FUNDAMENTALS OF LOSSY IMAGE COMPRESSION

- The decompression yields an imperfect reconstruction of the original image data.
- Given the level of image loss (or distortion) *D*, there is always a bound on the minimum bit rate of the compressed bit stream.
- A common measure for *D* is the ***mean square error*** between the encoded and decoded images, normalized by the variance of the input signal

Spatial Correlation Model in Images

- Compression is achieved by exploiting
  - *1. Spatial redundancies*
  - *2. Perceptual characteristics of the human visual system*
- For any image X, the spatial redundancy can be modeled by a two-dimensional covariance function:

$$Cov_X(i, j) = \sigma^2 e^{-\alpha\sqrt{i^2+j^2}}$$

  - where σ^2 is the covariance of the image and *i and j* refer to the distance from the reference pixel about which the covariance function is defined.
- Observation: the covariance function decays rapidly and is quite small beyond *i*,*j* > 8. It means that spatial redundancy need not to consider blocks of pixels larger than ***eight*** pixels

Rate-Distortion Function for Images

- **Terminology:** *Rate(R)* is the bit rate of the compressed bit stream, *Distortion (D)* is normalized by the variance of the encoder input.
- **Rate-Distortion Theory**: Establishes the theoretical minimum bit rate R_{min} so that the compressed input can be reconstructed within the allowed distortion *D*.
  - For a given *D*, the **rate-distortion function** *R(D)* is defined as the minimum possible rate necessary to achieve average distortion *D* or less.
  - R(D) is independent of the particular compression method and depends only on the underlying stochastic model for the input images and the distortion measure.
- **The source-coding theorem states:**
  - in lossy compression, it is possible to design a coding-decoding scheme of rate R > R(D) so that the average distortion is D or less,
  - if a coding-decoding system has rate R < R(D), then it is impossible to achieve average distortion D or less with this system.

Rate-Distortion Function R(D)

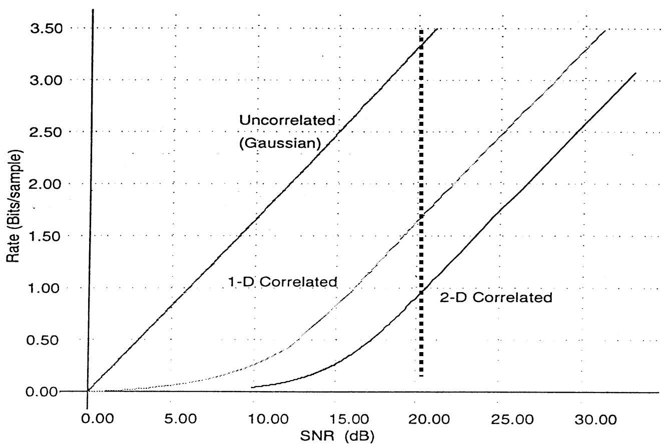


Figure 3.1 Rate-distortion performance for various image models.

Practical usage of *R(D)*

- Determine whether the exploitation of spatial redundancy would be beneficial by the correlation of input image.
- Determine the efficiency of the practical coder, by measuring D and the corresponding rate R and compare them against the theoretical limit $R(D)$.
- Use to optimize the design of functional blocks within the encoder and decoder.

Basic Coding Schemes

- Sample-Based Coding (or Pixel-Based coding)
  - the image samples are compressed on a sample-by-sample basis
  - the samples can be either in the spatial domain or temporal domain
- Block-Based Coding
- Object-Based Coding
- Model-Based Coding
- … …

Sample-Based Coding :
---DPCM Differential pulse code modulation)

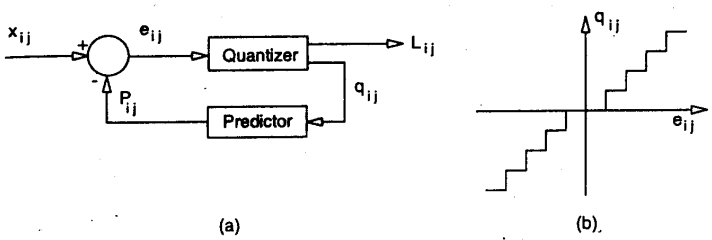


Figure 3.2 DPCM compression method. (a) Encoding block diagram; (b) input-output characteristics of a typical quantizer.

Ex. Practical DPCM systems use three previously decoded pixels near x_{ij},

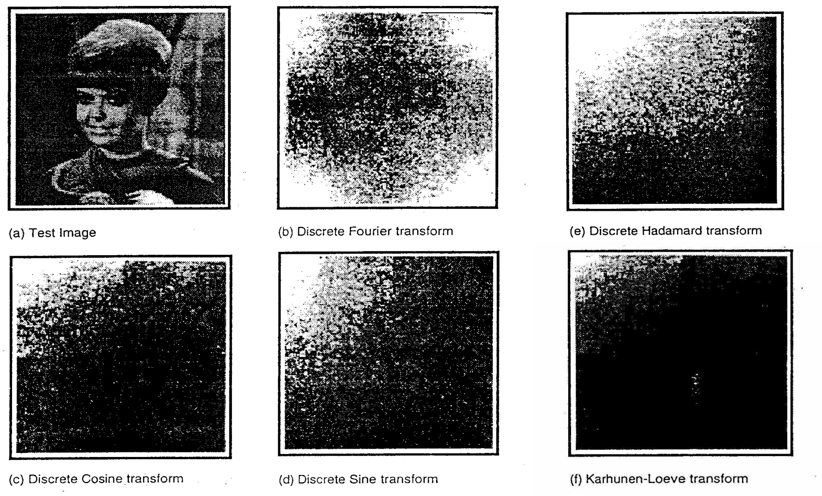
$$P_{ij} = w_1 x_{ij-1} + w_2 x_{i-1j-1} + w_3 x_{i-1j}$$

$$x_{ij} = P_{ij} + q_{ij}$$

Block-Based Coding

- **From source-coding theorem**: As the size of the coding block increases, (in other words, if a source is coded as an infinitely large block), then it is possible to find a block-coding scheme with rate *R(D)* that can achieve distortion *D*.
- Techniques:
  - Spatial-domain block coding: the pixels are grouped into blocks and the blocks are then compressed in the spatial domain, ex. VQ.
  - Transform-domain block coding: Pixels are grouped into blocks and the blocks are transformed to another domain, such as frequency domain, ex. DFT, DST, DCT,DHT,KLT.

Compaction efficiency for various image transforms



DCT-Based Coding

- Optimal transform is KLT, but
  - KLT is image dependent
  - complex computing complexity
- DCT-based coding,
  - is image independent, unlike KLT for highly correlated image data,
  - DCT compaction efficiency is close to KLT.
  - Computations of DCT can be performance with fast algorithms which can be easily implemented on parallel architectures.

Discrete Cosine Transform

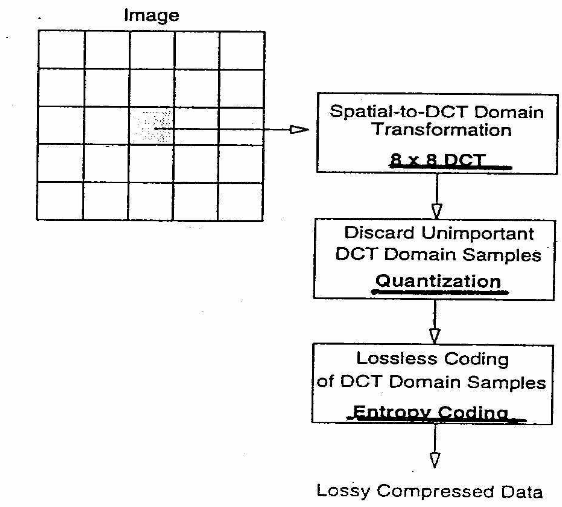
- Block size: 8 x 8
- Two-dimensional DCT:

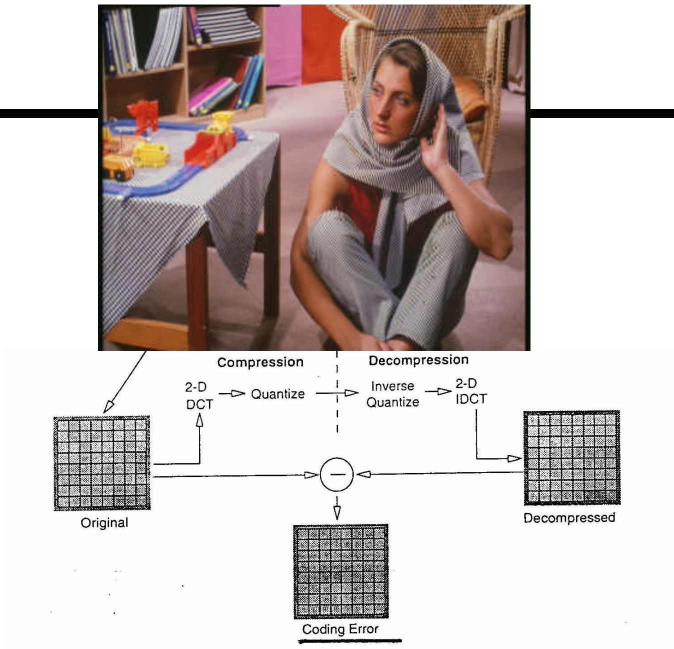
$$F(u,v) = \frac{2}{N} C(u)C(v) \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} f(x,y) \cos\frac{2p(2x+1)u}{4N} \cos\frac{2p(2y+1)v}{4N}$$

$$C(u), C(v) = \begin{cases} 1/\sqrt{2}, & u,v=0 \\ 1, & \text{otherwise} \end{cases}$$

  - Separable -- row-column method
  - Most large coefficients concentrate on the upper-left corner
  - Quantization and zig-zag scan

Generic DCT-based Image Coding System





@NTUEE
DSP/IC Lab

Original

$$X = \begin{bmatrix} 168 & 161 & 161 & 150 & 154 & 168 & 164 & 154 \\ 171 & 154 & 161 & 150 & 157 & 171 & 150 & 164 \\ 171 & 168 & 147 & 164 & 164 & 161 & 143 & 154 \\ 164 & 171 & 154 & 161 & 157 & 157 & 147 & 132 \\ 161 & 161 & 157 & 154 & 143 & 161 & 154 & 132 \\ 164 & 161 & 161 & 154 & 150 & 157 & 154 & 140 \\ 161 & 168 & 157 & 154 & 161 & 140 & 140 & 132 \\ 154 & 161 & 157 & 150 & 140 & 132 & 136 & 128 \end{bmatrix}$$

After DCT

$$Y = \begin{bmatrix} 214 & 49 & -3 & 20 & -10 & -1 & 1 & -6 \\ 34 & -25 & 11 & 13 & 5 & -3 & 15 & -6 \\ -6 & -4 & 8 & -9 & 3 & -3 & 5 & 10 \\ 8 & -10 & 4 & 4 & -15 & 10 & 6 & 6 \\ -12 & 5 & -1 & -2 & -15 & 9 & -5 & -1 \\ 5 & 9 & -8 & 3 & 4 & -7 & -14 & 2 \\ 2 & -2 & 3 & -1 & 1 & 3 & -3 & -4 \\ -1 & 1 & 0 & 2 & 3 & -2 & -4 & -2 \end{bmatrix}$$

@NTUEE
DSP/IC Lab

It is the process of quantization which leads to compression in DCT domain coding. The process of quantization of y_{kl} is expressed as

$$z_{kl} = round\left(\frac{y_{kl}}{q_{kl}}\right) = \left\lfloor \frac{y_{kl} \pm \lfloor \frac{q_{kl}}{2} \rfloor}{q_{kl}} \right\rfloor, k, l = 0, 1, ..., 7, \tag{3.28}$$

where q_{kl} denotes the kl-th element of an 8×8 quantization matrix Q. ($\lfloor x \rfloor$ denotes the largest integer smaller or equal to x.) In order to ensure that the same type of clipping is performed for either positive or negative valued y_{kl}, in (3.28), if $y_{kl} \geq 0$, then the two terms in the nominator are added; otherwise they are subtracted. For this example, if the 8×8 quantization matrix is given

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}, \tag{3.29}$$

$$Z = \begin{bmatrix} 13 & 4 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & -2 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Decompression begins with the entropy decoding of the coded bitstream. Since entropy coding is a lossless compression scheme, the decoder should be able to reconstruct an exact version of Z. Inverse quantization on Z is simply performed as

$$\hat{z}_{kl} = z_{kl} q_{kl} \; . \tag{3.30}$$

In this example, the inverse quantizer output matrix $\hat{Z}$ is

$$\hat{Z} = \begin{bmatrix} 208 & 44 & 0 & 16 & 0 & 0 & 0 & 0 \\ 36 & -24 & 14 & 19 & 0 & 0 & 0 & 0 \\ 0 & 0 & 16 & 0 & 0 & 0 & 0 & 0 \\ 14 & -17 & 0 & 0 & 0 & 0 & 0 & 0 \\ -18 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} .$$

After the inverse quantization, the decoder computes an 8×8 IDCT of $\hat{Z}$ using (3.22). The 8×8 IDCT output is given by

$$\tilde{X} = \begin{bmatrix} 171 & 160 & 149 & 149 & 158 & 166 & 166 & 162 \\ 174 & 164 & 155 & 154 & 160 & 164 & 161 & 156 \\ 171 & 164 & 157 & 156 & 158 & 158 & 151 & 145 \\ 161 & 157 & 154 & 154 & 155 & 151 & 144 & 137 \\ 156 & 155 & 155 & 156 & 156 & 152 & 145 & 140 \\ 159 & 160 & 160 & 160 & 157 & 153 & 148 & 145 \\ 161 & 161 & 160 & 156 & 150 & 144 & 141 & 139 \\ 159 & 158 & 155 & 148 & 139 & 132 & 129 & 128 \end{bmatrix} .$$

@NTUEE
DSP/IC Lab

$$X = \begin{bmatrix} 197 & 184 & 144 & 103 & 130 & 133 & 70 & 51 \\ 200 & 158 & 111 & 141 & 179 & 151 & 70 & 73 \\ 172 & 110 & 111 & 179 & 192 & 135 & 95 & 144 \\ 118 & 77 & 139 & 193 & 156 & 102 & 128 & 193 \\ 73 & 75 & 151 & 163 & 110 & 84 & 154 & 197 \\ 54 & 84 & 142 & 122 & 73 & 90 & 160 & 162 \\ 50 & 95 & 130 & 71 & 52 & 101 & 146 & 117 \\ 68 & 115 & 106 & 55 & 63 & 116 & 118 & 72 \end{bmatrix}, \quad \hat{X} = \begin{bmatrix} 198 & 182 & 153 & 136 & 145 & 145 & 95 & 32 \\ 182 & 159 & 146 & 153 & 152 & 129 & 98 & 81 \\ 153 & 124 & 135 & 174 & 159 & 105 & 104 & 150 \\ 120 & 95 & 125 & 180 & 153 & 86 & 112 & 203 \\ 88 & 84 & 120 & 159 & 130 & 81 & 121 & 211 \\ 62 & 93 & 120 & 114 & 92 & 92 & 131 & 173 \\ 45 & 112 & 123 & 64 & 52 & 110 & 139 & 114 \\ 37 & 127 & 126 & 31 & 27 & 123 & 143 & 72 \end{bmatrix} .$$

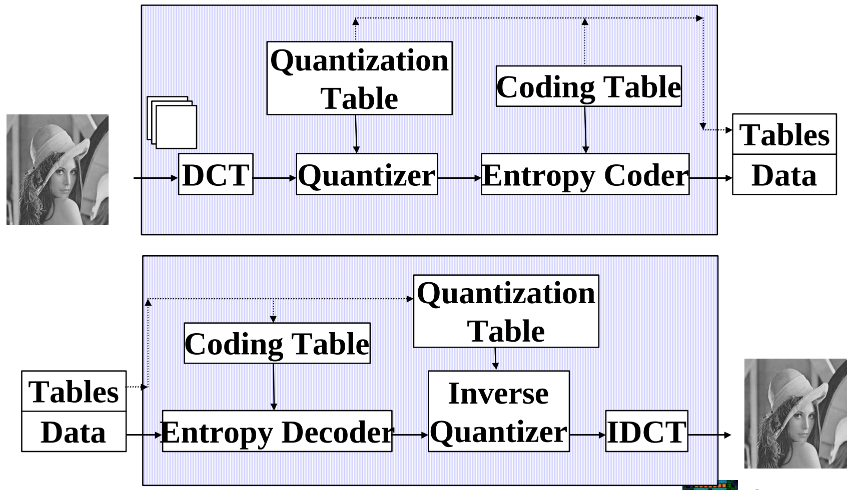
ıe corresponding DCT output is

$$Y = \begin{bmatrix} -60 & -5 & -14 & -38 & 17 & 15 & 15 & 7 \\ 127 & 139 & -40 & 103 & 102 & -41 & 12 & -13 \\ -76 & 123 & 22 & 110 & -105 & -46 & 1 & -8 \\ -20 & -5 & 29 & -53 & -54 & 18 & -1 & 11 \\ -4 & 4 & 5 & -25 & -6 & 4 & 2 & 5 \\ -3 & -10 & 9 & -19 & -5 & 8 & 7 & 6 \\ 3 & 0 & 1 & 1 & 4 & 0 & -1 & -2 \\ -1 & -4 & 5 & -4 & -2 & -2 & 1 & 4 \end{bmatrix}$$

More active area(block)

@NTUEE
DSP/IC Lab

Basic Block Diagram



An Example

- Original 8*8 block

140	144	147	140	140	155	179	179
144	152	140	147	140	148	167	179
152	155	136	167	163	162	152	172
168	145	156	160	152	155	136	160
162	148	156	148	140	136	147	162
147	167	140	155	155	140	136	162
136	156	123	167	162	144	140	147
148	155	136	155	152	147	147	136

- Shifted block(-128)

12	16	19	12	11	27	51	47
16	24	12	19	12	20	39	51
24	27	8	39	35	34	24	44
40	17	28	32	24	27	8	32
34	20	28	20	12	8	19	34
19	39	12	27	27	12	8	34
3	28	-5	39	34	16	12	19
20	27	8	27	24	19	19	8

- Block after FDCT

185	-17	14	-8	23	-9	-13	-18
20	-34	26	-9	-10	10	13	6
-10	-23	-1	6	-18	3	-20	0
-8	-5	14	-14	-8	-2	-3	8
-3	9	7	1	-11	17	18	15
3	-2	-18	8	8	-3	0	-6
8	0	-2	3	-1	-7	-1	-1
0	-7	-2	1	1	4	-6	0

- Q-Table (p/q)

3	5	7	9	11	13	15	17
5	7	9	11	13	15	17	19
7	9	11	13	15	17	19	21
9	11	13	15	17	19	21	23
11	13	15	17	19	21	23	25
13	15	17	19	21	23	25	27
15	17	19	21	23	25	27	29
17	19	21	23	25	27	29	31

An Example(Con't)

- Block after quantization

61	-3	2	0	2	0	0	-1
4	-4	2	0	0	0	0	0
-1	-2	0	0	-1	0	-1	0
0	0	1	0	0	0	0	0
0	0	0	0	0	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

- Zig-zag sequence
  - 61,-3,4,-1,-4,2,0,2,-2, 0,0,0,0,0,2,0,0,0,1,0,0, 0,0,0,0,-1,0,0,-1,0,0,0, 0,-1,0,0,0,0,0,0,0,-1,0, 0,0,0,0,0,0,0,0,0,0,0,0, 0,0,0,0,0,0,0,0,0

An Example (Con't)

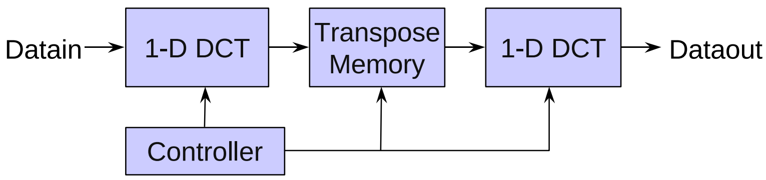
- Intermediate symbol sequence
  - (6)(61),(0,2)(-3), (0,3)(4),(0,1)(-1), (0,3)(-4),(0,2)(2), (1,2)(2),(0,2)(-2), (0,2)(-2),(5,2)(2), (3,1)(1),(6,1)(-1), (2,1)(-1),(4,1)(-1), (7,1)(-1),(0,0)
- Encoded bit sequence (total 98 bits)
  - (110)(111101)(01)(00) (100)(100)(00)(0)(100) (001)(01)(10)(11011)(10)(01)(01)(01)(01)(11 111110111)(10)(11101 0)(1)(1111011)(0)(111 00)(0)(111011)(0)(111 11010)(0)(1010)

FAST ALGORITHMS FOR THE DCT

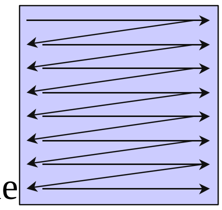
- An y_{kl} required 64 multiplications and 64 additions.
- 4096 multiply accumulate operations are needed for each 8x8 block.
- Use the row-column decomposition, only 16 1-D DCTs (8 for row and 8 for column) is needed (total 1024 multiply-accumulate operations)

1-D DCT/IDCT Approach

- Row-Column Method



- Input format
  - one word per clock cycle



Implementation

Example : row-column decomposition (1)

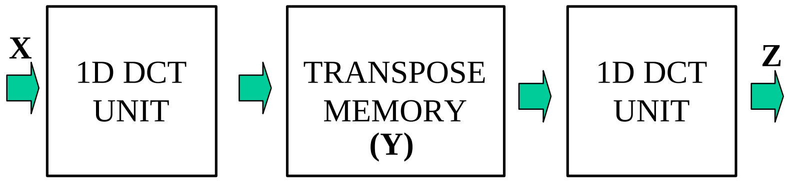
Separable,
row-column decomposition

$$a(k,n)=\sqrt{\frac{2}{N}}\sum_{n=0}^{N-1}c(k)\cos[\frac{2\pi(2n+1)k_1}{4N}]$$

$$k,n=0,1,\ldots,N-1$$

$$Z = AXA^T$$

$$c(0)=\sqrt{\frac{1}{2}} \quad \text{and} \quad c(k)=1 \quad \text{for} \quad k\neq 0$$



Y=AX ***Z=YA***

@NTUEE DSP/IC Lab

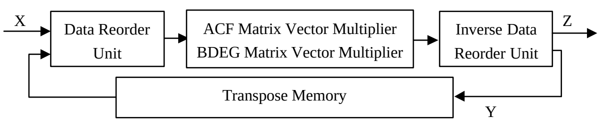
Implementation

Example : row-column decomposition (2)

$$A=\begin{bmatrix} a & a & a & a & a & a & a & a\\ b & d & e & g & -g & -e & -d & -b\\ c & f & -f & -c & -c & -f & f & c\\ d & -g & -b & -e & e & b & g & -d\\ a & -a & -a & a & a & -a & -a & a\\ e & -b & g & d & -d & -g & b & -e\\ f & -c & c & -f & -f & c & -c & f\\ g & -e & d & -b & b & -d & e & -g \end{bmatrix}, \quad \begin{bmatrix} a\\ b\\ c\\ d\\ e\\ f\\ g \end{bmatrix}=\sqrt{\frac{2}{N}}\begin{bmatrix} \cos\frac{\pi}{4}\\ \cos\frac{\pi}{16}\\ \cos\frac{\pi}{8}\\ \cos\frac{3\pi}{16}\\ \cos\frac{5\pi}{16}\\ \cos\frac{3\pi}{8}\\ \cos\frac{7\pi}{16} \end{bmatrix}$$

$$\begin{bmatrix} Y(0)\\ Y(2)\\ Y(4)\\ Y(6) \end{bmatrix}=\begin{bmatrix} a & a & a & a\\ c & f & -f & -c\\ a & -a & -a & a\\ f & -c & c & -f \end{bmatrix}\begin{bmatrix} X(0)+X(7)\\ X(1)+X(6)\\ X(2)+X(5)\\ X(3)+X(4) \end{bmatrix}$$

$$\begin{bmatrix} Y(1)\\ Y(3)\\ Y(5)\\ \vdots \end{bmatrix}=\begin{bmatrix} b & d & e & g\\ d & -g & -b & -e\\ e & -b & g & d\\ \vdots \end{bmatrix}\begin{bmatrix} X(0)-X(7)\\ X(1)-X(6)\\ X(2)-X(5)\\ \vdots \end{bmatrix}$$



@NTUEE DSP/IC Lab

Rate-Distortion Performance of the DCT

- For the same SNR, the resulting bit rate is 2.37 bits per sample lower than the rate of uncorrelated source.
- At very low distortions, a DPCM system is 0.4 bits per sample worse than the DCT system.