ISO IS-10918 - Understanding JPEG Image Compression

Introduction

The complete picture of the image starts from the color representation. Thus, we begin with analysis of color spaces and then start the examination of transform image coding, quantization, and entropy coding.

Color Space

Since Newton's time, it has been known that a wide spectrum of colors can be generated from a set of three primaries. An artist, for example, can paint most colors from layering pigments of the subtractive primaries: red, yellow, and blue. Television displays generate colors by mixing lights of the additive primaries: red, green, and blue. Although two primary systems can be used-- most generally a red-orange and a green-blue-- the image rendered is not lifelike, and two color systems never became successful in either motion picture or pre-standard television[3].

The color space obtained through combining the three colors can be determined by drawing a triangle on a special color chart with each of the base colors as an endpoint. The classic color chart used in early television specifications was established in 1931 by the Commission Internationale de L'Eclairage (CIE).

One of the special concepts introduced by the 1931 CIE chart was the isolation of the luminance, or brightness, from the chrominance, or hue. Using the CIE chart as a guideline, the National Television System Committee (NTSC) defined the transmission of signals in a luminance and chrominance format, rather than a format involving the three color components of the television phosphors. The new color space was labeled YIQ, representing the luminance, in-phase chrominance, and quadrature chrominance coordinates respectively.

1 2 3

In Europe, two television standards were later established, the phase-alternation-line (PAL) format and the Sequentiel couleura memoire (SECAM) format, both with an identical color space, YUV. The only change between the PAL/SECAM YUV color space and the NTSC YIQ color space is a 33 degree rotation in UV space[4].

The conversion between the standard Red Green Blue (RGB) format to YUV format is slightly different for digital signals than for analog signals. For the JPEG JFIF[13] format convention, the full range of 8 bits is used for Y, U, and V. The digital conversion insures that if the RGB inputs are between 0 and 255, (normalized so that equal values represents reference white), then the Y output has values between 0-255 and the U and V have values between 0 and $\pm$128. The values are stored as 8 bit unsigned characters, thus the U and V values are level shifted by adding 128 so that the values are always non-negative, and if U and V equal 256, they are clamped to 255. The conversion, written in a matrix form, is

$R, G, B \in [0, 1]$, $I \in [-0.5957, 0.5957]$, $Q \in [-0.5226, 0.5226]$

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.595716 & -0.274453 & -0.321263 \\ 0.211456 & -0.522591 & 0.311135 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1 & 0.9563 & 0.6210 \\ 1 & -0.2721 & -0.6474 \\ 1 & -1.1070 & 1.7046 \end{bmatrix} \begin{bmatrix} Y \\ I \\ Q \end{bmatrix}$$

4.2. Image Compression -- JPEG

Reference: W.B. Pennebaker, J.L. Mitchell, "The JPEG Still Image Data Compression Standard", Van Nostrand Reinhold, 1993.

4.2.1. Overview of JPEG

What is JPEG?

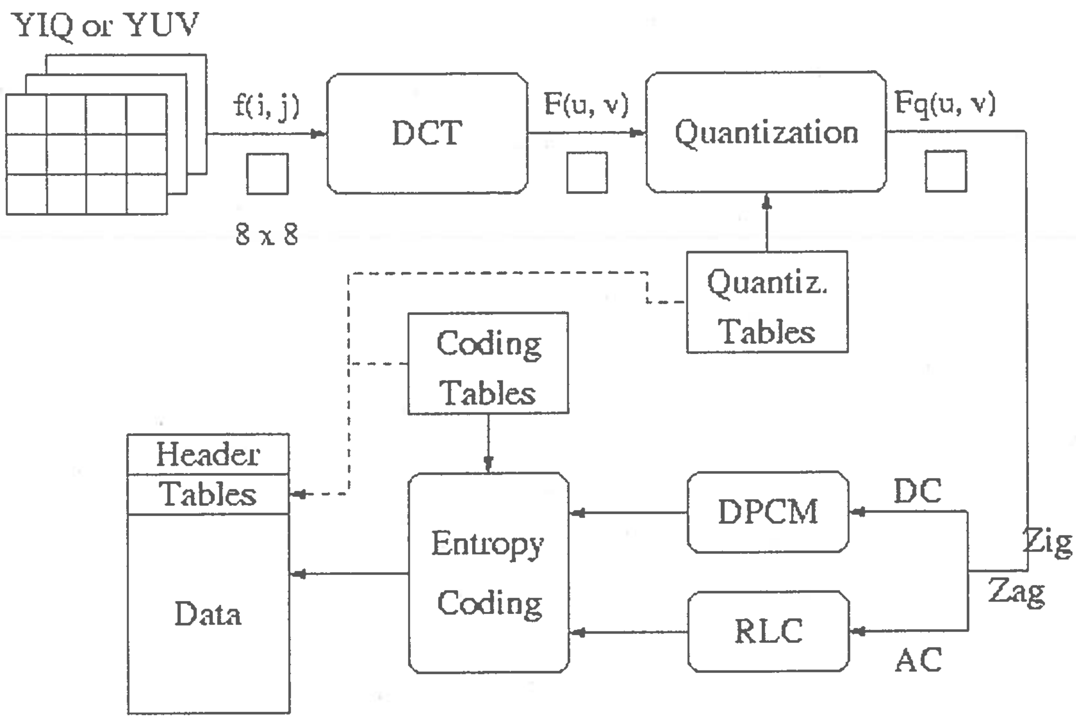
- "Joint Photographic Expert Group". Voted as international standard in 1992.
- Works with color and grayscale images, e.g., satellite, medical, ...

Motivation

- The *compression ratio* of lossless methods (e.g., Huffman, Arithmetic, LZW) is not high enough for image and video compression, especially when the distribution of pixel values is relatively flat.
- JPEG uses *transform coding*, it is largely based on the following observations:
  - Observation 1: A large majority of useful image contents change relatively slowly across images, i.e., it is unusual for intensity values to alter up and down several times in a small area, for example, within an 8 x 8 image block. Translate this into the spatial frequency domain, it says that, generally, lower spatial frequency components contain more information than the high frequency components which often correspond to less useful details and noises.
  - Observation 2: Pshchophysical experiments suggest that humans are more receptive to the loss of higher spatial frequency components than the loss of lower frequency components.

JPEG overview

- Encoding



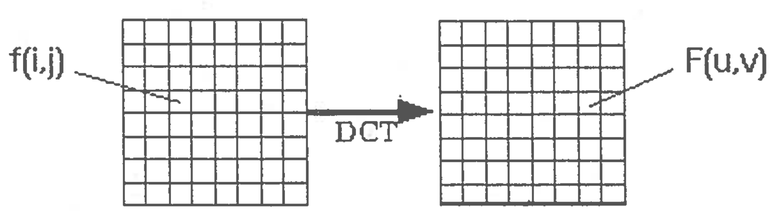
- Decoding -- Reverse the order

4.2.2. Major Steps

- DCT (Discrete Cosine Transformation)
- Quantization
- Zigzag Scan
- DPCM on DC component
- RLE on AC Components
- Entropy Coding

1. Discrete Cosine Transform (DCT)

- From spatial domain to frequency domain:



- **DEFINITIONS**

  Discrete Cosine Transform (DCT):

$$F(u,v) = \frac{\Lambda(u)\Lambda(v)}{4} \sum_{i=0}^{7}\sum_{j=0}^{7} \cos\frac{(2i+1)\cdot u\pi}{16} \cdot \cos\frac{(2j+1)\cdot v\pi}{16} \cdot f(i,j)$$

$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & for\ \xi = 0 \\ 1 & otherwise \end{cases}$$

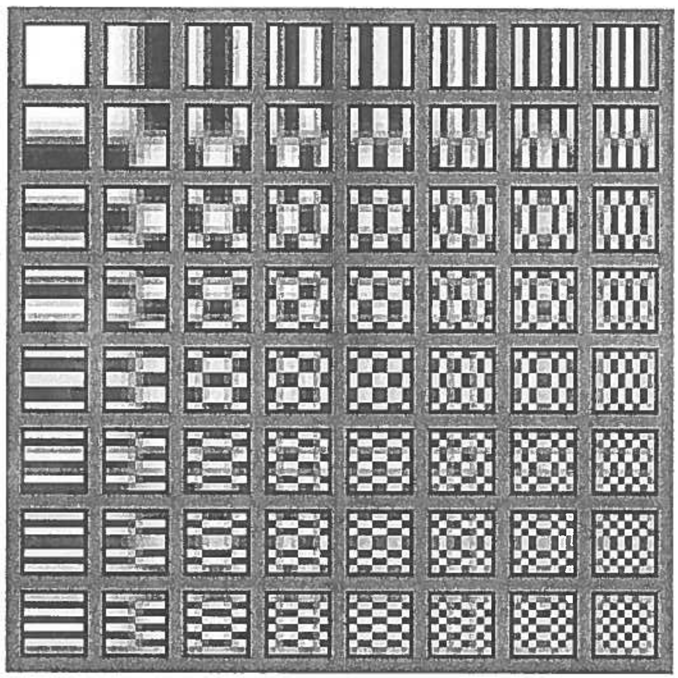
Inverse Discrete Cosine Transform (IDCT):

$$\tilde{f}(i,j) = \frac{1}{4} \sum_{u=0}^{7}\sum_{v=0}^{7} \Lambda(u)\Lambda(v) \cos\frac{(2i+1)\cdot u\pi}{16} \cdot \cos\frac{(2j+1)\cdot v\pi}{16} \cdot F(u,v)$$

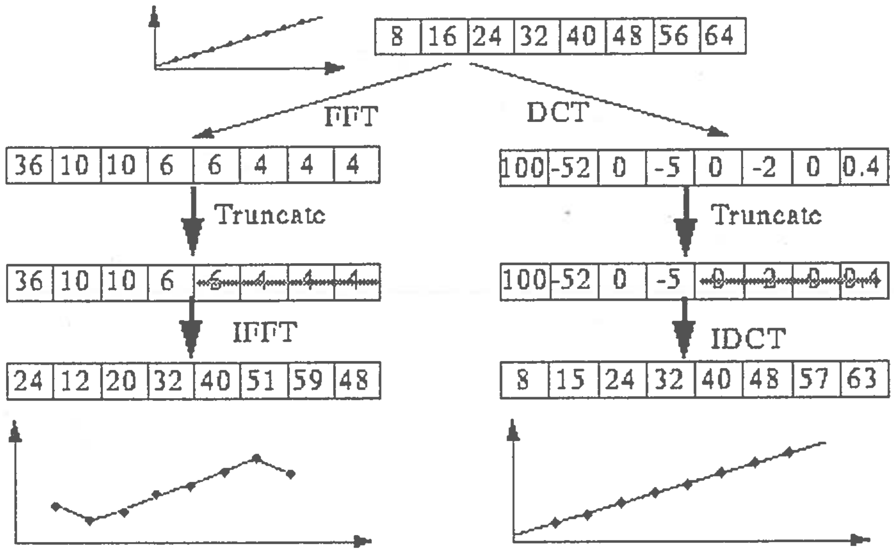
$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & for\ \xi = 0 \\ 1 & otherwise \end{cases}$$

Question: What are the DC and AC components, e.g., what is F[0,0]?

- The 64 (8 x 8) DCT basis functions:



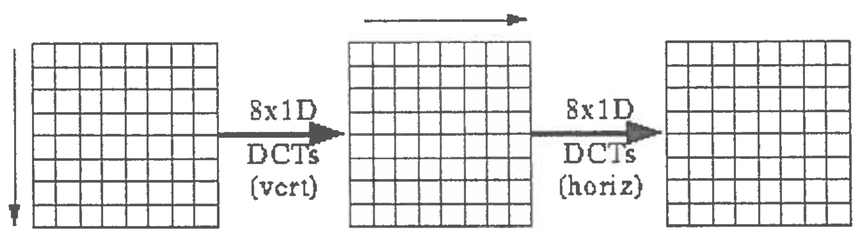
- Why DCT not FFT? -- DCT is like FFT, but can approximate linear signals well with few coefficients.



- Computing the DCT
  - Factoring reduces problem to a series of 1D DCTs:

$$F[u,v] = \frac{1}{2}\sum_{i} A(u)\cos\frac{(2i+1)u\pi}{16}G[i,v]$$

$$G[i,v] = \frac{1}{2}\sum_{j} A(v)\cos\frac{(2j+1)v\pi}{16}f[i,j]$$



  - Most software implementations use fixed point arithmetic. Some fast implementations approximate coefficients so all multiplies are shifts and adds.

2. Quantization

- F'[u, v] = round (F[u, v] / q[u, v]).

  Why? -- To reduce number of bits per sample

  Example: 101101 = 45 (6 bits).
  q[u, v] = 4 --> Truncate to 4 bits: 1011 = 11.

- Quantization error is the main source of the Lossy Compression.

Uniform Quantization

- Each F[u,v] is divided by the same constant *N*.

Non-uniform Quantization -- Quantization Tables

- Eye is most sensitive to low frequencies (upper left corner), less sensitive to high frequencies (lower right corner)

- The *Luminance Quantization Table* q(u, v) The *Chrominance Quantization Table* q(u, v)

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

17	18	24	47	99	99	99	99
18	21	26	66	99	99	99	99
24	26	56	99	99	99	99	99
47	66	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99
99	99	99	99	99	99	99	99

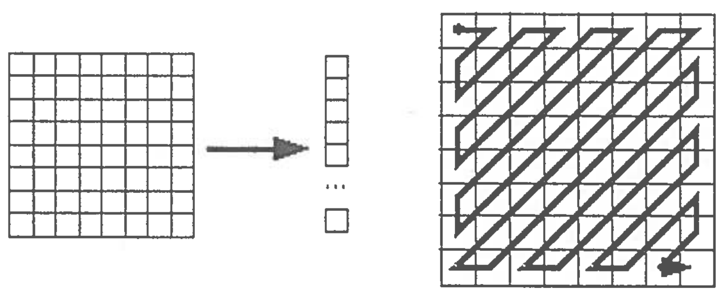
The numbers in the above quantization tables can be scaled up (or down) to adjust the so called <u>*quality factor*</u>.

Custom quantization tables can also be put in image/scan header.

3. Zig-zag Scan

- Why? -- to group low frequency coefficients in top of vector.

- Maps 8 x 8 to a 1 x 64 vector



4. Differential Pulse Code Modulation (DPCM) on DC component

- DC component is large and varied, but often close to previous value.

- Encode the difference from previous 8 x 8 blocks -- DPCM

5. Run Length Encode (RLE) on AC components

- 1 x 64 vector has lots of zeros in it

- Keeps *skip* and *value*, where *skip* is the number of zeros and *value* is the next non-zero component.

- Send (0,0) as end-of-block sentinel value.

6. Entropy Coding

- Categorize DC values into SIZE (number of bits needed to represent) and actual bits.

SIZE	Value
1	-1, 1
2	-3, -2, 2, 3
3	-7..-4, 4..7
4	-15..-8, 8..15
.	.
.	.
.	.
10	-1023..-512, 512..1023

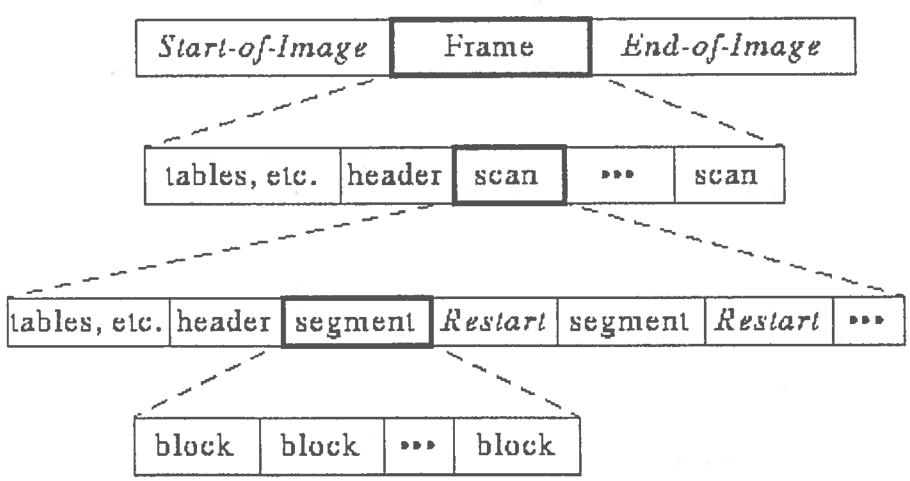
  Example: if DC value is 4, 3 bits are needed.

  Send off SIZE as Huffman symbol, followed by actual 3 bits.

- For AC components two symbols are used: Symbol_1: (*skip, SIZE*), Symbol_2: actual bits. Symbol_1 (*skip, SIZE*) is encoded using the Huffman coding, Symbol_2 is not encoded.

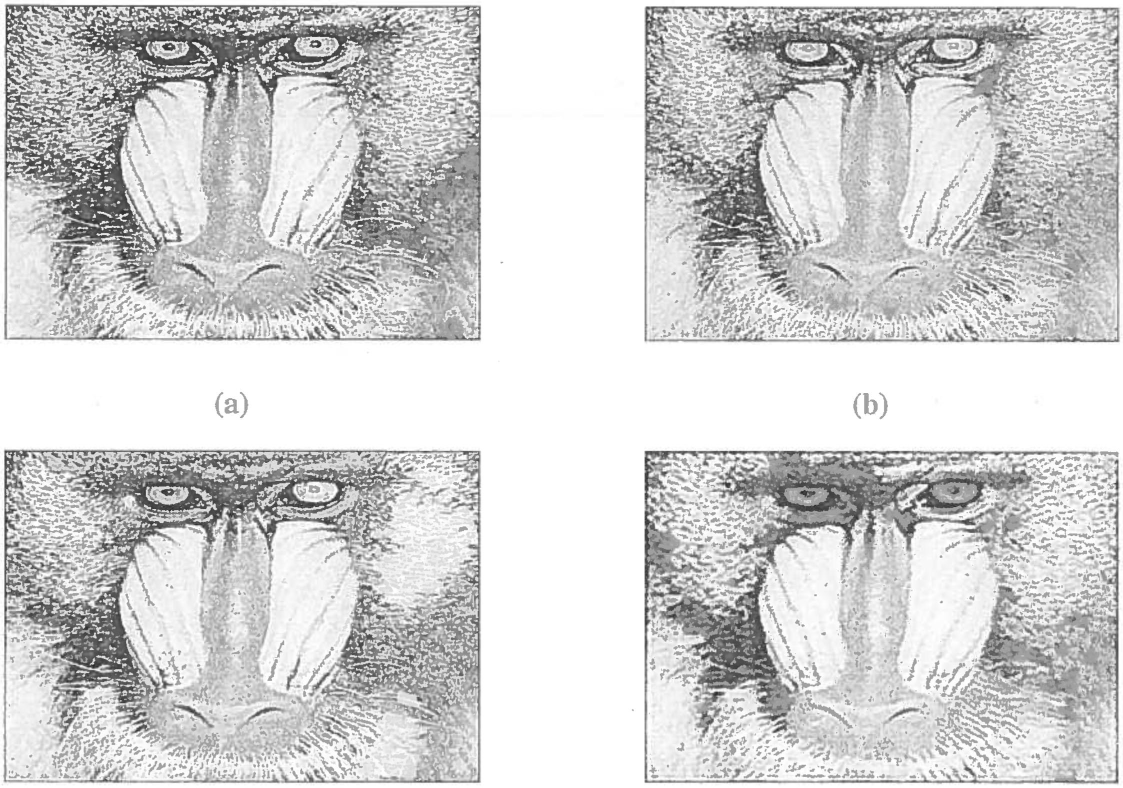
- Huffman Tables can be custom (sent in header) or default.

4.2.3. A Glance at the JPEG Bitstream



- A "Frame" is a picture, a "scan" is a pass through the pixels (e.g., the red component), a "segment" is a group of blocks, a "block" is an 8 x 8 group of pixels.

- Frame header:
  sample precision



(a) (b) (c) (d)

Figure 14. Inverse DCT of Baboon; (a) DCT(100%); (b) DCT(75%); (c) DCT(50%); (d) DCT(25%).