JPEG

Joint Photographic Expert Group

JPEG

- Resolution independence (handles arbitrary resolutions, images are padded to make their dimensions multiples of 8)
- Precision (8 or 12 bit DCT)
- No absolute bitrate targets
- Luminance-chrominance separability
- Extensible (no bounds on progressive stages)

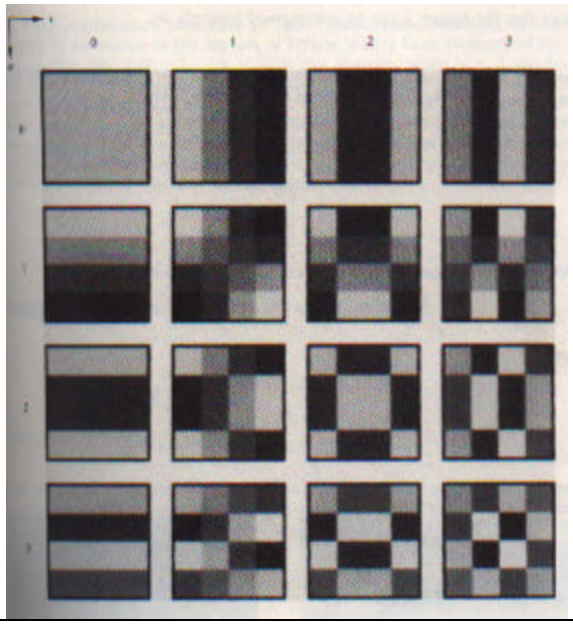
Discrete Cosine Transform

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos\left[\frac{(2x+1)u\pi}{2N}\right]\cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$a(u) = \begin{cases} \sqrt{\frac{1}{N}} & u = 0 \\ \sqrt{\frac{2}{N}} & u = 1,2,\ldots N-1 \end{cases}$$

DCT Bases Functions



Example

$$I_{i,j} = \begin{bmatrix} 139 & 144 & 149 & 153 & 155 & 155 & 155 & 155 \\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156 \\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156 \\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159 \\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 155 \\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157 \\ 162 & 162 & 161 & 161 & 162 & 157 & 157 & 157 \\ 162 & 162 & 161 & 161 & 163 & 158 & 158 & 158 \end{bmatrix}$$

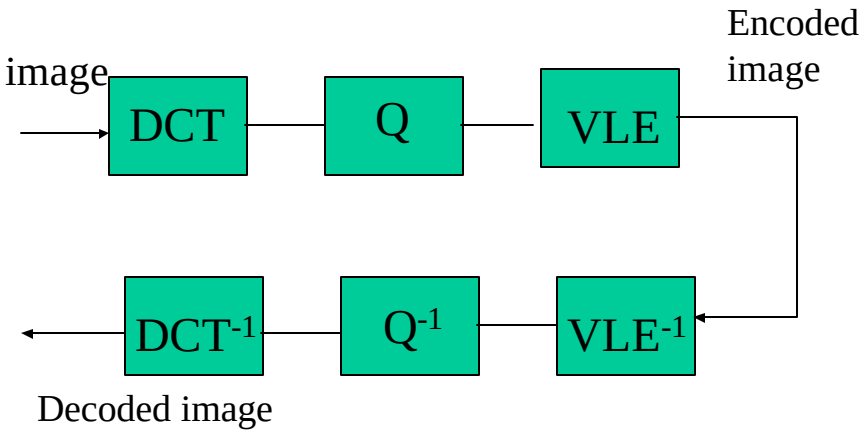
image

Example

$$F_{u,v} = \begin{bmatrix} 1260 & -1 & -12 & -5 & 2 & -2 & -3 & 1 \\ -23 & -17 & -6 & -3 & -3 & 0 & 0 & 1 \\ -11 & -9 & -2 & 2 & 0 & -1 & -1 & 0 \\ -7 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & -1 & 1 & 1 \\ 2 & 0 & 2 & 0 & -1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 2 & 1 & -1 \\ -3 & 2 & -4 & -2 & 2 & 1 & -1 & 0 \end{bmatrix}$$

DCT

JPEG BLOCK DIAGRAM



JPEG Modes

- Baseline
- Progressive
- Lossless
- Hierarchical

JPEG Baseline Coding

- Divide image into blocks of size 8X8.
- Level shift all 64 pixels values in each block by subtracting $2^{n-1,}$ (where 2^n is the maximum number of gray levels).
- Compute 2D DCT of a block.
- Quantize DCT coefficients using quantization table.

JPEG Baseline Coding

- Zig-zag scan the quantized DCT coefficients to form 1-D sequence.
- Code 1-D sequence (AC and DC) using JPEG Huffman variable length codes.

JPEG Baseline Decoding

- Decode 1-D sequence (AC and DC) using JPEG Huffman variable length codes.
- Convert 1-D sequence of coefficients into 2-D array using Zig-zag scan.
- Dequantize DCT coefficients using quantization table.
- Compute inverse DCT of a block.
- Level shift all 64 inverse transformed pixels values in each block by adding $2^{n-1,.}$

JPEG Quantization Tables

- The DCT basis functions were viewed with
  - Luma resolution of 720X576 and
  - chroma resolution of 360X288
  - Viewing distance of six times the screen width

JPEG Quantization Table (Luma)

$$Q_{u,v} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 51 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

JPEG Quantization Table (Chroma)

$$\begin{bmatrix} 17 & 18 & 24 & 47 & 99 & 99 & 99 & 99 \\ 18 & 21 & 26 & 66 & 99 & 99 & 99 & 99 \\ 24 & 26 & 56 & 99 & 99 & 99 & 99 & 99 \\ 47 & 66 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \\ 99 & 99 & 99 & 99 & 99 & 99 & 99 & 99 \end{bmatrix}$$

JPEG ZIG-ZAG SCAN

$$\begin{bmatrix} 1 & 2 & 6 & 7 & 15 & 16 & 28 & 29 \\ 3 & 5 & 8 & 14 & 17 & 27 & 30 & 43 \\ 4 & 9 & 13 & 18 & 26 & 31 & 42 & 44 \\ 10 & 12 & 19 & 25 & 32 & 41 & 45 & 54 \\ 11 & 20 & 24 & 33 & 40 & 46 & 53 & 55 \\ 21 & 23 & 34 & 39 & 47 & 52 & 56 & 61 \\ 22 & 35 & 38 & 48 & 51 & 57 & 60 & 62 \\ 36 & 37 & 49 & 50 & 58 & 59 & 63 & 64 \end{bmatrix}$$

JPEG Coefficient Coding Categories

Range	DC	AC
0	0	N/A
-1,1	1	1
-3,-2,2,3	2	2
-7,..,-4,4,..,7	3	3
-15,..,-8,8,..,15	4	4
….	..	..
-32767,..,32767	F	N/A

JPEG DC Code

Cat	Base Code	Length
0	010	3
1	011	4
2	100	5
3	00	5
4	101	7
5	110	8
6	1110	10
7	11110	12
8	111110	14
9	1111110	16
A	11111110	18
B	111111110	20

JPEG AC Code

Run/Cat	Base Code	Length
(0,0)	1010(EOB)	4
(0,1)	00	3
(0,2)	01	3
(0,3)	100	6
(0,4)	1011	8
(0,5)	11010	10
(0,6)	111000	12
(0,7)	1111000	14
(0,8)	1111110110	18
(0,9)	1111111110000010	25
…	…	…

Construction of JPEG Code

- Compute difference between the current DC coefficient and that of previously encoded block.
- Determine DC category of difference, and use the base code.
- Generate remain bits of code from the LSB (Least Significant Bits) of the difference.

Example (Encoding)

$$I = \begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 66 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix} \quad I' = \begin{bmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -62 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -65 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{bmatrix}$$

$$DCT = \begin{bmatrix} -415 & -29 & -62 & 25 & 55 & -20 & -1 & 3 \\ 7 & -21 & -62 & 9 & 11 & -7 & -6 & 6 \\ -46 & 8 & 77 & -25 & -30 & 10 & 7 & -5 \\ -50 & 13 & 35 & -15 & -9 & 6 & 0 & 3 \\ -11 & -8 & -13 & -2 & -1 & 1 & -4 & 1 \\ -10 & 1 & 3 & -3 & -1 & 0 & 2 & -1 \\ -4 & -1 & 2 & -1 & 2 & -3 & 1 & -2 \\ -1 & -1 & -1 & -2 & -1 & -1 & 0 & -1 \end{bmatrix} \quad Q' = \begin{bmatrix} -26 & -3 & -6 & 2 & 2 & 0 & 0 & 0 \\ 1 & -2 & -4 & 0 & 0 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Example (Encoding)

1-D coefficeints

[-26 -3 1 -3 -2 -6 2 -4 1 -4 1 1 5
0 -1 2 0 0 0 0 0 -1 -1 EOB]

Coded array

1010110 0100 001 0100 0101 100001
0110 100011 001 100011 001 001
100101 11100110 110110 0110 11110100
000 1010 92 bits, 512/92 =5.6:1

Determining Code (DC)

- The DC DCT is “-26”.
- The DC DCT of previous block was “-17”.
- The difference is: -26-(-17)=-9
- DC category for “-9” is “4”, with base code “101”, and code length “7 bits”.
- The difference (-9)=$(0110)_2$.
- The code for (-26) is 1010110.

Determining Code (AC)

- "-3" is AC category 2, preceded by "0" zeros
- Base code for 0/2 is "01", length is "4" bits
- Two LSB of (-3)=$(100)_2$ are "00"
- The code of "-3" is "0100"

Example (Decoding)

$$P=\begin{bmatrix}-26&-3&-6&2&2&0&0&0\\1&-2&-4&0&0&0&0&0\\-3&1&5&-1&-1&0&0&0\\-4&1&2&-1&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\end{bmatrix}$$

$$P'=\begin{bmatrix}-416&-33&-60&32&48&0&0&0\\12&-24&-56&0&0&0&0&0\\-42&13&80&-24&-40&0&0&0\\-56&17&44&-29&0&0&0&0\\18&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\\0&0&0&0&0&0&0&0\end{bmatrix}$$

$$P''=\begin{bmatrix}-70&-64&-61&-64&-69&-66&-58&-50\\-72&-73&-61&-39&-30&-40&-54&-59\\-68&-78&-58&-9&13&-12&-48&-64\\-59&-77&-57&0&22&-13&-51&-60\\-52&-71&-72&-54&-54&-71&-71&-54\\-42&-50&-70&-68&-67&-67&-61&-50\\-45&-59&-70&-68&-67&-67&-61&-50\\-35&-47&-61&-66&-60&-48&-44&-44\end{bmatrix}$$

$$P'''=\begin{bmatrix}58&64&67&64&59&62&70&78\\56&55&67&89&98&88&74&69\\60&50&70&119&141&116&80&64\\69&51&71&128&149&115&77&68\\74&53&64&105&115&84&65&72\\76&57&56&74&75&57&57&74\\83&69&59&60&61&61&67&78\\93&81&67&62&69&80&84&84\end{bmatrix}$$

Comparison

$$I = \begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 66 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

Original Image

$$P'' = \begin{bmatrix} 58 & 64 & 67 & 64 & 59 & 62 & 70 & 78 \\ 56 & 55 & 67 & 89 & 98 & 88 & 74 & 69 \\ 60 & 50 & 70 & 119 & 141 & 116 & 80 & 64 \\ 69 & 51 & 71 & 128 & 149 & 115 & 77 & 68 \\ 74 & 53 & 64 & 105 & 115 & 84 & 65 & 72 \\ 76 & 57 & 56 & 74 & 75 & 57 & 57 & 74 \\ 83 & 69 & 59 & 60 & 61 & 61 & 67 & 78 \\ 93 & 81 & 67 & 62 & 69 & 80 & 84 & 84 \end{bmatrix}$$

Decoded Image

Difference

$$Diff = \begin{bmatrix} -6 & -9 & -6 & 2 & 11 & -1 & -6 & -5 \\ 7 & 4 & -1 & 1 & 11 & -3 & -5 & 3 \\ 2 & 9 & -2 & -6 & -3 & -12 & -14 & 9 \\ -6 & 7 & 0 & -4 & -5 & -9 & -7 & 1 \\ -7 & 8 & 4 & -1 & 11 & 4 & 3 & 2 \\ 3 & 8 & 4 & -4 & 2 & 11 & 1 & 1 \\ 2 & 2 & 5 & -1 & -6 & 0 & -2 & 5 \\ -6 & -2 & 2 & 6 & -4 & -4 & -6 & 10 \end{bmatrix}$$

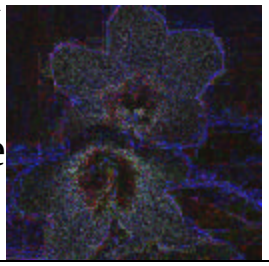
JPEG

Original 64K

13K

5K

Difference



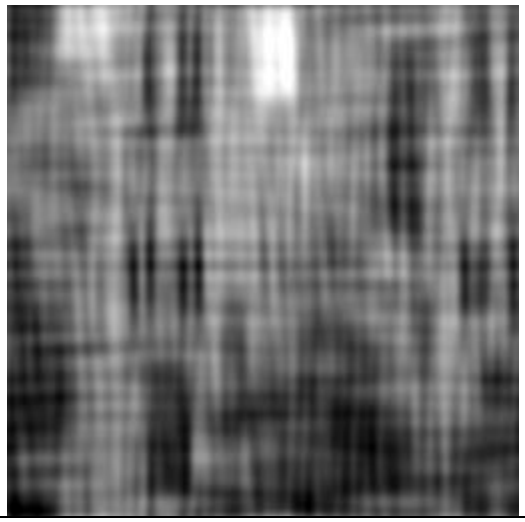
Progressive

- Coding of DCT coefficients in multiple passes.
  - Spectral selection
    - send lower frequency DCTs first, followed by higher frequency coefficients
    - E.G., First send DC, then first AC, Second AC,…
    - Reasonable quality after 5 AC

Progressive

– Successive approximation
  - send DC coefficient first
  - start with MSB of AC coefficients, followed by LSB
  - gives better quality images at low bitrate

First Update (.14%)



Third Update (4.12%)

Sixth Update (8.24%)

Lossless

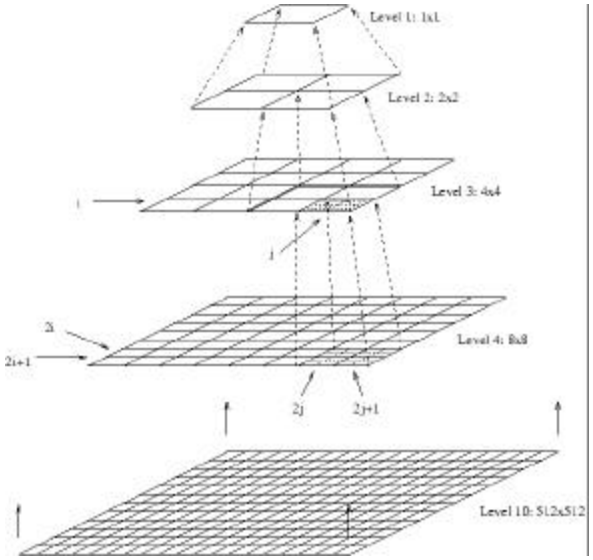
- Uses predictive coding
- Integer prediction errors are VLC coded
- 8 different predictions can be used

0	Differential
1	A
2	B
3	C
4	A+B+C
5	A-(B-C)/2
6	B-(A-C)/2
7	(A+B)/2

Hierarchical

- Increasing special resolutions between progressive stages.
  - First the lowest resolution of pyramid is coded
  - Output of each stage is upsampled and taken as a prediction for the next stage.
  - In the next stage the difference between the actual second level and the predicted second level is encoded and transmitted.
  - The procedure continues until highest resolution

Pyramid


Level 1: 1x1
Level 2: 2x2
Level 3: 4x4
i
j
2i
2i+1
Level 4: 8x8
2j
2j+1
Level 10: 512x512