Multimedia Software Systems
CS4551

JPEG Image Compression Algorithm

Image Revisited

- We consider here *still images*, for example:
  - Photographs (color or grayscale)
  - Fax (bi-level and multilevel)
  - Documents (text, handwriting, graphics and photographs)
  - …
- Components of Images
  - Gray Images – single component
  - Color Images – r, g, b components
  - Sometimes images also have an alpha component.
  - Each component is represented as a number of bits per pixel (application dependent).

Why Compression?

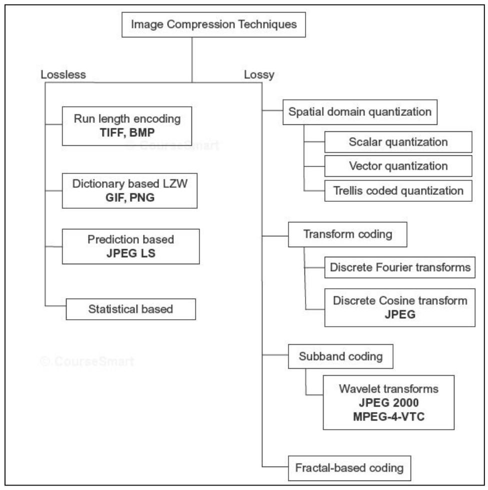
Multimedia image data	Grayscale image	Color image	HDTV video frame	Medical image	Super High Definition (SHD) image
Size/duration	512 × 512	512 × 512	1280 × 720	2048 × 1680	2048 × 2048
Bits/pixel or bits/sample	8 bpp	24 bpp	12 bpp	12 bpp	24 bpp
Uncompressed size (B for bytes)	262 KB	786 KB	1.3 MB	5.16 MB	12.58 MB
Transmission bandwidth (b for bits)	2.1 Mb/image	6.29 Mb/ image	8.85 Mb/ frame	41.3 Mb/ image	100 Mb/ image
Transmission time (56 K modem)	42 seconds	110 seconds	158 seconds	12 min.	29 min.
Transmission time (780 Kb DSL)	3 seconds	7.9 seconds	11.3 seconds	51.4 seconds	2 min.

Some Compression Formats

- There are many different uncompressed and compressed formats used in the industry! Some of commonly used compressed formats are
  - RIFF –Resource Interchange File Format
  - GIF – Graphics Interchange File Format
  - PNG –Portable Network Graphics
  - JPEG –Joint Photographic Expert Groups
  - JPEG 2000
  - MPEG4-VTC (Visual Texture Coding)
  - …

Types of Image Compression

- Makes use of Lossless or Lossy Schemes or a Hybrid
- Lossless Schemes
  - Already discussed in the previous lectures
- Lossy Schemes
  - Frequency Domain Based
    - Discrete Fourier Transforms
    - **Discrete Cosine Transform**
    - Wavelet Transforms



CSULA CS451 Multimedia Software Systems by Eun-Young Kang

JPEG Background – Requirements

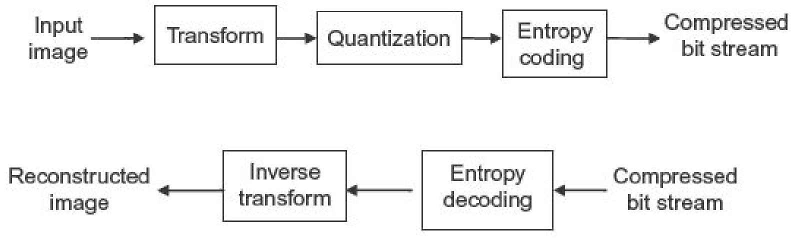
- Be at or near the start of art with regards to compression rate and accompanying fidelity
- Make no assumptions about the type of image –should be applicable to practically any kind of continuous-tone digital source image.
- Have a tractable computational complexity
- Support flexibility by allowing the following modes of operation
  - Sequential Encoding
  - Lossless Encoding
  - Progressive Encoding
  - Hierarchical Encoding

CSULA CS451 Multimedia Software Systems by Eun-Young Kang

JPEG Background –Selection Process and Proposed Standard

- JPEG is the standard algorithm for compression of *still* images
  - Started in June 1987 for first selection (12->3)
  - Second selection in 1988
  - Finalized and Accepted in 1991
- It is of reasonably low computational complexity, is capable of producing compressed images of high quality, and can provide both *lossless* and *lossy* compression of arbitrarily sized grayscale and color images

JPEG Coding Scheme



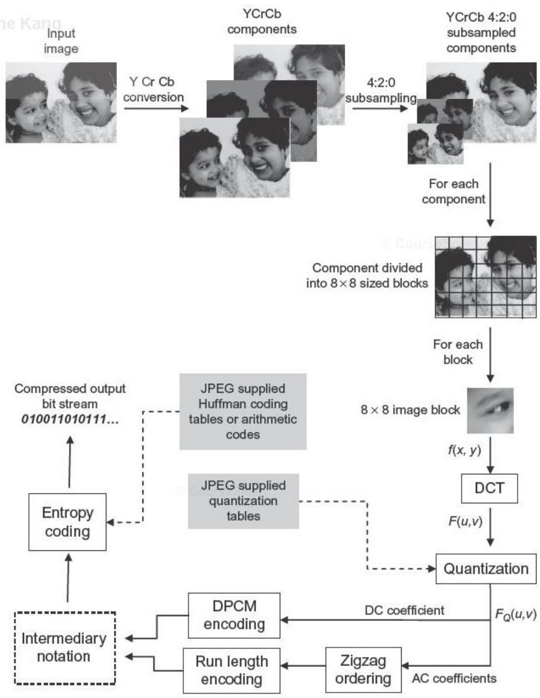
JPEG Compression Algorithm

1. The RGB color vectors of the input image are converted into YCrCb values and each channel is encoded independently
2. Each channel image is converted into a series of 8x8 pixel blocks, which are then processed in a raster scan sequence from left to right and from top to bottom
3. Each 8x8 block of pixels is spectrally analyzed using DCT (transform coding) and coefficients are quantized
4. After DCT coding and quantization, coefficients are entropy coded

JPEG Compression Algorithm

- Supplement diagram.

JPEG Encoding



Conversion b/w RGB and YCbCr

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} + \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Ranges:
R/G/B [0 ... 255]
Y/Cb/Cr [0 ... 255]

www.equasys.de

RGB to full-range YCbCr color conversion

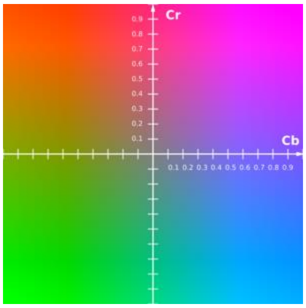
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.000 & 1.400 \\ 1.000 & -0.343 & -0.711 \\ 1.000 & 1.765 & 0.000 \end{bmatrix} \cdot \begin{bmatrix} Y \\ (Cb - 128) \\ (Cr - 128) \end{bmatrix}$$

Ranges:
Y/Cb/Cr [0 ... 255]
R/G/B [0 ... 255]

www.equasys.de

Full-range YCbCr to RGB color conversion

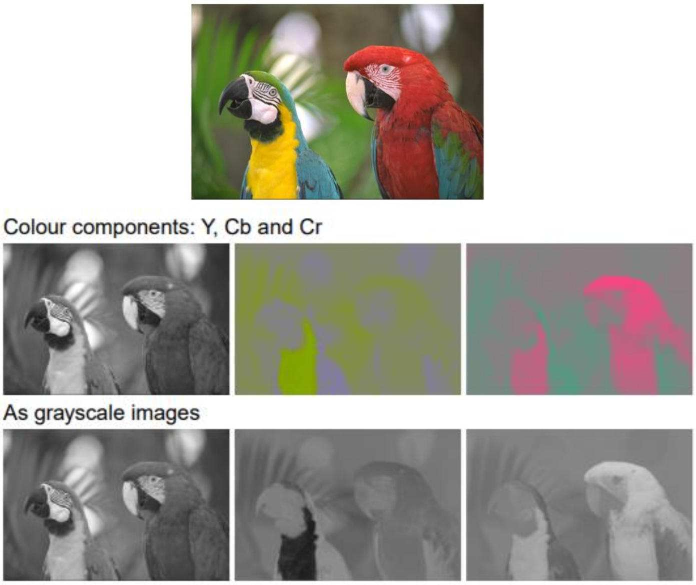
YCbCr



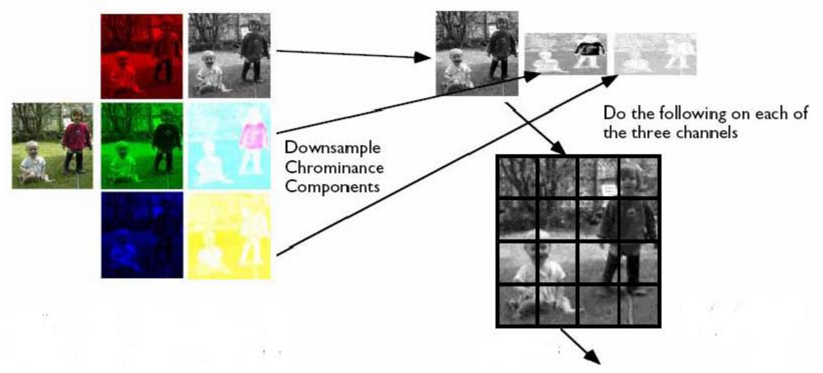
The CbCr plane at constant luma Y′=0.5

Colour components: R, G and B

As grayscale images



Color Sub-Sampling



Color Sub-Sampling (2)

- Each of Cb and Cr has been down sampled by 2. The reconstructed image (right) is not too different from the original (left).

image where Cb and Cr have been downsampled a factor 2 both horizontally and vertically, ie half of the image information has been removed.

2D DCT

Do the following to each block

Pixel values

49	61	69	61	78	89	100	112
68	60	51	42	62	69	80	89
90	81	58	49	69	72	68	69
100	91	79	72	69	68	59	58
111	100	101	91	82	71	59	49
131	119	120	102	90	90	81	59
148	140	129	99	92	78	59	39
151	140	142	119	98	90	72	39

Forward DCT

672	101	13	21	-13	0	6	9
-104	-144	35	-5	9	-17	-3	0
31	-21	-1	-7	-9	0	0	4
22	-21	-9	-13	-2	3	2	2
9	-2	-17	-11	0	5	1	-2
4	6	1	0	-2	-7	0	-1
21	-12	-3	2	-4	3	0	-1
-5	8	-3	2	1	-6	0	2

2D DCT Formula

- Discrete Cosine Transform (DCT) and Inverse DCT

** DCT*

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

** IDCT*

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

$$\text{where } i, j, u, v = 0, 1, \ldots 7$$

$$C(x) = \begin{cases} \frac{\sqrt{2}}{2} & \text{if } x = 0 \\ 1 & \text{otherwise} \end{cases}$$

Examples of 8x8 DCTs

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

Source image sample

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

DCT coefficients

DCT coefficient values are calculated for the shifted input image samples. (shifted by -128)

8x8 DCT Coefficients

- The DCT coefficients represent the *spatial frequency components* within a 8x8 image block
- The (0, 0) coefficient is called ***DC** coefficient.* **DC coefficient** is the **coefficient** with zero frequency in both dimensions.
- AC **coefficients** are remaining 63 **coefficients** with non-zero frequencies.
  - As we move to the right of the block, the coefficients represent the energy in higher *horizontal frequencies*
  - As we move down the block, the coefficients represent the energy in higher *vertical frequencies*

DCT Coefficient Quantization

- Each DCT coefficient is quantized independently using a uniform quantizer
- Quantization is defined as division of each DCT coefficient by its corresponding quantizer step size followed by rounding to the nearest integer.

$$F^Q(u,v) = \textit{Integer Round}\,\left(\frac{F(u,v)}{Q(u,v)}\right)$$

- The number of quantization intervals per each DCT channel is chosen independently for each coefficient.
- De-quantization $$F^{Q'}(u,v) = F^Q(u,v) * Q(u,v)$$

Rules for DCT Coefficient Quantization

- Low-frequency coefficients usually have more energy than high-frequency coefficients - therefore require more quantization intervals.
- The Human Visual System is more sensitive to low frequencies. Hence, low-frequency coefficients require more quantization intervals
- The Human Visual System is more sensitive to luminance channels than to chrominance channels.
- Hence, luminance channels require more quantization intervals than chrominance channels
- *“Note: the quantization table is not standardized!”*

Example

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

Source image sample

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

DCT coefficients

DCT coefficient values are calculated for the shifted input image samples. (shifted by -128)

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

Quantization table

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

Quantized coefficients

CSULA CS451 Multimedia Software Systems by Eun-Young Kang

Example – Quantization Tables

The Luminance Quantization Table (left) and the chrominance quantization table (right)

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

Example – DCT and Quantization

- Used the quantization tables in the previous slide.
- DCT coefficient values are calculated for the shifted input image samples. (shifted by -128)

An 8 × 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

$F(u, v)$

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

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

CSULA CS451 Multimedia Software Systems by Eun-Young Kang

Example – DCT and Quantization

- Used the quantization tables in the previous slide.
- DCT coefficient values are calculated for the shifted input image samples. (shifted by -128)

Another 8 × 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

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

178	187	183	175	178	177	150	183
191	174	171	182	176	171	170	188
199	153	128	177	171	167	173	183
195	178	158	167	167	165	166	177
190	186	158	155	159	164	158	178
194	184	137	148	157	158	150	173
200	194	148	151	161	155	148	167
200	195	172	159	159	152	156	154

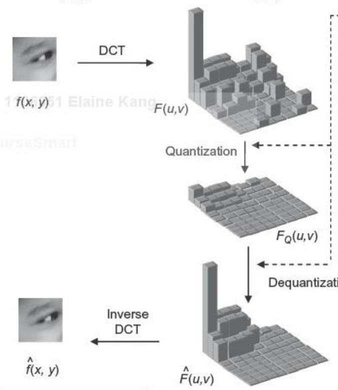
Pixel values $f(x, y)$

1359	46	61	26	38	−21	−5	−18
31	−35	−25	−11	13	10	12	−3
13	20	−17	−14	−11	−7	6	5
−5	5	2	−8	−11	−26	8	−4
10	15	−10	−16	−21	−7	8	7
−6	1	0	7	5	−7	−1	−3
−13	−8	1	10	8	4	−3	−4
−5	−5	−2	5	5	0	0	−3

DCT values $F(u,v)$

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

Quantization table



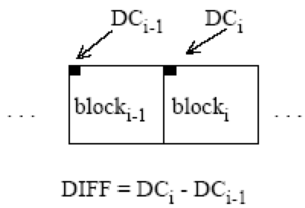
85	4	6	2	2	−1	0	0
3	−3	−2	−1	1	0	0	0
1	2	−1	−1	0	0	0	0
0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

192	185	178	152	193	162	155	190
181	172	162	154	187	164	159	192
183	168	150	158	181	167	162	190
198	177	150	155	177	169	161	182
202	180	148	148	171	167	159	175
193	176	145	141	162	162	156	170
195	184	155	145	159	156	150	164
209	200	170	154	160	153	144	156

1360	44	60	32	48	−40	0	0
36	−36	−28	−19	26	0	0	0
14	26	−16	−24	0	0	0	0
0	0	0	0	0	0	0	0
18	22	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

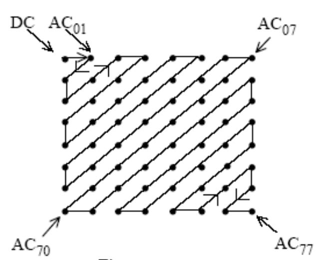
Preparation for Entropy Coding – Differential Encoding for DCs

- DC is a measure of the average value of 64 image samples
- There is usually strong correlation between the DC coefficients of adjacent blocks
- After quantization, the DC coefficient of a block is encoded as the difference from the DC term of the previous block in the processing order.
- Differential encoding
  - Diff(i) = DC(i) – DC(i-1)



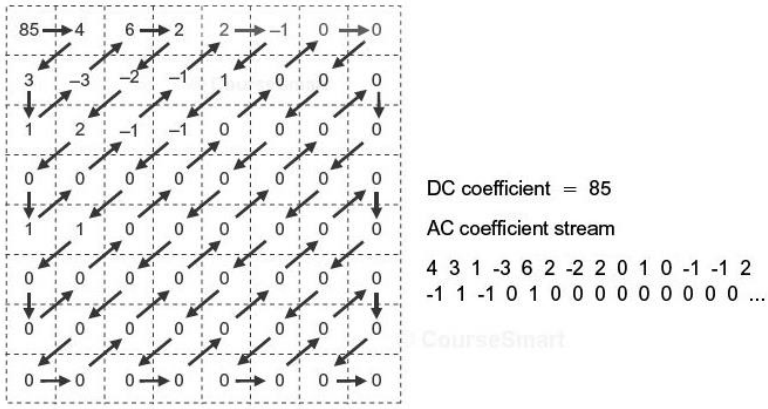
Preparation for Entropy Coding – Zigzag Order for ACs

- The quantized DCT coefficients are first processed in *zigzag order*



Zig–zag sequence

Preparation for Entropy Coding – Zigzag Order for ACs



Entropy Coding

- Zigzag ordering of coefficients places low frequency coefficients before high-frequency coefficients.
  - Low frequency coefficients are more likely to be non-zero
  - High frequency coefficients are more likely to be null after quantization
- In this way, the sequence of quantized coefficients is such that there often are long sequences of null values
- **Two steps of entropy coding**
  - Step1: Intermediate sequence of symbol
  - Step2: Assignment of variable length codes to the symbols

Intermediate Representation - AC coefficients (1)

- AC coefficients are represented in an *intermediate runlength representation, which* encodes the runs of zero preceding any non-null coefficient
- Each non-zero AC coefficient is represented in combination with the run-length of the null coefficients before it, using a pair of symbols:

Symbol-1	**Symbol-2**
(RUNLENGTH,SIZE)	*(AMPLITUDE)*

- **Symbol- 1** - *RUNLENGTH* is the number of consecutive null coefficients before the non-zero symbol; *SIZE* is the size of the symbol used to encode the non-null symbol
- **Symbol- 2** - *AMPLITUDE* is the actual (encoded) value

Intermediate Representation - AC coefficients (2)

- *RUNLENGTH* must be ≤15. To represent a run of >15 zeros, use (15,0) that is interpreted as the extension symbol with runlength=16. There can be up to 3 consecutive (15,0) symbol-1 extensions
- If the last run of zeros contains the last DCT coefficient, then the special EOB (End-Of-Block) symbol-1 value (0,0) terminates the representation.

Intermediate Representation - DC coefficient

- DC coefficients are represented by only one pair:

Symbol-1	Symbol-2
(SIZE)	*(AMPLITUDE)*

- **Symbol- 1** - *SIZE* is the size of the symbol used to encode the non-null symbol
- **Symbol- 2** - *AMPLITUDE* is the actual (encoded) value

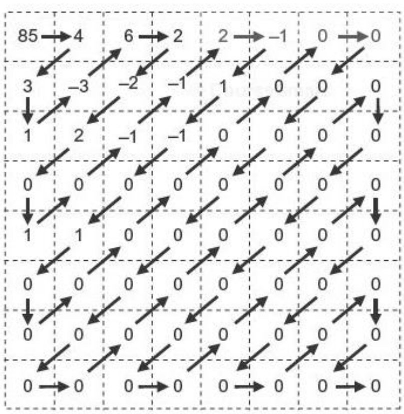
Intermediate Representation – Symbol 2

SIZE	AMPLITUDE
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

Table 3. Baseline Entropy Coding Symbol-2 Structure

Entropy Coding – Step 2

- For both DC and AC coefficients, each symbol-1 is Huffman encoded (variable length prefix code)
  - Huffman tables are not specified in the standard and must be input to the encoder
- Each symbol-2 is encoded with a Variable Length Integer (VLI) code
  - It is different from a Huffman code because the symbol length is known in advance
  - The coding table is hard-wired in the proposal



DC coefficient = 85

AC coefficient stream

4 3 1 -3 6 2 -2 2 0 1 0 -1 -1 2
-1 1 -1 0 1 0 0 0 0 0 0 0 0 0 ...

DC coefficient representation:

symbol -1 <SIZE>

symbol -2 <AMPLITUDE>

AC coefficient representation:

symbol -1 <RUNLENGTH, SIZE>

symbol -2 <AMPLITUDE>

Intermediary stream

<2><3> <0,3><4> <0,2><3> <0,1><1> <0,2><-3> <0,3><6>
<0,2><2> <0,2><-2> <0,2><2> <1,1><1> <1,1><-1> <0,1><-1>
<0,2><2> <0,1><-1> <0,1><1> <0,1><-1> <1,1><1> EOB

Intermediary symbol	Binary representation of first symbol (prefixed Huffman Codes)	Binary representation of second symbol (non-prefixed variable integer codes)
<2> <3>	011	11
<0,3> <4>	100	100
<0,2> <3>	01	11
<0,1> <1>	00	1
<0,2> <−3>	01	00
<0,3> <6>	100	110
<0,2> <2>	01	10
<0,2> <−2>	01	01
<0,2> <2>	01	10
<1,1> <1>	11	1
<1,1> <−1>	11	0
<0,1> <−1>	00	0
<0,2> <2>	01	10
<0,1> <−1>	00	0
<0,1> <1>	00	1
<0,1> <−1>	00	0
<1,1> <1>	11	1
EOB	1010	

Binary Stream:

011111001000111001010010011001100101011011111000001100000010001111010

Example of Entropy Coding

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

Example of Entropy Coding

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

(d) normalized quantized coefficients

- **If the DC term of the previous block is 12, then the difference is +3. Thus the intermediate representation is (2)(3), size=2, amplitude=3.**
- **The intermediate sequence of symbols for this example 8x8 block is**

  (2)(3), (1,2)(-2), (0,1)(-1), (0,1)(-1), (0,1)(-1) (2,1)(-1), (0,0)
- **Differential DC VLC**

(2) 011

- **AC luminance VLCs (generated by Huffman coding) are**

(0,0) 1010
(0,1) 00
(1,2) 11011
(2,1) 11100

- **VLIs (related to two's compliment representation)**

(3) 11
(-2) 01
(-1) 0

- **Then the 64 coefficients becomes 0111111011010000000001110001010 (31 bits)**

JPEG Performances

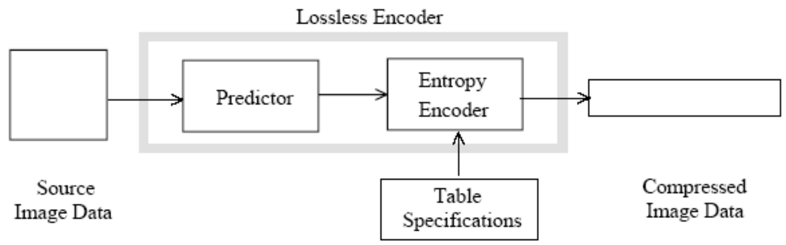
- Assume the pixels of an arbitrary color image are digitized to 8 bits for luminance and chrominance channels with 4:2:2 color sampling scheme.
- Then effectively the source image has 16 bits/pixel

Bits/Pixel	***Quality***	***Compression Ratio***
0.25	*Fair*	*64:1*
0.5	*Good*	*32:1*
0.75	*Very Good*	*21:1*
1.5	*Excellent*	*10:1*
≥2	*Indistinguishable*	*8:1*

Artifacts of JPEG – Original 380KB

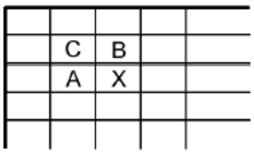
Artifacts of JPEG – Compressed 40KB

Lossless JPEG Mode



- It is wholly independent of DCT processing
- Each sample value is *predicted* from a combination of nearby sample values, and the prediction residual is entropy coded (Huffman)
- Prediction effectively reduces the entropy of the residual, so that small average symbol length can be achieved.

Lossless JPEG Mode

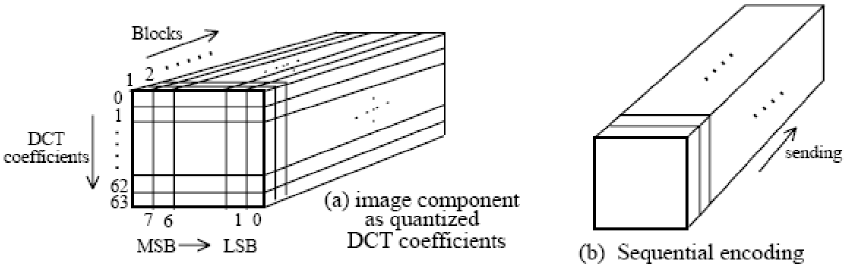


X can be predicted by a combination of A,B,C

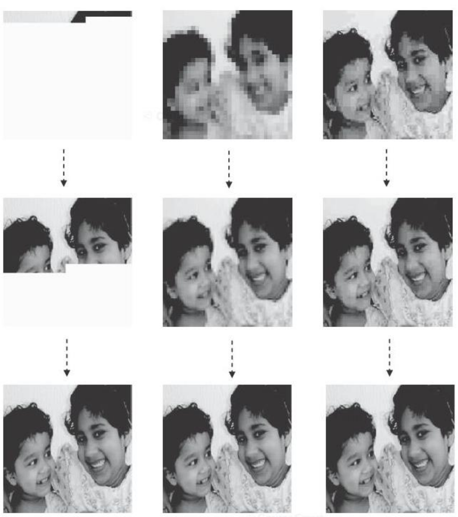
selection-value	prediction
0	no prediction
1	A
2	B
3	C
4	A+B-C
5	A+((B-C)/2)
6	B+((A-C)/2)
7	(A+B)/2

Table 1. Predictors for Lossless Coding

Sequential Encoding



Progressive Encoding



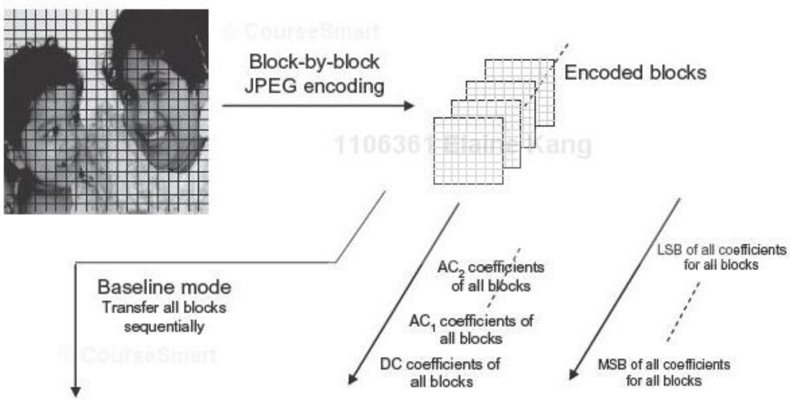
Progressive Encoding (1)

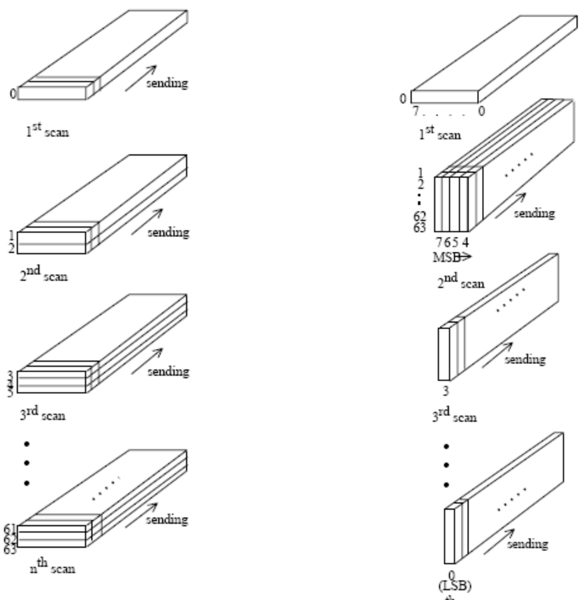
- Layering or progressive encoding: an image can be transmitted at a low rate (and a low quality) and then progressively improved by subsequent transmission
- Convenient for browsing applications, where a low-quality or low-resolution image is adequate browsing
- Three forms of JPEG progressive encoding:
  - Spectral selection
  - Successive approximation (SNR scalability)
  - Hierarchical (pyramidal) mode

Progressive Encoding (2)

- Spectral selection
  - Initial transmission sends low-frequency DCT coefficients, followed progressively by the higher frequency coefficients, according to the zig-zag scan
  - Simple to implement but reconstructed images from the early scans are blurred
- Successive approximation (SNR scalability)
  - Only the most significant bits of each coefficient are sent in the first scan, followed by the next most significant bits and so on until all bits have been transmitted

Progressive Encoding (3)





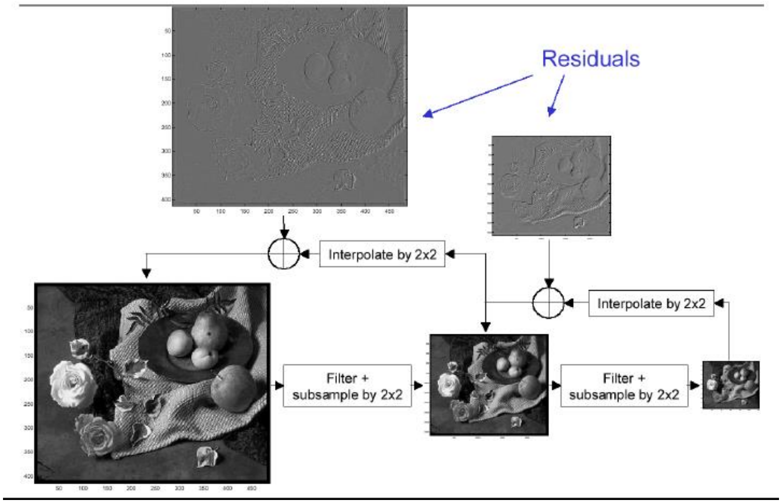
c) progressive encoding: spectral selection

d) progressive encoding: successive approximation

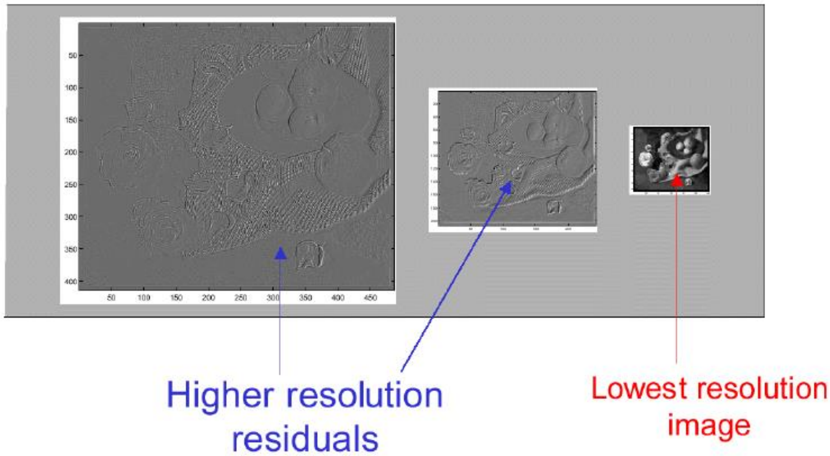
Progressive Encoding (4)

- Hierarchical (pyramidal) mode
  - Image can be sent in one of several *resolution modes* to accommodate different kinds of displays
  - Different resolution modes achieved by *filtering and down sampling* the image in multiples of two in each direction. The resulting image is interpolated and subtracted from the next level to create a *residual*
  - The different resolution modes are transmitted together with the residuals

Pyramid Decomposition (1)



Pyramid Decomposition (2)



CSULA CS451 Multimedia Software Systems by Eun-Young Kang

Hierarchical (pyramidal) mode – Encoding/Decoding

- Encoder for three level hierarchical mode (f -> f_2 -> f_4)
  - Reduction of image resolution : Reduce resolution of the input image by a factor of 2 in each dimension. Repeat this to obtain f_4
  - Compress lowest-resolution image f_4 using JPEG sequential mode ~> F4
  - Compress difference image d2 = f_2 - f'_4 where f'_4 is the interpolated (expanded) images of the decoded f_4 image ~> D2
  - Compress difference image d = f – f'_2 where f'_2 is the interpolated (expanded) images of the decoded f_4 image ~> D
- Decoder for three level hierarchical mode
  - Decompress F4
  - Restore image f'2 using F'4 + D2 ~> F2
  - Restore image f using F'2 + D ~> F

CSULA CS451 Multimedia Software Systems by Eun-Young Kang