Data Compression

Transform Coding and JPEG

中央大學資工系
蘇柏齊

Transform

- Transform is a fixed procedure that changes one chunk of data into another chunk of data.
- Types of transforms commonly used
  - Fourier transform
  - Laplace transform
  - Z transform
  - Hadamard transform
  - Haar transform
  - Discrete Cosine transform
  - Wavelet transform
  - …

Usage of Transforms

- Extract features from signals
  - The average value or dc term is proportional to the average image amplitude. The high-frequency terms (ac term) give an indication of the amplitude and orientation of edges within an image.
- Dimensionality reduction in computation
  - The transform coefficients that are small may be excluded from processing operations, such as filtering, without much loss in processing accuracy.
- Transform image coding
  - Achieve bandwidth reduction by discarding or grossly quantizing low-magnitude transform coefficients.

Transform

- Linear Transformation:
  - Forward transform **c=Ax**
  - Inverse transform $\mathbf{x}=\mathbf{A}^{-1}\mathbf{c}$
    - **x** is represented as linear combination of "basis function"
- Orthonormality: $A^{-1}=A^T$

$$\begin{bmatrix} a_{00} & \cdots & a_{0,N-1} \\ \vdots & & \vdots \\ a_{N-1,0} & \cdots & a_{N-1,N-1} \end{bmatrix} \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} c_0 \\ \vdots \\ c_{N-1} \end{bmatrix}$$

$$A^T c = x$$

$$\begin{bmatrix} a_{00} & \cdots & a_{N-1,0} \\ \vdots & & \vdots \\ a_{0,N-1} & \cdots & a_{N-1,N-1} \end{bmatrix} \begin{bmatrix} c_0 \\ \vdots \\ c_{N-1} \end{bmatrix} = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

$$\begin{bmatrix} a_{00} \\ \vdots \\ a_{0,N-1} \end{bmatrix} c_0 + \cdots + \begin{bmatrix} a_{N-1,0} \\ \vdots \\ a_{N-1,N-1} \end{bmatrix} c_{N-1} = \begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix}$$

basis vectors

coefficients

Properties of Transform

- Orthonomality
  - The energy of data is equal to the energy of coefficients

$$\sum_{i=0}^{N-1} c_i^2 = c^T c = (Ax)^T (Ax)$$

$$= (x^T A^T)(Ax) = x^T (A^T A) x = x^T x = \sum_{i=0}^{N-1} x_i^2$$

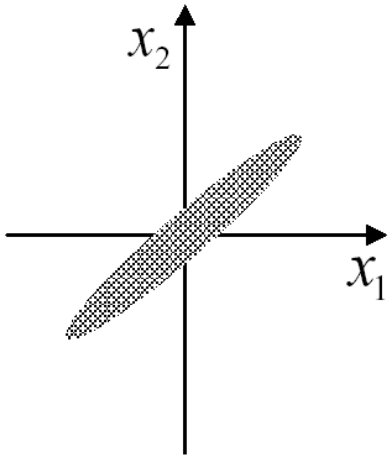
- Interpretation: every orthonormal transform is simply a rotation of the coordinate system

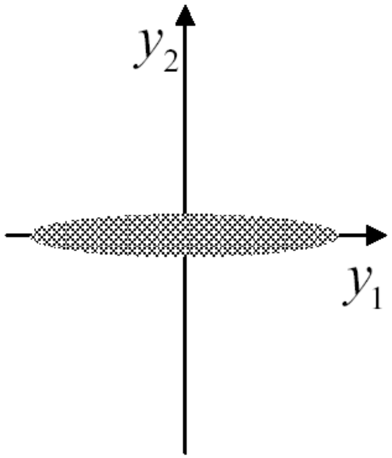
Properties of Transform

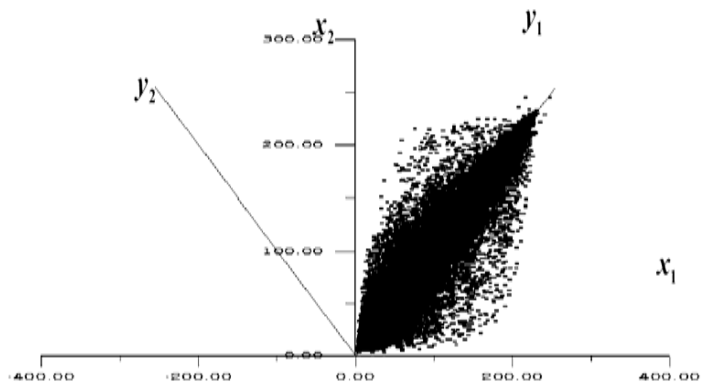
- 2-D Orthonormal Transform

$$\mathbf{A} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}$$

Example: image pixel pairs







Strongly correlated samples,
equal energies

<u>After transform:</u>
uncorrelated samples,
most of the energy in
first coefficient

Properties of Transform

- Transforms can be viewed as decomposition of data into basis functions
  - DFT
    - Basis functions include sine and cosine of different freq.
  - Hadamard Transform
    - Square waves of different freq.
- Rotation of coordinates
  - Transform itself does not compress data
- Energy Reallocation (Energy compaction)
  - Change statistical properties
  - Decrease the correlation between samples
  - Transform coding gain
    G_{TC}=arithmetic mean of variances /geometric mean of variance

Hadamard Transform

- Use square functions as its basis functions
- Based on the *Hadamard matrix*
  - A square array of +1's & -1's whose rows and columns are orthogonal.
- $H_1=[1]$
- Hadamard matrix with size 2N

$$\mathbf{H}_2 = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$\mathbf{H}_{2N} = \frac{1}{\sqrt{2}}\begin{bmatrix} \mathbf{H}_N & \mathbf{H}_N \\ \mathbf{H}_N & -\mathbf{H}_N \end{bmatrix}$$

$$\begin{array}{cc} & \text{Sign Changes} \\ \mathbf{H}_4 = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} & \begin{matrix} 0 \\ 3 \\ 1 \\ 2 \end{matrix} \end{array}$$

$$\begin{array}{cc} & \text{Sign Changes} \\ \mathbf{H}_8 = \frac{1}{2\sqrt{2}}\left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right] & \begin{matrix} 0 \\ 7 \\ 3 \\ 4 \\ 1 \\ 6 \\ 2 \\ 5 \end{matrix} \end{array}$$

Hadamard Transform

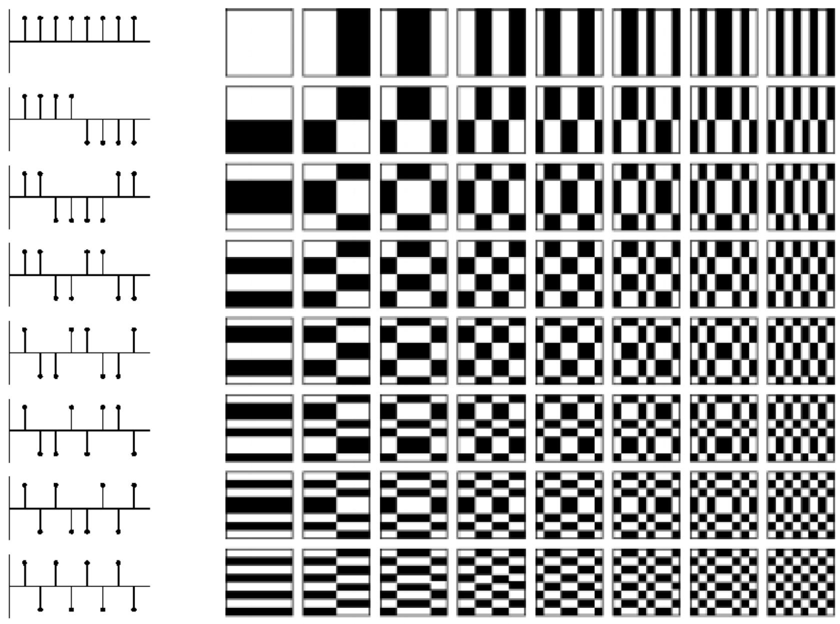
- It is possible to construct a Hadamard matrix of order whose number of sign changes per row increases from 0 to $N-1$

Natural				Sequential	
decimal	binary	Gray		bit reversed	decimal
0	000	000	→	000	0
1	001	001	→	100	4
2	010	011	→	110	6
3	011	010	→	010	2
4	100	110	→	011	3
5	101	111	→	111	7
6	110	101	→	101	5
7	111	100	→	001	1

$$\begin{array}{c} \text{row} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}
\begin{array}{c} \text{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{pmatrix} \end{array}
\quad
\begin{array}{c} \text{row} \\ 0 \\ 4 \\ 6 \\ 2 \\ 3 \\ 7 \\ 5 \\ 1 \end{array}
\begin{array}{c} \text{matrix} \\ \begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \end{pmatrix} \end{array}
\quad
\begin{array}{c} \text{sign changes} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{array}$$

Hadamard Transform

- Only addition and subtraction operations are needed
- Popular when efficiency consideration dominates
- Performance moderate



Discrete Fourier Transform

- DFT: $\mathbf{X}$[k]=DFT{$\mathbf{x}$[n]}= $\sum_{n=0}^{N-1} \mathbf{x}[n] e^{-j\frac{2\pi}{N}kn}$
- IDFT: $\mathbf{x}$[n]=IDFT{$\mathbf{X}$[k]}= $\frac{1}{N}\sum_{k=0}^{N-1} \mathbf{X}[k] e^{j\frac{2\pi}{N}kn}$
- x is represented in terms of superposition of N complex exponentials {$\mathbf{e}_0$, $\mathbf{e}_1$, …, $\mathbf{e}_{N-1}$}

$$\mathbf{e}_\mathbf{k}[n] = e^{j\frac{2\pi}{N}kn} = W_N^{-kn} \quad (e^{-j\frac{2\pi}{N}} = W_N)$$

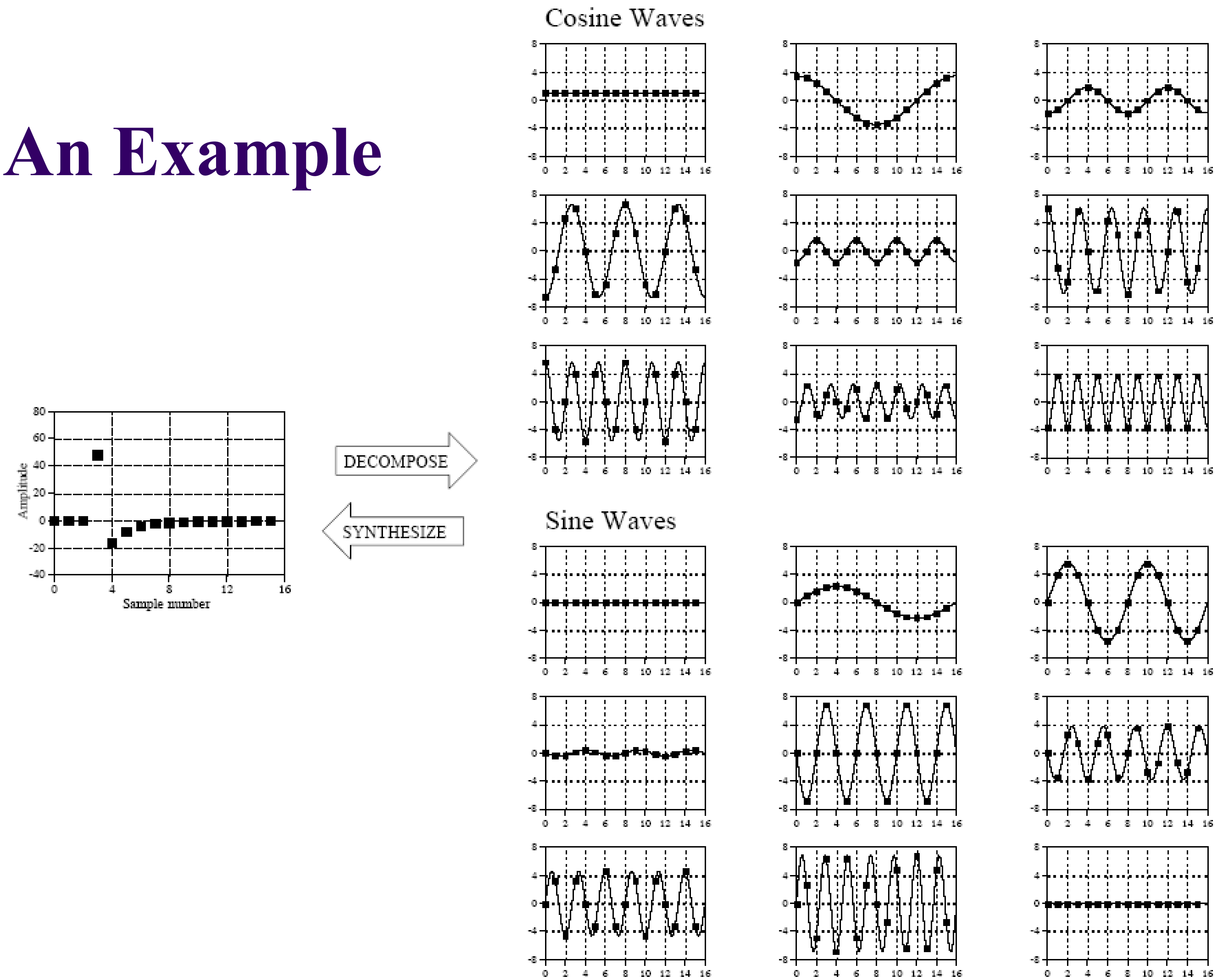
$$\mathbf{x} = \begin{bmatrix} \mathbf{x}[0] \\ \mathbf{x}[1] \\ \vdots \\ \mathbf{x}[N-1] \end{bmatrix}, \mathbf{X} = \begin{bmatrix} \mathbf{X}[0] \\ \mathbf{X}[1] \\ \vdots \\ \mathbf{X}[N-1] \end{bmatrix}, \mathbf{e}_\mathbf{k} = \begin{bmatrix} W_N^0 \\ W_N^{-k} \\ \vdots \\ W_N^{-k(N-1)} \end{bmatrix} \Rightarrow \mathbf{X}[k] = \mathbf{e}_\mathbf{k}^{*\mathbf{T}}\mathbf{x}$$

$$\mathbf{X} = \begin{bmatrix} \mathbf{e}_\mathbf{0}^{*\mathbf{T}} \\ \mathbf{e}_\mathbf{1}^{*\mathbf{T}} \\ \vdots \\ \mathbf{e}_\mathbf{N-1}^{*\mathbf{T}} \end{bmatrix} \mathbf{x} = \begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^1 & \cdots & W_N^{N-1} \\ \vdots & \vdots & \cdots & \vdots \\ W_N^0 & W_N^{N-1} & \cdots & W_N^{(N-1)(N-1)} \end{bmatrix} \mathbf{x} = \mathbf{W}_\mathbf{N}\mathbf{x}$$

Discrete Fourier Transform

- $$\mathbf{e}_p^{*T}\mathbf{e}_m = \left[W_N^0 \ W_N^p \cdots W_N^{p(N-1)}\right]\begin{bmatrix} W_N^0 \\ W_N^{-m} \\ \vdots \\ W_N^{-m(N-1)} \end{bmatrix} = \sum_{k=0}^{N-1} W_N^{(p-m)k} = \frac{1-W_N^{(p-m)N}}{1-W_N^{(p-m)}}$$
  $$= \begin{cases} \text{N, if p=m} \\ \text{0, otherwise} \end{cases}$$
- $$\mathbf{W}_\mathbf{N}\mathbf{W}_\mathbf{N}^{*T} = \begin{bmatrix} \mathbf{e}_\mathbf{0}^{*T} \\ \vdots \\ \mathbf{e}_\mathbf{N-1}^{*T} \end{bmatrix}\left[\mathbf{e}_\mathbf{0} \ldots \mathbf{e}_\mathbf{N-1}\right] = N\mathbf{I}_\mathbf{NxN} \Rightarrow \mathbf{W}_\mathbf{N}^{-1} = \frac{1}{N}\mathbf{W}_\mathbf{N}^{*T}$$
- $$\mathbf{X} = \mathbf{W}_\mathbf{N}\mathbf{x} \Rightarrow \mathbf{x} = \mathbf{W}_\mathbf{N}^{-1}\mathbf{X} = \frac{1}{N}\mathbf{W}_\mathbf{N}^{*T}\mathbf{X} = \frac{1}{N}\begin{bmatrix} W_N^0 & W_N^0 & \cdots & W_N^0 \\ W_N^0 & W_N^* & \cdots & W_N^{*(N-1)} \\ \vdots & \vdots & \cdots & \vdots \\ W_N^0 & W_N^{*(N-1)} & \cdots & W_N^{*(N-1)(N-1)} \end{bmatrix}\mathbf{X}$$

An Example



Cosine Waves
Amplitude
Sample number
DECOMPOSE
SYNTHESIZE
Sine Waves

2D Discrete Fourier Transform

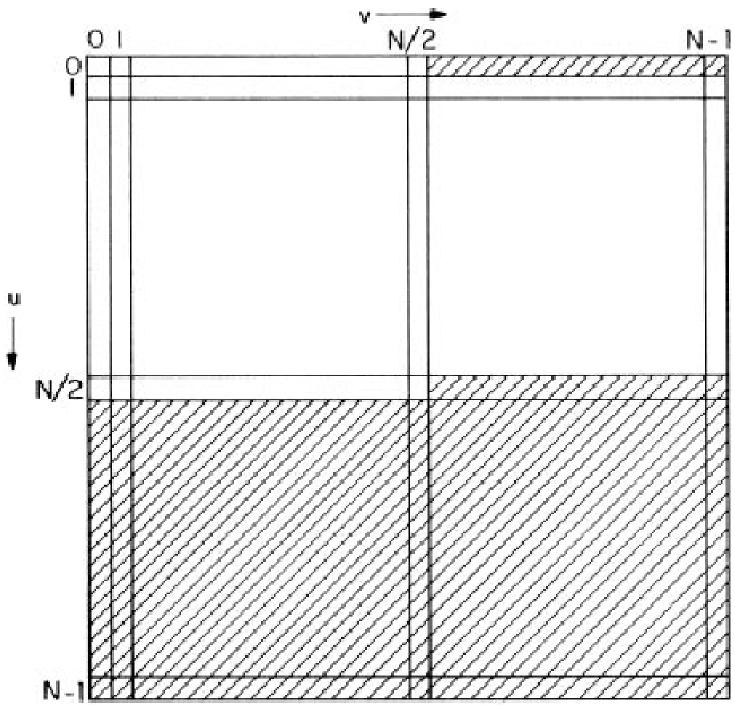
- Use *cosθ* and *sinθ* as its basis functions

$$F(u,v) = \frac{1}{N}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y) e^{\frac{-2\pi i(ux+vy)}{N}}$$

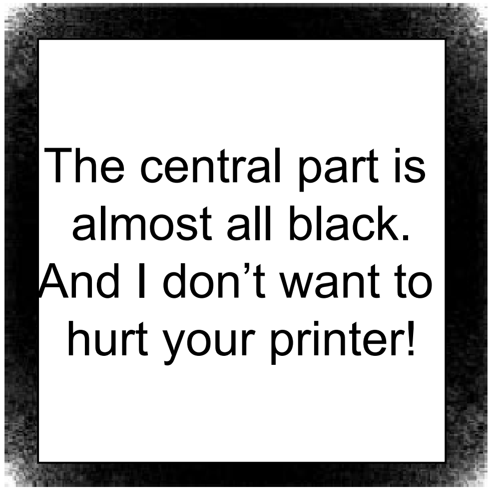
$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v) e^{\frac{2\pi i(ux+vy)}{N}}$$

- Fast Fourier Transform (FFT) -> $O(N\log N)$
- Represented by *Real+Complex* or *Magnitude+Phase*
- BTW, F(u,v)=F*(-u+mN,-v+nN)

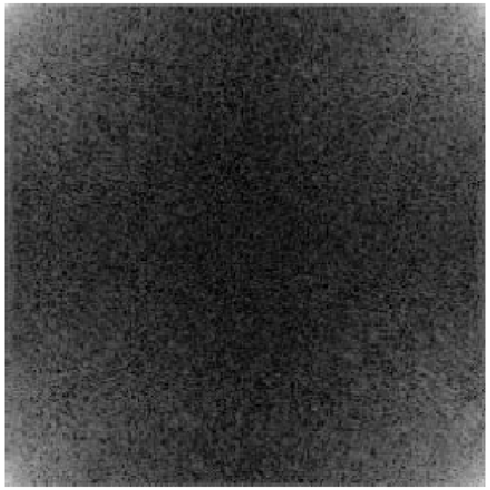
DFT on Digital Images



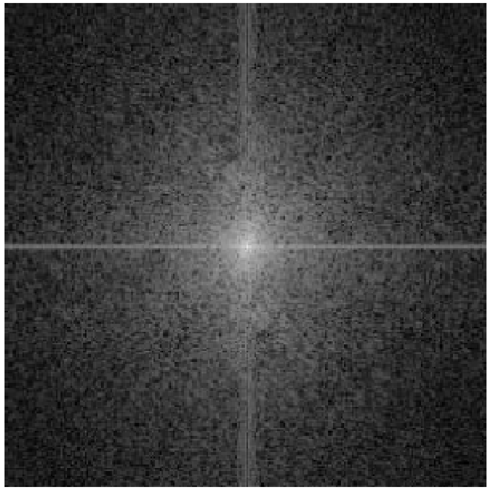
(a) Original



(b) Clipped magnitude, nonordered



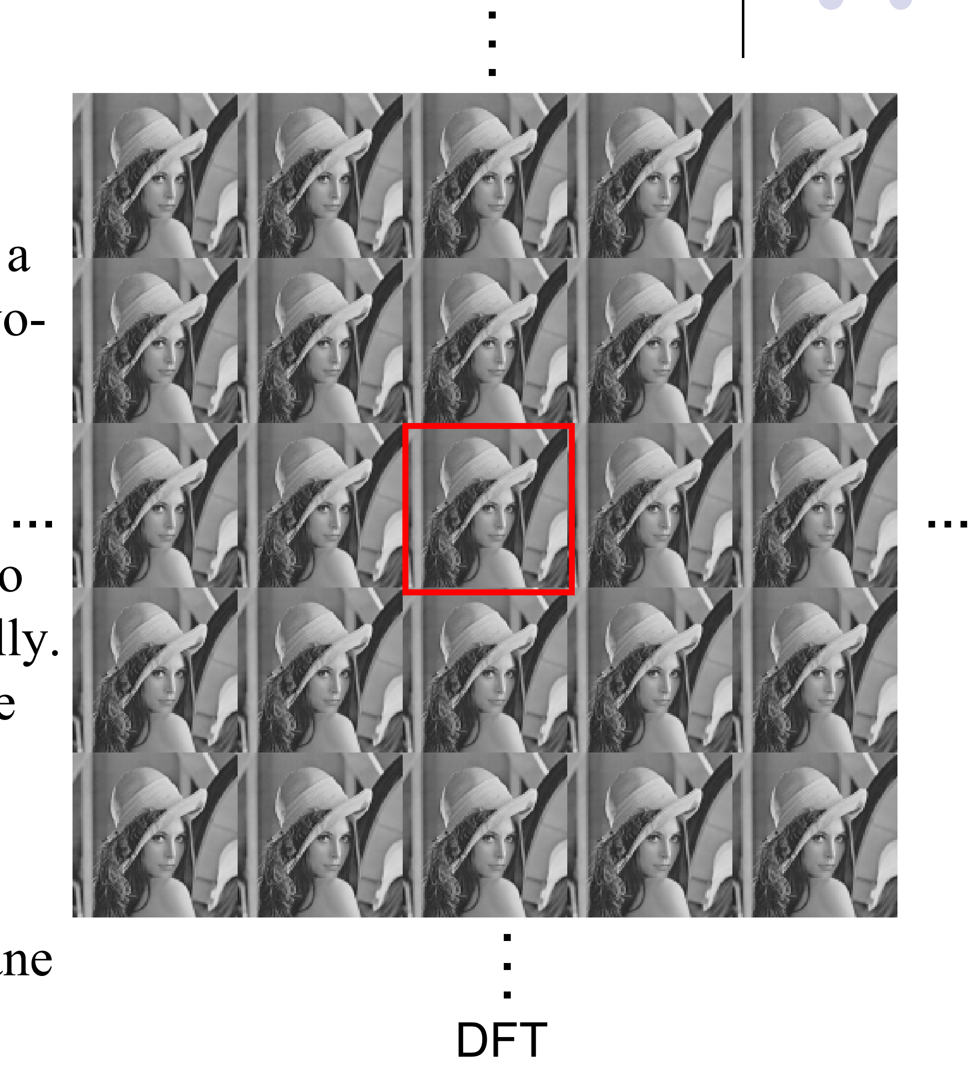
(c) Log magnitude, nonordered



(d) Log magnitude, ordered

DFT on Digital Images

- Periodically tiled images:
  - The two-dimensional Fourier transform of an image is essentially a Fourier series representation of a two-dimensional field. For the Fourier series representation to be valid, the field must be periodic. Thus, the original image must be considered to be periodic horizontally and vertically. The right side of the image therefore abuts the left side, and the top and bottom of the image are adjacent. Spatial frequencies along the coordinate axes of the transform plane arise from these transitions.



Is DFT suitable to compression?

- Maybe not. Sharp discontinuities are introduced at the beginning and the end of the sequence. To represent these sharp discontinuities, the DFT needs nonzero coefficients of the high-frequency components. Because these components are needed only at the two endpoints of the sequence, their effect needs to be cancelled out at other points in the sequence. Thus, the DFT adjust other coefficients accordingly. When we discard the high-frequency coefficients (which should not have been there anyway) during the compression process, the coefficients that were canceling out the high-frequency effect in other parts of the sequence result in the introduction of additional distortion.

Karhunen Loève Transform (KLT)

- Karhunen Loève Transform (KLT): basis functions are eigenvectors of the covariance matrix $R_{\boldsymbol{XX}}$ of the input signal
- KLT yields decorrelated transform coefficients (covariance matrix $R_{\boldsymbol{YY}}$ is diagonal).
- KLT achieves optimum energy concentration
- Disadvantage:
  - KLT dependents on signal statistics
  - KLT not separable for image blocks
  - Transform matrix cannot be factored into sparse matrices -> computation is complex

Karhunen-Loeve Transform
(Hotelling Transform, Eigenvector Transform)

- Procedure:
  - Consider a vector $\mathbf{z}=[z_1,z_2,\ldots,z_N]^T$.
  - Take vectors from the signal to calculate its mean $\mathbf{m}=[m_1,\ldots,m_N]^T$.
  - Calculate the covariance matrix of the signal: $\mathbf{C}_z=E[(\mathbf{z}-\mathbf{m})(\mathbf{z}-\mathbf{m})^T]_{NxN}$, where $c_{i,j}=E[(z_i-m_i)(z_j-m_j)]=\mathrm{Cov}(z_i,z_j)$
  - Calculate eigenvectors of $\mathbf{C}_z$. $(\mathbf{C}_z\mathbf{e_i}=\lambda_\mathbf{i}\mathbf{e_i})$. $\mathbf{e_i}=[e_{i1},e_{i2},\ldots,e_{iN}]^T$. There are N eigenvectors with the corresponding eigenvalues.
  - Let $\mathbf{\Phi}_{NxN}=(\mathbf{e_1},\mathbf{e_2},\ldots,\mathbf{e_N})^T$. KLT: $\mathbf{y}=\mathbf{\Phi}(\mathbf{z}-\mathbf{m})$, IKLT: $\mathbf{z}=\mathbf{\Phi}^T\mathbf{y}+\mathbf{m}$.
- Facts
  - $\mathbf{m}_y=\mathbf{0}$
  - $\mathbf{C}_y=E[\mathbf{y}\mathbf{y}^T]=E[\mathbf{\Phi}(\mathbf{z}-\mathbf{m})\,(\mathbf{z}-\mathbf{m})^T\mathbf{\Phi}^T]=\mathbf{\Phi}\mathbf{C}_z\mathbf{\Phi}^T=\mathbf{\Lambda}=$ $\begin{bmatrix} \lambda_1 & & & & 0 \\ & \lambda_2 & & & \\ & & \ddots & & \\ & & & \ddots & \\ 0 & & & & \lambda_n \end{bmatrix}$
  - Arrange the eigenvalues (variances) in a non-increasing order. Choose L<N to get a new transform matrix, $\mathbf{\Omega}_{LxN}$
    $\mathbf{w}_{Lx1}=\mathbf{\Omega}(\mathbf{z}-\mathbf{m})$, $\mathbf{\Omega}_{LxN}=(\mathbf{e_1},\mathbf{e_2},\ldots,\mathbf{e_L})^T$
    $\mathbf{z}'=\mathbf{\Omega}^T\mathbf{w}+\mathbf{m}$.

$$\mathbf{C}_z\mathbf{\Phi}^T=\mathbf{\Phi}^T\mathbf{\Lambda}$$

$$MSE=\sum_{i=L+1}^{N}\sigma_{y,i}^2.$$

The mean square reconstruction error equals the sum of variances of the discarded components

Discrete Cosine Transform

- Use cosine function as its basis function
- Performance approaches KLT
- DCT causes less blocking effect than DFT
- Close to FFT -> Fast algorithm exists
- Most popular in image compression application

Discrete Cosine Transform

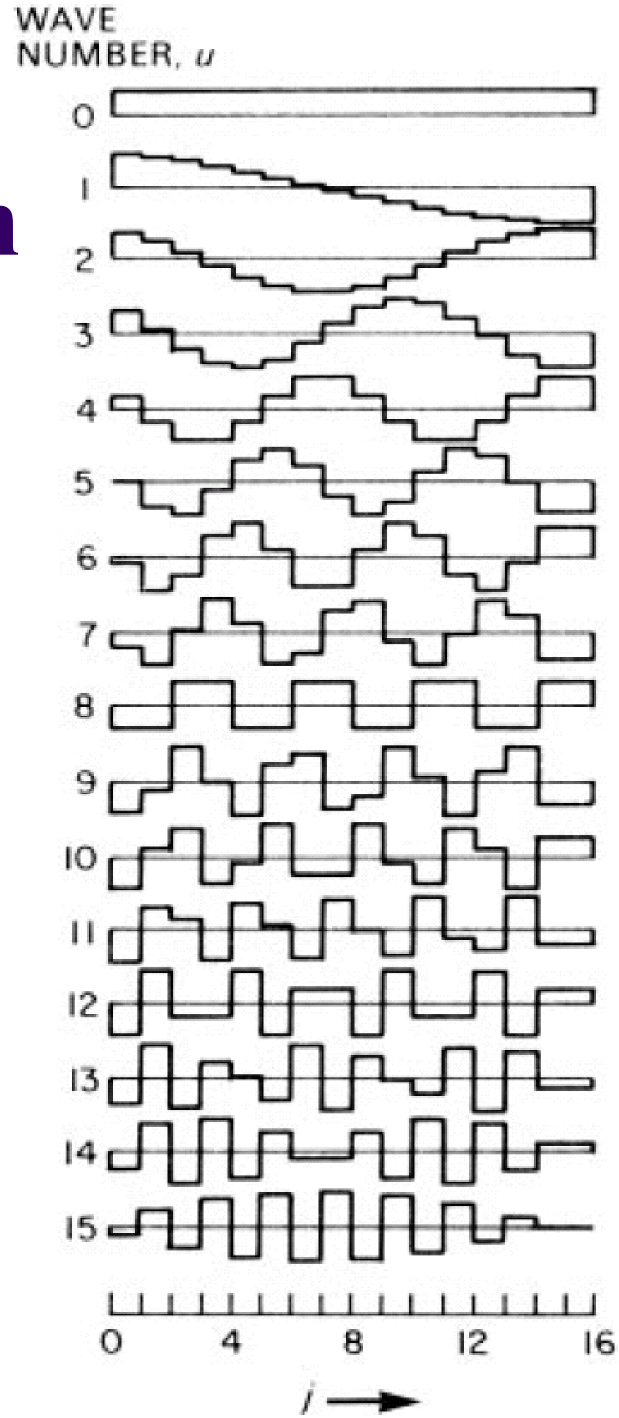
FDCT:

$$X[k] = \sum_{n=0}^{N-1} C_k \times x[n] \times \cos\left(\frac{2\pi}{2N} k(n+\frac{1}{2})\right)$$

IDCT:

$$x[n] = \sum_{k=0}^{N-1} C_k \times X[k] \times \cos\left(\frac{2\pi}{2N} k(n+\frac{1}{2})\right)$$

$$\text{where } C_k = \begin{cases} \sqrt{\frac{1}{N}}, \text{ if k=0} \\ \sqrt{\frac{2}{N}}, \text{ if } k \neq 0 \end{cases}$$



Discrete Cosine Transform

- N=4

$FDCT:$

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix} = 1/\sqrt{2} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2} & 1/\sqrt{2}) \\ \cos(\pi/8) & \cos(3\pi/8) & \cos(5\pi/8) & \cos(7\pi/8) \\ \cos(2\pi/8) & \cos(6\pi/8) & \cos(10\pi/8) & \cos(14\pi/8) \\ \cos(3\pi/8) & \cos(9\pi/8) & \cos(15\pi/8) & \cos(21\pi/8) \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}$$

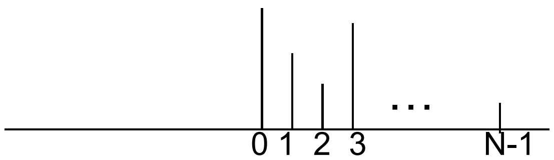
$$= \begin{bmatrix} a & a & a & a \\ b & c & -c & -b \\ a & -a & -a & a \\ c & -b & b & -c \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix}, a = 1/2, b = \frac{1}{\sqrt{2}}\cos(\frac{\pi}{8}), c = \frac{1}{\sqrt{2}}\cos(\frac{3\pi}{8}).$$

$IDCT:$

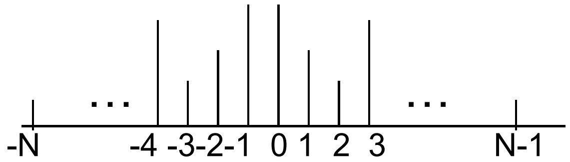
$$\begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \end{bmatrix} = 1/\sqrt{2} \begin{bmatrix} 1/\sqrt{2} & \cos(\pi/8) & \cos(2\pi/8) & \cos(3\pi/8) \\ 1/\sqrt{2} & \cos(3\pi/8) & \cos(6\pi/8) & \cos(9\pi/8) \\ 1/\sqrt{2} & \cos(5\pi/8) & \cos(10\pi/8) & \cos(15\pi/8) \\ 1/\sqrt{2} & \cos(7\pi/8) & \cos(14\pi/8) & \cos(21\pi/8) \end{bmatrix} \begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \end{bmatrix}$$

Discrete Cosine Transform

x[n]



Mirror-extended x[n]



x[-1-n]=x[n]
n=0,1,…,N-1

(Define x[n] = x[n], N-1>n>=0
= x[-1-n], -1>=n>=-N)

$$X[k]=\sum_{n=-N}^{N-1} x[n]\times e^{-j\frac{2\pi}{2N}k(n+\frac{1}{2})}$$

$$=\sum_{n=-N}^{-1} x[n]\times e^{-j\frac{2\pi}{2N}k(n+\frac{1}{2})}+\sum_{n=0}^{N} x[n]\times e^{-j\frac{2\pi}{2N}k(n+\frac{1}{2})}$$

For the first term, let m=-1-n $\Rightarrow n=-1-m$

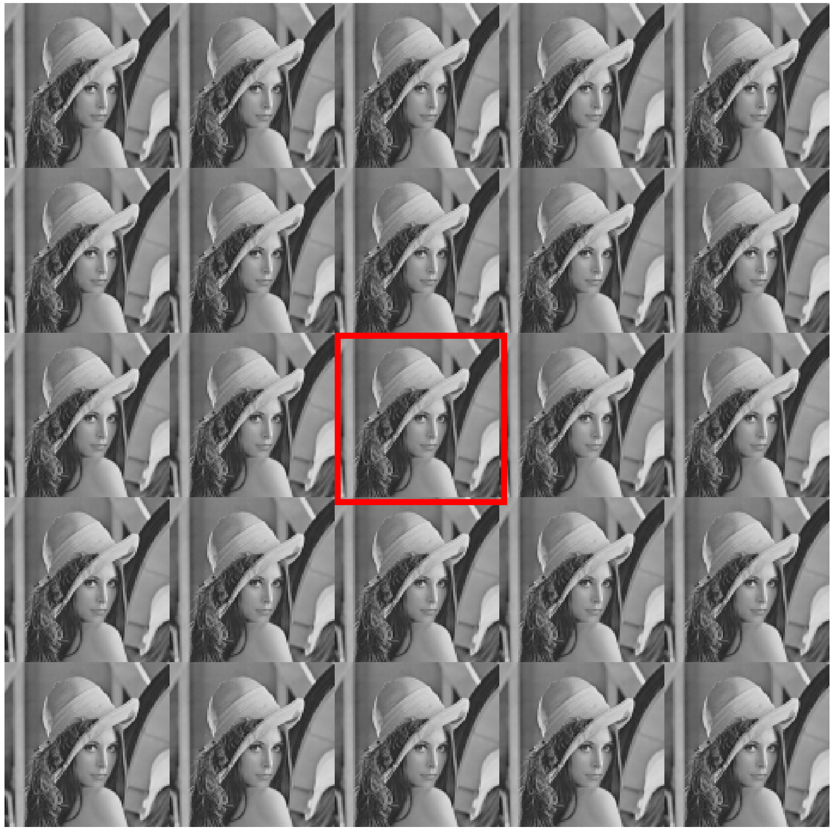
$$\sum_{n=-N}^{-1} x[n]\times e^{-j\frac{2\pi}{2N}k(n+\frac{1}{2})}=\sum_{m=0}^{N-1} x[-1-m]\times e^{-j\frac{2\pi}{2N}k(-m-\frac{1}{2})}$$

$$=\sum_{m=0}^{N-1} x[m]\times e^{j\frac{2\pi}{2N}k(m+\frac{1}{2})}$$

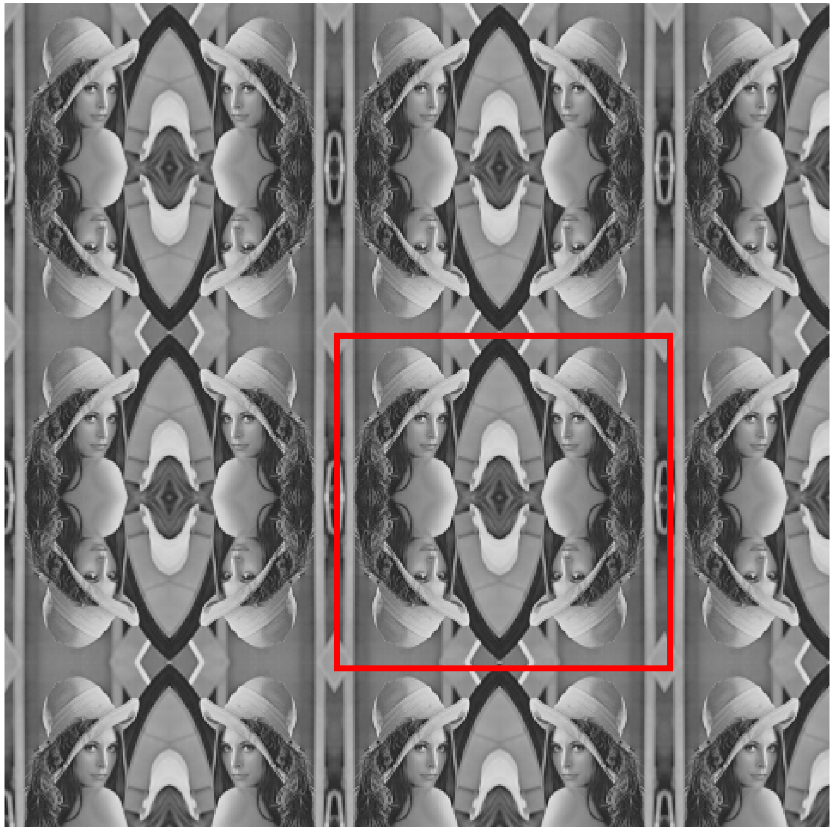
$$X[k]=\sum_{n=0}^{N-1} x[n]\times\left(e^{j\frac{2\pi}{2N}k(n+\frac{1}{2})}+e^{-j\frac{2\pi}{2N}k(n+\frac{1}{2})}\right)$$

$$=\sum_{n=0}^{N-1} x[n]\times 2\cos\left(\frac{2\pi}{2N}k(n+\frac{1}{2})\right)$$

DFT vs. DCT



DFT



DCT

Matrix Representation of 2-D 8x8 DCT

- How to apply 1-D DCT to a 2D-image?
  - Apply 1-D DCT horizontally, and then apply 1-D DCT vertically
- Matrix representation of 2-D 8x8 DCT
  - DCT : $S_{8x8} = C_{8x8}\, s_{8x8}\, C_{8x8}^{\ T}$
  - IDCT : $s_{8x8} = C_{8x8}^{\ T}\, S_{8x8}\, C_{8x8}$

2 Dimensional 8x8 DCT

DCT :

$$S(v,u)=\frac{C(v)}{2}\frac{C(u)}{2}\sum_{y=0}^{7}\sum_{x=0}^{7}s(y,x)\cos[\frac{(2x+1)u\pi}{16}]\cos[\frac{(2y+1)v\pi}{16}]$$

$$u,v=0\ldots7$$

IDCT :

$$s(y,x)=\sum_{v=0}^{7}\frac{C(v)}{2}\sum_{u=0}^{7}\frac{C(u)}{2}S(y,x)\cos[\frac{(2x+1)u\pi}{16}]\cos[\frac{(2y+1)v\pi}{16}]$$

$$y,x=0\ldots7$$

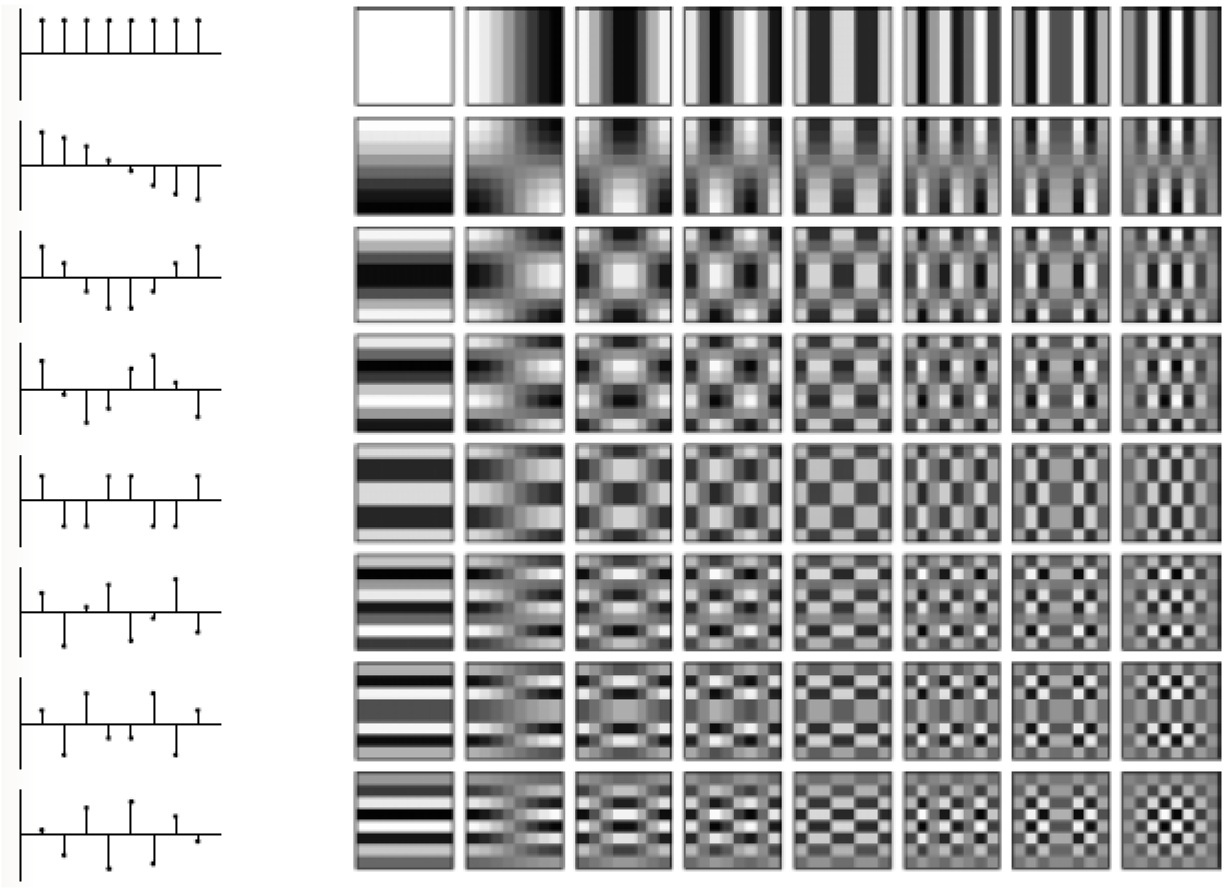
$$C(u)=\begin{matrix}1/\sqrt{2} & ,u=0\\ 1 & ,u>0\end{matrix}$$

$$C(v)=\begin{matrix}1/\sqrt{2} & ,v=0\\ 1 & ,v>0\end{matrix}$$

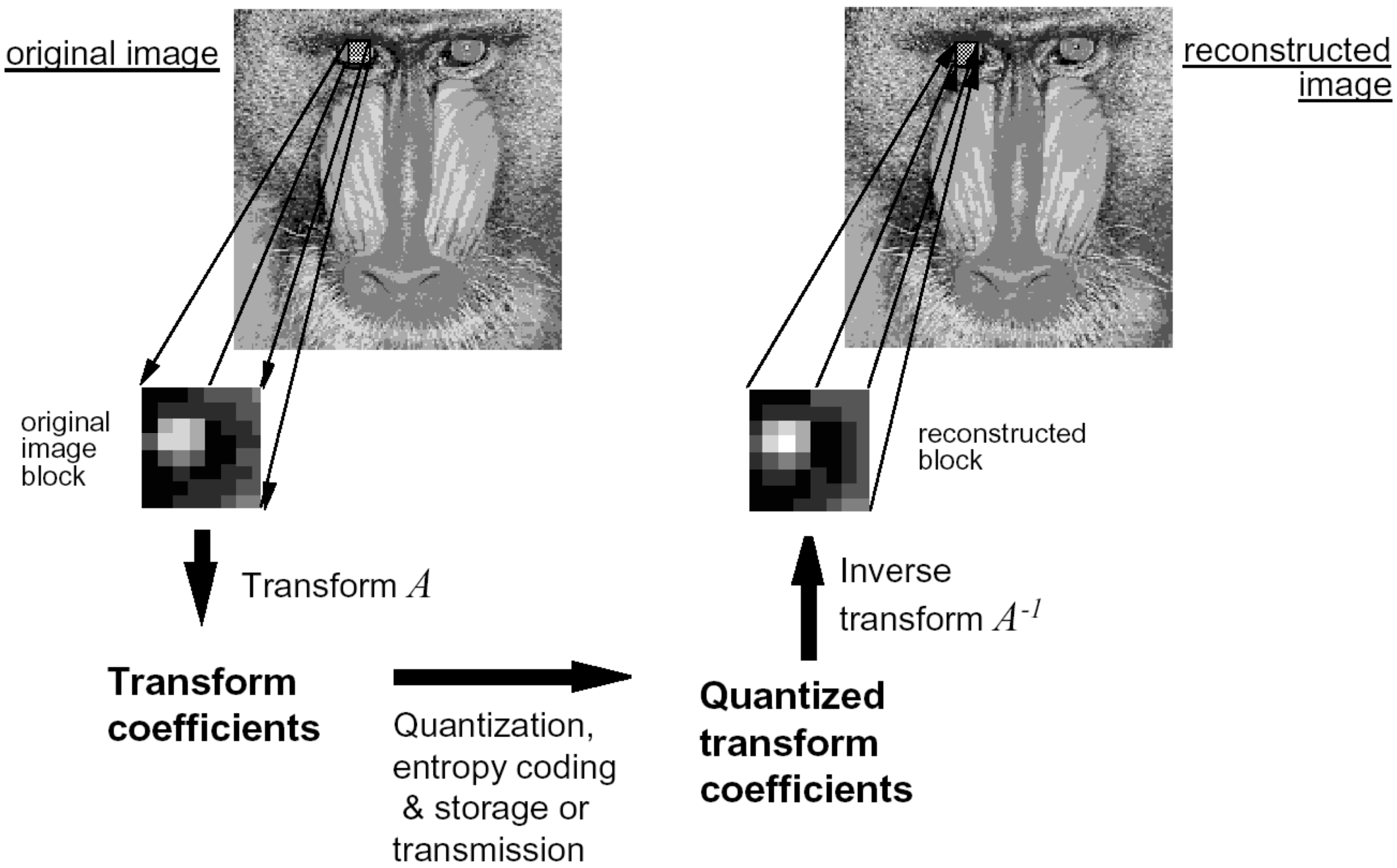
s(y,x) : input 8x8 data

S(v,u) : 8x8 DCT coefficient data

Basis function for 2D 8x8 DCT



Transform Coding



Transform Coding

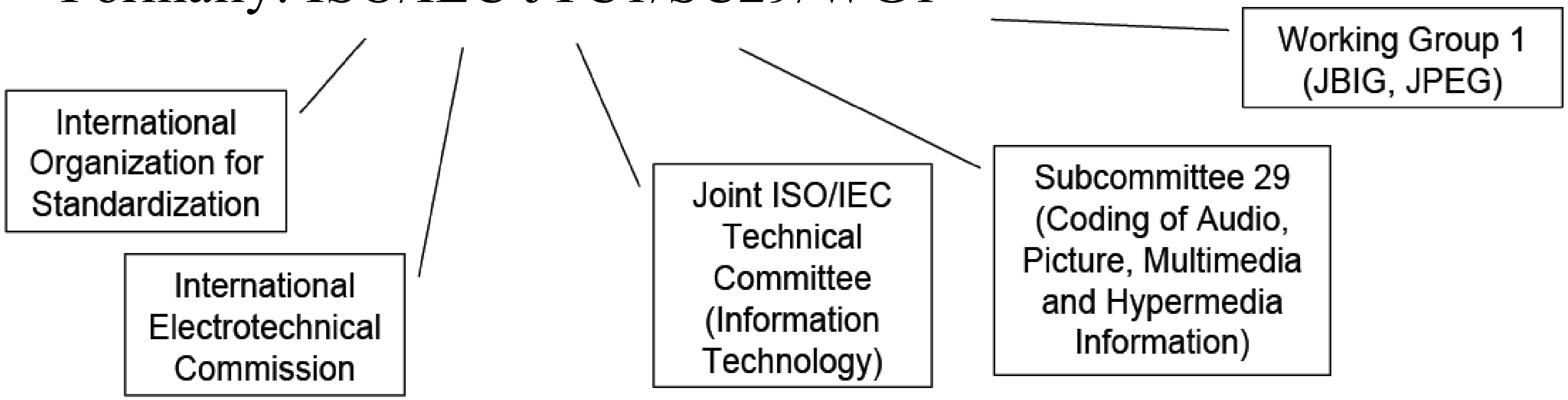
- Encoding:
  - Transform the input pixels $x_0, x_1,..., x_{N-1}$ into coefficients $c_0, c_1,..., c_{N-1}$ (real values):
    - The coefficients have the property that most of them are near zero.
    - Most of the "energy" is compacted into a few coefficients.
  - (Scalar) quantize the coefficient:
    - This is bit allocation!
    - Important coefficients should have more quantization levels (= represented with more accuracy).
  - Entropy encode the quantized values.
- Decoding
  - Entropy decode the quantized values.
  - Compute approximate coefficients $c'_0, c'_1,..., c'_{N-1}$ from the quantized values.
  - Inverse transform $c'_0, c'_1,..., c'_{N-1}$ to $x'_0, x'_1,..., x'_{N-1}$ which is (we hope) a good approximation of the original $x_0, x_1,..., x_{N-1}$.

Note on Transform Coding

- Image Coding:
  - JPEG –uses DCT.
- Video Coding:
  - MPEG –uses DCT.
  - H.261/263/264 –use DCT.
- Audio Coding:
  - MP3 = MPEG 1- Layer 3 uses DCT.
- Alternative Transforms:
  - Lapped transforms remove some of the blocking artifacts.
  - Wavelet transforms do not need to use blocks at all.

JPEG Still Image Compression

- JPEG:
  - Joint Photographic Experts Group
  - Formally: ISO/IEC JTC1/SC29/WG1



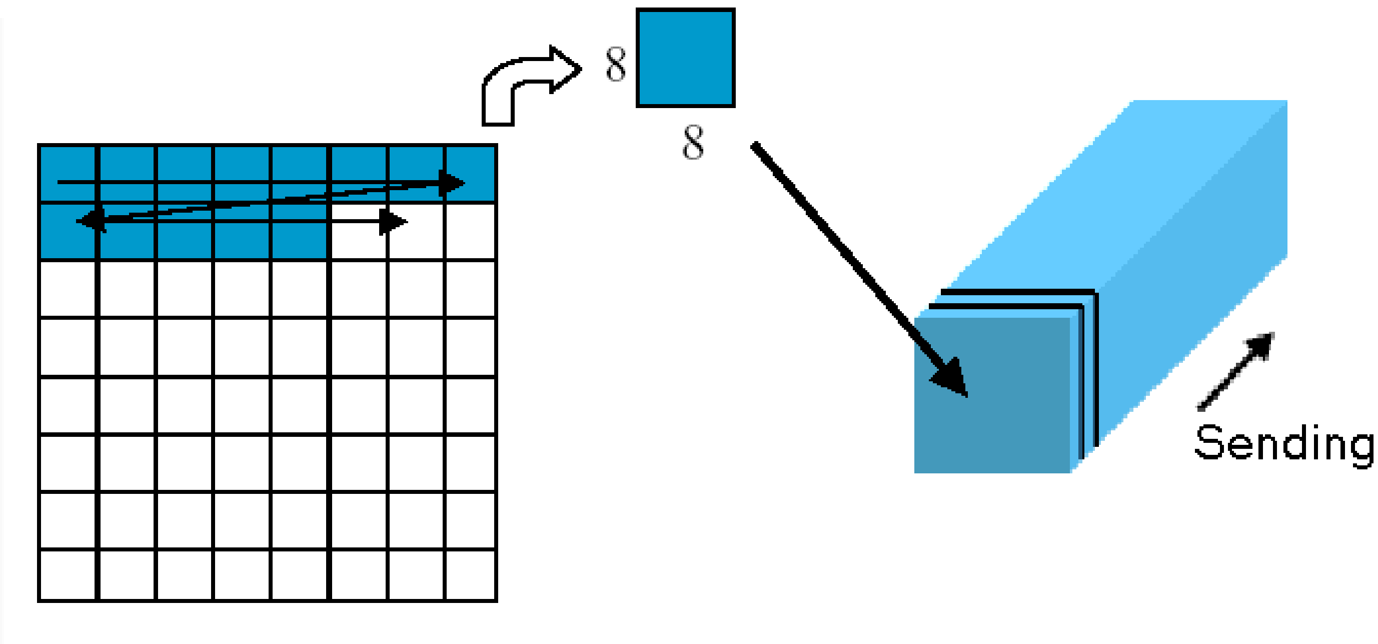
- Work commenced in mid-1980’s and draft international standard in 1991
- Widely used for image exchange
- Motion-JPEG is de facto standard for digital video

Modes of Operation

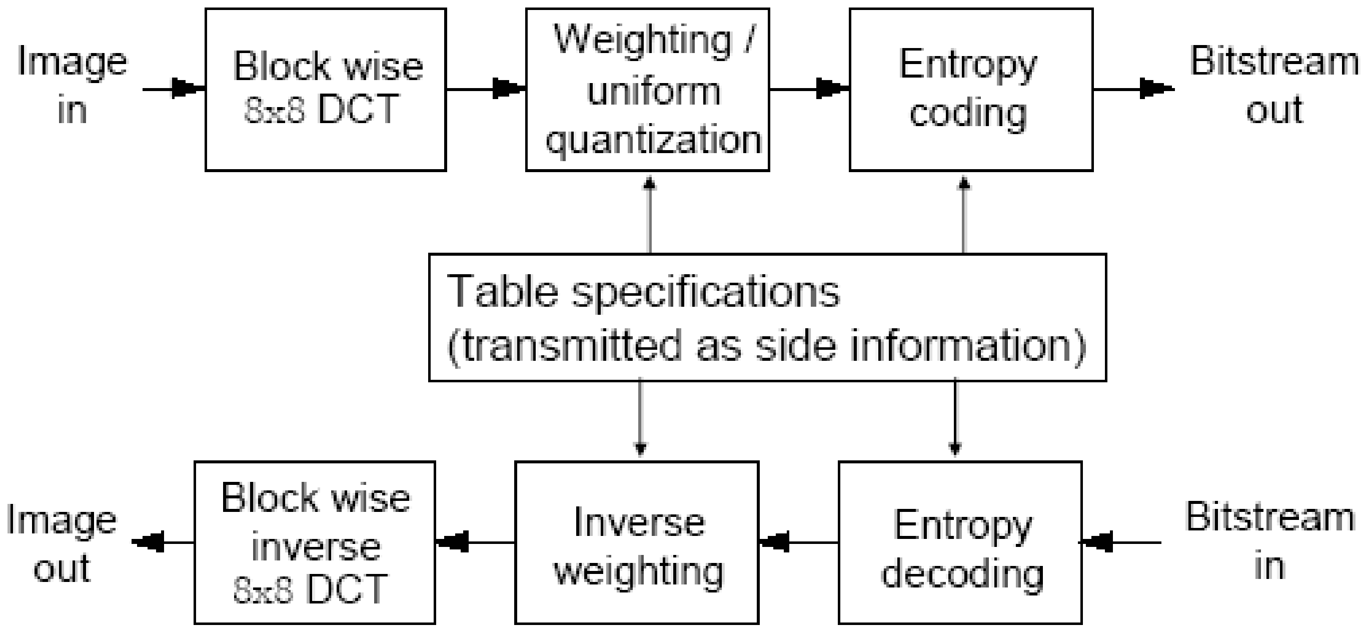
- Baseline Sequential DCT-based
- Progressive DCT-based Mode
- Loss-less Mode
- Hierarchical Mode

Baseline Sequential DCT-based

- Each image component is encoded in a single left-to-right top-to-bottom scan



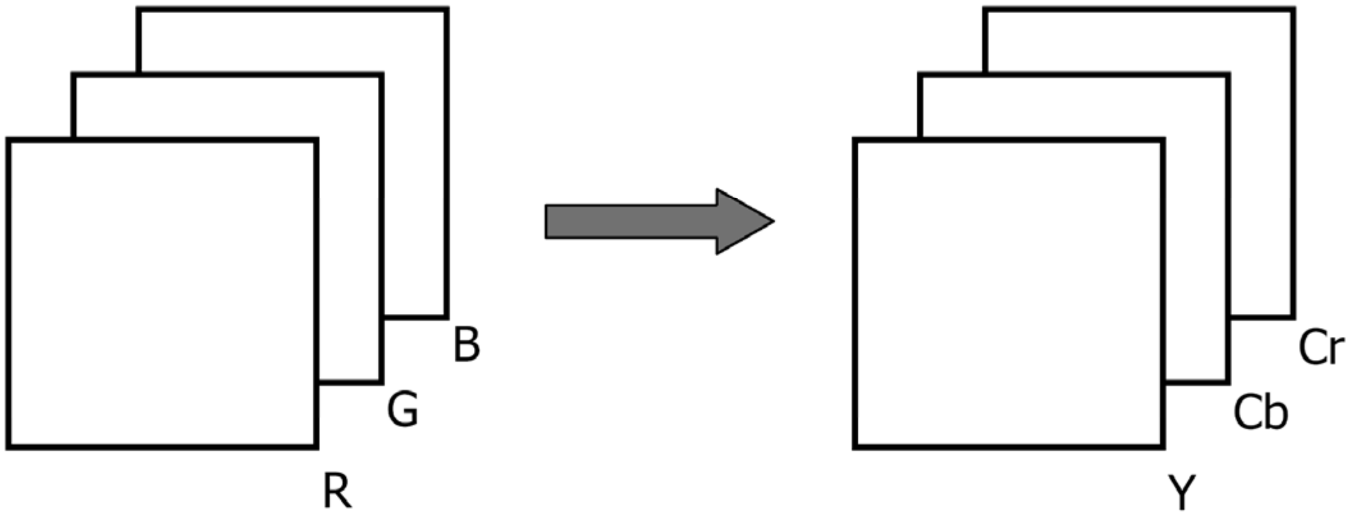
Block Diagram of JPEG-baseline



Baseline Sequential DCT-based Encoding Process

- Color space conversion
- Partition
- Subsampling
- Encoding flow control
  - Discrete Cosine Transform (DCT)
  - Quantization
  - Entropy Encoding (Huffman)

Color Space Conversion



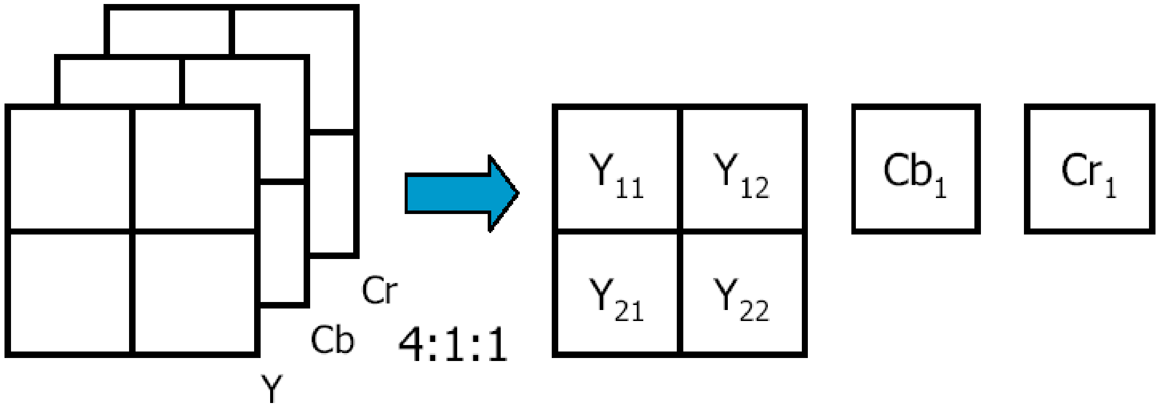
Y – Luminance

Cb,Cr – Chrominance

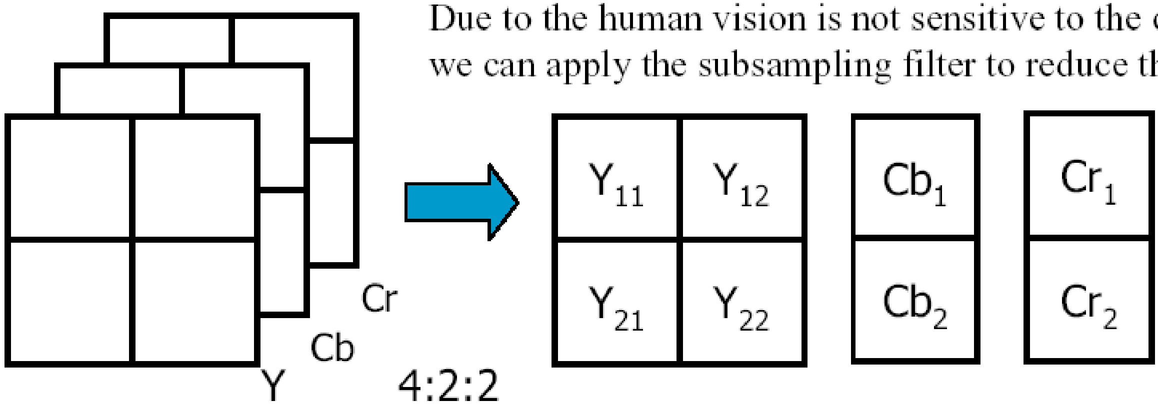
Y=0.299R'+0.587G'+0.114B'
Cb=-0.16875R'-0.33126G'+0.5B'
Cr=0.5R'-0.41869G'-0.08131B

R'=Y+1.402Cr
G'=Y-0.34413Cb-0.71414Cr
B'=Y+1.772Cb

Subsampling

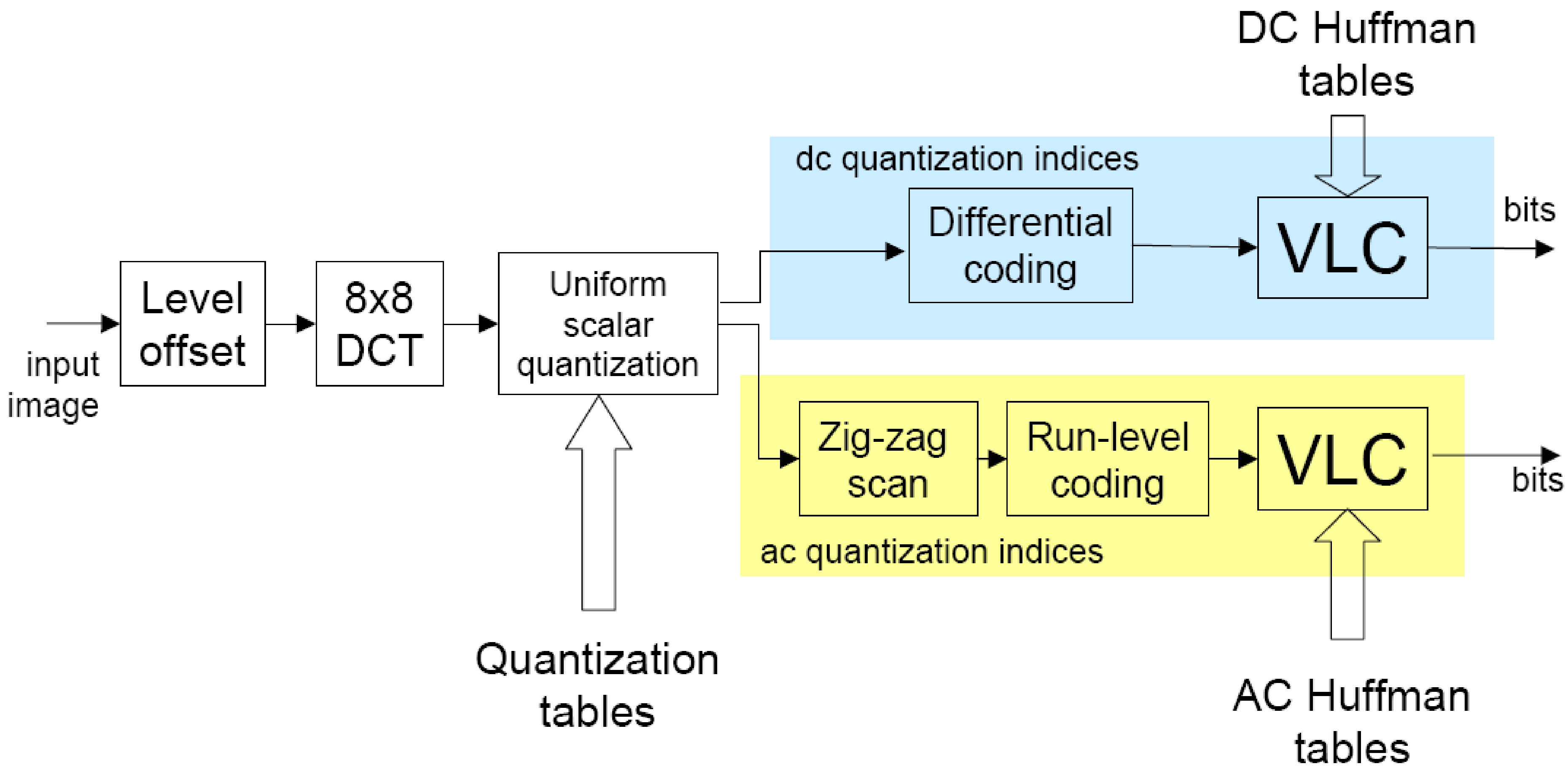


Due to the human vision is not sensitive to the chrominance domain, we can apply the subsampling filter to reduce the data size.



Note: If there is no subsampling, we called this 4:4:4 mode

Closer View of JPEG Encoding



Example of 2-D 8x8 DCT/IDCT

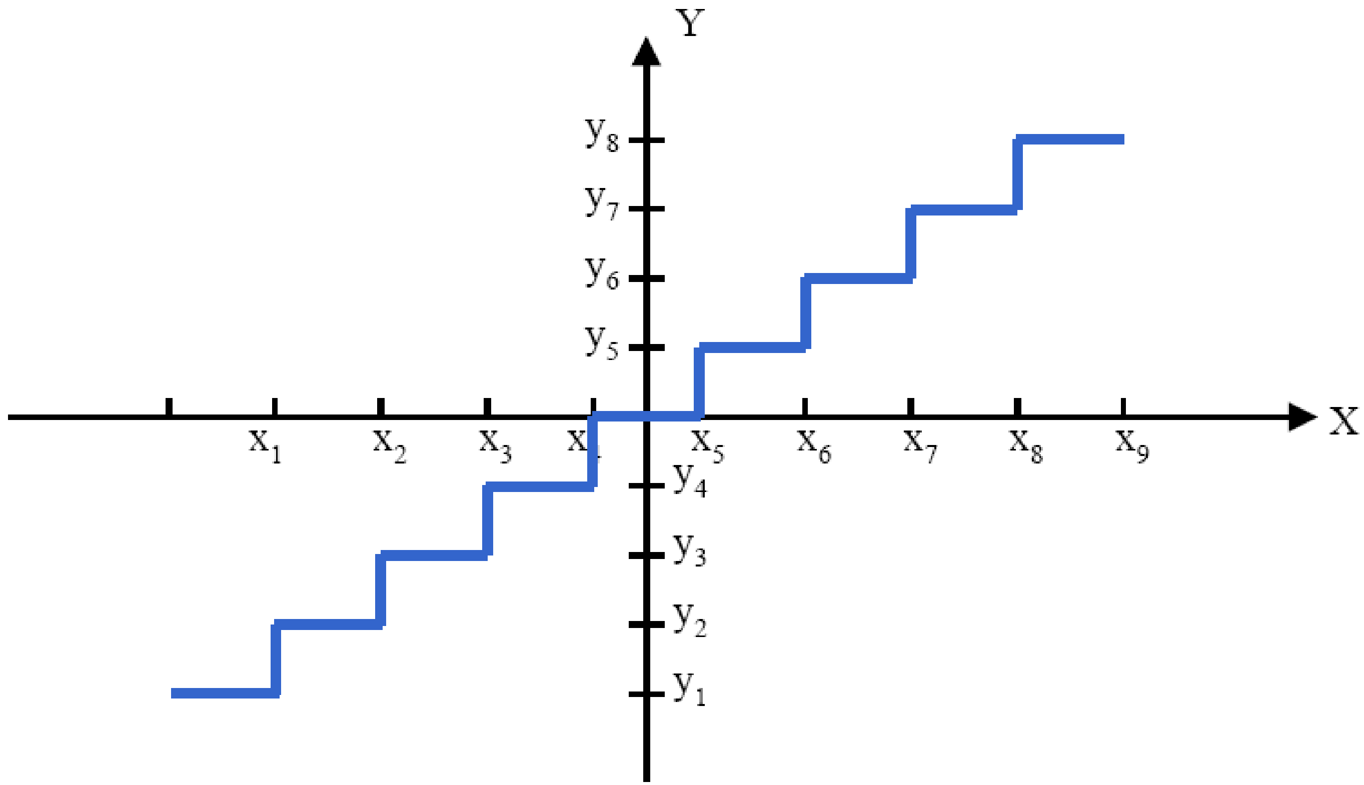
$$\begin{bmatrix} 139 & 144 & 149 & 153 & 155 & 155 & 155 & 155 \\ 144 & 151 & 153 & 156 & 159 & 156 & 156 & 156 \\ 150 & 155 & 160 & 163 & 158 & 156 & 156 & 156 \\ 159 & 161 & 162 & 160 & 160 & 159 & 159 & 159 \\ 159 & 160 & 161 & 162 & 162 & 155 & 155 & 155 \\ 161 & 161 & 161 & 161 & 160 & 157 & 157 & 157 \\ 162 & 162 & 161 & 163 & 162 & 157 & 157 & 157 \\ 162 & 162 & 161 & 161 & 163 & 158 & 158 & 158 \end{bmatrix} \xrightarrow[\text{then DCT}]{\text{-128}} \begin{bmatrix} 236 & -1 & -12 & -5 & 2 & -2 & -3 & 1 \\ -23 & -17 & -6 & -3 & -3 & 0 & 0 & -1 \\ -11 & -9 & -2 & 2 & 0 & -1 & -1 & 0 \\ -7 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & -1 & 1 & 1 \\ 2 & 0 & 2 & 0 & -1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 2 & 1 & -1 \\ -3 & 2 & -4 & -2 & 2 & 1 & -1 & 0 \end{bmatrix}$$

Pixel domain

Frequency domain

Quantization

- Less bits are needed to code the quantized coefficient
- DCT process is lossless but quantization process is lossy



Example of Quantization

$$\begin{bmatrix} 236 & -1 & -12 & -5 & 2 & -2 & -3 & 1 \\ -23 & -17 & -6 & -3 & -3 & 0 & 0 & -1 \\ -11 & -9 & -2 & 2 & 0 & -1 & -1 & 0 \\ -7 & -2 & 0 & 1 & 1 & 0 & 0 & 0 \\ -1 & -1 & 1 & 2 & 0 & -1 & 1 & 1 \\ 2 & 0 & 2 & 0 & -1 & 1 & 1 & -1 \\ -1 & 0 & 0 & -1 & 0 & 2 & 1 & -1 \\ -3 & 2 & -4 & -2 & 2 & 1 & -1 & 0 \end{bmatrix}$$

DCT matrix

$$\div$$

$$\begin{bmatrix} 15 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -2 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Quantized DCT matrix

$$\begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

Quantization matrix

Default Quantization Table

Luminance

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	36	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Chrominance

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

factor

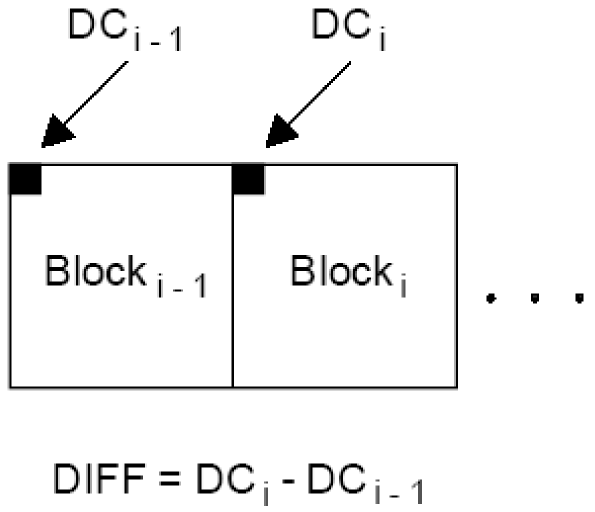
=5000/QF if QF<50

=200-2QF if QF>=50

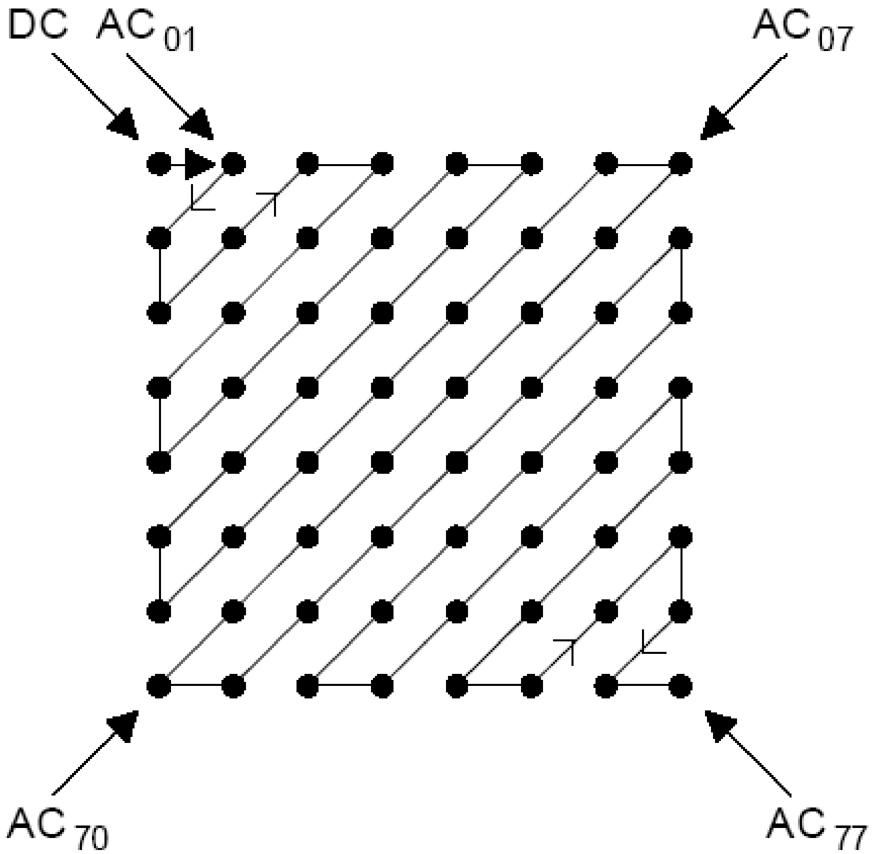
$q_{ij}= s_{ij}$ *factor/100

(Default QF=50)

DC/AC Processing

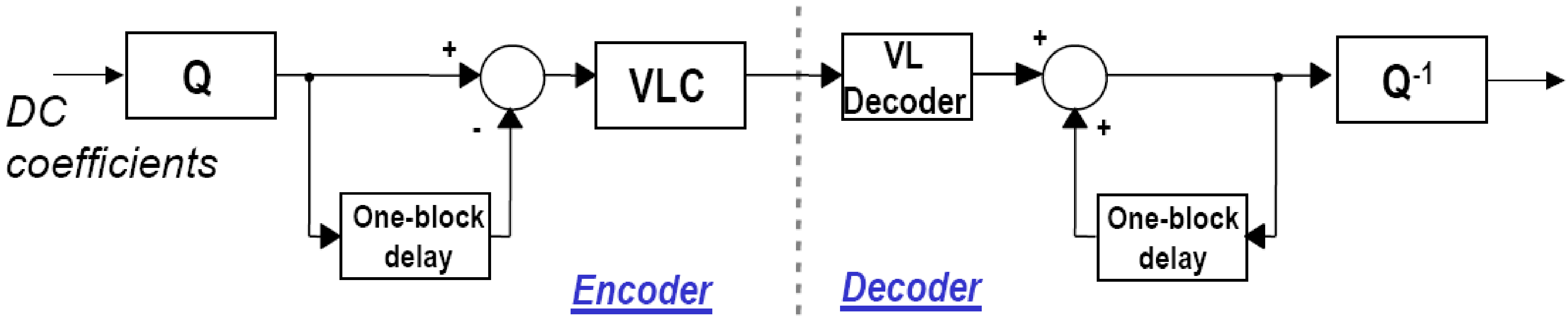


Differential DC encoding

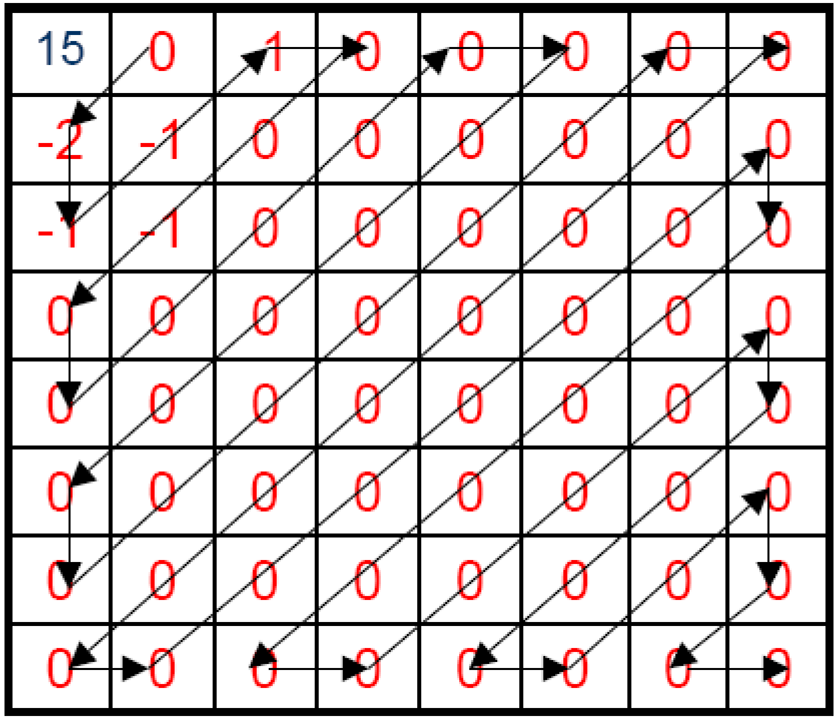


Zig-zag order

DC Differential Coding



AC Run-length Coding



15 <u>0 -2</u> <u>-1</u> <u>-1</u> <u>-1</u> <u>0 0 -1</u> 0 0 ... 0

<1,-2><0,-1><0,-1><0,-1><2,-1><EOB>

JPEG Coefficient Coding Categories

Range	DC Difference Category	AC Category
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
−31, . . . , −16, 16, . . . , 31	5	5
−63, . . . , −32, 32, . . . , 63	6	6
−127, . . . , −64, 64, . . . , 127	7	7
−255, . . . , −128, 128, . . . , 255	8	8
−511, . . . , −256, 256, . . . , 511	9	9
−1023, . . . , −512, 512, . . . , 1023	A	A
−2047, . . . , −1024, 1024, . . . , 2047	B	B
−4095, . . . , −2048, 2048, . . . , 4095	C	C
−8191, . . . , −4096, 4096, . . . , 8191	D	D
−16383, . . . , −8192, 8192, . . . , 16383	E	E
−32767, . . . , −16384, 16384, . . . , 32767	F	N/A

AC

DC

JPEG Default DC Code

	Luminance	DC	Chrominance	DC
SSSS	Code length	Codeword	Code length	Codeword
0	2	00	2	00
1	3	010	2	01
2	3	011	2	10
3	3	100	3	110
4	3	101	4	1110
5	3	110	5	11110
6	4	1110	6	111110
7	5	11110	7	1111110
8	6	111110	8	11111110
9	7	1111110	9	111111110
10	8	11111110	10	1111111110
11	9	111111110	11	11111111110

DC Differential Coding Category

SSSS	DIFF values
0	0
1	-1,1
2	-3,-2,2,3
3	-7..-4,4..7
4	-15..-8,8,15
5	-13..-16,16..31
....	....

DC Huffman Table

Category	Code length	Code word
0	2	00
1	3	010
2	3	011
3	3	100
4	3	101
5	3	110
....	....	....

DC code word =
Category code word
+ DIFF value code word

DIFF values	Code word
-1,1	0,1
-3,-2,2,3	00,01,10,11

15 -> 101 1111

VLC Table for Luminance AC Coefficients in JPEG

Run	Size	Code word	Run	Size	Code word
0	0	1010 (EOB)	3	5	1111 1111 1001 0000
0	1	00	3	6	1111 1111 1001 0001
0	2	01	3	7	1111 1111 1001 0010
0	3	100	3	8	1111 1111 1001 0011
0	4	1011	3	9	1111 1111 1001 0100
0	5	1101 0	3	A	1111 1111 1001 0101
0	6	1111 000	4	1	1110 11
0	7	1111 1000	4	2	1111 1110 00
0	8	1111 1101 10	4	3	1111 1111 1001 0110
0	9	1111 1111 1000 0010	4	4	1111 1111 1001 0111
0	A	1111 1111 1000 0011	4	5	1111 1111 1001 1000
1	1	1100	4	6	1111 1111 1001 1001
1	2	1101 1	4	7	1111 1111 1001 1010
1	3	1111 001	4	8	1111 1111 1001 1011
1	4	1111 1011 0	4	9	1111 1111 1001 1100
1	5	1111 1110 110	4	A	1111 1111 1001 1101
1	6	1111 1111 1000 0100	5	1	1111 010
1	7	1111 1111 1000 0101	5	2	1111 1110 111
1	8	1111 1111 1000 0110	5	3	1111 1111 1001 1110
1	9	1111 1111 1000 0111	5	4	1111 1111 1001 1111
1	A	1111 1111 1000 1000	5	5	1111 1111 1010 0000
2	1	1110 0	5	6	1111 1111 1010 0001
2	2	1111 1001	5	7	1111 1111 1010 0010
2	3	1111 1101 11	5	8	1111 1111 1010 0011
2	4	1111 1111 0100	5	9	1111 1111 1010 0100
2	5	1111 1111 1000 1001	5	A	1111 1111 1010 0101
2	6	1111 1111 1000 1010	6	1	1111 011
2	7	1111 1111 1000 1011	6	2	1111 1111 0110
2	8	1111 1111 1000 1100	6	3	1111 1111 1010 0110
2	9	1111 1111 1000 1101	6	4	1111 1111 1010 0111
2	A	1111 1111 1000 1110	6	5	1111 1111 1010 1000
3	1	1110 10	6	6	1111 1111 1010 1001
3	2	1111 1011 1	6	7	1111 1111 1010 1010
3	3	1111 1111 0101	6	8	1111 1111 1010 1011
3	4	1111 1111 1000 1111	6	9	1111 1111 1010 1100
6	A	1111 1111 1010 1101	A	6	1111 1111 1100 1011
7	1	1111 1010	A	7	1111 1111 1100 1100
7	2	1111 1111 0111	A	8	1111 1111 1100 1101
7	3	1111 1111 1010 1110	A	9	1111 1111 1100 1110
7	4	1111 1111 1010 1111	A	A	1111 1111 1100 1111
7	5	1111 1111 1011 0000	B	1	1111 1110 01
7	6	1111 1111 1011 0001	B	2	1111 1111 1101 0000

Run	Size	Code word	Run	Size	Code word
7	7	1111 1111 1011 0010	B	3	1111 1111 1101 0001
7	8	1111 1111 1011 0011	B	4	1111 1111 1101 0010
7	9	1111 1111 1011 0100	B	5	1111 1111 1101 0011
7	A	1111 1111 1011 0101	B	6	1111 1111 1101 0100
8	1	1111 1100 0	B	7	1111 1111 1101 0101
8	2	1111 1111 1000 000	B	8	1111 1111 1101 0110
8	3	1111 1111 1011 0110	B	9	1111 1111 1101 0111
8	4	1111 1111 1011 0111	B	A	1111 1111 1101 1000
8	5	1111 1111 1011 1000	C	1	1111 1110 10
8	6	1111 1111 1011 1001	C	2	1111 1111 1101 1001
8	7	1111 1111 1011 1010	C	3	1111 1111 1101 1010
8	8	1111 1111 1011 1011	C	4	1111 1111 1101 1011
8	9	1111 1111 1011 1100	C	5	1111 1111 1101 1100
8	A	1111 1111 1011 1101	C	6	1111 1111 1101 1101
9	1	1111 1100 1	C	7	1111 1111 1101 1110
9	2	1111 1111 1011 1110	C	8	1111 1111 1101 1111
9	3	1111 1111 1011 1111	C	9	1111 1111 1110 0000
9	4	1111 1111 1100 0000	C	A	1111 1111 1110 0001
9	5	1111 1111 1100 0001	D	1	1111 1111 000
9	6	1111 1111 1100 0010	D	2	1111 1111 1110 0010
9	7	1111 1111 1100 0011	D	3	1111 1111 1110 0011
9	8	1111 1111 1100 0100	D	4	1111 1111 1110 0100
9	9	1111 1111 1100 0101	D	5	1111 1111 1110 0101
9	A	1111 1111 1100 0110	D	6	1111 1111 1110 0110
A	1	1111 1101 0	D	7	1111 1111 1110 0111
A	2	1111 1111 1100 0111	D	8	1111 1111 1110 1000
A	3	1111 1111 1100 1000	D	9	1111 1111 1110 1001
A	4	1111 1111 1100 1001	D	A	1111 1111 1110 1010
A	5	1111 1111 1100 1010	E	1	1111 1111 1110 1011
E	2	1111 1111 1110 1100	F	1	1111 1111 1111 0101
E	3	1111 1111 1110 1101	F	2	1111 1111 1111 0110
E	4	1111 1111 1110 1110	F	3	1111 1111 1111 0111
E	5	1111 1111 1110 1111	F	4	1111 1111 1111 1000
E	6	1111 1111 1111 0000	F	5	1111 1111 1111 1001
E	7	1111 1111 1111 0001	F	6	1111 1111 1111 1010
E	8	1111 1111 1111 0010	F	7	1111 1111 1111 1011
E	9	1111 1111 1111 0011	F	8	1111 1111 1111 1100
E	A	1111 1111 1111 0100	F	9	1111 1111 1111 1101
F	0	1111 1111 001 (ZRL)	F	A	1111 1111 1111 1110

AC Huffman Encoding

AC coefficient magnitude category

SSSS	AC coefficients
1	-1,1
2	-3,-2,2,3
3	-7..-4,4..7
4	-15..-8,8,15
5	-13..-16,16..31
6	-127..-
….	64,64..127
	….

AC Huffman Table

Run/Size	Code length	Code word
0/0 (EOB)	4	1010
0/1	2	00
..	..	..
1/2	5	11011
..	..	..
2/1	5	11100
….	….	….

AC code word = Run/Size code word + AC coefficient code word

<1,-2>	<0,-1>	<0,-1>	<0,-1>	<2,-1>	<EOB>
11011 01	00 0	00 0	00 0	11100 0	1010

(Run/Category for the following nonzero coefficient + fixed length code)

AC Huffman Encoding

- Another example:

```
39 -3  1  0  0
 2 -1  0  0  0
 1  0  0  0
 0 -1 ...
 0 ...
```

=> {39 -3 2 1 -1 1 0 0 0 0 0 -1 EOB}

Assume the DC of previous block is 34, error=5

100101/0100/0110/001/000/001/11110100/1010

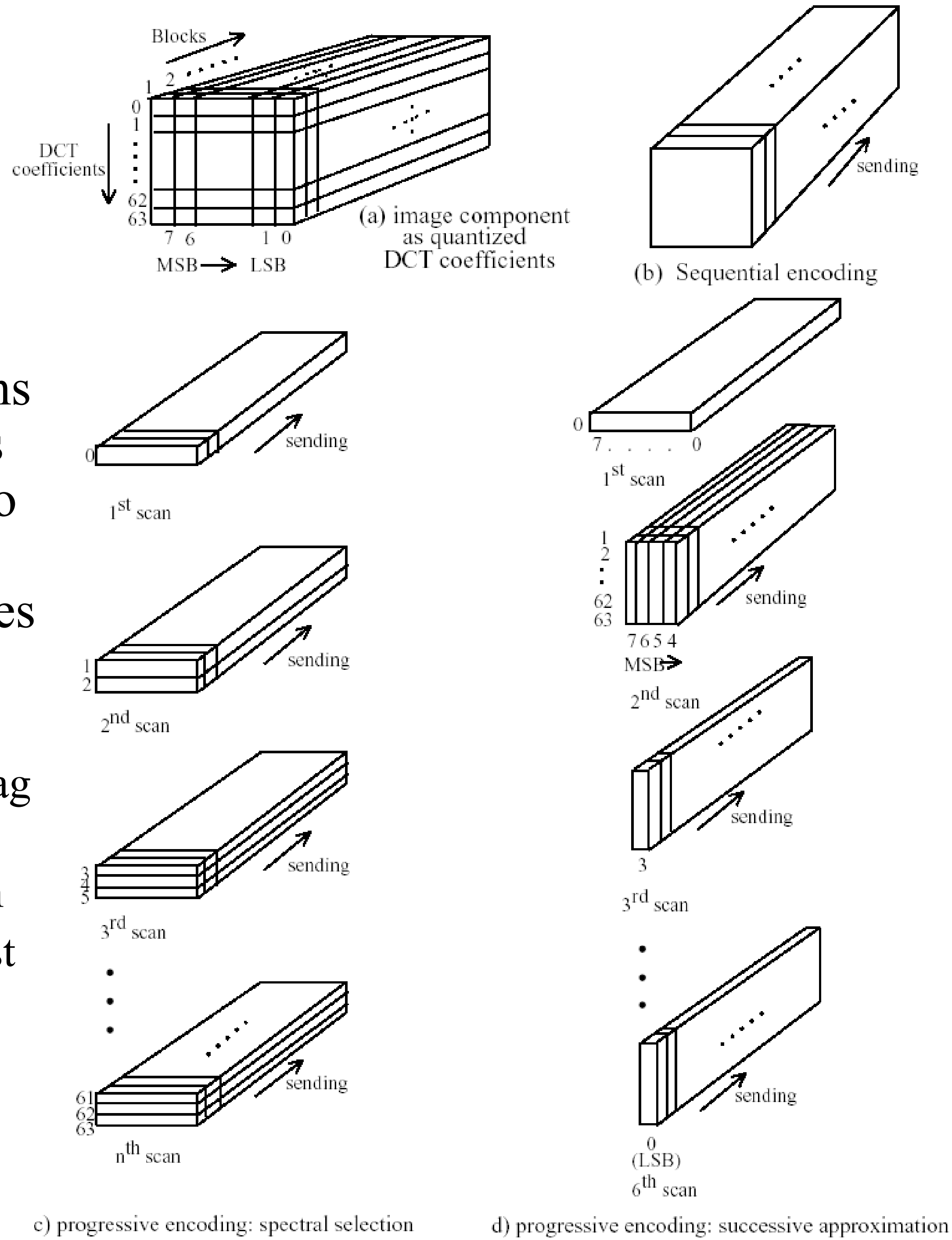
DC_DIFF (0,-3) (0,2) (0,1) (0,-1) (0,1) (5,-1) EOB

Beyond JPEG-baseline

- Huffman code tables can be optionally replaced by arithmetic coder (rarely supported)
- Hierarchical mode for progressive image transmission
- Up to 255 image components
- Lossless mode: prediction with Huffman coding of residual

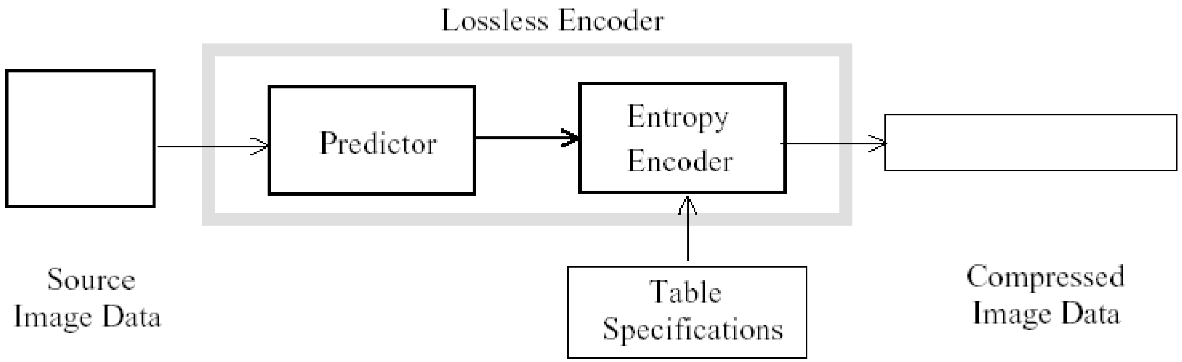
Progressive Mode

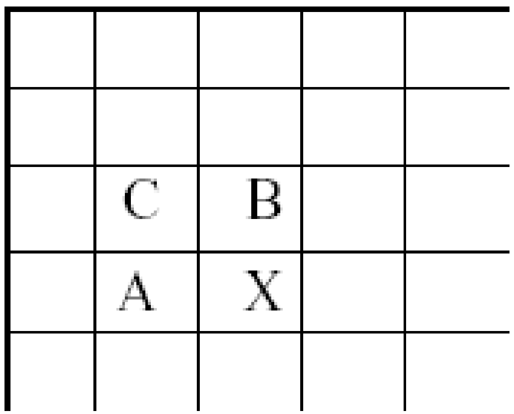
- The image is encoded in multiple scans for applications in which transmission time is long and the viewer prefers to watch the image built up in multiple coarse-to-clear passes
  - Spectral selection
    - Specify a band of coefficients from the zig-zag sequence.
  - Successively Approximation
    - A specified number of most significant bits is encoded first (Bit-plane coding)



Lossless Mode

- Perfect recovery of each pixel
- No DCT but predictive method is used
- Lossless compression ratio: 2:1
- Redundancy can be processed by Huffman or arithmetic code
- Huffman table is suggested

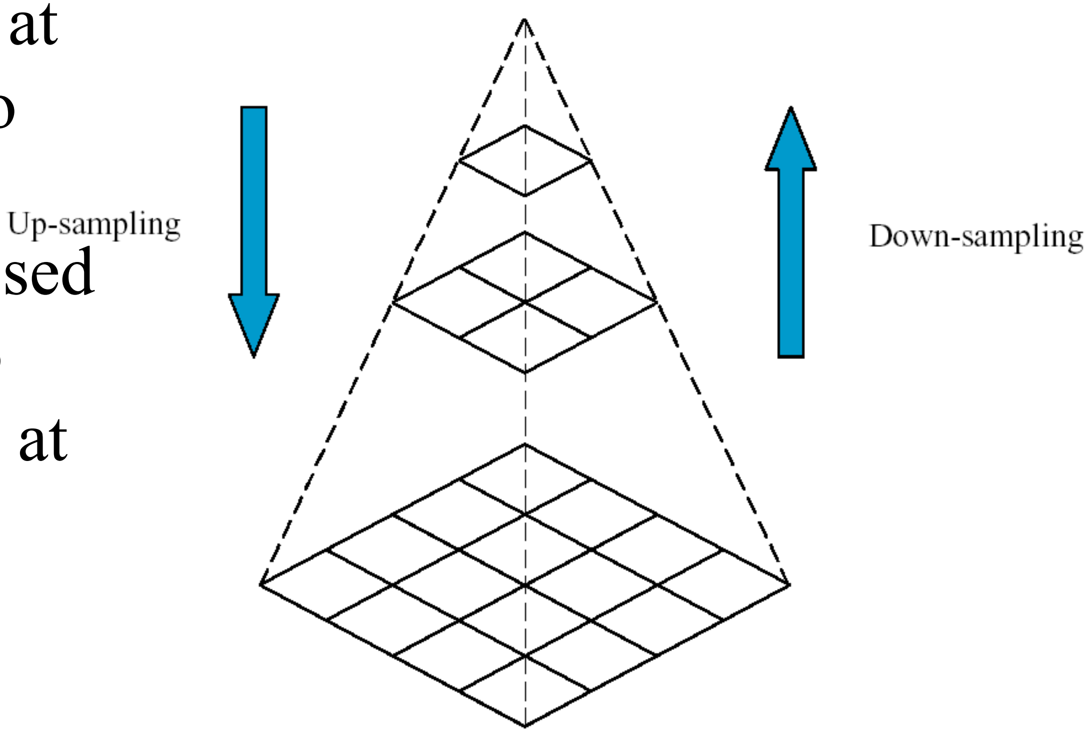




selection-value	prediction
0	no prediction
1	A
2	B
3	C
4	A+B-C
5	A+((B-C)/2)
6	B+((A-C)/2)
7	(A+B)/2

Hierarchical Mode

- The image is encoded at multiple resolutions so that lower-resolution versions may be accessed without first having to decompress the image at its full resolution



Hierarchical Mode

- Filter and down-sample the original image by the desired number of multiples of 2 in each dimension.
- Encode this reduced-size image using one of the sequential DCT, progressive DCT, or lossless encoders.
- 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.
- 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.
- Repeat until the full resolution of the image has been encoded.

Hierarchical Mode