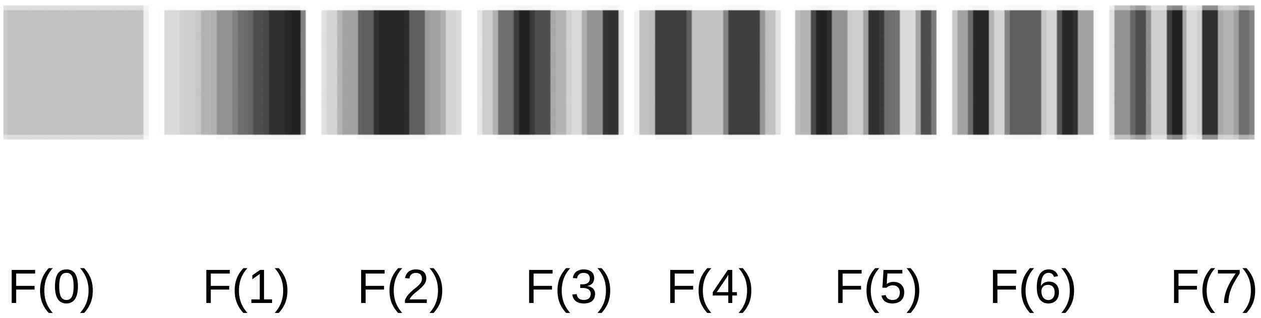
1D Forward DCT

- Given a list of *n* intensity values *I(x)*, where *x* = *0, …, n-1*
- Compute the *n* DCT coefficients:

$$F(u) = \sqrt{\frac{2}{n}} \quad C(u) \sum_{x=0}^{n-1} I(x) \cos \frac{(2x+1)\mu\pi}{2n}, u = 0 \ldots n-1$$

$$where \qquad C(u) = \begin{cases} \frac{1}{\sqrt{2}} & for \quad u = 0, \\ 1 & otherwise \end{cases}$$

Visualization of 1D DCT Basic Functions



1D Inverse DCT

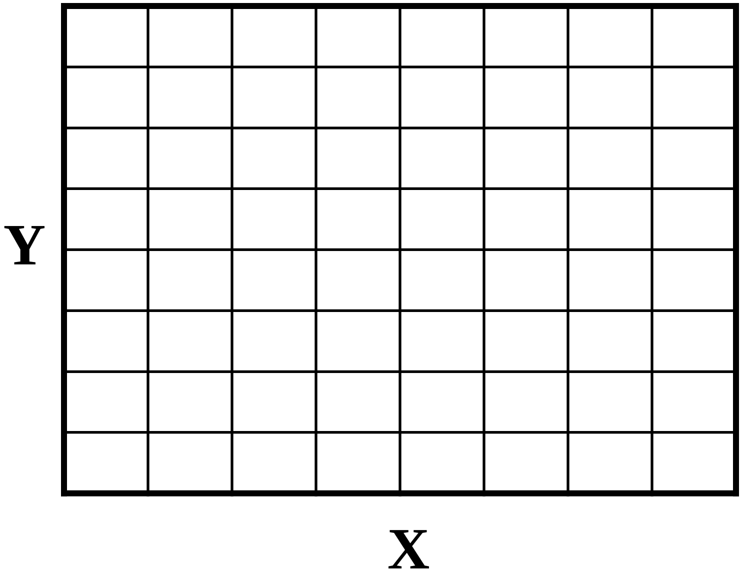
- Given a list of *n* DCT coefficients F(u), where u = 0, …, n-1
- Compute the *n* intensity values:

$$I(x) = \sqrt{\frac{2}{n}} \sum_{u=0}^{n-1} F(u)C(u)\cos\frac{(2x+1)\mu\pi}{2n}, x = 0 \ldots n-1$$

$$where \quad C(u) = \begin{cases} \frac{1}{\sqrt{2}} & for \quad u = 0, \\ 1 & otherwise \end{cases}$$

Extend DCT from 1D to 2D

- Perform 1D DCT on each row of the block
- Again for each column of 1D coefficients
  - alternatively, transpose the matrix and perform DCT on the rows



Equations for 2D DCT

- Forward DCT:

$$F(u,v)=\frac{2}{\sqrt{nm}}C(u)C(v)\sum_{y=0}^{m-1}\sum_{x=0}^{n-1}I(x,y)*\cos\left(\frac{(2x+1)u\pi}{2n}\right)*\cos\left(\frac{(2y+1)v\pi}{2m}\right)$$

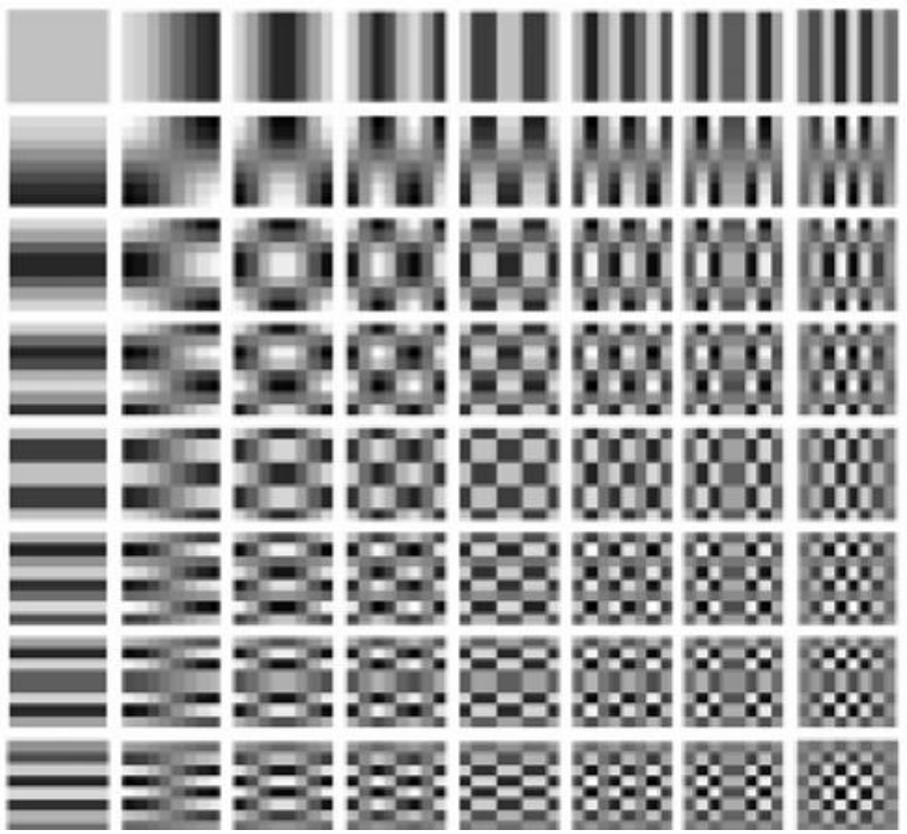
- Inverse DCT:

$$I(y,x)=\frac{2}{\sqrt{nm}}\sum_{v=0}^{m-1}\sum_{u=0}^{n-1}F(v,u)C(u)C(v)\cos\left(\frac{(2x+1)u\pi}{2n}\right)*\cos\left(\frac{(2y+1)v\pi}{2m}\right)$$

Visualization of 2D DCT Basis Functions

Increasing frequency

Increasing frequency



Coefficient Differentiation

- F(0,0)
  - includes the lowest frequency in both directions
  - is called **DC coefficient**
  - Determines fundamental color of the block
- F(0,1) …. F(7,7)
  - are called **AC coefficients**
  - Their frequency is non-zero in one or both directions

3. Quantization

- Throw out bits
- Consider example: 101101_2 = 45 (6 bits)
  - □ We can truncate this string to 4 bits: 1011_2 = 11
  - □ We can truncate this string to 3 bits: 101_2 = 5 (original value 40) or 110_2 = 6 (original value 48)
- Uniform quantization is achieved by dividing DCT coefficients by N and round the result (e.g., above we used N=4 or N=8)
- In JPEG – use quantization tables
  - □ $Fq(u,v) = F(u,v)/Q_{uv}$
  - □ Two quantization tables – one for luminance and one for two chrominance components

De facto Quantization Table

Eye becomes less sensitive

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

Eye becomes less sensitive

4. Entropy Encoding

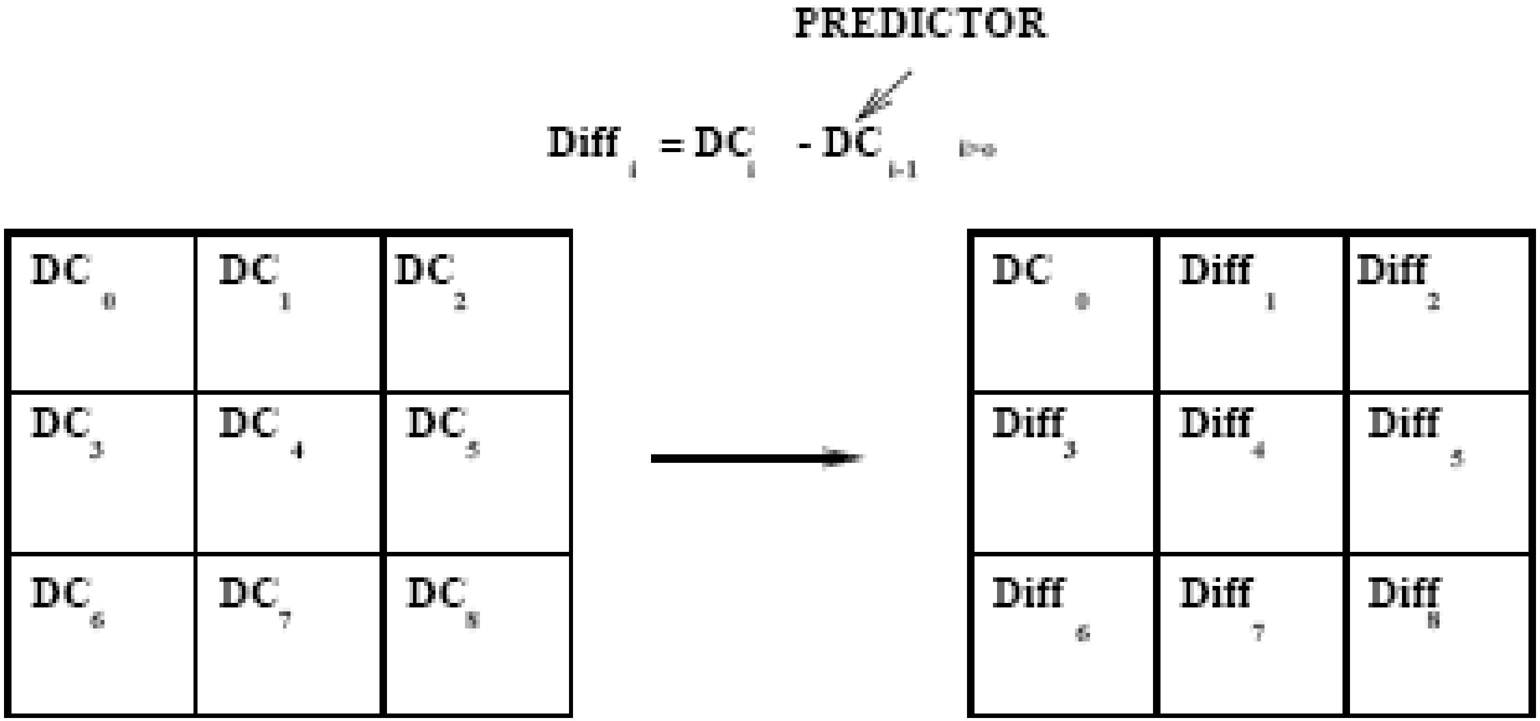
- Compress sequence of **quantized DC and AC coefficients** from quantization step
  - further increase compression, without loss
- Separate DC from AC components
  - DC components change slowly, thus will be encoded using difference encoding

DC Encoding

- DC represents **average intensity of a block**
  - encode using difference encoding scheme

- Because difference tends to be near zero, can use less bits in the encoding
  - categorize difference into difference classes
  - send the index of the difference class, followed by bits representing the difference

Difference Coding applied to DC Coefficients



PREDICTOR

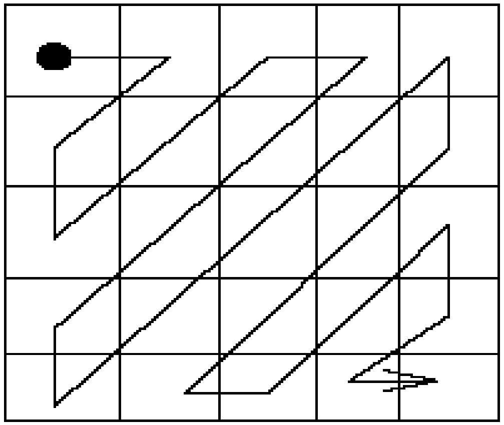
$\text{Diff}_i = \text{DC}_i - \text{DC}_{i-1} \quad i>0$

DC_0	DC_1	DC_2
DC_3	DC_4	DC_5
DC_6	DC_7	DC_8

DC_0	Diff_1	Diff_2
Diff_3	Diff_4	Diff_5
Diff_6	Diff_7	Diff_8

AC Encoding

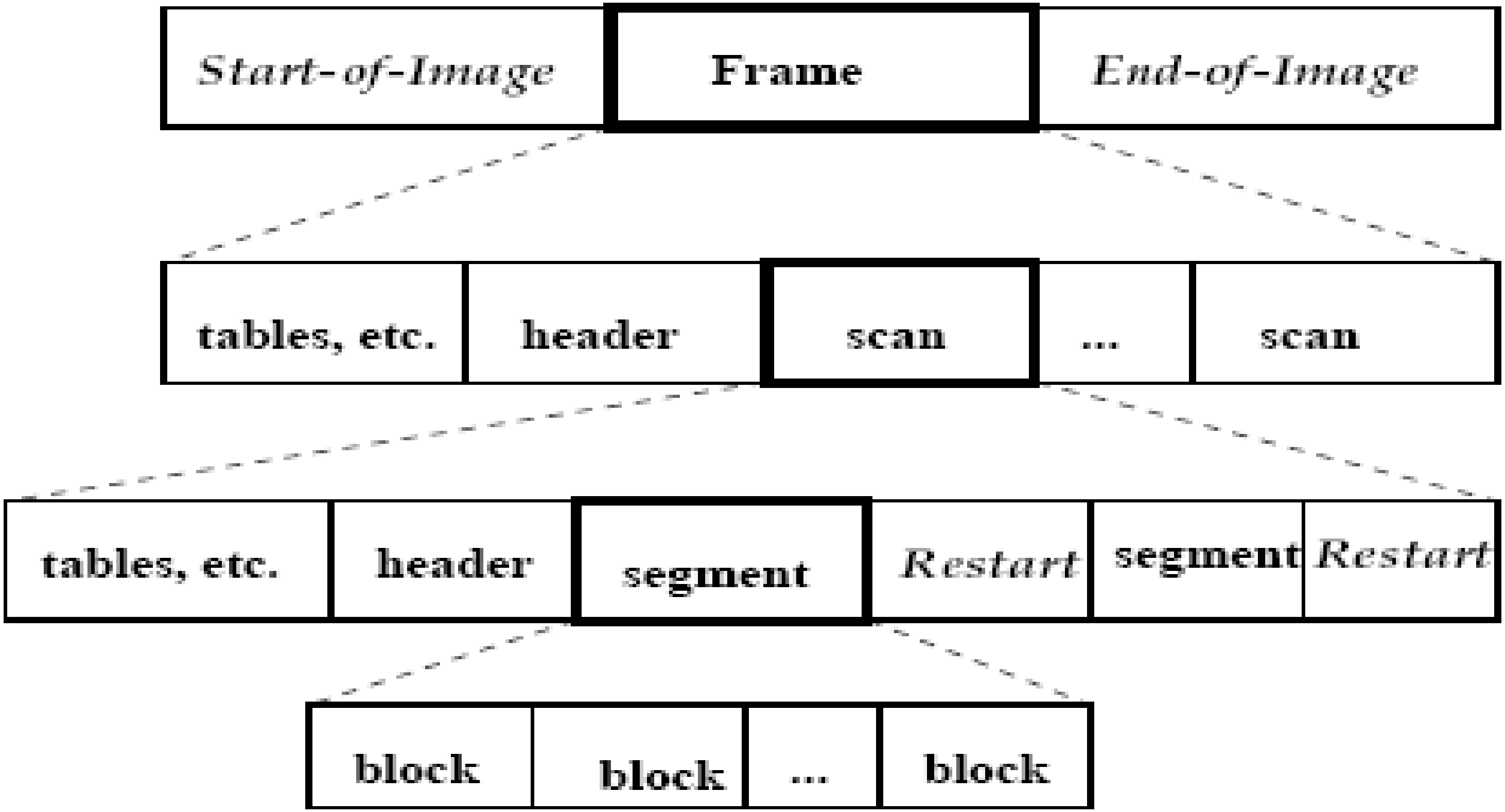
- Use **zig-zag ordering** of coefficients
  - orders frequency components from low->high
  - produce maximal series of 0s at the end
  - Ordering helps to apply efficiently entropy encoding
- Apply **Huffman coding**
- Apply **RLE** on AC zero values



Huffman Encoding

- Sequence of DC difference indices and values along with RLE of AC coefficients
- Apply Huffman encoding to sequence
- Attach appropriate headers
- Finally have the JPEG image!

Interchange Format of JPEG



Start-of-Image **Frame** *End-of-Image*

tables, etc. **header** **scan** **...** **scan**

tables, etc. **header** **segment** *Restart* **segment** *Restart*

block **block** **...** **block**

Example - One Everyday Photo

2.76M

Example - One Everyday Photo

600K

Example - One Everyday Photo

350K

Example - One Everyday Photo

240K

Example - One Everyday Photo

144K

Example - One Everyday Photo

88K

Discussion

- What types of image content would JPEG work best (worst) for?
- Is image compression solved?
- What’s missing from JPEG?