Lecture 5
Compression Basics

- Information theory for source coding
- Lossy vs lossless coding
- Binary encoding
- Quantization
- Transform coding

What makes compression possible?

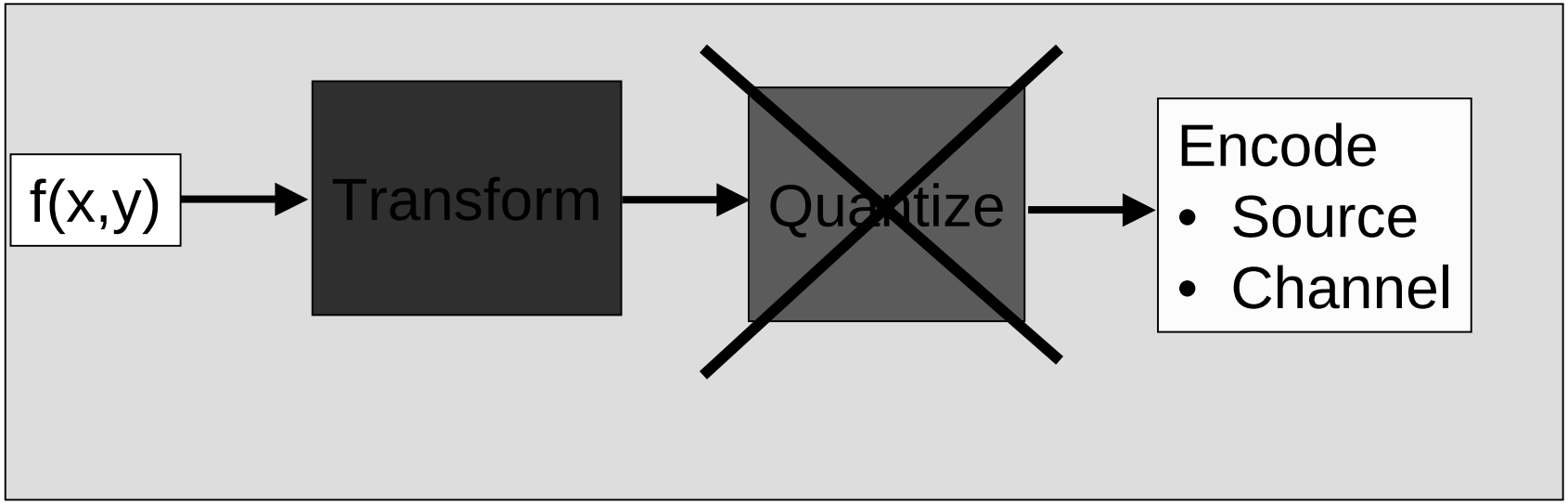
- Image signals contain a high degree of
  - *spatial redundancy* (correlation)
  - *psychovisual redundancy*

- The higher the redundancy, the higher the achievable compression.

How does compression work?

- Exploit statistical redundancy
  - Take advantage of patterns in the signal
  - Describe frequently occuring events effectively
- Introduce acceptable deviations
  - Omit ‘irrelevant’ details that humans cannot percieve
  - Match the signal resolution to the application

Compression model



Lossless compression

- Minimize number of bits required to represent original digital image samples w/o any loss of information
- All B bits of each sample must be reconstructed perfectly
- Achievable compression usually rather limited
- Applications: Binary images, medical images, master copy before editing

Lossy compression

- Some deviation of decompressed image from original is acceptable
  - HVS might not percieve loss
  - Digital input is inperfect representation of real-world scene
- Much higher compression than with lossless

Typical bit-rates after compression

- Dependent on image content
- Natural images
  - Lossless: 5 bits/pixel
  - Lossy: High quality: 1 bpp

    Moderate q : 0.5 bpp

    Usable q : 0.25 bpp

Distortion measure

Most commonly employed: Mean Squared Error

$$\mathrm{MSE} \triangleq \frac{1}{N_1 N_2} \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} \left(x[n_1, n_2] - \hat{x}[n_1, n_2] \right)^2$$

. . . or, equivalently, Peak Signal to Noise Ratio

$$\mathrm{PSNR} \triangleq 10 \log_{10} \frac{\left(2^B - 1\right)^2}{\mathrm{MSE}} \ \mathrm{dB}$$

Advantages

- Easy calculation
- Mathematical tractability in optimization problems

Disadvantage

- Neglects properties of human vision

Measures of compression

Image represented by "bit-stream" $\mathbf{c}$ of length $\|\mathbf{c}\|$.
Compare no. of bits w/ and w/o compression

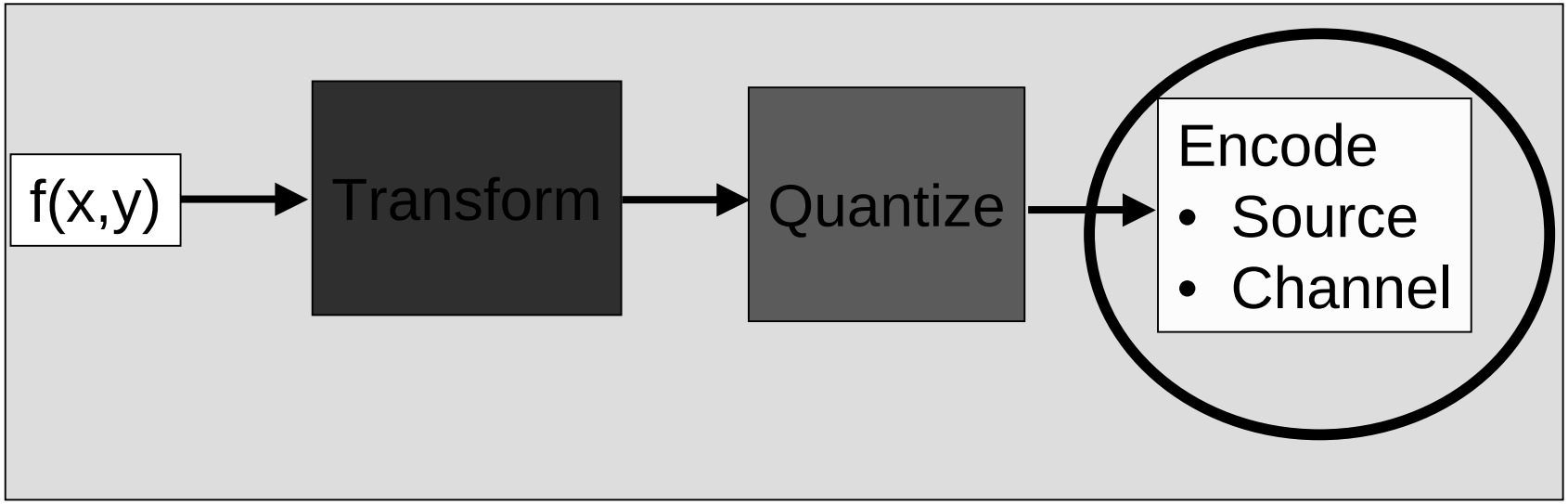
$$\text{compression ratio} \triangleq \frac{N_1 N_2 B}{\|\mathbf{c}\|}$$

Alternatively

$$\text{bit-rate} \triangleq \frac{\|\mathbf{c}\|}{N_1 N_2} \text{ bits/pixel}$$

For lossy compression, bit-rate more meaningful than compression ratio, as B is somewhat arbitrary.

Compression model



Statistical redundancy

- Trivial example: Given two B-bit integers (e.g., representing two adjacent pixels)

$$x_1, x_2 \in \left\{0,1,\ldots,2^B - 1\right\}$$

- Assume that x_1, x_2 only takes on values $\{0,1\}$
  - Compression to 1 bpp
- Further assume, that $x_1 = x_2$
  - Compression to 0.5 bpp
- Hope: bit-rate increases only slightly, as long as the above assumptions hold with high probability

Entropy

- Statistical redundancy is formalized by the concept of entropy of Shanon's information thory

The *entropy H* of a source ***U*** is the average information per symbol; that is, the average code length per symbol, given by

$$H(\mathbf{U}) = \sum_s p(s) \log_2 \frac{1}{p(s)} = -\sum_s p(s) \log_2 p(s)$$

where *s* stands for the symbols in the alphabet.

Lossless Coding Theorem [Shannon, 1948]
The performance of a VLC can be measured by

$$\min R = H(\boldsymbol{U}) + \varepsilon \text{ bits/symbol}$$

where *R* is the average rate, *H(**U**)* is the entropy of the source, and ε is a positive number that is arbitrarily close to zero.

Symbol coding

- Fixed-Length Coding (Fixed length codes to Fixed-length blocks)
- Variable-Length (Entropy) Coding
  - Huffman coding (Variable length codes to Fixed-length blocks)
  - Arithmetic coding (Variable length codes to Variable-length blocks)

If some symbols are more likely than others, then employ VLC.

Fixed length code

Symbol	*Code*
a_1	00
a_2	01
a_3	10
a_4	11

Average codeword length is 2; entropy of the source is 2.

- Fixed-length coding is optimal (the entropy of the source equal to the fixed codeword length) only if:
  1) the number of symbols is equal to a power of 2, and
  2) all the symbols are equiprobable.

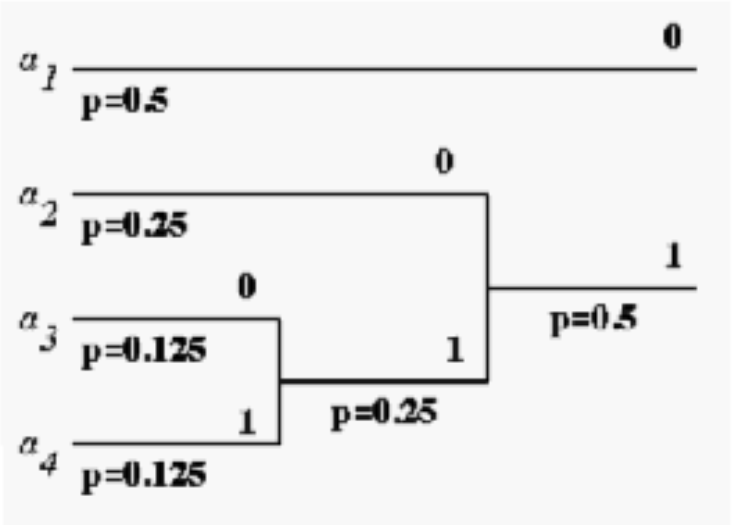
Huffman Coding

2-Step Process

- Reduction
  - List symbols in descending order of probability
  - Reduce the two least probable symbols into one symbol equal to their combined probability
  - Reorder in descending order of probability at each stage
  - Repeat until only two symbols remain
- Splitting
  - Assign 0 and 1 to the final two symbols remaining and work backwards
  - Expand code at each split by appending a 0 or 1 to each code word

Huffman Coding - Example1

Symbol	*Probability*	*Information*
a_1	0.50	1 bit
a_2	0.25	2 bits
a_3	0.125	3 bits
a_4	0.125	3 bits



The average codeword length is

$R = 0.5 \times 1 + 0.25 \times 2 + (0.125 \times 3) \times 2 = 1.75$

The entropy of the source is

$H = -0.5 \log_2 0.5 - 0.25 \log_2 0.25 - (0.125 \log_2 0.125) \times 2 = 1.75$

- Huffman coding achieves the entropy of the source of when the symbol probabilities are powers of 2.

Huffman Coding - Example2

Symbol	*Probability*	*Information*
a_1	0.40	1.32 bits
a_2	0.25	2 bits
a_3	0.15	2.73 bits
a_4	0.15	2.73 bits
a_5	0.05	4.32 bits

Average codeword length is 2.15; entropy of the source is 2.07.

Original Source		*Reduced Source 1*		*Reduced Source 2*		*Reduced Source 3*	
Prob.	Code	Prob.	Code	Prob.	Code	Prob.	Code
0.40	1	0.40	1	0.40	1	0.60	**0**
0.25	01	0.25	01	0.35	0**0**	0.40	**1**
0.15	001	0.20	00**0**	0.25	0**1**		
0.15	000**0**	0.15	00**1**				
0.05	000**1**						

Decoding of Huffman codes

- Huffman codes are uniquely decodable, with proper synchronization, because no codeword is a prefix of another.

Example: A received binary string

001101101110000…

can be decoded uniquely as

$a_3\ a_1\ a_2\ a_1\ a_2\ a_1\ a_1\ a_4$ …

Block Huffman coding

- Huffman codes can also be assigned to blocks of L symbols from the original alphabet at a time.
- Requires building a new block alphabet with all possible combinations of the L symbols from the original alphabet and computing their probabilities.
- Coding efficiency improves as L gets larger, although the design of Huffman codes gets more complicated.

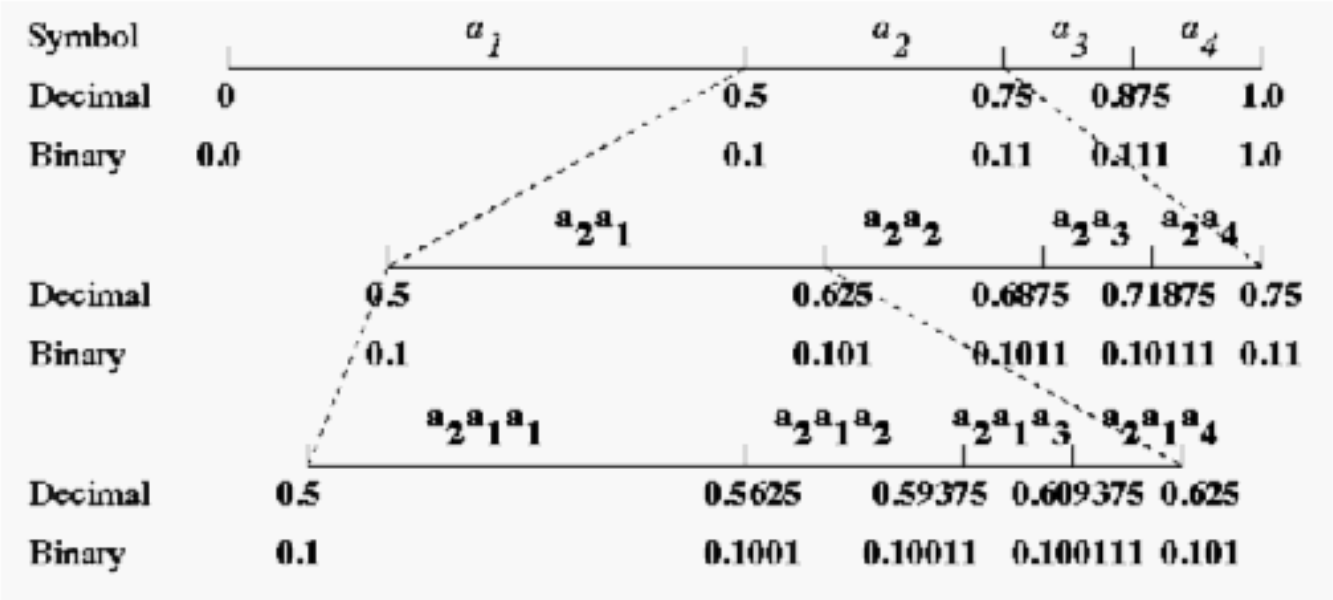
Arithmetic coding

- There is no one-to-one correspondence between the symbols of an alphabet ***A*** and the codewords.
- A sequence of symbols $\mathbf{x}= \{x_1,..,x_N\}$ is associated with a subinterval of [0,1) whose length equals the probability p(**x**) of the sequence.
- The encoder processes the input symbols one by one. Bits are sequentially sent to the channel starting from the most significant bit towards the least significant bit as they are determined.
- The final transmitted bitstream is a uniquely decodable codeword representing the source, which is a binary number pointing to the subinterval associated with this sequence.
- Because integer-length codes are not assigned to fixed-length blocks of symbols, arithmetic coding can achieve the lower bound established by the noiseless coding theorem as $N \rightarrow \infty$

Arithmetic coding - Example

Symbol	*Probability*	*Information*
a_1	0.50	1 bit
a_2	0.25	2 bits
a_3	0.125	3 bits
a_4	0.125	3 bits

We wish to determine an arithmetic code to represent a sequence of symbols, $a_2\ a_1\ a_3\ \ldots$

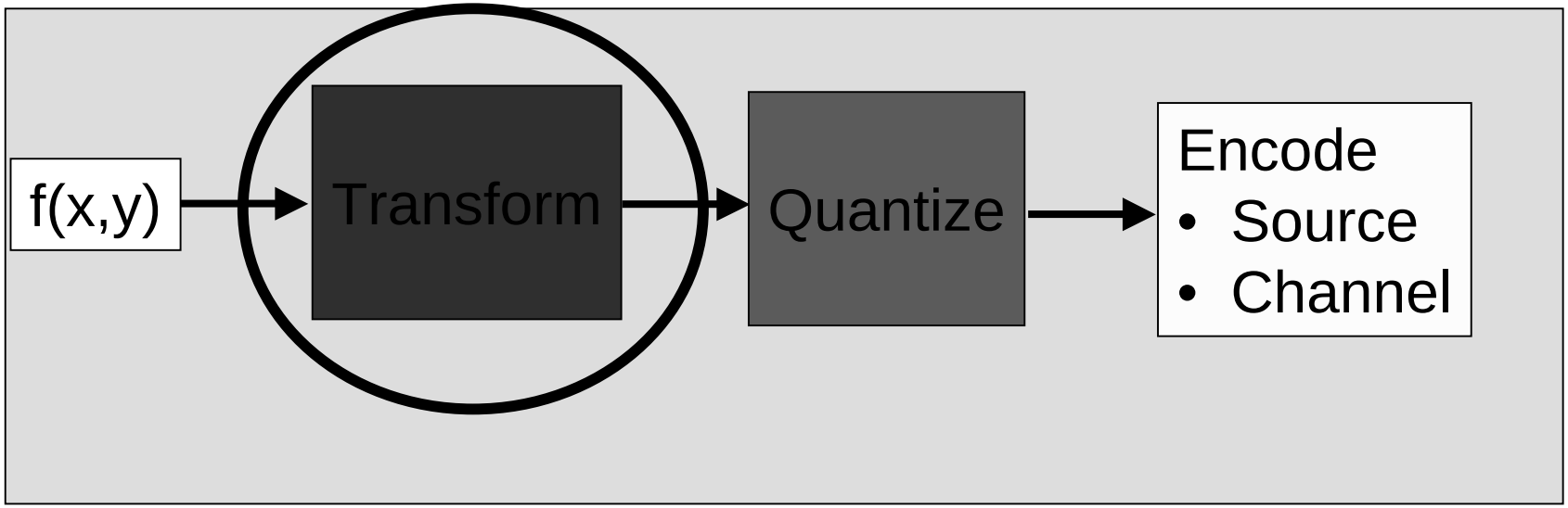


Arithmetic coding - Decoder operation

Received Bit	*Interval*	*Symbol*
1	[0.5,1)	
0	[0.5,0.75)	a2
0	[0.5,0.625)	a1
1	[0.5625,0.625)	
1	[0.59375,0.625)	
.	.	.

The first bit restricts the interval to [0.5,1). However, there are three symbols within this range; thus, the first bit does not contain sufficient information. After receiving the second bit, we have "10" which points to the interval [0.5,0.75). All possible combinations of two symbols pointing to this range start with a_2. Hence, we can now decode the first symbol as a_2.

Compression model



Basis vectors and images

- A basis for a vector space ~ a set of vectors
  - Linearly independent ~ $\Sigma a_i \underline{v}_i = \underline{0}$ if and only if all $a_i = 0$
  - Uniquely represent every vector in the space by their linear combination ~ $\Sigma b_i \underline{v}_i$ ("*spanning set*" $\{\underline{v}_i\}$)
- Orthonormal basis
  - Orthogonality ~ inner product $\langle \underline{x}, \underline{y} \rangle = \underline{y}^{*T} \underline{x} = 0$
  - Normalized length ~ $\| \underline{x} \|^2 = \langle \underline{x}, \underline{x} \rangle = \underline{x}^{*T} \underline{x} = 1$
- 2-D inner product
  - $\langle F, G \rangle = \Sigma_m \Sigma_n f(m,n)\, g^*(m,n) = \underline{G_I}^{*T} \underline{F_I}$ (rewrite matrix into vector)
    - *!! Don't do* FG *~ may not even be a valid operation for MxN matrices!*
- 2D Basis Matrices (Basis Images)
  - Represent any images of the same size as a linear combination of basis images

Transformations

- DFT
- DCT
- KL
- Haar

Why DCT?

- Orthogonal, separable transform. Fast algorithms exist.
- Near-optimum energy compaction property with image-independent basis functions.
  - The performance of the DCT approaches that of the KLT for stationary 1st order Markov sources with a high correlation coefficient.
- Availability of special purpose hardware.

1-D Discrete Cosine Transform

$$\begin{cases} Z(k) = \sum_{n=0}^{N-1} z(n) \cdot \alpha(k) \cos\left[\dfrac{\pi(2n+1)k}{2N}\right] \\ z(n) = \sum_{k=0}^{N-1} Z(k) \cdot \alpha(k) \cos\left[\dfrac{\pi(2n+1)k}{2N}\right] \end{cases}$$

$$\alpha(0) = \frac{1}{\sqrt{N}}, \alpha(k) = \sqrt{\frac{2}{N}}$$

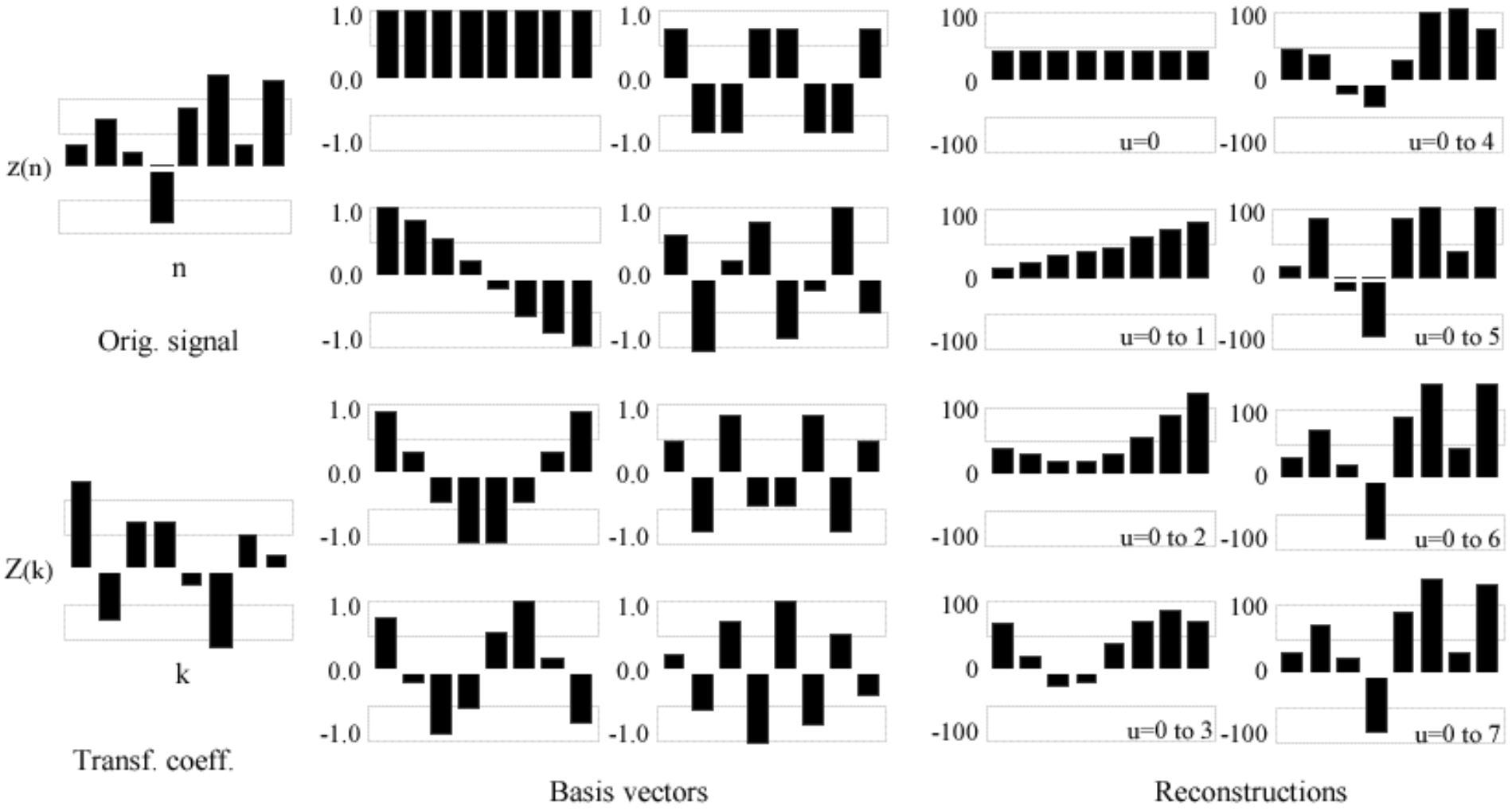
Transform matrix *C*

- $c(k,n) = \alpha(0)$ for k=0
- $c(k,n) = \alpha(k)\ cos[\pi(2n+1)/2N]$ for k>0

C is real and orthogonal

- rows of *C* form orthonormal basis
- *C* is not symmetric!
- DCT is <u>not</u> the real part of unitary DFT!
  - *related to DFT of a symmetrically extended signal*

DCT - Example


z(n)
n
Orig. signal
Z(k)
k
Transf. coeff.
1.0
0.0
-1.0
Basis vectors
100
0
-100
u=0
u=0 to 1
u=0 to 2
u=0 to 3
u=0 to 4
u=0 to 5
u=0 to 6
u=0 to 7
Reconstructions

2-D DCT (1/4)

Separable orthogonal transform

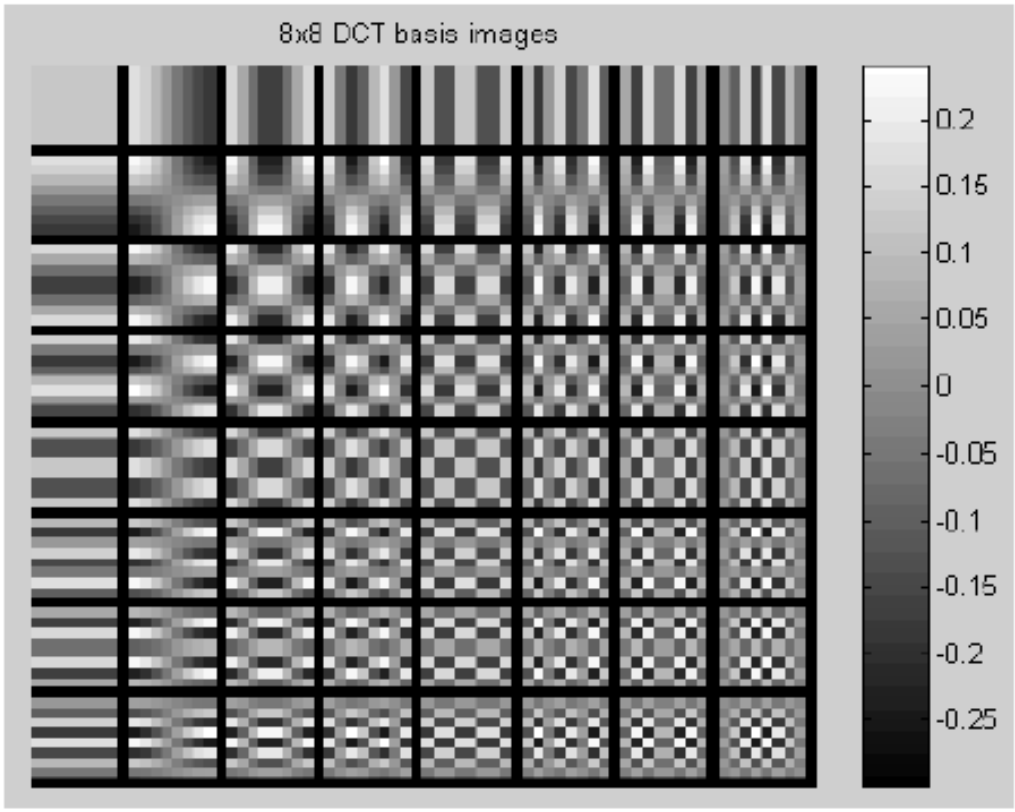
$Y = C\,X\,C^T \Leftrightarrow X = C^T\,Y\,C$

$$S(k_1,k_2) = \sqrt{\frac{4}{N^2}} C(k_1) C(k_2) \sum_{n_1=0}^{N-1} \sum_{n_2=0}^{N-1} s(n_1,n_2) \cos\left(\frac{\pi(2n_1+1)k_1}{2N}\right) \cos\left(\frac{\pi(2n_2+1)k_2}{2N}\right)$$

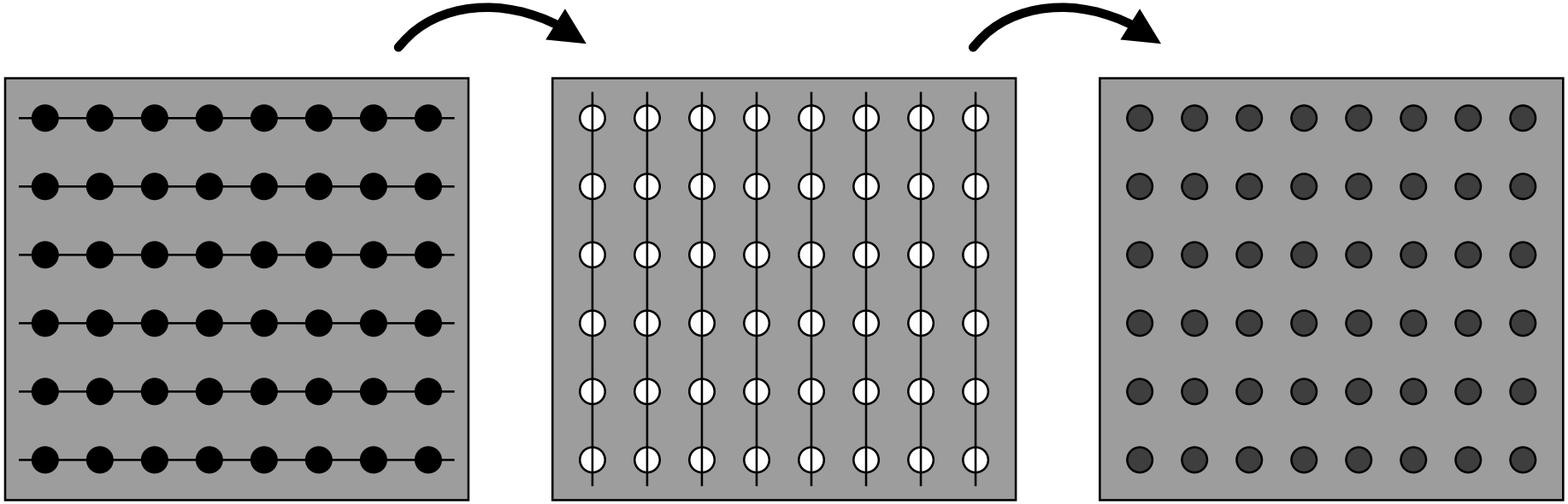
where

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

DCT basis images



2-D DCT (2/4)

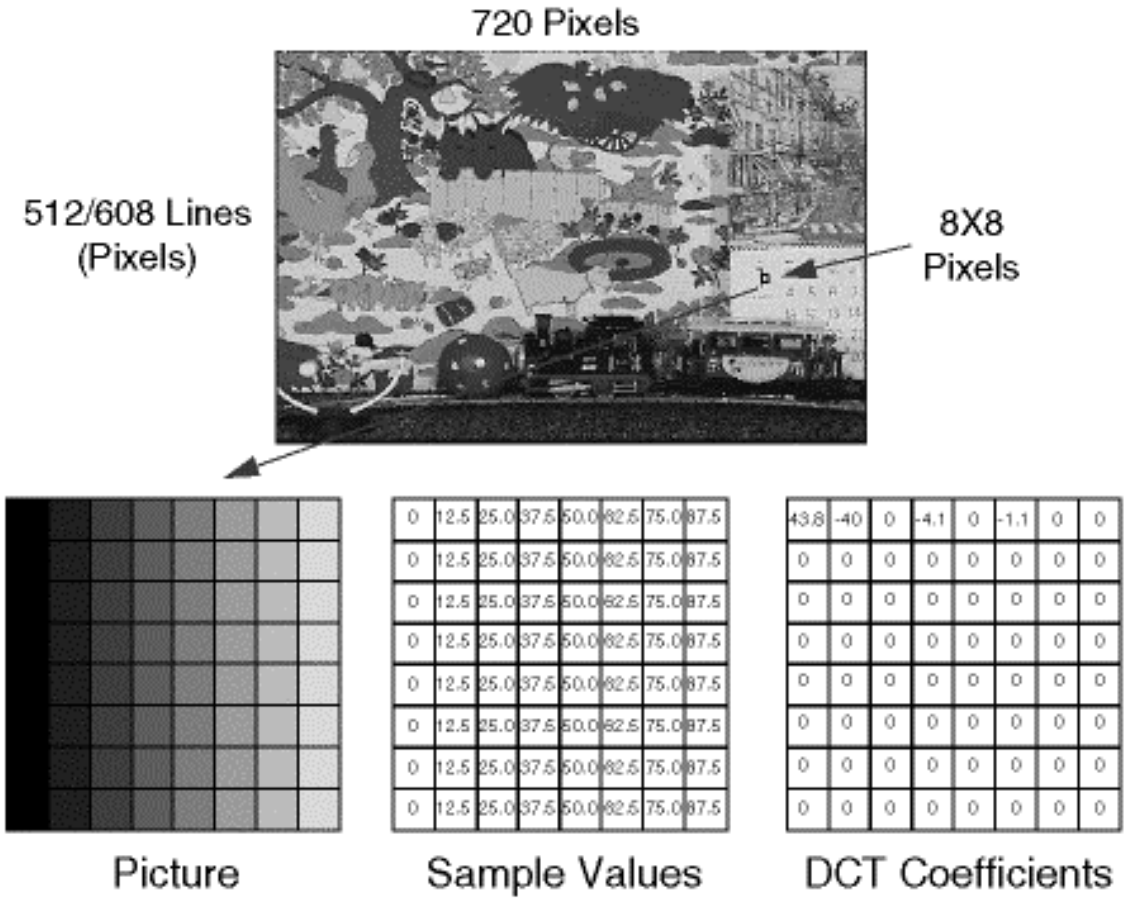


2-D DCT (3/4)

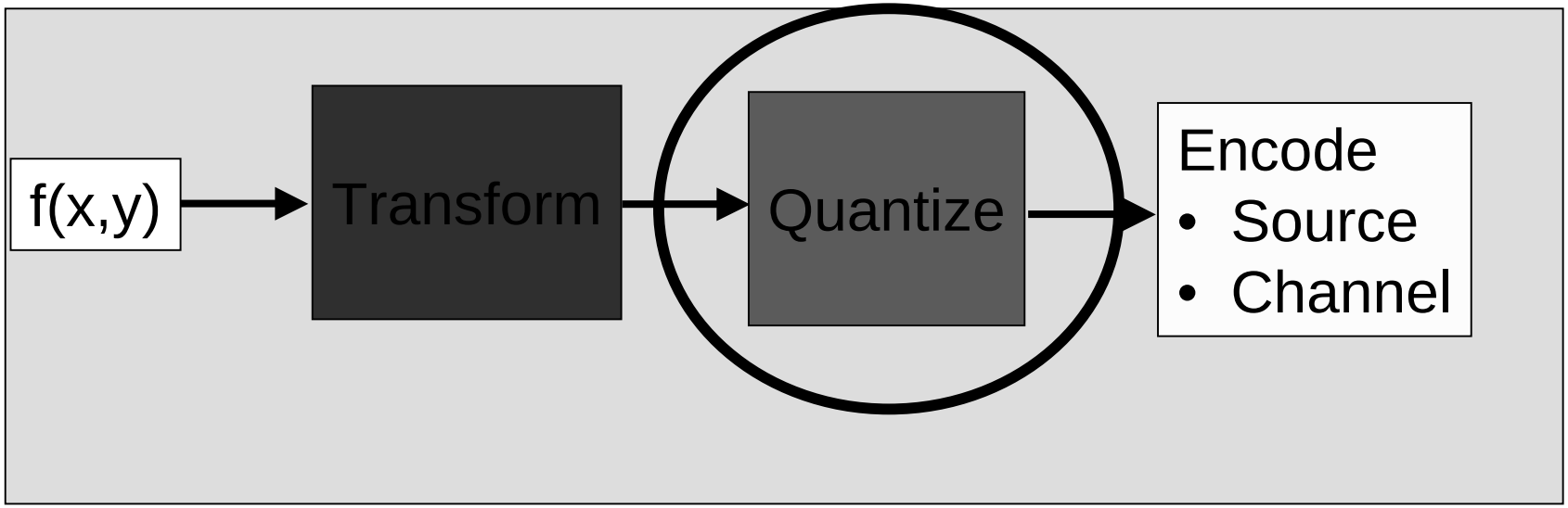
139	144	149	153	155	155	155	155
144	151	153	156	159	156	156	156
150	155	160	163	158	156	156	156
159	160	162	160	160	159	159	159
159	160	161	162	162	155	155	155
161	161	161	161	160	157	157	157
162	162	161	163	162	157	157	157
162	162	161	162	163	158	158	158

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

2-D DCT (4/4)


720 Pixels
512/608 Lines
(Pixels)
8X8
Pixels
Picture
Sample Values
DCT Coefficients

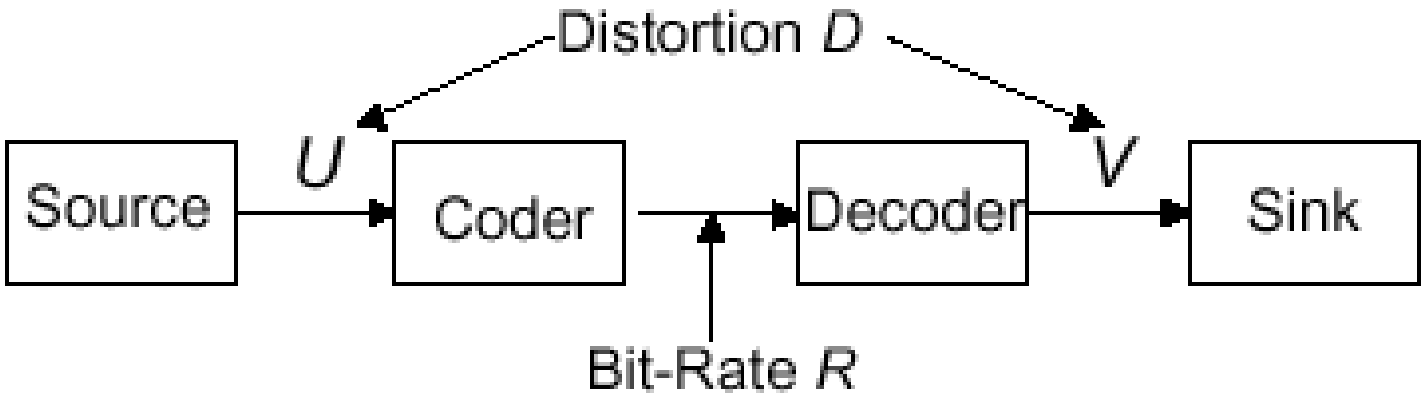
Compression model



Rate distortion theory

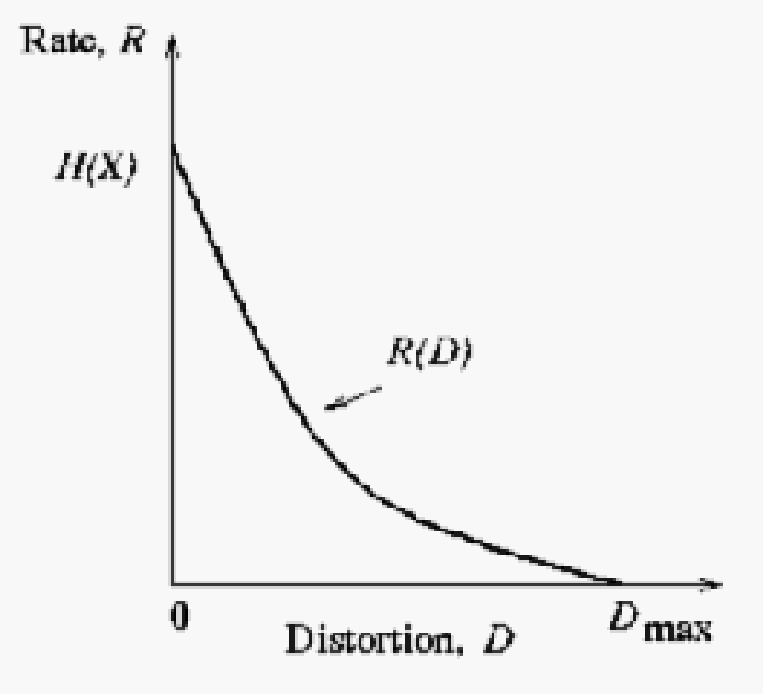
•Theoretical discipline treating data compression from the viewpoint of information theory.

•RD theory calculates minimum transmission bitrate for a given distortion D.



Rate distortion function

For a given distortion D,
$R(D)$ bits/symbol are sufficient to enable source reconstruction with an average distortion that is arbitrarily close to D.
The actual rate R satisfies $R \geq R(D)$,
where $R(0) = H(X)$.

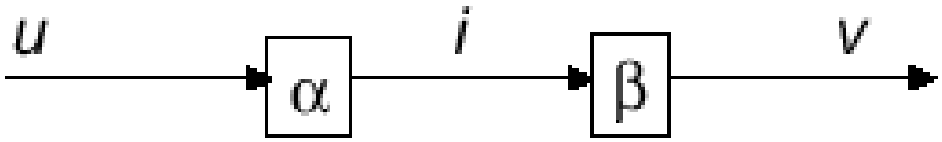


The rate distortion function can be computed analytically for simple source and distortion models. Computer algorithms exist to compute R(D) when analytical methods are impractical.

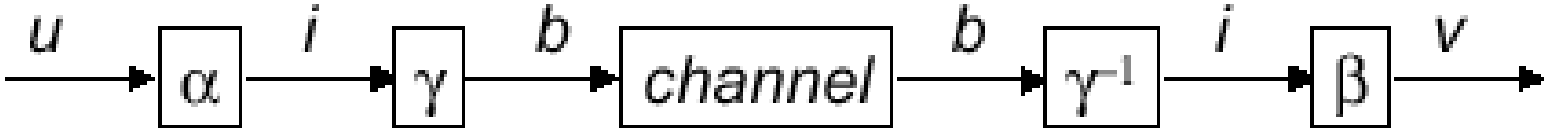
Quantization

- Structure

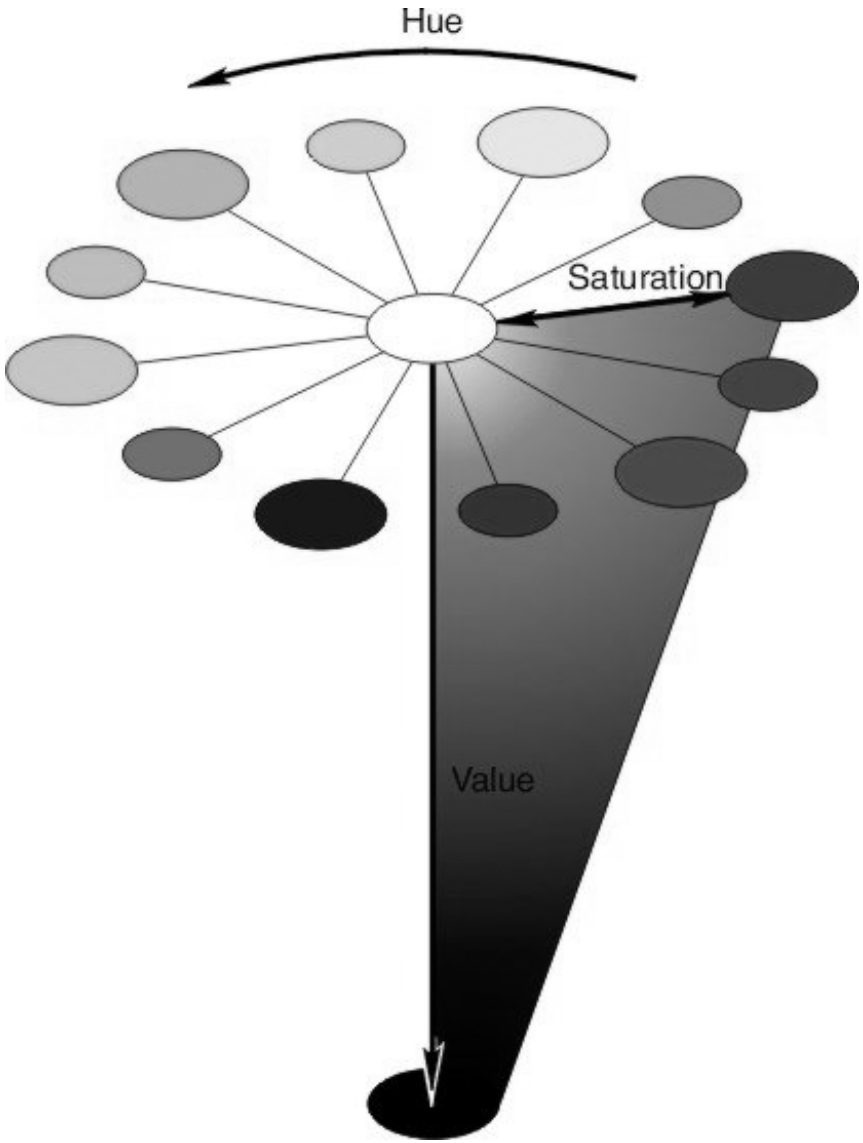
- Alternative: coder (α) / decoder (β) structure



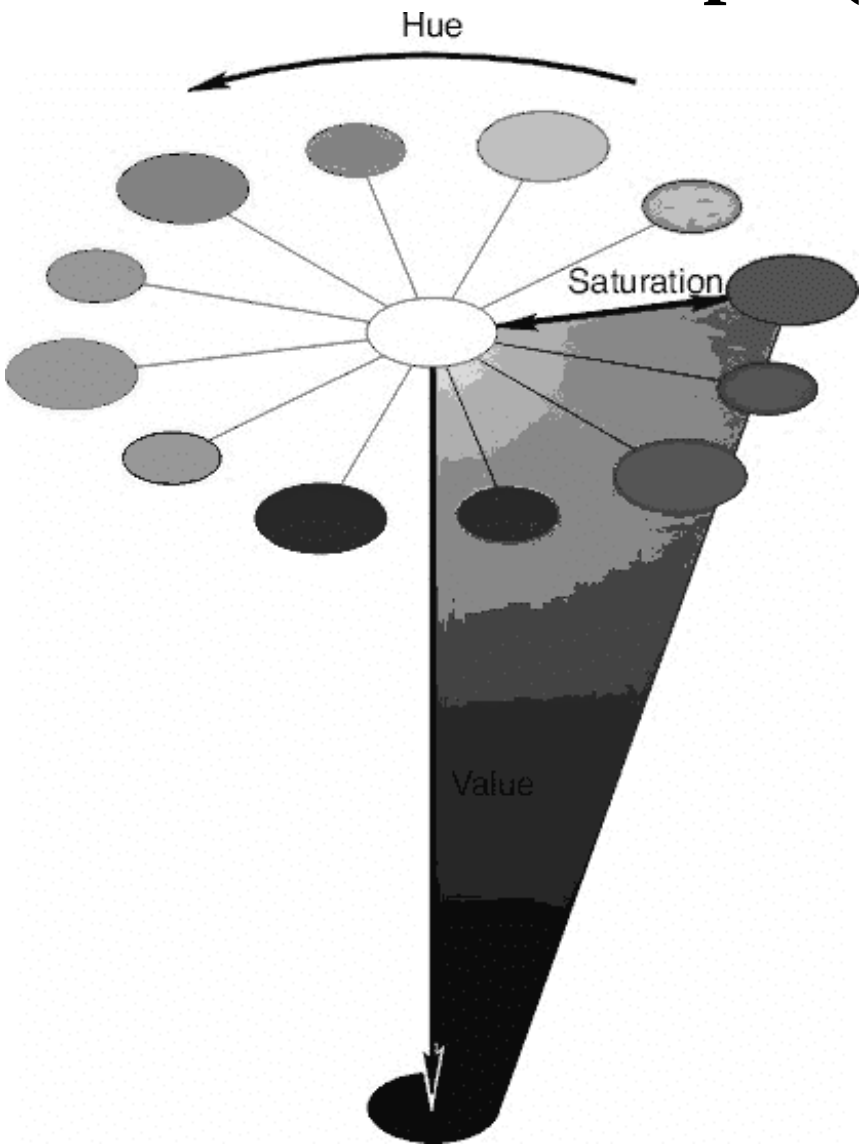
- Insert entropy coding (γ) and transmission channel



Quantization - Example (1/2)



Quantization - Example (2/2)



Quantization techniques

- Scalar
  - Uniform
  - Non-linear
  - Llyod-Max
  - Weighted quantization
- Vector
  - LBG
  - Adaptive
  - Lattice

Scaler quantization

- Average distortion

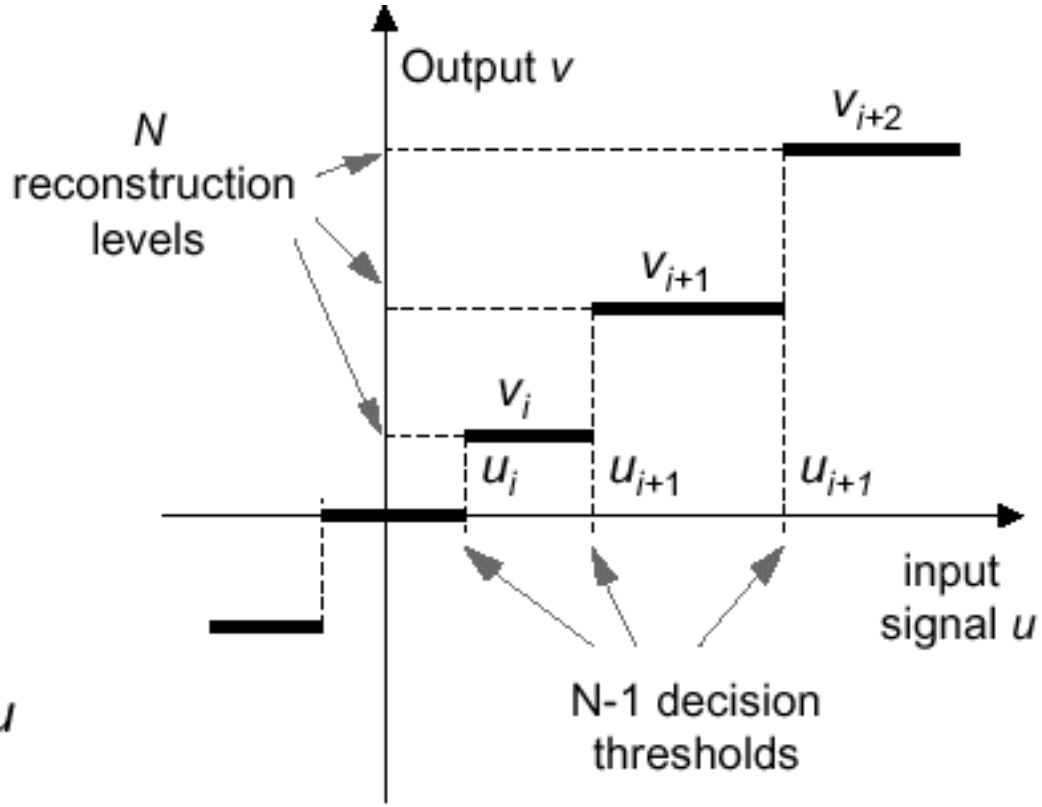
$$D = E\{d(U,V)\}$$

$$= \sum_{k=0}^{N-1} \int_{u_k}^{u_{k+1}} d(u,v_k) \cdot f_U(u)\mathrm{d}u$$

- Assume MSE

$$d(u,v_k) = (u - v_k)^2$$

$$D = \sum_{k=0}^{N-1} \int_{u_k}^{u_{k+1}} (u - v_k)^2 \cdot f_U(u)\mathrm{d}u$$



Uniform quantization

- A uniform quantizer is a quantizer where all reconstruction levels are equally spaced, that is,

  $v_{i+1} - v_i = \theta$, $1 \le i \le L-1$ and θ is the stepsize.

- If the input s is uniformly distributed over a finite interval [A,B], then a uniform quantizer with fixed L is given by

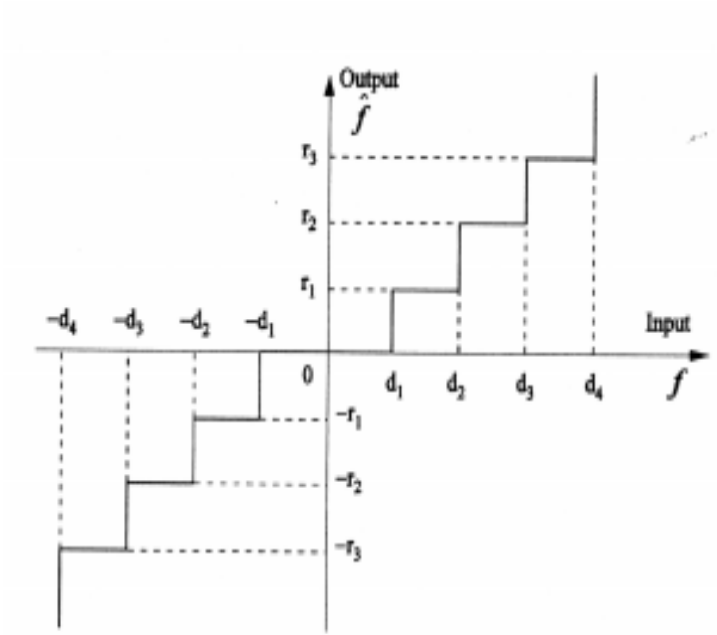
$$\theta = \frac{B - A}{L}$$

$$u_i = A + i\theta \quad \text{for } 0 \le i \le L$$

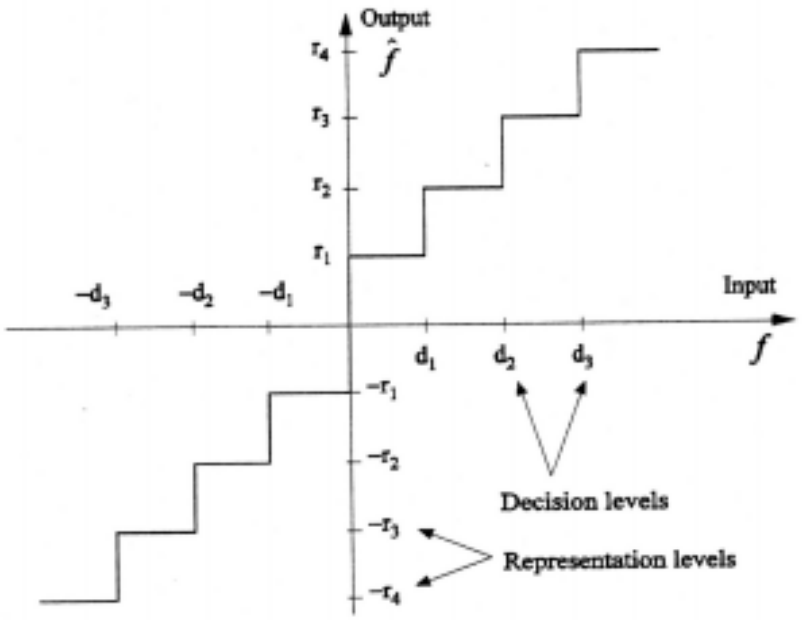
and

$$v_i = d_{i-1} + \frac{\theta}{2} \quad \text{for } 1 \le i \le L$$

Uniform Symmetric Quantizers



Midtreader



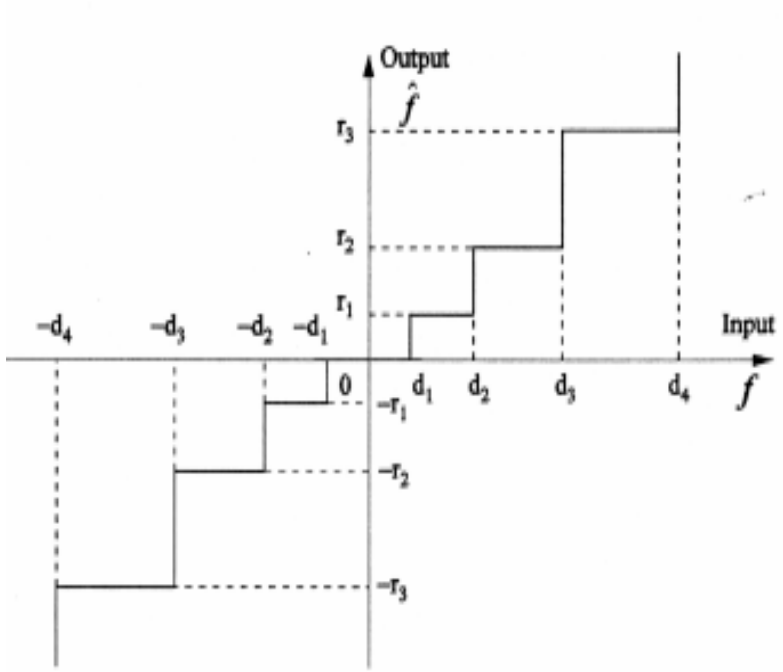
Midriser

Step size is constant

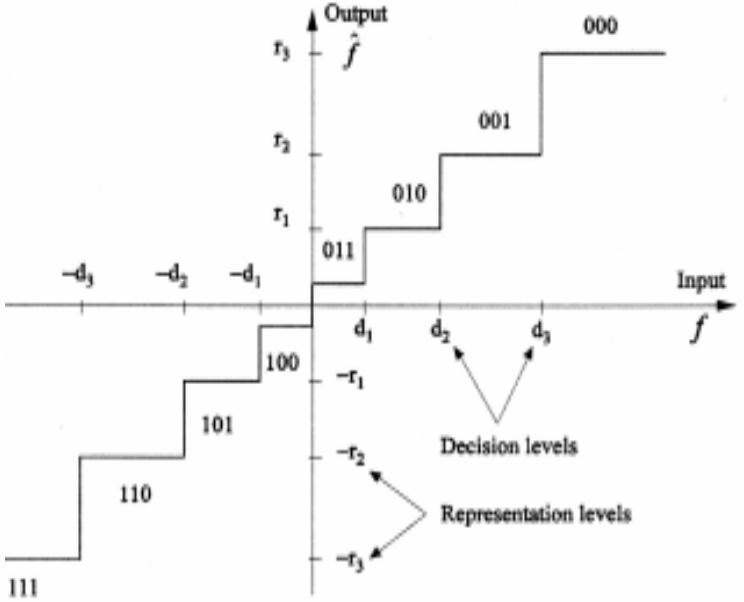
r_i: output levels

d_i: input levels

Nonuniform Symmetric Quantizers



Midtreader



Midriser

r_i: output levels

d_i: input levels

Optimal scalar quantization

- Lloyd-Max Quantization:

Given the number of levels L, select u_i and v_i , i=1,…, L,

in order to minimize the mean square quantization error,

$D= E\{(u-v_i)^2\}$, for a given input probability density

function p(s).

- Entropy-Constrained Quantization:

For a fixed output entropy H(u) = C, where C is a constant,
find u_i and v_i (L is unknown) in order to minimize

a distortion measure D.

Lloyd-Max quantizer design

- Non-uniform quantizer: Minimize with respect to r_i and d_i

$$E\left\{(s-r_i)^2\right\}=\sum_{i=1}^{L}\int_{d_{i-1}}^{d_i}(s-r_i)^2\,p(s)ds$$

- The necessary conditions are given by

$$r_i=\frac{\int_{d_{i-1}}^{d_i} s\,p(s)ds}{\int_{d_{i-1}}^{d_i} p(s)ds},\quad 1\le i\le L \qquad\qquad d_i=\frac{r_i+r_{i+1}}{2},\quad 1\le i\le L-1$$

$$d_0=-\infty \quad \text{and} \quad d_L=\infty$$

- Uniform quantizer: Include the additional constraint

$$r_i - r_{i-1} = \theta$$

The resulting nonlinear equations are solved numerically.

Vector quantization (1/2)

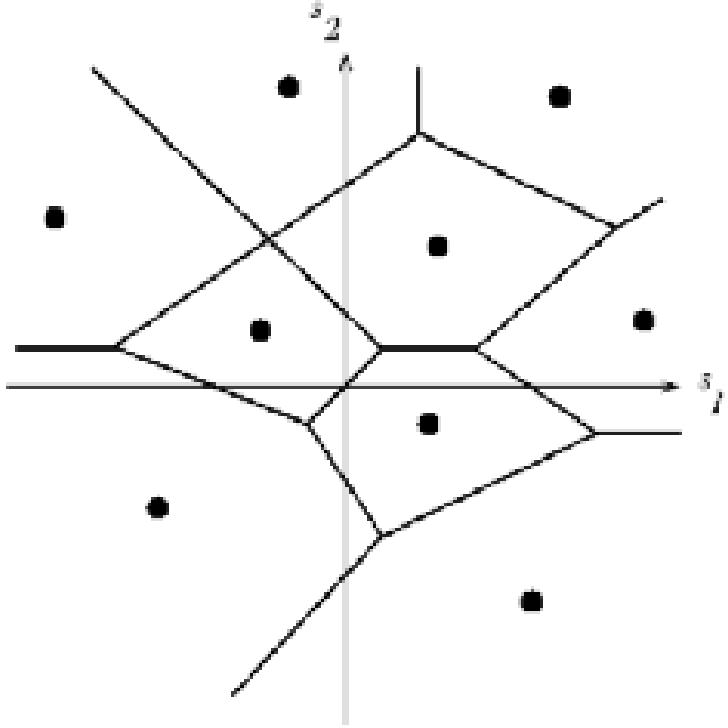
Let $\mathbf{s}=[s_1, s_2, \ldots, s_N]^T$ denote an N-dimensional vector corresponding to a block of pixels.

In VQ the N-dimensional space is quantized into L regions $\boldsymbol{R}_i$, i=1,…, L, where each region is represented by a single vector $\mathbf{r}^{(i)} = [r_1^{(i)}, r_2^{(i)}, \ldots, r_N^{(i)}]^T$.

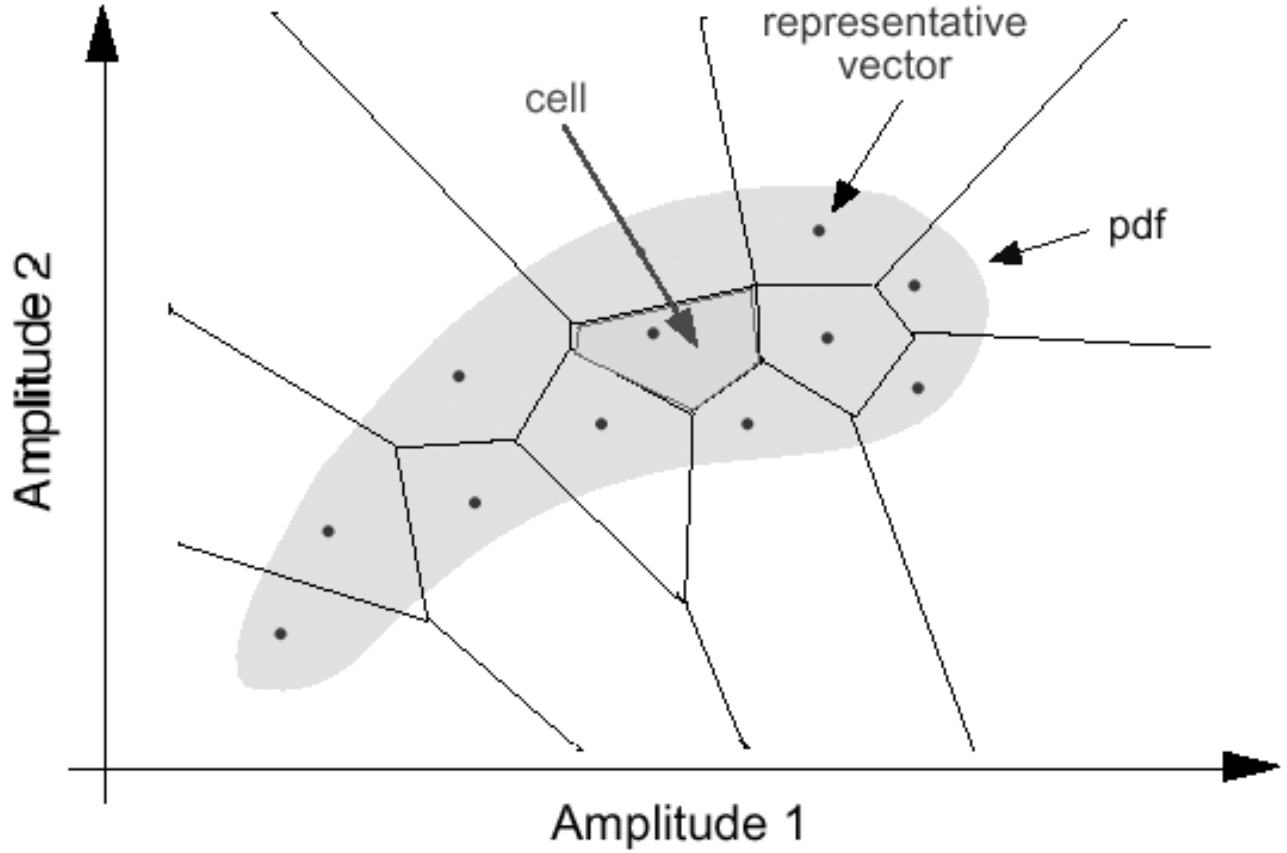
The set of vectors

$\mathbf{r}^{(i)} = [r_1^{(i)}, r_2^{(i)}, \ldots, r_N^{(i)}]^T$, i=1,…, L

is called as the *codebook*.



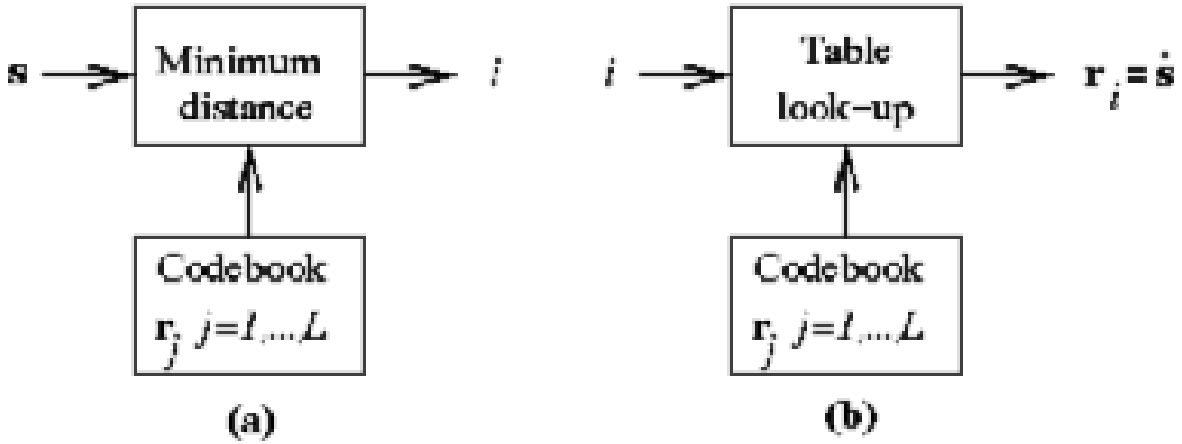
Vector quantization (1/2)



Vector quantization - Implementation

A vector quantizer can be described by

$$\dot{\mathbf{s}} = VQ(\mathbf{s}) = \mathbf{r}^{(i)} \quad \text{if } \mathbf{s} \in \mathbf{R}_i$$



The computational complexity resides in the encoder which performs a search for the closest element in the codebook to the input data vector.

Codebook design (1/2)

0) Choose an initial codebook $\boldsymbol{C}^{(j)}= \mathbf{r}_1, \ldots, \mathbf{r}_L$. Set j=0.

1) Determine the regions $\boldsymbol{R}_i$, i=1, …, L using $\boldsymbol{C}^{(j)}$ and a minimum distance classifier.

2) If $\frac{D^{(j-1)} - D^{(j)}}{D^{(j)}} < \varepsilon$, stop. $D^{(j)}$ denotes the overall distortion in data representation using $\boldsymbol{C}^{(j)}$.

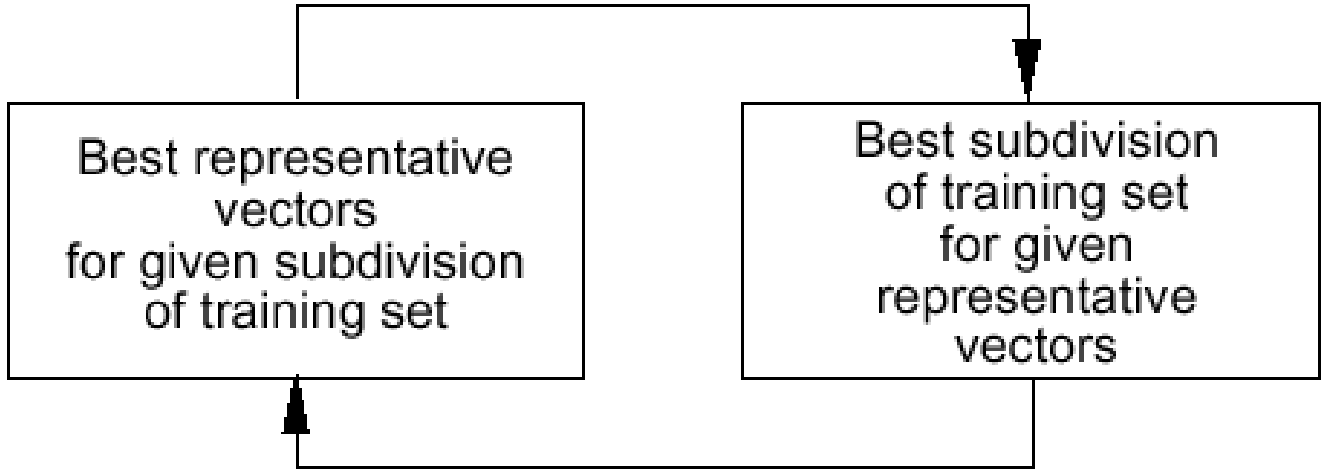
3) Increment j. Update $\boldsymbol{C}^{(j)}$ by computing the centroids of $\boldsymbol{R}_i$, i=1,…, L.

Comments:

- The overall distortion decreases at each step of the algorithm, but convergence is guaranteed only to a local minimum.
- Rule of thumb: 100 training vectors are required per codevector.
- Typically, L < 5000.

Codebook design (2/2)

Linde, Buzo, Gray, 1980: Lloyd algorithm generalized for VQ



Assumption: fixed code word length

Code book unstructured: full search

Methods, Standards (1/2)

- Lossless Image Compression Methods
  - Predictive coding
  - Bit-plane RL coding
  - Ziv-Lempel coding
- Lossless Image Compression Standards
  - Fax Standards: ITU-T G3, G4 and ISO JBIG
  - ISO JPEG LS

Methods, Standards (2/2)

- JPEG
- JPEG 2000