PROGETTAZIONE E PRODUZIONE MULTIMEDIALE

Prof. Alberto Del Bimbo
Dip. Sistemi e Informatica
Univ. degli Studi di Firenze

INTRODUCTION - Part II

- C – Image encoding

Compression techniques

- La scelta della tecnica di compressione da impiegare dipende dal contesto applicativo e da una serie di parametri legati al tempo di compressione al fattore di compressione ed alla fedelta' dell'informazione ripristinata. In generale:
  - documenti testuali: compressione lossless
  - dati per analisi numerica: compressione lossless
  - Programmi: compressione lossless
  - immagini destinate a tipografia: compressione lossless
  - immagini destinate al WEB: compressione lossy
  - Video: compressione lossy
  - Audio: compressione lossy

- L'impiego di tecniche lossy e' in molti casi obbligatorio e rappresenta l'unica possibile soluzione qualora si vogliano dei fattori di compressione molto elevati.

- **Immagine 146 x 184 con 75 colori. Dimensione file:**
  - **Teorica con 8 bpp: 26864 byte**
  - **Teorica con 24 bpp: 80592 byte**
  - **PPM (24 bpp) : 80674 byte**
  - **GIF (8 bpp): 3585 byte (FC=22.48)**
  - **JPG (24 bpp): 4805 byte (FC=16.77)**
  - **PPM.ZIP (24 bpp): 3698 byte (FC=21.79)**

Immagine 244 x 334 con 31322 colori. Dimensione file:

Teorica con 8 bpp: 81496 byte
Teorica con 24 bpp: 244488 byte
PPM (24 bpp) : 244620 byte
GIF (8 bpp): 49613 byte (FC=4.92)
JPG (24 bpp): 16352 byte (FC=14.95)
PPM.ZIP (24 bpp): 190977 byte (FC=1.28)

Lossless compression

RUN-LENGTH ENCODING

- Con questa tecnica si sostituisce una sequenza di simboli uguali con un solo simbolo accompagnato dal numero di volte che questo compare consecutivamente:

  ppppppppppp ppppppppppp ppppppppppp diventa $11p $11p $11p

- E' necessario definire l'impiego di un carattere speciale (in questo caso $). L'intero file puo' essere codificato con la sequenza:

  $11r $11r $11r
  $11p $11p $11p
  $11c $11c $11c

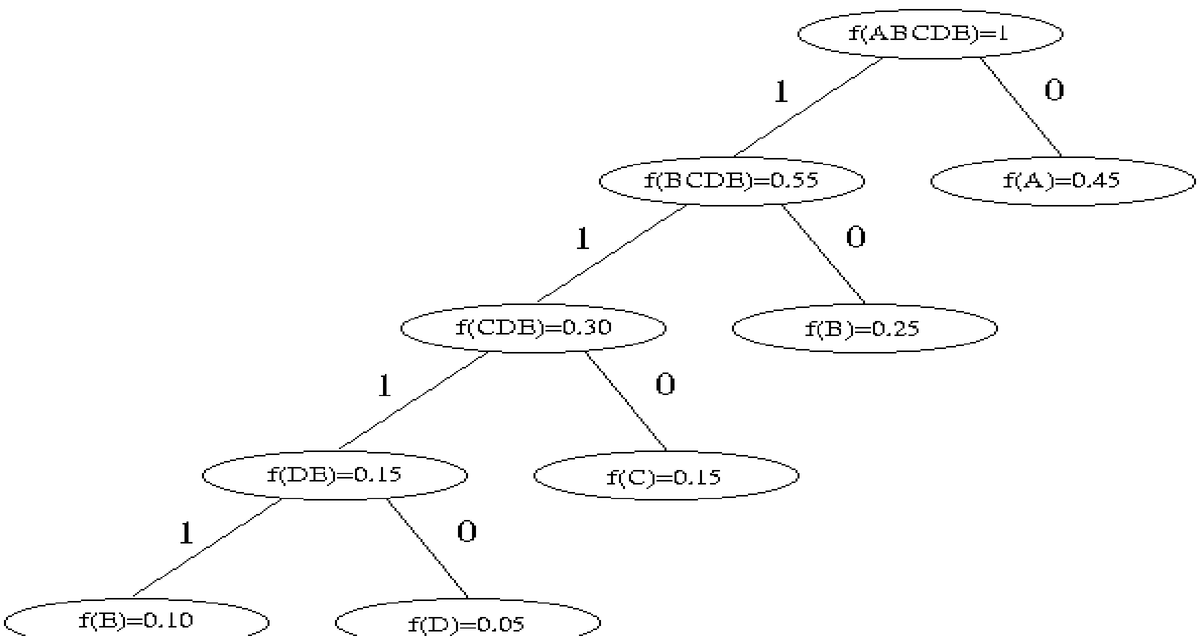
  Totale caratteri: 9+9+9 = 27
  Totale numeri: 3+3+3=9
  Dimensione file: 36Byte = 288bit

- Questa tecnica puo' essere impiegata anche per la codifica di immagini, traendo vantaggio dall'occorrenza di pixel consecutivi di ugual colore.

HUFFMAN ENCODING

- Invece di rappresentare ogni carattere con un Byte, e' possibile associare ad ogni carattere un codice. Ai caratteri piu' frequenti viene associato un codice di pochi bit, mentre la lunghezza del codice cresce al diminuire della frequenza del carattere.
- Il codice da associare ad ogni carattere puo' essere determinato costruendo una struttura ad albero binario sulla base delle frequenze dei singoli caratteri.

- Si consideri il caso di un documento in cui compaiono 5 caratteri A, B, C, D, E. La frequenza dei singoli caratteri e': f(A)=0.45, f(B)=0.25, f(C)=0.15, f(D)=0,05, f(E)=0.10.
  Il corrispondente albero binario e'**:**
- Pertanto i 5 caratteri vengono cosi' codificati:
  - A: 0
  - B: 10
  - C: 110
  - D: 1110
  - E: 1111



ARITHMETIC ENCODING

- E' una variante della codifica di Huffman in grado di consentire maggiori rapporti di compressione a fronte della perdita di poter accedere non sequenzialmente al file compresso.
- A questa categoria appartiene un particolare algoritmo di compressione chiamato LZW (Lempel-Zif-Welsh) ed impiegato nel formato GIF di compressione delle immagini.

- **SOFTWARE DI COMPRESSIONE**
- **UNIX:**
  - **compress / uncompress (filename.Z)**
  - **gzip / gunzip (filename.gz)**
  - **zip / unzip (filename.zip)**
- **MSwindows:**
  - **pkzip / pkunzip (filename.zip)**
  - **zip / unzip (filename.zip)**
  - **rar (filename.rar)**
  - **arj (filename.arj)**
  - **zoo (filename.zoo)**
  - **lharc (filename.lha)**
- **Macintosh:**
  - **stuffit (filename.sit)**
  - **sea (filename.sea)**
  - **zip / unzip (filename.zip)**

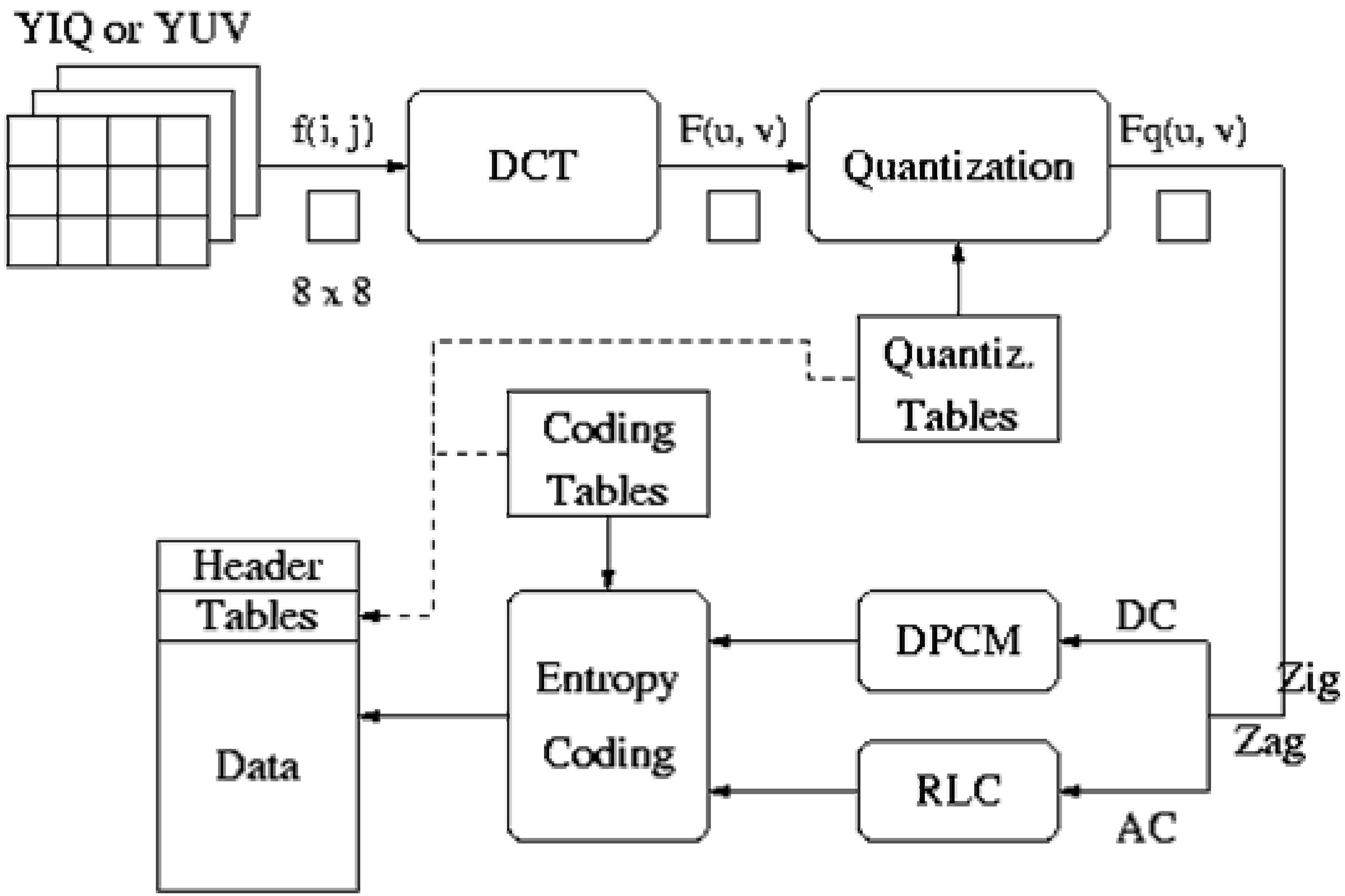
Lossy compression

SECT A JPEG ENCODING

- 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.

- **Major Steps**
  - DCT (Discrete Cosine Transformation)
  - Quantization
  - Zigzag Scan
  - Entropy coding
    - Coefficient encoding
      - DPCM on DC component
      - RLE on AC Components
    - Huffman Coding

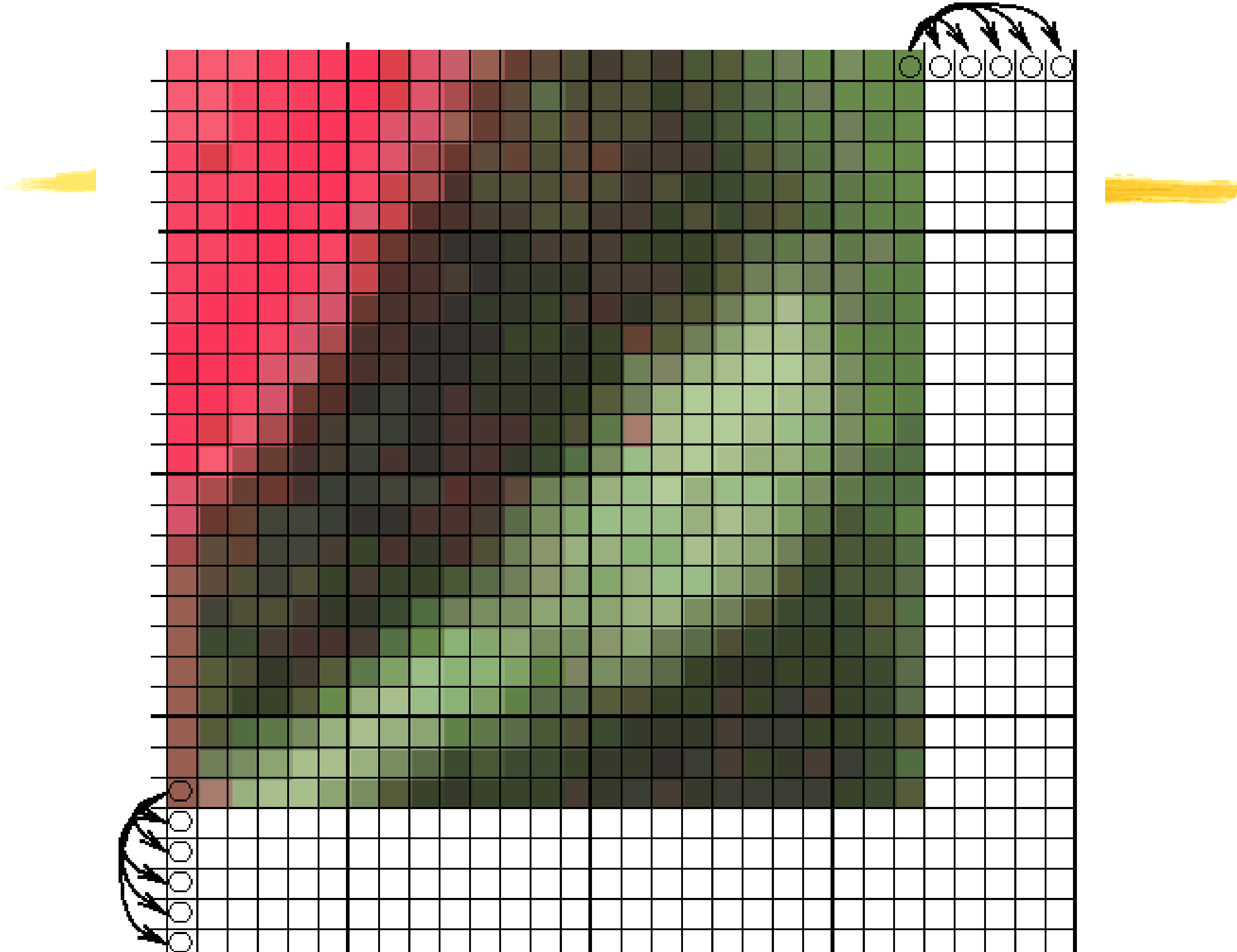
JPEG chain



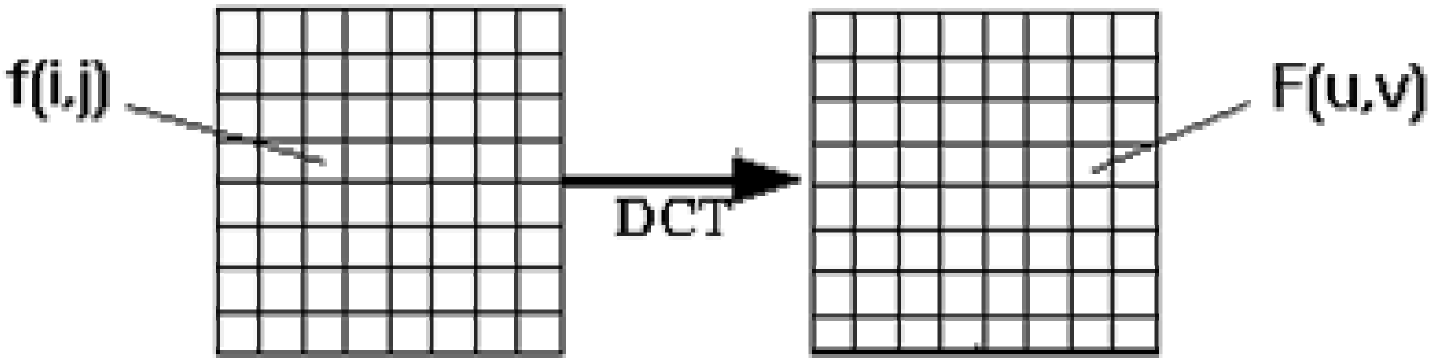
YIQ or YUV
f(i, j)
8 x 8
DCT
F(u, v)
Quantization
Fq(u, v)
Quantiz.
Tables
Coding
Tables
Header
Tables
Data
Entropy
Coding
DPCM
RLC
DC
AC
Zig
Zag

DCT (Discrete Cosine Transformation)

- Apply DCT to 8x8 image blocks
- If the image size is not a multiple of 8, then add copies of the last row or column until a multiple of 8 is reached. This makes both tone and luminance of the 8x8 block not change too much after DCT, as it would be if these elements were set to 0.



DCT allows to shift from spatial domain to frequency domain:



F(u,v) is the DCT coefficient of the 8x8 block in the (u,v) position of the 8x8 matrix that encodes the transformed coefficients.
f(i,j) is the value that is present in the (i,j) position of the 8x8 block

Discrete Cosine Transform (DCT):

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

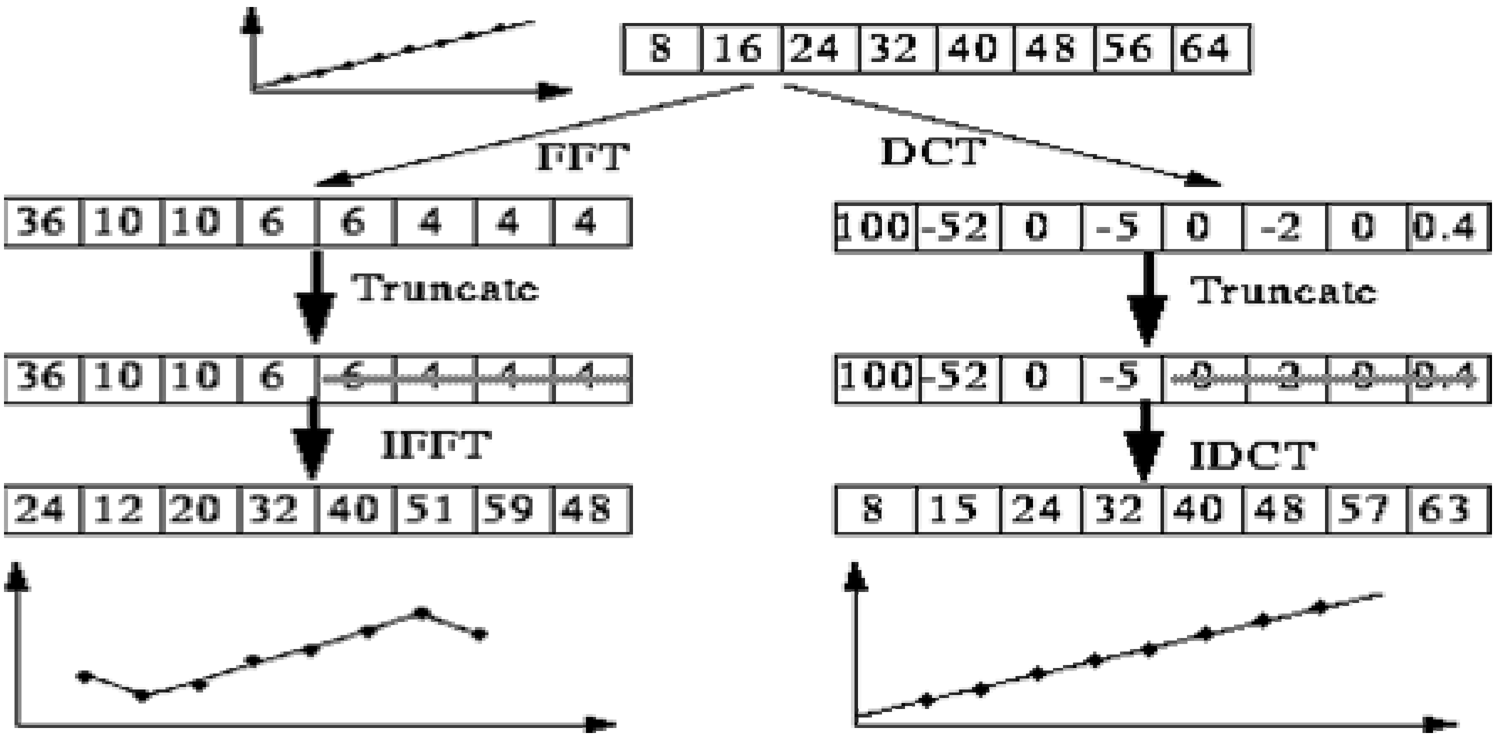
$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & for\ \xi = 0 \\ 1 & otherwise \end{cases}$$

Inverse Discrete Cosine Transform (IDCT)

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

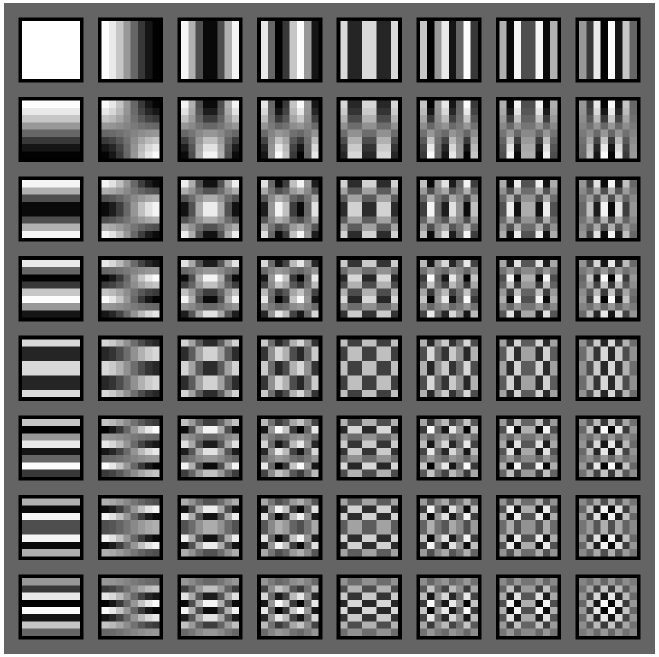
$$\Lambda(\xi) = \begin{cases} \frac{1}{\sqrt{2}} & for\ \xi = 0 \\ 1 & otherwise \end{cases}$$

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



The 64 (8 x 8) DCT basis functions

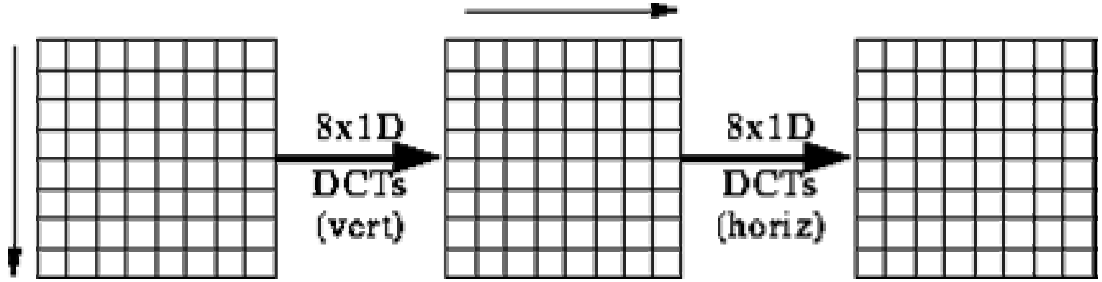
F[0,0]



- To compute DCT, factoring reduces the problem to a series of 1D DCTs:

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

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



Quantization

- To reduce number of bits per sample quantization is used:
  F'[u, v] = round (F[u, v] / q[u, v])
  Where q(u,v) is the quantization matrix and F(u,v) is the DCT coefficient matrix. Different quantization matrices are used for different color codes.

- 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** 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)

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

The Chrominance Quantization Table q(u, v)

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

139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	161	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	161	163	158	158	158

(a) source image samples

235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3
-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2
-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1
-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3
-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3
1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0
-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8
-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4

(b) forward DCT coefficients

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

(c) quantization table

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

(c) quantization table

235.6	−1.0	−12.1	−5.2	2.1	−1.7	−2.7	1.3
−22.6	−17.5	−6.2	−3.2	−2.9	−0.1	0.4	−1.2
−10.9	−9.3	−1.6	1.5	0.2	−0.9	−0.6	−0.1
−7.1	−1.9	0.2	1.5	0.9	−0.1	0.0	0.3
−0.6	−0.8	1.5	1.6	−0.1	−0.7	0.6	1.3
1.8	−0.2	1.6	−0.3	−0.8	1.5	1.0	−1.0
−1.3	−0.4	−0.3	−1.5	−0.5	1.7	1.1	−0.8
−2.6	1.6	−3.8	−1.8	1.9	1.2	−0.6	−0.4

Coefficienti DCT

$F(u,v)$

Quantizzazione (fase di codifica)

15	0	−1	0	0	0	0	0
−2	−1	0	0	0	0	0	0
−1	−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	0	0	0	0	0	0	0

Coefficienti quantizzati

$$F^Q(u,v) = \text{round int}\left(\frac{F(u,v)}{Q(u,v)}\right)$$

Dequantizzazione (fase di decodifica)

240	0	−10	0	0	0	0	0
−24	−12	0	0	0	0	0	0
−14	−13	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	0	0	0	0	0	0	0

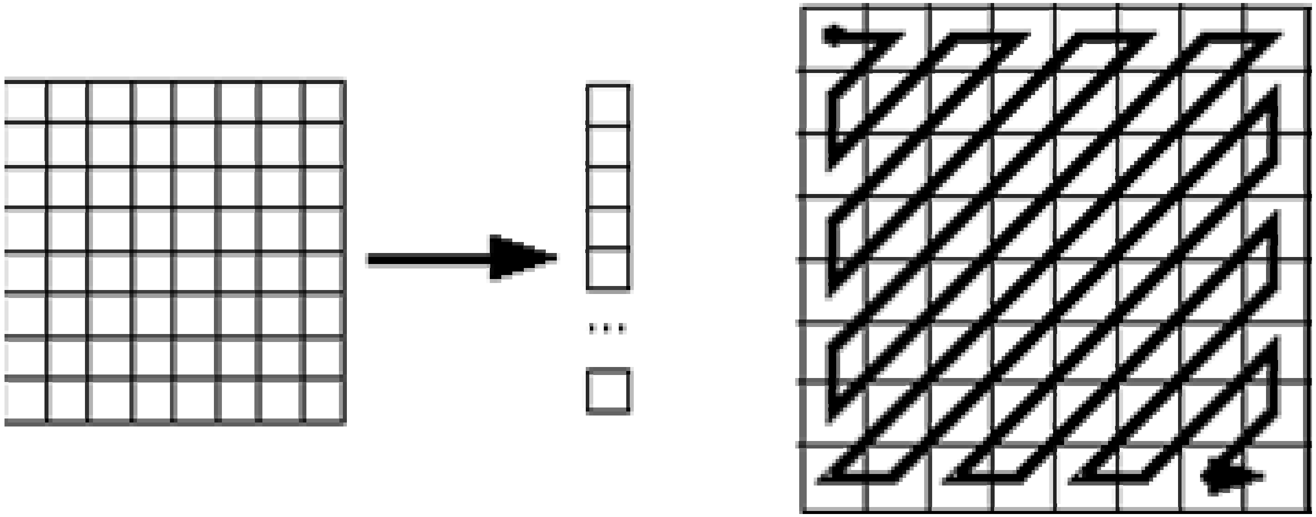
Coefficienti dequantizzati

$$F^{Q'}(u,v) = F^Q(u,v) \cdot Q(u,v)$$

- As a result of quantization we have a 8x8 matrix with many elements equal to 0. non null coefficients are all in the upper left corner.
- This suggests to transform the 8x8 matrix into a 64 element vector using a zig-zag order.

Zig-zag scan

Zig-Zag scan is used to group low frequency coefficients in top of the vector: Maps 8x8 to 1x64 vector



Coefficient encoding

- Differential Pulse Code Modulation (DPCM) on the DC component
  - DC component is large and varied, but often close to the value of the DC component of the previous block.
  - According to this we encode the difference between the previous and the current 8 x 8 block (DC diff)

- Run Length Encoding (RLE) on AC components
  - Many of the AC coefficients are equal to 0.
  - According to this they are encoded using RLE, which counts the number of consecutive 0s. A minimum of 0 to a maximum of 16 consecutive 0s is allowed. In the latter case the special symbol (15,0) is used. The end of block is encoded with (0,0).

- For the DC component (DC diff) we build the pair: (SIZE, AMPLITUDE)
- SIZE is the number of bits needed to represent the DC diff coefficient; AMPLITUDE is the DC diff.

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.

- For the AC components we use the following representation: (RUNLENGHT, SIZE) (AMPLITUDE)
- RUNLENGHT is the number of consecutive 0 (from 0 to 15), SIZE has the same meaning as for the DC coefficient, AMPLITUDE is the actual value for nonzero AC coefficients-

1 Coefficiente DC

63 Coefficienti AC

15	0	−1	0	0	0	0	0
−2	−1	0	0	0	0	0	0
−1	−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	0	0	0	0	0	0	0

Coefficienti della DCT quantizzati

DC del blocco precedente = 12,   15 − 12 = +3

3, (1 zero), −2, −1, −1, −1, (2 zeri), −1, (55 zeri)

	simbolo-1	simbolo-2
DC Coeff.	(size)	(ampiezza)
AC Coeff.	(runlength,size)	(ampiezza)

Simboli speciali	(simbolo-1)
(15,0)	(15,0) – runlength=16
(0,0)	(0,0) – EOB (end of block)

Size	Ampiezza	
1	−1	1
2	−3,−2	2,3
3	−7...−4	4..7
4	−15..−8	8..15
5	−31..−16	16..31
6	−63..−32	32..63
7	−127..−64	64..127
8	−255..−128	128..255
9	−511..−256	256..511
10	−1023..−512	512..1023

Baseline Entropy Coding

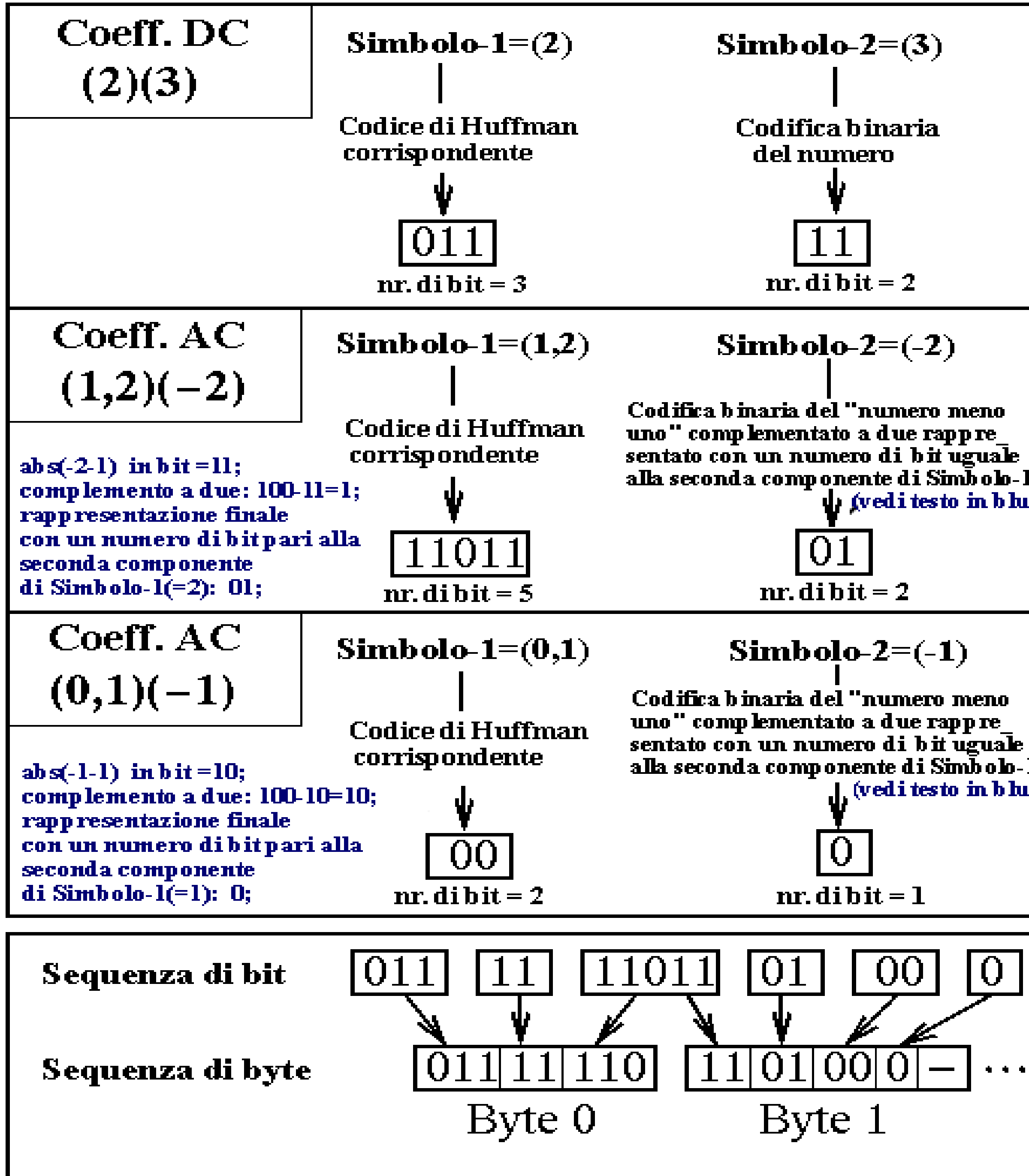
(2)(3), (1,2)(−2), (0,1)(−1), (0,1)(−1), (0,1)(−1), (2,1)(−1), (0,0)

Sequenza di simboli codificati

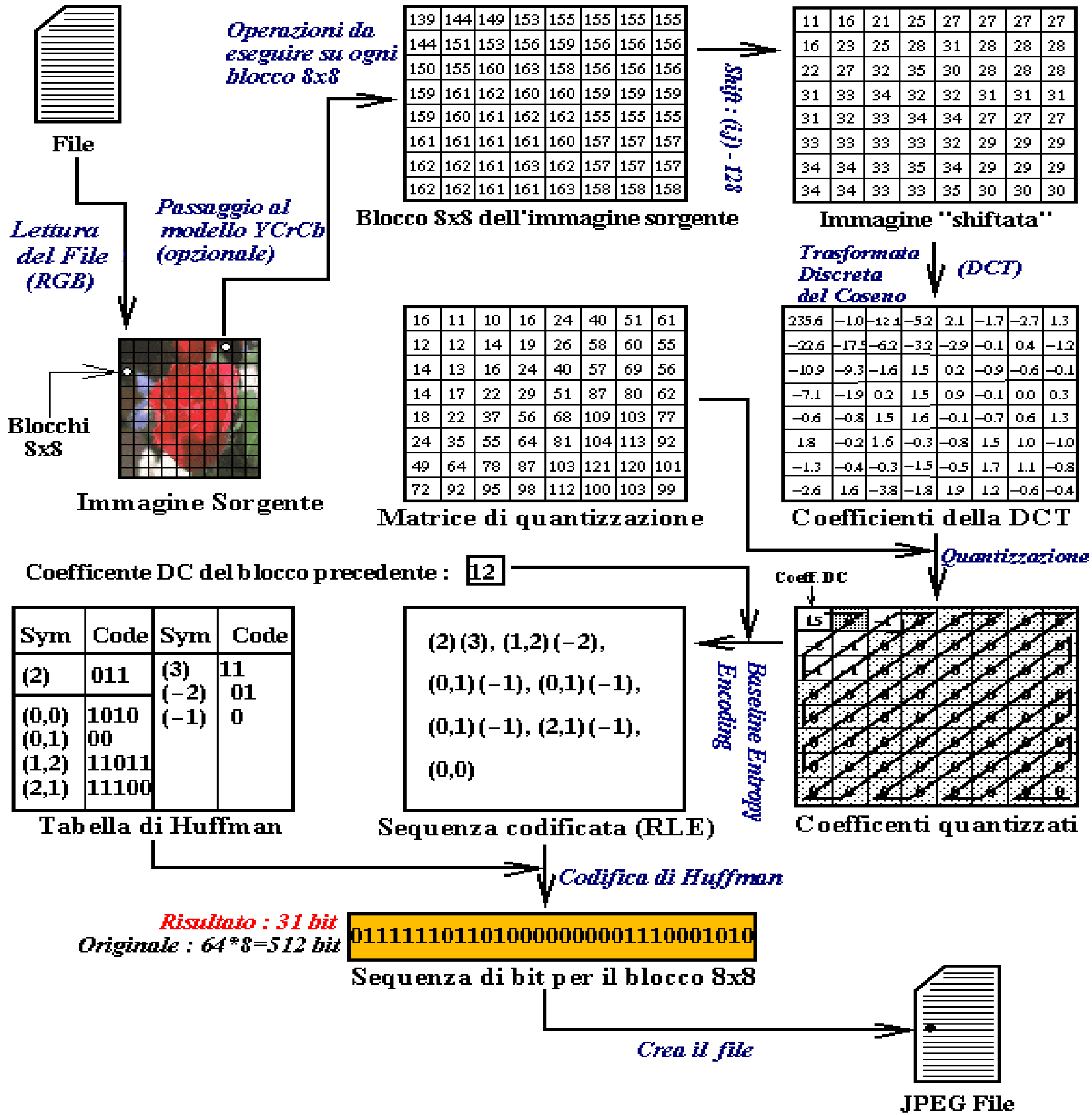
Huffman encoding

- After we have encoded every block, we have a sequence of symbols:
  - Symbol 1: (SIZE) or (RLE, SIZE)
  - Symbol 2: (AMPLITUDE)
- These symbols are further encoded using the Huffman encoding to reduce the number of data.
- Most frequent symbols are encode with shorter codes. Less frequent with longer ones. Huffman tables provide codes for every symbol of the sequence. Huffman tables are different for DC and AC symbol 1.
- Huffman Tables can be custom (sent in header) or default.

Symbol1 and Symbol2 encoding



Byte stuffing


File
Lettura del File (RGB)
Passaggio al modello YCrCb (opzionale)
Operazioni da eseguire su ogni blocco 8x8
139 144 149 153 155 155 155 155
144 151 153 156 159 156 156 156
150 155 160 163 158 156 156 156
159 161 162 160 160 159 159 159
159 160 161 162 162 155 155 155
161 161 161 161 160 157 157 157
162 162 161 163 162 157 157 157
162 162 161 161 163 158 158 158
Blocco 8x8 dell'immagine sorgente
Shift : (i,j) - 128
11 16 21 25 27 27 27 27
16 23 25 28 31 28 28 28
22 27 32 35 30 28 28 28
31 33 34 32 32 31 31 31
31 32 33 34 34 27 27 27
33 33 33 33 32 29 29 29
34 34 33 35 34 29 29 29
34 34 33 33 35 30 30 30
Immagine "shiftata"
Trasformata Discreta del Coseno
(DCT)
Blocchi 8x8
Immagine Sorgente
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
Matrice di quantizzazione
235.6 −1.0 −12.1 −5.2 2.1 −1.7 −2.7 1.3
−22.6 −17.5 −6.2 −3.2 −2.9 −0.1 0.4 −1.2
−10.9 −9.3 −1.6 1.5 0.2 −0.9 −0.6 −0.1
−7.1 −1.9 0.2 1.5 0.9 −0.1 0.0 0.3
−0.6 −0.8 1.5 1.6 −0.1 −0.7 0.6 1.3
1.8 −0.2 1.6 −0.3 −0.8 1.5 1.0 −1.0
−1.3 −0.4 −0.3 −1.5 −0.5 1.7 1.1 −0.8
−2.6 1.6 −3.8 −1.8 1.9 1.2 −0.6 −0.4
Coefficienti della DCT
Quantizzazione
Coefficente DC del blocco precedente : 12
Coeff. DC
Sym Code Sym Code
(2) 011 (3) 11
(−2) 01
(0,0) 1010 (−1) 0
(0,1) 00
(1,2) 11011
(2,1) 11100
Tabella di Huffman
(2) (3), (1,2) (−2),
(0,1) (−1), (0,1) (−1),
(0,1) (−1), (2,1) (−1),
(0,0)
Sequenza codificata (RLE)
Baseline Entropy Encoding
Coefficenti quantizzati
Codifica di Huffman
Risultato : 31 bit
Originale : 64*8=512 bit
0111111011010000000001110001010
Sequenza di bit per il blocco 8x8
Crea il file
JPEG File

Progressive JPEG

- The DCT progressive mode of operation consists of the same FDCT and Quantization steps that are used by DCT sequential mode. The key difference is that each image component is encoded in multiple scans rather than in a single scan.

- After each block of DCT coefficients is quantized, it is stored in a coefficient buffer memory. The buffered coefficients are then partially encoded in each of multiple scans.
- The first scan(s) encode a rough but recognizable version of the image which can be transmitted quickly in comparison to the total transmission time, and are refined by succeeding scans until reaching a level of picture quality that was established by the quantization tables.

- There are two complementary methods by which a block of quantized DCT coefficients may be partially encoded.
  - First, only a specified band of coefficients from the zig-zag sequence need be encoded within a given scan. This procedure is called .spectral selection, because each band typically contains coefficients which occupy a lower or higher part of the spatial-frequency spectrum.
  - Secondly, the coefficients within the band need not be encoded to their full quantized accuracy in a given scan (first the N most significant bits and the less significant in successive scans)

- The quantized DCT coefficient information can be viewed as a rectangle for which the axes are the DCT coefficients (in zig-zag order) and their amplitudes.

- Spectral selection slices the information in one dimension and successive approximation in the other.

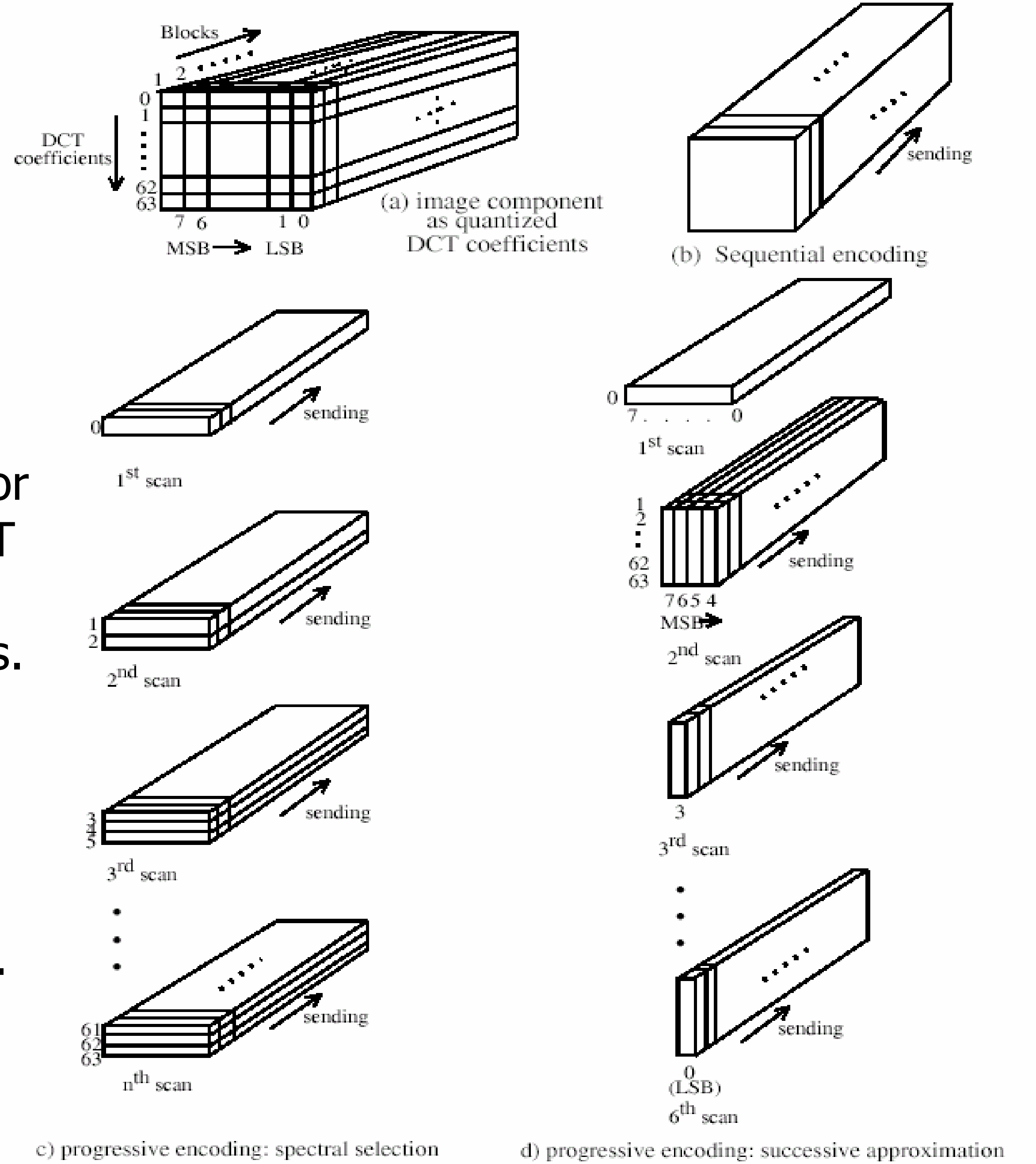


Figure 11. Spectral Selection and Successive Approximation Methods of Progressive Encoding

Hierarchical JPEG

- The hierarchical mode provides a pyramidal. encoding of an image at multiple resolutions, each differing in resolution from its adjacent encoding by a factor of two in either the horizontal or vertical dimension or both.

The encoding procedure can be summarized as follows:

- 1) Filter and down-sample the original image by the desired number of multiples of 2 in each dimension.
- 2) Encode this reduced-size image using one of the sequential DCT, progressive DCT, or lossless encoders described previously.
- 3) Decode this reduced-size image and then interpolate and up-sample it by 2 horizontally and/or vertically, using the identical interpolation filter which the receiver must use.
- 4) Use this up-sampled image as a prediction of the original at this resolution, and encode the difference image using one of the sequential DCT, progressive DCT, or lossless encoders described previously.
- 5) Repeat steps 3) and 4) until the full resolution of the image has been encoded.

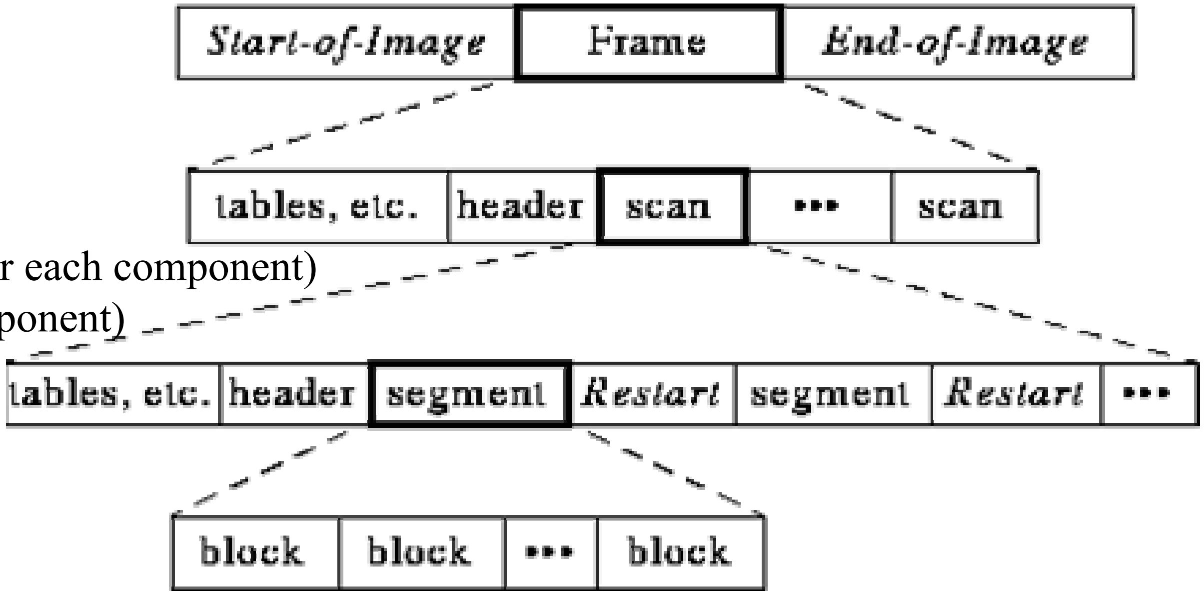
- Hierarchical encoding is useful in applications in which a very high resolution image must be accessed by a lower-resolution display.
- An example is an image scanned and compressed at high resolution for a very high-quality printer, where the image must also be displayed on a low-resolution PC video screen.

Lossless JPEG

- Lossless JPEG doesnot use DCT.
- It encodes the difference between the actual value of each pixel and its predicted value. The predicted value is provided by an appropriate function (e.g. the average) based on the modified values of the pixels above and left.
- The sequence is encoded with the Huffman code

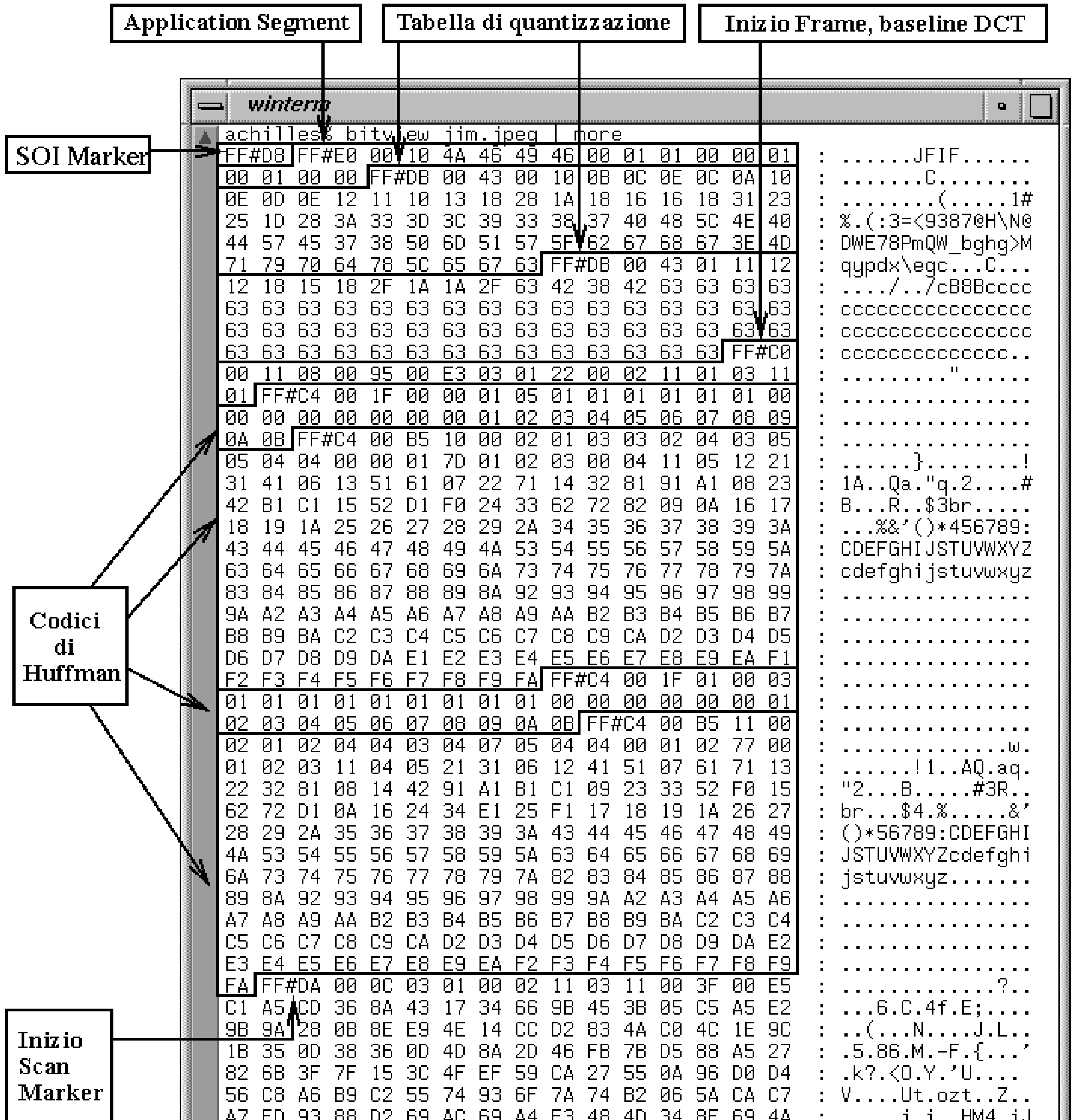
The JPEG Bitstream

Frame header:
sample precision
(width, height) of image
number of components
unique ID (for each component)
horizontal/vertical sampling factors (for each component)
quantization table to use (for each component)



Scan header
Number of components in scan
component ID (for each component)
Huffman table for each component (for each component)

Misc. (can occur between headers)
Quantization tables
Huffman Tables
Arithmetic Coding Tables
Comments
Application Data


Application Segment
Tabella di quantizzazione
Inizio Frame, baseline DCT
SOI Marker
Codici di Huffman
Inizio Scan Marker
winterm
achilles% bitview jim.jpeg | more
FF#D8 FF#E0 00 10 4A 46 49 46 00 01 01 00 00 01 : ......JFIF......
00 01 00 00 FF#DB 00 43 00 10 0B 0C 0E 0C 0A 10 : ....C........
0E 0D 0E 12 11 10 13 18 28 1A 18 16 16 18 31 23 : ....(.....1#
25 1D 28 3A 33 3D 3C 39 33 38 37 40 48 5C 4E 40 : %.(:3=<9387@H\N@
44 57 45 37 38 50 6D 51 57 5F 62 67 68 67 3E 4D : DWE78PmQW_bghg>M
71 79 70 64 78 5C 65 67 63 FF#DB 00 43 01 11 12 : qypdx\egc...C...
12 18 15 18 2F 1A 1A 2F 63 42 38 42 63 63 63 63 : ..../../cB8Bcccc
63 63 63 63 63 63 63 63 63 63 63 63 63 63 63 63 : cccccccccccccccc
63 63 63 63 63 63 63 63 63 63 63 63 63 63 63 63 : cccccccccccccccc
63 63 63 63 63 63 63 63 63 63 63 63 63 63 FF#C0 : cccccccccccccc..
00 11 08 00 95 00 E3 03 01 22 00 02 11 01 03 11 : ...."......
01 FF#C4 00 1F 00 00 01 05 01 01 01 01 01 01 00 : ....
00 00 00 00 00 00 00 01 02 03 04 05 06 07 08 09 : ....
0A 0B FF#C4 00 B5 10 00 02 01 03 03 02 04 03 05 : ....
05 04 04 00 00 01 7D 01 02 03 00 04 11 05 12 21 : ......}........!
31 41 06 13 51 61 07 22 71 14 32 81 91 A1 08 23 : 1A..Qa."q.2....#
42 B1 C1 15 52 D1 F0 24 33 62 72 82 09 0A 16 17 : B...R..$3br......
18 19 1A 25 26 27 28 29 2A 34 35 36 37 38 39 3A : ...%&'()*456789:
43 44 45 46 47 48 49 4A 53 54 55 56 57 58 59 5A : CDEFGHIJSTUVWXYZ
63 64 65 66 67 68 69 6A 73 74 75 76 77 78 79 7A : cdefghijstuvwxyz
83 84 85 86 87 88 89 8A 92 93 94 95 96 97 98 99 : ....
9A A2 A3 A4 A5 A6 A7 A8 A9 AA B2 B3 B4 B5 B6 B7 : ....
B8 B9 BA C2 C3 C4 C5 C6 C7 C8 C9 CA D2 D3 D4 D5 : ....
D6 D7 D8 D9 DA E1 E2 E3 E4 E5 E6 E7 E8 E9 EA F1 : ....
F2 F3 F4 F5 F6 F7 F8 F9 FA FF#C4 00 1F 01 00 03 : ....
01 01 01 01 01 01 01 01 01 00 00 00 00 00 00 01 : ....
02 03 04 05 06 07 08 09 0A 0B FF#C4 00 B5 11 00 : ....
02 01 02 04 04 03 04 07 05 04 04 00 01 02 77 00 : ....w.
01 02 03 11 04 05 21 31 06 12 41 51 07 61 71 13 : ......!1..AQ.aq.
22 32 81 08 14 42 91 A1 B1 C1 09 23 33 52 F0 15 : "2...B.....#3R..
62 72 D1 0A 16 24 34 E1 25 F1 17 18 19 1A 26 27 : br...$4.%.....&'
28 29 2A 35 36 37 38 39 3A 43 44 45 46 47 48 49 : ()*56789:CDEFGHI
4A 53 54 55 56 57 58 59 5A 63 64 65 66 67 68 69 : JSTUVWXYZcdefghi
6A 73 74 75 76 77 78 79 7A 82 83 84 85 86 87 88 : jstuvwxyz.......
89 8A 92 93 94 95 96 97 98 99 9A A2 A3 A4 A5 A6 : ....
A7 A8 A9 AA B2 B3 B4 B5 B6 B7 B8 B9 BA C2 C3 C4 : ....
C5 C6 C7 C8 C9 CA D2 D3 D4 D5 D6 D7 D8 D9 DA E2 : ....
E3 E4 E5 E6 E7 E8 E9 EA F2 F3 F4 F5 F6 F7 F8 F9 : ....
FA FF#DA 00 0C 03 01 00 02 11 03 11 00 3F 00 E5 : ....?..
C1 A5 CD 36 8A 43 17 34 66 9B 45 3B 05 C5 A5 E2 : ...6.C.4f.E;....
9B 9A 28 0B 8E E9 4E 14 CC D2 83 4A C0 4C 1E 9C : ..(...N....J.L..
1B 35 0D 38 36 0D 4D 8A 2D 46 FB 7B D5 88 A5 27 : .5.86.M.-F.{...'
82 6B 3F 7F 15 3C 4F EF 59 CA 27 55 0A 96 D0 D4 : .k?.<O.Y.'U....
56 C8 A6 B9 C2 55 74 93 6F 7A 74 B2 06 5A CA C7 : V....Ut.ozt..Z..