

Chapter 8 : Image Compression





Applications of image compression

- Televideo conferencing,
- Remote sensing (satellite imagery),
- Document and medical imaging,
- Facsimile transmission (FAX),
- Control of remotely piloted vehicles in military, space, and hazardous waste management



Fundamentals

- What is *data* and hat is *information* ?
 - Data are the means by which information is conveyed. Various amounts of data may be used to represent the same amount of information
- Data redundancy
 - Coding redundancy
 - Interpixel redundancy
 - Psychovisual redundancy

Coding redundancy

- The graylevel histogram of an image can provide a great deal of insight into the construction of codes
- The average number of bits to represent each pixel

$$L_{avg} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k)$$

- $l(r_k)$ is the average number of bits to represent each value of r_k
- Variable length coding (VLC) : Higher probability, shorter bit length

r_k	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_0 = 0$	0.19	000	3	11	2
$r_1 = 1/7$	0.25	001	3	01	2
$r_2 = 2/7$	0.21	010	3	10	2
$r_3 = 3/7$	0.16	011	3	001	3
$r_4 = 4/7$	0.08	100	3	0001	4
$r_5 = 5/7$	0.06	101	3	00001	5
$r_6 = 6/7$	0.03	110	3	000001	6
$r_7 = 1$	0.02	111	3	000000	6

$$L_{avg} = 3.0 \text{ bits (code1)}$$

$$2.7 \text{ bits (code2)}$$



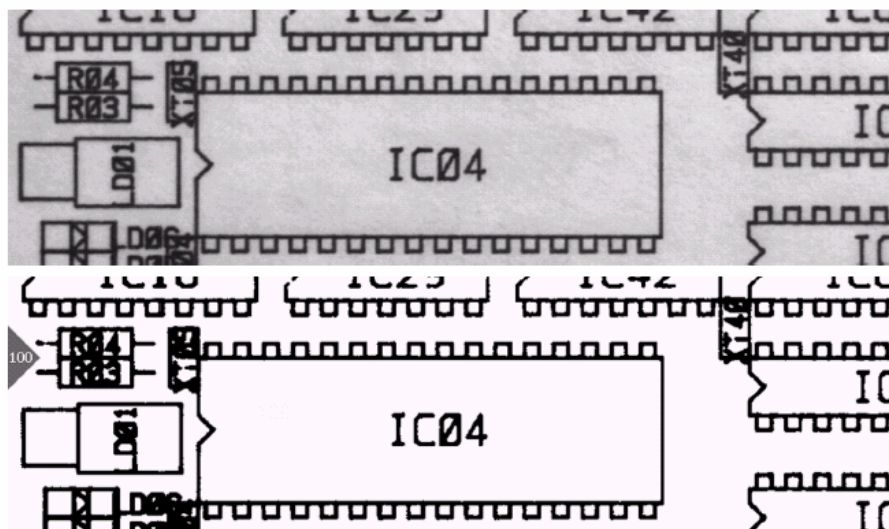
Interpixel redundancy

- Autocorrelation coefficients

$$\gamma(\Delta n) = \frac{A(\Delta n)}{A(0)} \quad A(\Delta n) = \frac{1}{N - \Delta n} \sum_{y=0}^{N-1-\Delta n} f(x, y) f(x, y + \Delta n)$$

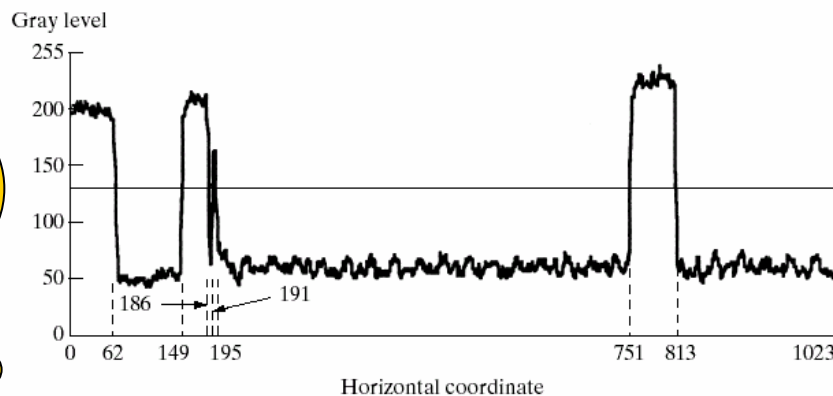
- Interpixel redundancy
 - Spatial redundancy
 - Geometrical redundancy
 - Interframe redundancy
- Larger autocorrelation, more interpixel redundancy

Run-length coding to remove spatial redundancy



$$C_R = 2.63$$

$$R_D = 1 - \frac{1}{2.63} = 0.62$$



Line 100: (1, 63) (0, 87) (1, 37) (0, 5) (1, 4) (0, 556) (1, 62) (0, 210)

ization
threshold.
(d) Run-length
code.

Psychovisual redundancy

- Certain information simply has less relative importance than others in normal visual processing – psychovisual redundancy
- The elimination of psychovisually redundant data results in a loss of quantitative information – called quantization
⇒ lossy data compression
 - E.g., quantization in graylevels
 - E.g., line interlacing in TV
(reduced video scanning rate)

(b) False contouring on quantization

(c) Improve graylevel quantization via dithering





Fidelity criteria

- Objective fidelity criterion

- Root-mean-square error

$$e_{rms} = \left[\frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2 \right]^{\frac{1}{2}}$$

- Mean-square signal-to-noise ratio

$$SNR_{ms} = \frac{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} \hat{f}(x, y)^2}{\sum_{x=0}^{M-1} \sum_{y=0}^{N-1} [\hat{f}(x, y) - f(x, y)]^2}$$

- Subjective fidelity criterion

- Much worse, worse, slightly worse, the same, slightly better, better, much better

Image compression model

- Source encoder
 - Remove input redundancies
- Channel encoder
 - Increase the noise immunity of the source encoder's output

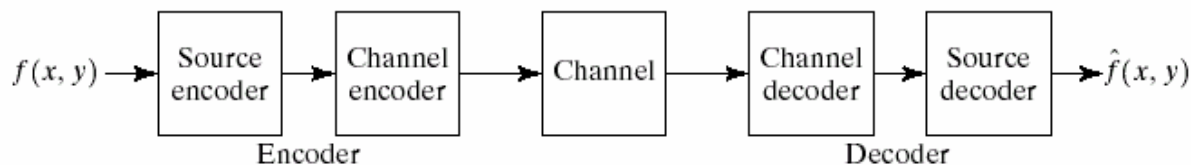


FIGURE 8.5 A general compression system model.

Source encoder model

■ Mapper

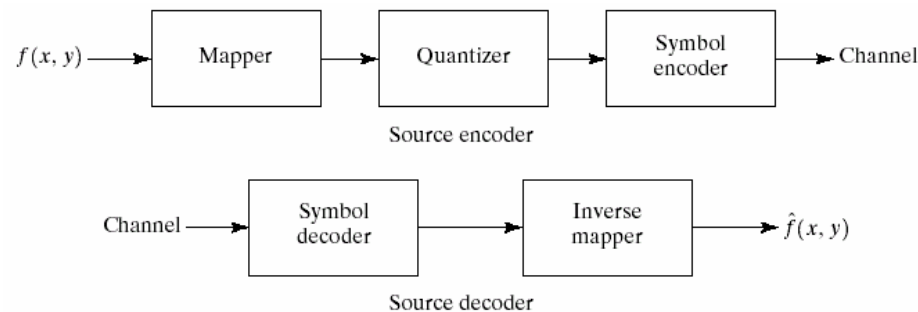
- Transform the input data into a format designed to reduce interpixel redundancies in the image (reversible)

■ Quantizer

- Reduce the accuracy of the mapper's output (reduce the psychovisual redundancies) – irreversible and must be omitted on error-free compression

■ Symbol encoder

- Fixed or variable-length coder (assign the shortest code words to the most frequently occurring output to reduce coding redundancies) -- reversible



a
b

FIGURE 8.6 (a) Source encoder and (b) source decoder model.



Channel encoder and decoder

- Designed to reduce the impact of channel noise by inserting a controlled form of redundancy into the source encoded data
- Hamming code as a channel code
 - 7-bit Hamming (7,4) – the minimum distance is 3 and could correct one bit error

h_1 h_2 h_4 are even
parity bits

$$h_1 = b_3 \oplus b_2 \oplus b_0 \quad h_3 = b_3$$

$$h_2 = b_3 \oplus b_1 \oplus b_0 \quad h_5 = b_2$$

$$h_4 = b_2 \oplus b_1 \oplus b_0 \quad h_6 = b_1$$

$$h_7 = b_0$$

- A single-bit error is indicated by a nonzero parity word $c_4c_2c_1$

$$c_1 = h_1 \oplus h_3 \oplus h_5 \oplus h_7$$

$$c_2 = h_2 \oplus h_3 \oplus h_6 \oplus h_7$$

$$c_4 = h_4 \oplus h_5 \oplus h_6 \oplus h_7$$



Information theory

- Explore the minimum amount of data that is sufficient to describe completely an image without loss of information
- Self-information of an event E with probability $P(E)$

$$I(E) = \log \frac{1}{P(E)} = -\log P(E)$$

$$P(E)=1 \Rightarrow I(E)=0$$

$$P(E)=1/2 \Rightarrow I(E)=1 \text{ bit}$$

The base of logarithm
determines the information units



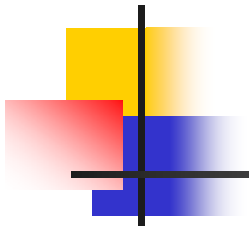
Information channel

- Source alphabet $A = \{a_1, a_2, \dots, a_J\}$ $\sum_{j=1}^J P(a_j) = 1$
- Symbol probability $\mathbf{z} = [P(a_1), P(a_2), \dots, P(a_J)]^T$
- Ensemble (A, \mathbf{z}) describes the information source completely
- Average information per source output

$$H(\mathbf{z}) = -\sum_{j=1}^J P(a_j) \log P(a_j)$$

$H(\mathbf{z})$ is the uncertainty or entropy of the source or the average amount of information (in m-ary units) obtained by observing a single source output

For equally probable source symbols, entropy is maximized

- 
- Channel alphabet $B = \{b_1, b_2, \dots, b_K\}$
 - Channel description $(B, \mathbf{v}) \quad \mathbf{v} = [P(b_1), P(b_2), \dots, P(b_K)]^T$

$$P(b_k) = \sum_{j=1}^J P(b_k | a_j) P(a_j)$$

$$\mathbf{Q} = \begin{bmatrix} P(b_1|a_1) & P(b_1|a_2) & \dots & \dots & P(b_1|a_J) \\ P(b_2|a_1) & \dots & \dots & \dots & \cdot \\ \cdot & \dots & \dots & \dots & \cdot \\ \cdot & \dots & \dots & \dots & \cdot \\ P(b_K|a_1) & P(b_K|a_2) & \dots & \dots & P(b_K|a_J) \end{bmatrix}$$

$$\mathbf{v} = \mathbf{Q}\mathbf{z}$$

\mathbf{Q} : Forward channel transition matrix, channel matrix



Conditional entropy function

- $$H(\mathbf{z}|b_k) = - \sum_{j=1}^J P(a_j|b_k) \log P(a_j|b_k)$$

$$H(\mathbf{z}|\mathbf{v}) = \sum_{k=1}^K H(\mathbf{z}|b_k) P(b_k)$$

$$H(\mathbf{z}|\mathbf{v}) = - \sum_{j=1}^J \sum_{k=1}^K P(a_j, b_k) \log P(a_j|b_k)$$

- $H(\mathbf{z}|\mathbf{v})$ is called the equivocation of \mathbf{z} wrt \mathbf{v} .
- The difference between $H(\mathbf{z})$ and $H(\mathbf{z}|\mathbf{v})$ is the average information received upon observing a single output symbol -- mutual information $I(\mathbf{z}, \mathbf{v})$

$$I(\mathbf{z}, \mathbf{v}) = H(\mathbf{z}) - H(\mathbf{z}|\mathbf{v})$$



Mutual information

- $$I(\mathbf{z}, \mathbf{v}) = \sum_{j=1}^J \sum_{k=1}^K P(a_j, b_k) \log \frac{P(a_j, b_k)}{P(a_j)P(b_k)}$$
$$= \sum_{j=1}^J \sum_{k=1}^K P(a_j) q_{kj} \log \frac{q_{kj}}{\sum_{i=1}^J P(a_i) q_{ki}}$$

$$\mathbf{Q} = \{q_{ij}\}$$

- The minimum possible value of $I(\mathbf{z}, \mathbf{v})$ is zero and occurs when the input and output symbols are statistically independent (i.e., $P(a_j, b_k) = P(a_j)P(b_k)$)
- The maximum value of $I(\mathbf{z}, \mathbf{v})$ for all possible \mathbf{z} is called the capacity C described by channel matrix \mathbf{Q}

$$C = \max_{\mathbf{z}} [I(\mathbf{z}, \mathbf{v})]$$



Channel capacity

- Capacity : the maximum rate at which information can be transmitted reliably through the channel
- Channel capacity does not depend on the input probability of the source (i.e., on how the channel is used) but is a function of the conditional probabilities defining the channel alone (i.e., \mathbf{Q})

Binary entropy function

- For binary source $\mathbf{z} = [p_{bs}, 1 - p_{bs}]^T$

To estimate source entropy

$$H(\mathbf{z}) = -p_{bs} \log_2 p_{bs} - \bar{p}_{bs} \log_2 \bar{p}_{bs}$$

$$\text{Max}(H_{bs}) = 1 \text{ bit} \quad \text{when } p_{bs} = 0.5$$

Plot_1

- Binary symmetrical channel (BSC)

- error probability : p_e

- channel matrix : $\mathbf{Q} = \begin{bmatrix} \bar{p}_e & p_e \\ p_e & \bar{p}_e \end{bmatrix}$

To estimate channel capacity

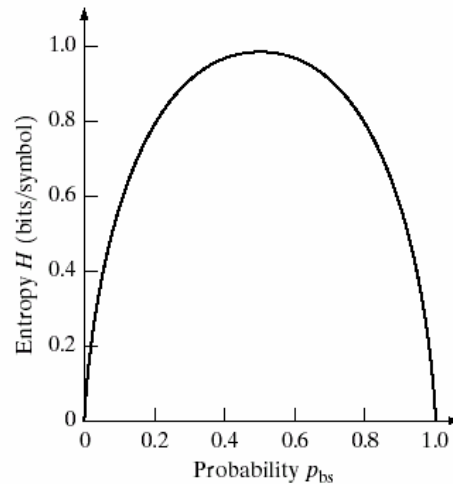
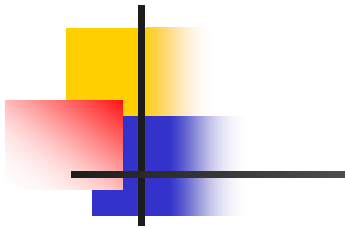
- probability of received symbols : $\mathbf{v} = \mathbf{Q}\mathbf{z} = \begin{bmatrix} \bar{p}_e p_{bs} + p_e \bar{p}_{bs} \\ p_e p_{bs} + \bar{p}_e \bar{p}_{bs} \end{bmatrix}$

- mutual information : $I(\mathbf{z}, \mathbf{v}) = H_{bs}(p_{bs} p_e + \bar{p}_{bs} \bar{p}_e) - H_{bs}(p_e)$

Plot_2

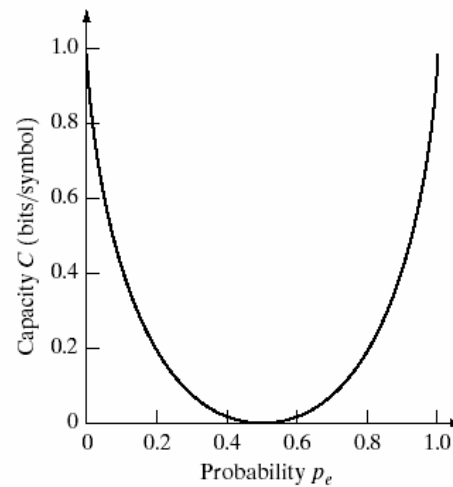
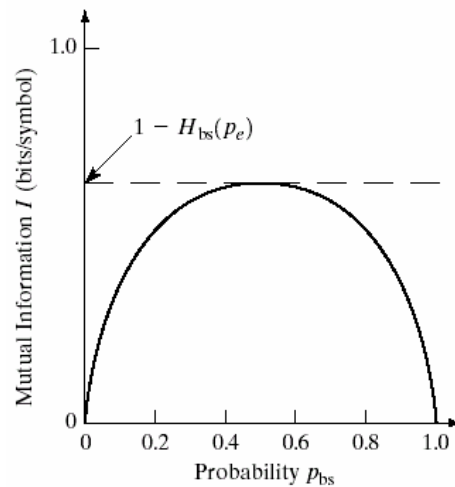
$$I(\mathbf{z}, \mathbf{v}) = 0 \text{ when } p_{bs} = 0 \text{ or } 1$$

$$I(\mathbf{z}, \mathbf{v}) = 1 - H_{bs}(p_e) = C \text{ when } p_{bs} = 0.5 \text{ and any } p_e$$



a
b c

FIGURE 8.8 Three binary information functions: (a) the binary entropy function; (b) the mutual information of a binary symmetric channel (BSC); (c) the capacity of the BSC.



when $p_e = 0$ or $1 \Rightarrow C = 1$ bit

when $p_e = 0.5 \Rightarrow C = 0$ bit (no information transmitted)

Shannon's first theorem for noiseless coding

- Define the minimum average code length per source symbol that can be achieved
- For a zero-memory source (with finite ensemble (A, \mathbf{z}) and statistically independent source symbols)
 - Single symbol : $A = \{a_1, a_2, \dots, a_J\}$
 - Block symbol : (n -tuple of symbols) $A' = \{\alpha_1, \alpha_2, \dots, \alpha_{J^n}\}$

$$P(\alpha_i) = P(a_{j_1})P(a_{j_2})\dots P(a_{j_n})$$

- $(A, \mathbf{z}) \rightarrow H(\mathbf{z})$
 - $(A', \mathbf{z}') \rightarrow H(\mathbf{z}')$
- $$H(\mathbf{z}') = -\sum_{i=1}^{J^n} P(\alpha_i) \log P(\alpha_i)$$

$$H(\mathbf{z}') = nH(\mathbf{z})$$

Entropy of zero-memory source with block symbols



N-extended source

- Use n symbols as a block symbol
- Average word length of the n-extended code can be

$$L'_{avg} = \sum_{i=1}^{J^n} P(\alpha_i) l(\alpha_i) = \sum_{i=1}^{J^n} P(\alpha_i) \left\lceil \log \frac{1}{P(\alpha_i)} \right\rceil$$

$$\Rightarrow H(\mathbf{z}') \leq L'_{avg} < H(\mathbf{z}') + 1$$

$$H(\mathbf{z}) \leq \frac{L'_{avg}}{n} < H(\mathbf{z}) + \frac{1}{n}$$

$$\lim_{n \rightarrow \infty} \left[\frac{L'_{avg}}{n} \right] = H(\mathbf{z})$$

It is possible to make L'_{avg} / n arbitrarily close to $H(\mathbf{z})$ by coding infinitely long extensions of the source

Coding efficiency

- Coding efficiency : $\eta = n \frac{H(z)}{L'_{avg}}$
- Example :
 - Original source : $P(a_1) = 2/3, \quad P(a_2) = 1/3$
 $H(z) = 1.83 \text{ bits / symbol}$
 $L'_{avg} = 1 \text{ bit / symbol} \Rightarrow \eta = 0.918/1 = 0.918$
 - Second extension source : $\mathbf{z}' = \{4/9, 2/9, 2/9, 1/9\}$
 $L'_{avg} = 17/9 = 1.89 \text{ bits / symbol} \Rightarrow \eta = 1.83/1.89 = 0.97$

TABLE 8.4
Extension coding
example.

α_i	Source Symbols	$P(\alpha_i)$ Eq. (8.3-14)	$I(\alpha_i)$ Eq. (8.3-1)	$l(\alpha_i)$ Eq. (8.3-16)	Code Word	Code Length
<i>First Extension</i>						
α_1	a_1	2/3	0.59	1	0	1
α_2	a_2	1/3	1.58	2	1	1
<i>Second Extension</i>						
α_1	$a_1 a_1$	4/9	1.17	2	0	1
α_2	$a_1 a_2$	2/9	2.17	3	10	2
α_3	$a_2 a_1$	2/9	2.17	3	110	3
α_4	$a_2 a_2$	1/9	3.17	4	111	3



Shannon's second theorem for noisy coding

- For any $R < C$ (channel capacity of the zero-memory channel with Q), there exists codes of block length r and rate R such that the block decoding error can be arbitrarily small
- Rate distortion function
given a distortion D , the minimum rate at which information about the source can be conveyed to the user

Q : the probability from source symbols to output symbols by compression

$$Q_D = \{q_{kj} | d(Q) \leq D\}$$

$$R(D) = \min_{Q \in Q_D} [I(\mathbf{z}, \mathbf{v})]$$

$d(Q)$: average distortion

D : distortion threshold

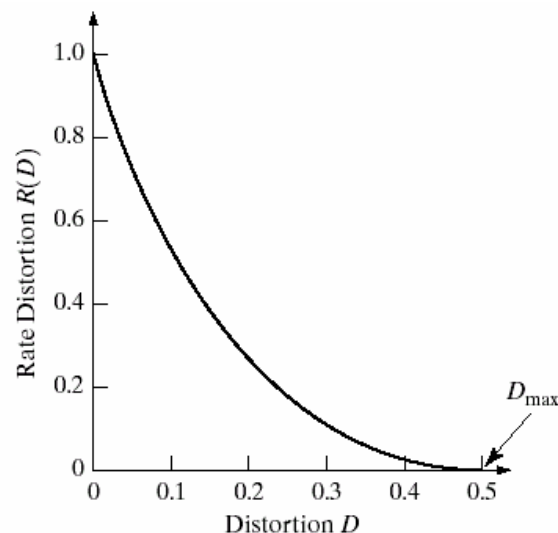
Example of Rate-distortion function

- Consider a zero-memory binary source with equally probable source symbols $\{0,1\}$

→ $R(D) = 1 - H_{bs}(D)$

- $R(D)$ is monotonically decreasing and convex in the interval $(0, D_{\max})$
- Shape of R-D function represents the coding efficiency of a coder

FIGURE 8.10 The rate distortion function for a binary symmetric source.

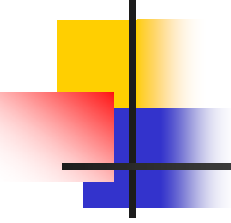


Estimate the information content (entropy) of an image

- Construct a source model based on the relative frequency of occurrence of the graylevels
 - First-order estimate (consider histogram of single pixels) : 1.81 bits/pixel
 - Second-order estimate (consider histogram of pairs of adjacent pixels) : 1.25 bits/pixel
 - Third-order, fourth-order ... : approaching the source entropy

21	21	21	95	169	243	243	243
21	21	21	95	169	243	243	243
21	21	21	95	169	243	243	243
21	21	21	95	169	243	243	243

Example image

- 
-
- The difference between the higher-order estimates of entropy and the first-order estimate indicates the presence of interpixel redundancy
 - If the pixels are statistically independent, the higher-order estimates are equivalent to the first-order estimate and variable length coding (VLC) provides optimal compression (i.e., no further mapping is necessary)
 - First-order estimate of difference mapped source : 1.41 bits/pixel
 $1.41 > 1.25 \Rightarrow$ there is an even better mapping

Error-free compression -- variable-length coding (VLC)

■ Huffman coding

- the most popular method to yield the smallest possible number of code symbols per source symbol
- construct the Huffman tree according to source symbol probabilities
- Code the Huffman tree
- Compute the source entropy, average code length, and code efficiency

Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

FIGURE 8.11
Huffman source reductions.

■ Huffman code is :

- a block code (each symbol is mapped to a fixed sequence of bits)
- instantaneous (decoded without referencing succeeding symbols)
- uniquely decodable (any code word is not a prefix of another)
- for example : 010100111100 $\rightarrow a_3 a_1 a_2 a_2 a_6$

$$L_{avg} = 2.2 \text{ bits / symbol}$$

$$H(z) = 2.14 \text{ bits/symbol}$$

$$\eta = 0.973$$

FIGURE 8.12
Huffman code
assignment
procedure.

Original source			Source reduction			
Sym.	Prob.	Code	1	2	3	4
a_2	0.4	1	0.4 1	0.4 1	0.4 1	0.6 0
a_6	0.3	00	0.3 00	0.3 00	0.3 00	0.4 1
a_1	0.1	011	0.1 011	0.2 010	0.3 01	
a_4	0.1	0100	0.1 0100	0.1 011		
a_3	0.06	01010	0.1 0101			
a_5	0.04	01011				

Variations of VLC

TABLE 8.5
Variable-length
codes.

Source symbol	Probability	Binary Code	Huffman	Truncated Huffman	B ₂ -Code	Binary Shift	Huffman Shift
<i>Block 1</i>							
a_1	0.2	00000	10	11	C00	000	10
a_2	0.1	00001	110	011	C01	001	11
a_3	0.1	00010	111	0000	C10	010	110
a_4	0.06	00011	0101	0101	C11	011	100
a_5	0.05	00100	00000	00010	C00C00	100	101
a_6	0.05	00101	00001	00011	C00C01	101	1110
a_7	0.05	00110	00010	00100	C00C10	110	1111
<i>Block 2</i>							
a_8	0.04	00111	00011	00101	C00C11	111 000	00 10
a_9	0.04	01000	00110	00110	C01C00	111 001	00 11
a_{10}	0.04	01001	00111	00111	C01C01	111 010	00 110
a_{11}	0.04	01010	00100	01000	C01C10	111 011	00 100
a_{12}	0.03	01011	01001	01001	C01C11	111 100	00 101
a_{13}	0.03	01100	01110	100000	C10C00	111 101	00 1110
a_{14}	0.03	01101	01111	100001	C10C01	111 110	00 1111
<i>Block 3</i>							
a_{15}	0.03	01110	01100	100010	C10C10	111 111 000	00 00 10
a_{16}	0.02	01111	010000	100011	C10C11	111 111 001	00 00 11
a_{17}	0.02	10000	010001	100100	C11C00	111 111 010	00 00 110
a_{18}	0.02	10001	001010	100101	C11C01	111 111 011	00 00 100
a_{19}	0.02	10010	001011	100110	C11C10	111 111 100	00 00 101
a_{20}	0.02	10011	011010	100111	C11C11	111 111 101	00 00 1110
a_{21}	0.01	10100	011011	101000	C00C00C00	111 111 110	00 00 1111
<i>Entropy</i> 4.0							
<i>Average length</i> 5.0 4.05 4.24 4.65 4.59 4.13							



Variations of VLC (cont.)

- Considered when a large number of source symbols are considered
- Truncated Huffman coding
 - Only partial symbols (with most probabilities, here the top 12) are encoded with Huffman code
 - A prefix code (whose probability was the sum of other symbols, here “10”) followed with a fixed-length code is used to represent all the other symbols
- B-code
 - Each code word is made up of *continuation* bits (C) and *information* bits (natural binary numbers)
 - C can be 1 or 0, but alternative
 - E.g., $a_{11}a_2a_7 \rightarrow 001010101000010$ or 101110001100110

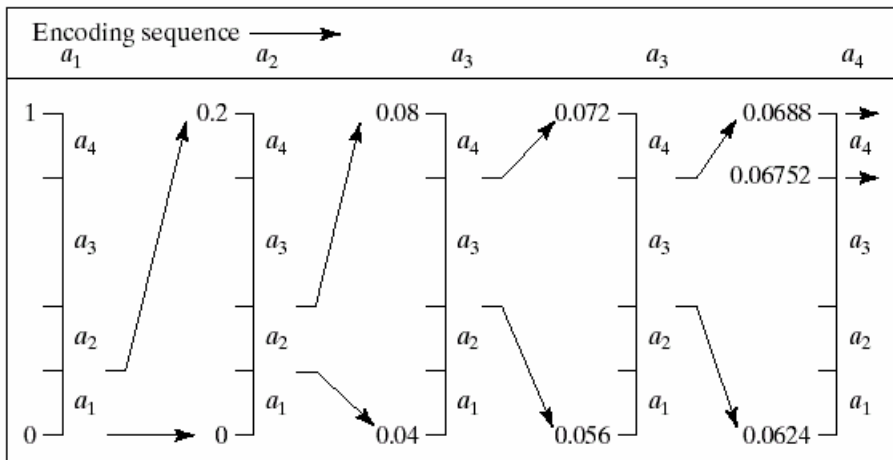


Variations of VLC (cont.)

- Binary shift code
 - Divide the symbols into blocks of equal size
 - Code the individual elements within all blocks identically
 - Add shift-up or shift-down symbols to identify each block (here, “111”)
- Huffman shift code
 - Select one reference block
 - Sum up probabilities of all other blocks and use it to determine the shift symbol by Huffman method (here, “00”)

Arithmetic coding

Source Symbol	Probability	Initial Subinterval
a_1	0.2	$[0.0, 0.2)$
a_2	0.2	$[0.2, 0.4)$
a_3	0.4	$[0.4, 0.8)$
a_4	0.2	$[0.8, 1.0)$



- AC generates non-block codes (no look-up table as in Huffman code)
- An entire sequence of source symbols is assigned a single code word
- The code word itself defines an interval of real numbers between 0 and 1. As the number of symbols in the message increases, the interval becomes smaller (more information bits to represent this real number)
- Multiple symbols are integrally coded with a set of bits \rightarrow number of bits per symbol is effectively fractional



Arithmetic coding (cont.)

- Example : coding of $a_1a_2a_3a_3a_4$
 - Any real number between $[0.06752, 0.0688]$ can represent the source symbols (e.g., $0.000100011=0.068359375$)
 - Theoretically, 0.068 is enough to represent the source symbol \rightarrow 3 decimal digits \rightarrow 0.6 decimal digits per symbol
 - Actually, $0.068359375 \rightarrow$ 9 bits for 5 symbols \rightarrow 1.8 bits per symbol
 - As the length of the source symbol sequence increases, the resulting arithmetic code approaches the Shannon's bound
 - Two factors limit the coding performance
 - An end-of-message indicator is necessary to separate one message sequence from another
 - The finite precision arithmetic



LZW (Lempel-Ziv-Welch) coding

- Assign fixed-length code words to variable-length sequences of source symbols but require no a priori knowledge of the probabilities of occurrence of symbols
- LZW compression has been integrated into GIF, TIFF, and PDF formats
- For a 9-bit (512 words) dictionary, the latter half entries are constructed in the encoding process. An entry of two or multiple pixels can be possible (i.e., assigned with a 9 bits code)
 - If the size of the dictionary is too small, the detection of matching gray-level sequences will be less likely.
 - If too large, the size of code words will adversely affect compression performance



LZW coding (cont.)

Currently Recognized Sequence	Pixel Being Processed	Encoded Output	Dictionary Location (Code Word)	Dictionary Entry
	39			
39	39	39	256	39-39
39	126	39	257	39-126
126	126	126	258	126-126
126	39	126	259	126-39
39	39			
39-39	126	256	260	39-39-126
126	126			
126-126	39	258	261	126-126-39
39	39			
39-39	126			
39-39-126	126	260	262	39-39-126-126
126	39			
126-39	39	259	263	126-39-39
39	126			
39-126	126	257	264	39-126-126
126		126		

- The LZW decoder builds an identical decompression dictionary as it decodes simultaneously the bit stream
- Notice to handle the dictionary overflow



Bit-plane coding

- Bit-plane decomposition

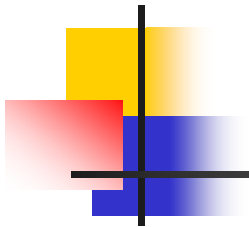
- Binary : bit change between two adjacent codes may be significant (e.g., 127 and 128)
- Gray code : only 1 bit is changed between any two adjacent codes

$$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \dots + a_12^1 + a_02^0$$

$$g_i = a_i \oplus a_{i+1}$$

$$g_{m-1} = a_{m-1} \quad 0 \leq i \leq m-2$$

- Gray-coded bit planes are less complex than the corresponding binary bit planes



Bit 7



Bit 7



Bit 3



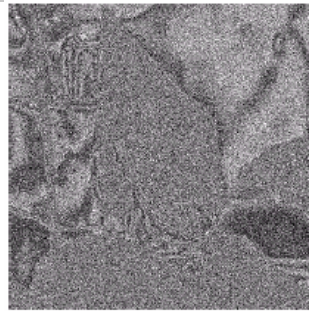
Bit 3



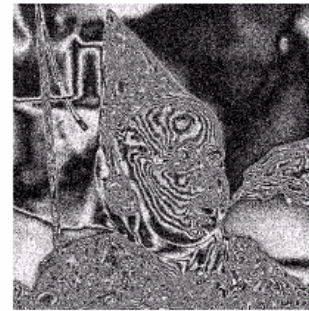
Bit 6



Bit 6



Bit 2



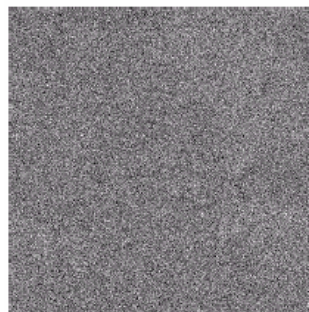
Bit 2



Bit 5



Bit 5



Bit 1



Bit 1



Bit 4



Bit 4



Bit 0



Bit 0



Lossless compression for binary images

- 1-D run-length coding
 - RLC+VLC according to run-lengths statistics
- 2-D run-length coding
 - used for FAX image compression
 - Relative address coding (RAC)
 - based on the principle of tracking the binary transitions that begin and end each black and white run
 - combined with VLC



Other methods for lossless compression of binary images

- White block skipping
 - code the solid lines as 0's and all other lines with a 1 followed by original bit patterns
- Direct contour tracing
 - represent each contour by a single boundary point and a set of directionals
- Predictive differential quantizing (PDO)
 - a scan-line-oriented contour tracing procedure
- Comparisons of various algorithms
 - only entropies after pixel mapping are computed, instead of real encoding (see next slide)

Comparison of various methods

Method	Bit-plane code rate (bits/pixel)								Code Rate	Compression Ratio
	7	6	5	4	3	2	1	0		
<i>Binary Bit-Plane Coding</i>										
CBC (4×4)	0.14	0.24	0.60	0.79	0.99	—	—	—	5.75	1.4:1
RLC	0.09	0.19	0.51	0.68	0.87	1.00	1.00	1.00	5.33	1.5:1
PDQ	0.07	0.18	0.79	—	—	—	—	—	6.04	1.3:1
DDC	0.07	0.18	0.79	—	—	—	—	—	6.03	1.3:1
RAC	0.06	0.15	0.62	0.91	—	—	—	—	5.17	1.4:1
<i>Gray Bit-Plane Coding</i>										
CBC (4×4)	0.14	0.18	0.48	0.40	0.61	0.98	—	—	4.80	1.7:1
RLC	0.09	0.13	0.40	0.33	0.51	0.85	1.00	1.00	4.29	1.9:1
PDQ	0.07	0.12	0.61	0.40	0.82	—	—	—	5.02	1.6:1
DDC	0.07	0.11	0.61	0.40	0.81	—	—	—	5.00	1.6:1
RAC	0.06	0.10	0.49	0.31	0.62	—	—	—	4.05	1.8:1

	WBS (1×8)	WBS (4×4)	RLC	PDQ	DDC	RAC
<i>Code rate (bits/pixel)</i>	0.48	0.39	0.32	0.23	0.22	0.23
<i>Compression ratio</i>	2.1:1	2.6:1	3.1:1	4.4:1	4.7:1	4.4:1

- Run-length coding proved to be the best coding method for bit-plane coded images
- 2-D techniques (PDQ, DDC, RAC) perform better when compressing binary images or higher bit-plane images
- Gray-coded images proved to gain additional 1 bit/pixel compression efficiency relative to binary-coded images

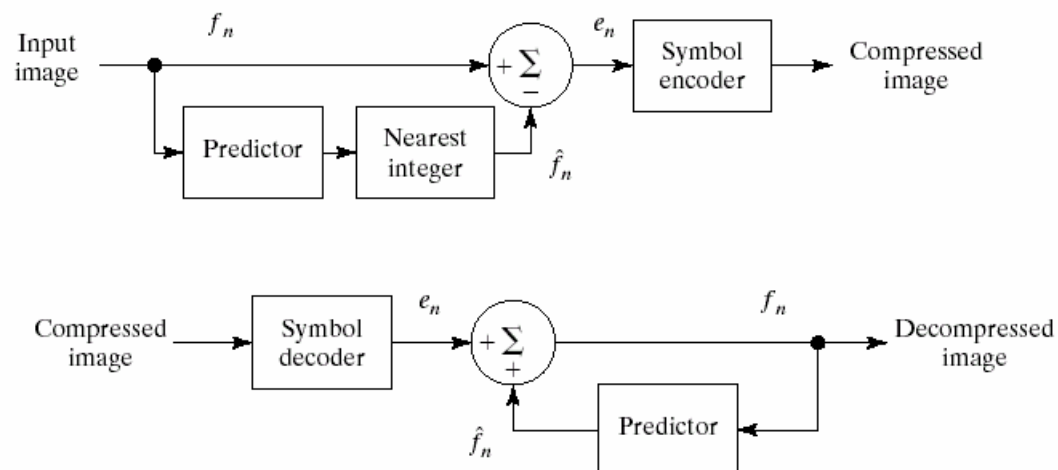
For binary image

Lossless predictive coding

- Eliminating the interpixel redundancies of closely spaced pixels by extracting and coding only the new information (the difference between the actual and predicted value) in each pixel
- System architecture
 - the same predictor in the encoder and decoder sides $e_n = f_n - \hat{f}_n$
 - rounding the predicted value
 - RLC symbol encoder

a
b

FIGURE 8.19 A lossless predictive coding model:
(a) encoder;
(b) decoder.





Methods of prediction

- Use local, global, or adaptive predictor to generate \hat{f}_n
- linear prediction : linear combination of m previous pixels

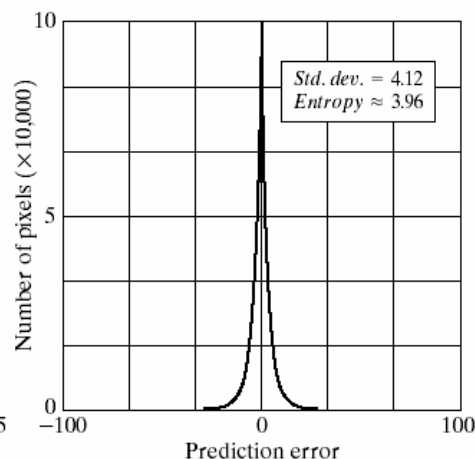
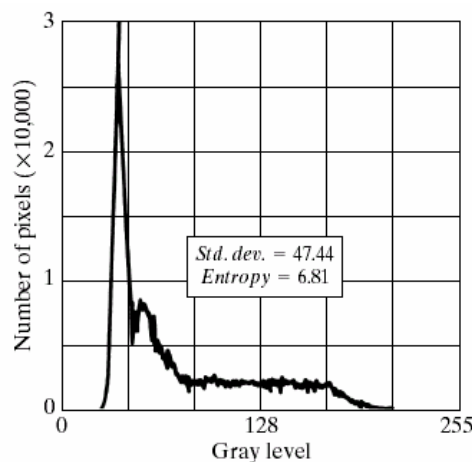
$$\hat{f}_n = \text{round} \left[\sum_{i=1}^m \alpha_i f_{n-i} \right]$$

- m : order of prediction
- α_i : prediction coefficients
- Use local neighborhoods (e.g., pixels-1,2,3,4) for prediction of pixel-X in 2-D images
 - special case : previous pixel predictor

2	3	4
1	X	

Entropy reduction by difference mapping

- Due to removal of interpixel redundancies by prediction, first-order entropy of difference mapping will be lower than the original image (3.96 bits/pixel vs. 6.81 bits/pixel)
- The probability density function of the prediction errors is highly peaked at zero and characterized by a relatively small variance (modeled by the zero mean uncorrelated Laplacian pdf)

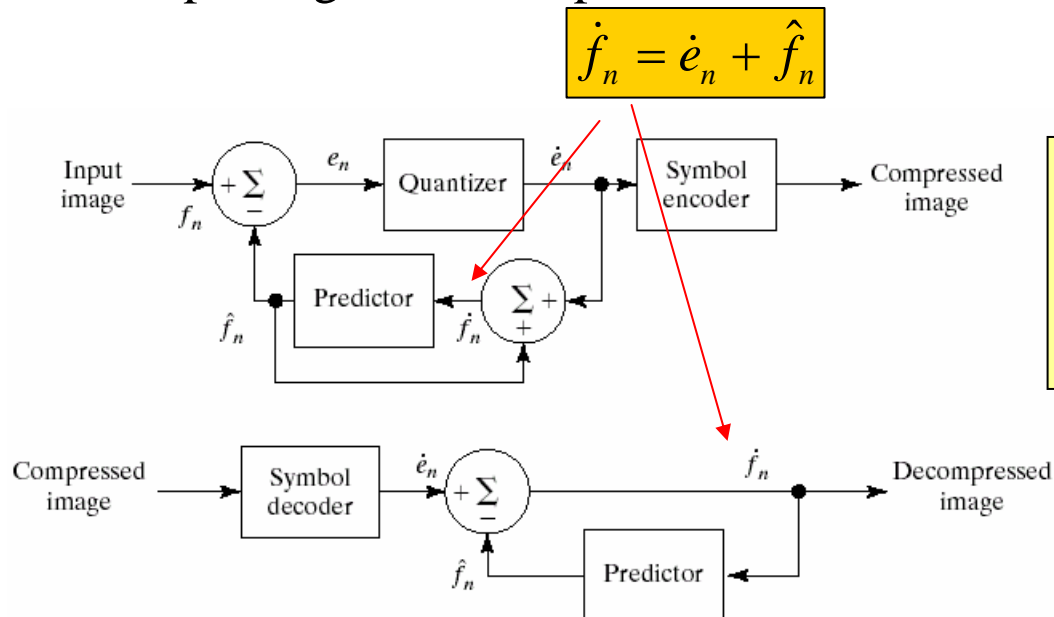


$$p_e(e) = \frac{1}{\sqrt{2}\sigma_e} e^{-\frac{\sqrt{2}|e|}{\sigma_e}}$$

$$p_e(e) = \frac{\lambda}{2} e^{-\lambda|e|}$$

Lossy predictive coding

- Error-free encoding of images seldom results in more than 3:1 reduction in data
- A lossy predictive coding model
 - the prediction in the encoder and decoder must be equivalent (same)
 - placing encoder's predictor within a feedback loop



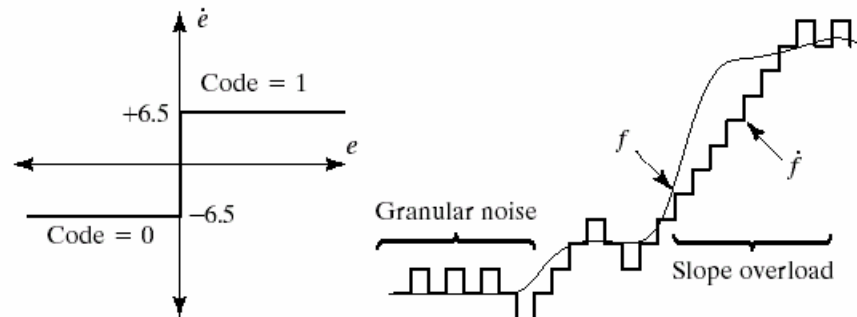
This closed loop configuration prevents error buildup at decoder's output



Delta modulation (DM)

- DM : $\hat{f}_n = \alpha \hat{f}_{n-1}$ $\dot{e}_n = \begin{cases} +\zeta & \text{for } e_n > 0 \\ -\zeta & \text{otherwise} \end{cases}$
 - the resulting bit rate is 1 bit/pixel
- Slope overload effect
 - ζ is too small to represent input's large changes, lead to blurred object edges
- Granular noise effect
 - ζ is too large to represent input's small change, lead to grainy or noisy surfaces

Delta modulation (cont)



a b
c

FIGURE 8.22 An example of delta modulation.

Input		Encoder				Decoder		Error
n	f	\hat{f}	e	\dot{e}	\dot{f}	\hat{f}	$\dot{\hat{f}}$	$[f - \dot{\hat{f}}]$
0	14	—	—	—	14.0	—	14.0	0.0
1	15	14.0	1.0	6.5	20.5	14.0	20.5	-5.5
2	14	20.5	-6.5	-6.5	14.0	20.5	14.0	0.0
3	15	14.0	1.0	6.5	20.5	14.0	20.5	-5.5
·	·	·	·	·	·	·	·	·
·	·	·	·	·	·	·	·	·
14	29	20.5	8.5	6.5	27.0	20.5	27.0	2.0
15	37	27.0	10.0	6.5	33.5	27.0	33.5	3.5
16	47	33.5	13.5	6.5	40.0	33.5	40.0	7.0
17	62	40.0	22.0	6.5	46.5	40.0	46.5	15.5
18	75	46.5	28.5	6.5	53.0	46.5	53.0	22.0
19	77	53.0	24.0	6.5	59.6	53.0	59.6	17.5
·	·	·	·	·	·	·	·	·
·	·	·	·	·	·	·	·	·

Optimal predictors

- Minimize encoder's mean-square prediction error

$$E\{e_n^2\} = E\{[f_n - \hat{f}_n]^2\}$$

with $\dot{f}_n = \bar{e}_n + \hat{f}_n \approx e_n + \hat{f}_n = f_n$ $\hat{f}_n = \sum_{i=1}^m \alpha_i f_{n-i}$ DPCM

\Rightarrow select m prediction coefficients to minimize

$$E\{e_n^2\} = E\left\{\left[f_n - \sum_{i=1}^m \alpha_i f_{n-i}\right]^2\right\}$$

- The solution : $\mathbf{a} = \mathbf{R}^{-1} \mathbf{r}$

$$\mathbf{r} = \begin{bmatrix} E\{f_n f_{n-1}\} \\ E\{f_n f_{n-2}\} \\ . \\ . \\ E\{f_n f_{n-m}\} \end{bmatrix} \quad \mathbf{R} = \begin{bmatrix} E\{f_{n-1} f_{n-1}\} & E\{f_{n-1} f_{n-2}\} & . & . & E\{f_{n-1} f_{n-m}\} \\ E\{f_{n-2} f_{n-1}\} & E\{f_{n-2} f_{n-2}\} & . & . & . \\ . & . & . & . & . \\ . & . & . & . & . \\ E\{f_{n-m} f_{n-1}\} & . & . & . & E\{f_{n-m} f_{n-m}\} \end{bmatrix} \quad \mathbf{a} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ . \\ . \\ \alpha_m \end{bmatrix}$$

Optimal predictors (cont)

- Variance of prediction error

$$\sigma_e^2 = \sigma^2 - \mathbf{a}^T \mathbf{r} = \sigma^2 - \sum_{i=1}^m E\{f_n f_{n-i}\} \alpha_i$$

- For a 2-D Markov source with separable autocorrelation function $E\{f(x, y) f(x-i, y-j)\} = \sigma^2 \rho_v^i \rho_h^j$

and 4-th order linear predictor

2	1
3	X
4	

$$\hat{f}(x, y) = \alpha_1 f(x, y-1) + \alpha_2 f(x-1, y-1) + \alpha_3 f(x-1, y) + \alpha_4 f(x-1, y+1)$$

→ $\alpha_1 = \rho_h, \alpha_2 = -\rho_v \rho_h, \alpha_3 = \rho_v, \alpha_4 = 0$

- We generally have

$$\sum_{i=1}^m \alpha_i \leq 1.0$$

to reduce the DPCM decoder's susceptibility to transmission noise



Examples of predictors

- Four examples :

$$\hat{f}(x, y) = 0.97 f(x, y - 1)$$

$$\hat{f}(x, y) = 0.5 f(x, y - 1) + 0.5 f(x - 1, y)$$

$$\hat{f}(x, y) = 0.75 f(x, y - 1) + 0.75 f(x - 1, y) - 0.5 f(x - 1, y - 1)$$

$$\hat{f}(x, y) = \begin{cases} 0.97 f(x, y - 1) & \text{if } \Delta h \leq \Delta v \\ 0.97 f(x - 1, y) & \text{otherwise} \end{cases}$$

Adaptive with local measure of directional property

- The visually perceptible error decreases as the order of the predictor increases



Optimal quantization

- The staircase quantization function is an odd function that can be described by the decision (s_i) and reconstruction (t_i) levels
- Quantizer design is to select the best s_i and t_i for a particular optimization criterion and input probability density function $p(s)$

L-level Lloyd-Max quantizer

- Optimal in the mean-square error sense

$$\int_{s_{i-1}}^{s_i} (s - t_i) p(s) ds = 0 \quad i = 1, 2, \dots, \frac{L}{2}$$

t_i is the centroid of each decision interval

$$s_i = \begin{cases} 0 & i = 0 \\ \frac{t_i + t_{i+1}}{2} & i = 1, 2, \dots, \frac{L}{2} - 1 \\ \infty & i = \frac{L}{2} \end{cases}$$

Decision levels are halfway of the reconstruction levels

Non-uniform quantizer

$$s_{-i} = -s_i, \quad t_{-i} = -t_i$$

How about an optimal uniform quantizer ?

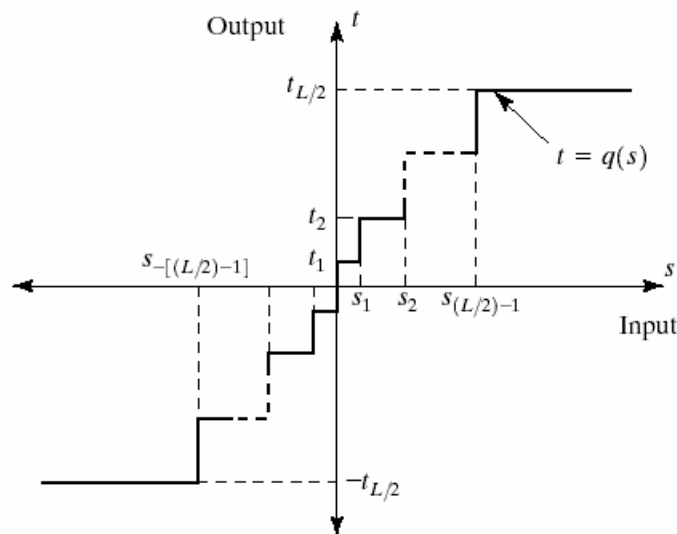
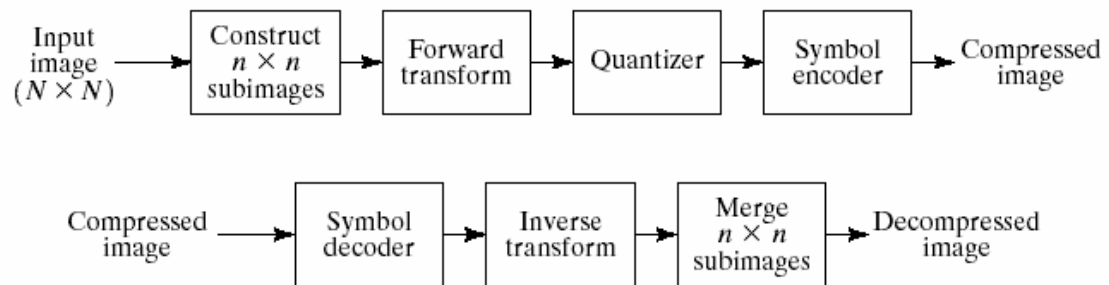


FIGURE 8.25 A typical quantization function.

Transform coding

- Subimage decomposition
- Transformation
 - Decorrelate the pixels of each subimage
 - Energy packing
- Quantization
 - Eliminate coefficients that carry the least information
- coding



a
b

FIGURE 8.28 A transform coding system: (a) encoder; (b) decoder.



Transform selection

- Requirements
 - orthonormal or unitary forward and inverse transformation kernels
 - basis function or basis images
 - separable and symmetric for the kernels
- Types
 - DFT
 - DCT
 - WHT (Walsh-Hadamard transform)
- Compression is achieved during the quantization of the transformed coefficients (not during the transformation step)



WHT

- The summation in the exponent is performed in modulo 2 arithmetic
- $b_k(z)$ is the k th bit (from right to left) in the binary representation of z
- $N = 2^m$

$$T(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) g(x, y, u, v)$$

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) h(x, y, u, v)$$

$$g(x, y, u, v) = h(x, y, u, v) = \frac{1}{N} (-1)^{\sum_{i=0}^{m-1} [b_i(x) p_i(u) + b_i(y) p_i(v)]}$$

DCT

$$\blacksquare \quad g(x, y, u, v) = h(x, y, u, v) = \alpha(u)\alpha(v) \cos\left[\frac{(2x+1)u\pi}{2N}\right] \cos\left[\frac{(2y+1)v\pi}{2N}\right]$$

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}} & \text{for } u = 0 \\ \sqrt{\frac{2}{N}} & \text{for } u = 1, 2, \dots, N-1 \end{cases}$$

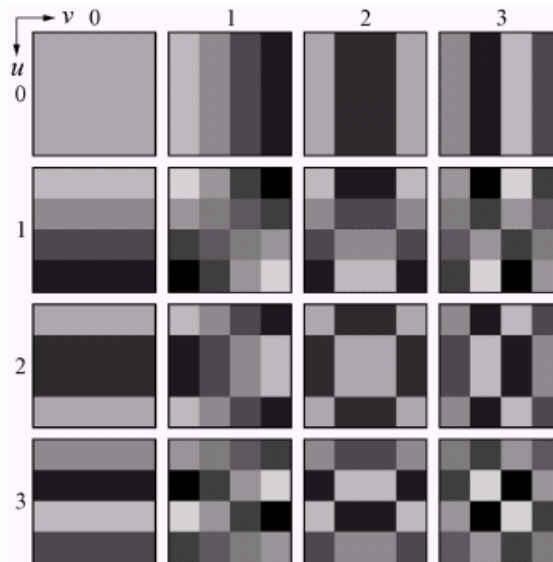
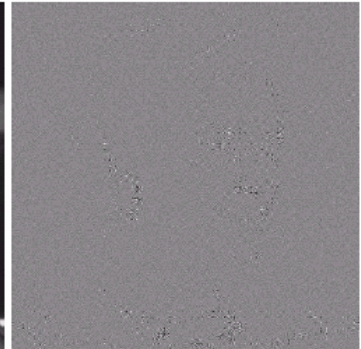
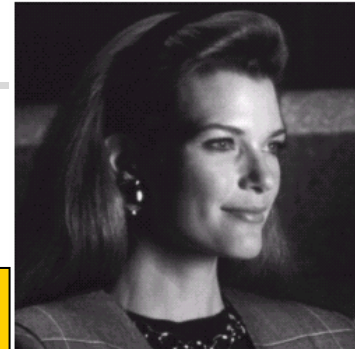


FIGURE 8.30 Discrete-cosine basis functions for $N = 4$. The origin of each block is at its top left.

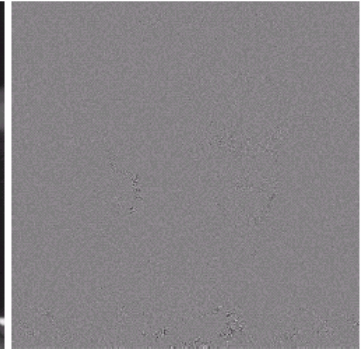
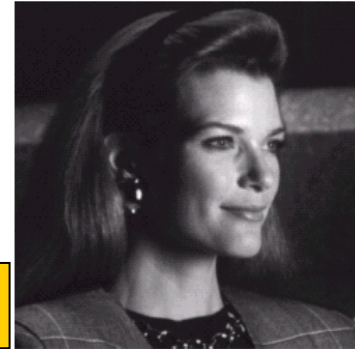
Approximations using DFT, WHT, and DCT

- Truncating 50% of the transformed coefficients based on maximum magnitudes
- The RMS error values are 1.28 (DFT), 0.86 (WHT), and 0.68 (DCT).

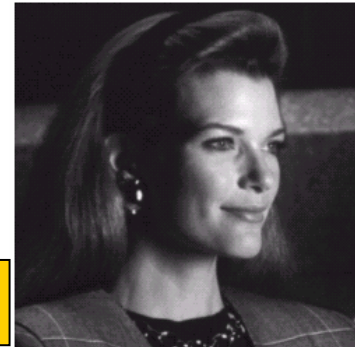
DFT



WHT



DCT





Basis images

- A linear combination of n^2 basis images : \mathbf{H}_{uv}

$$\mathbf{F} = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} T(u, v) \mathbf{H}_{uv}$$

$$\mathbf{H}_{uv} = \begin{bmatrix} h(0,0,u,v) & h(0,1,u,v) & \cdot & \cdot & h(0,n-1,u,v) \\ h(1,0,u,v) & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ h(n-1,0,u,v) & h(n-1,1,u,v) & \cdot & \cdot & h(n-1,n-1,u,v) \end{bmatrix}$$



Transform coefficient masking

- Masking function : $\gamma(u, v)$


$$\hat{\mathbf{F}} = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} \gamma(u, v) T(u, v) \mathbf{H}_{uv}$$

- Optimizing masking function :

$$\begin{aligned} e_{ms} &= E\{\|\mathbf{F} - \hat{\mathbf{F}}\|^2\} = E\left\{\left\|\sum_{u=0}^{n-1} \sum_{v=0}^{n-1} T(u, v) \mathbf{H}_{uv} - \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} \gamma(u, v) T(u, v) \mathbf{H}_{uv}\right\|^2\right\} \\ &= \dots = \sum_{u=0}^{n-1} \sum_{v=0}^{n-1} \sigma_{T(u, v)}^2 [1 - \gamma(u, v)] \end{aligned}$$

The total mean-square approximation error is the sum of the variances of the discarded transform coefficients

Transformations that pack the most information into the fewest coefficients provide the best approximation

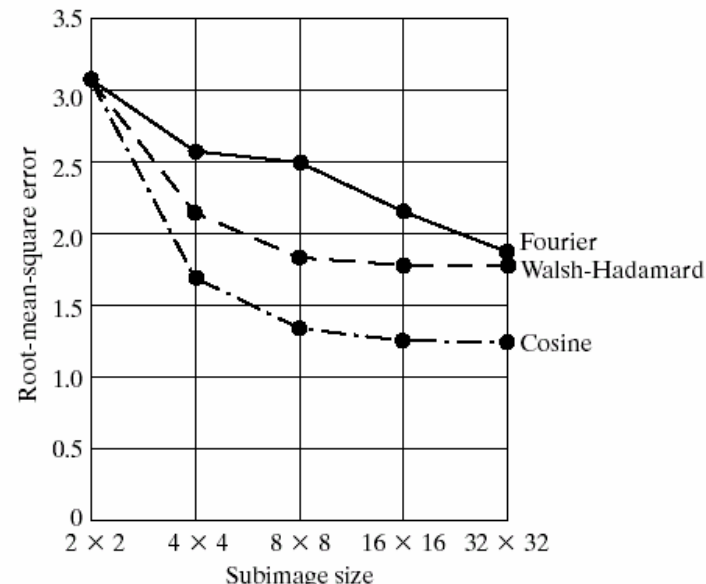


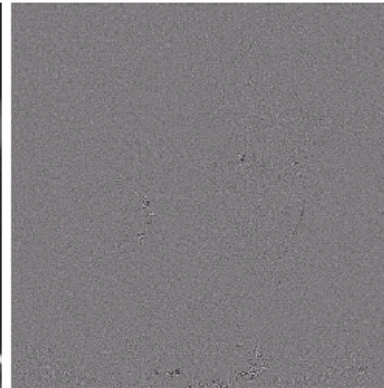
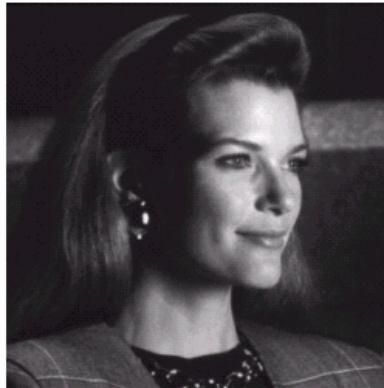
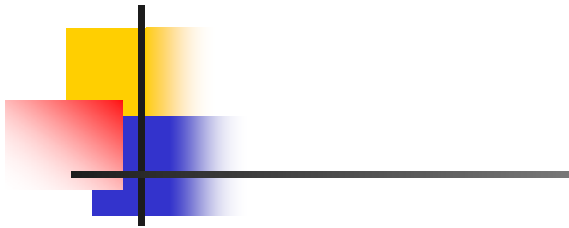
KLT or
DCT

Subimage size selection

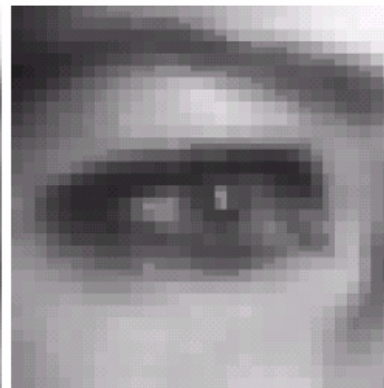
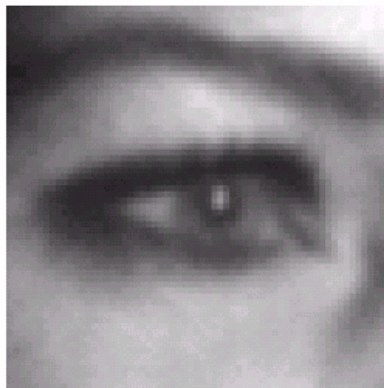
- The most popular subimage sizes are 8×8 and 16×16 .
- A comparison of $n \times n$ subimage for $n=2,4,8,16$
 - Truncating 75% of resulting coefficients
 - Curves of WHT and DCT flatten as size of subimage becomes greater than 8×8 , while FT does not
 - Block effect decreases when subimage size increases

FIGURE 8.33
Reconstruction
error versus
subimage size.



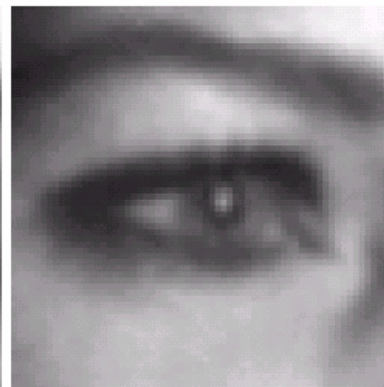
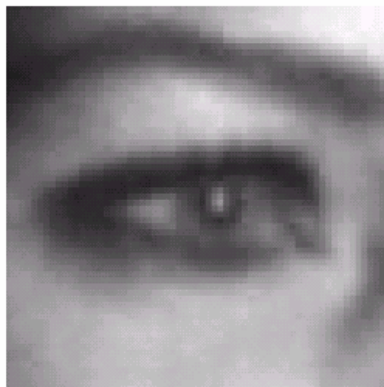


original



2x2
subimage

4x4
subimage



8x8
subimage



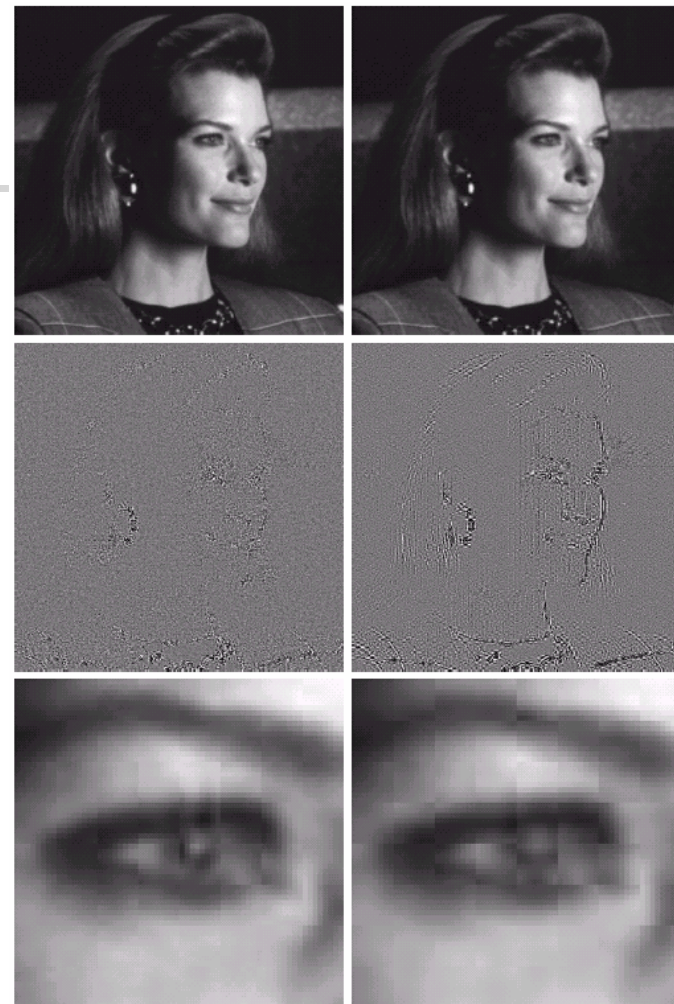
Zonal vs. threshold coding

- The coefficients are retained based on
 - Maximum variance : zonal coding
 - Maximum magnitude : threshold coding
- Zonal coding
 - Each DCT coefficient is considered as a random variable whose distribution could be computed over the ensemble of all transformed subimages
 - The variances can also be based on an assumed image model (e.g., a Markov autocorrelation function)
 - Coefficients of maximum variances are usually located around the origin
- Threshold coding
 - Select coefficients which have the largest magnitudes
 - Causing far less error than the zonal coding result

Bit allocation for zonal coding

- The retained coefficients are allocated bits according to :
 - The dc coefficient is modeled by a Rayleigh density function
 - The ac coefficients are often modeled by a Laplacian or Gaussian density
- The number of bits is made proportional to $\log_2 \sigma_{T(u,v)}^2$

The information content of a Gaussian random variable is proportional to $\log_2(\sigma^2 / D)$



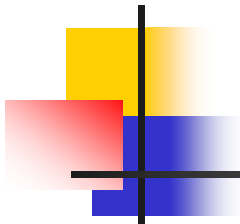
Threshold
coding

Zonal
coding



Bit allocation in threshold coding

- The transform coefficients of largest magnitude make the most significant contribution to reconstructed subimage quality
- Inherently adaptive in the sense that the location of the retained transform coefficients vary from one subimage to another
- Retained coefficients are recorded in a 1-D zigzag ordering pattern
 - The mask pattern is run-length coded



a b
c d

FIGURE 8.36 A typical (a) zonal mask, (b) zonal bit allocation, (c) threshold mask, and (d) thresholded coefficient ordering sequence. Shading highlights the coefficients that are retained.

Zonal mask

1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	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

1	1	0	1	1	0	0	0
1	1	1	1	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	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Bit allocation

8	7	6	4	3	2	1	0
7	6	5	4	3	2	1	0
6	5	4	3	3	1	1	0
4	4	3	3	2	1	0	0
3	3	3	2	1	1	0	0
2	2	1	1	1	0	0	0
1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0

0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

Thresholded and retained coefficients

Zigzag ordering pattern

Bit allocation in threshold coding (cont)

- Thresholding the transform coefficients -- the threshold can be varied as a function of the location of each coefficient within the subimage
 - Result in a variable code rate, but offer the advantage that thresholding and quantization can be combined by replacing the masking (i.e., $\gamma(u,v)T(u,v)$) with

$$\hat{T}(u,v) = \text{round}\left[\frac{T(u,v)}{Z(u,v)}\right]$$

$Z(u,v)$ is the transform normalization array

- $\dot{T}(u,v) = \hat{T}(u,v)Z(u,v)$
- The element $Z(u,v)$ can be scaled to achieve a variety of compression levels

Recovered or denormalized value

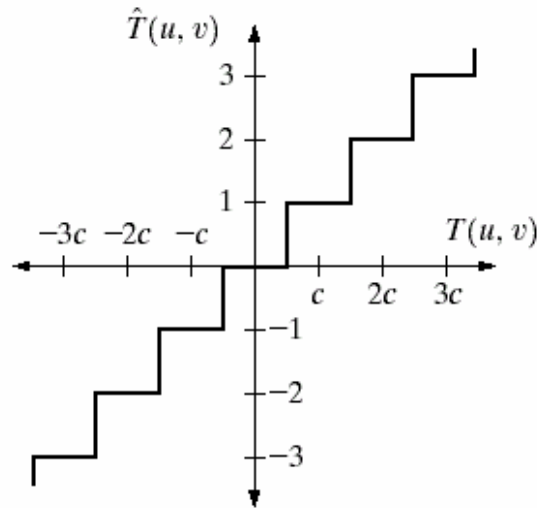
Normalization (quantization) matrix

- Used in JPEG standard

a b

FIGURE 8.37

(a) A threshold coding quantization curve [see Eq. (8.5-40)].
(b) A typical normalization matrix.



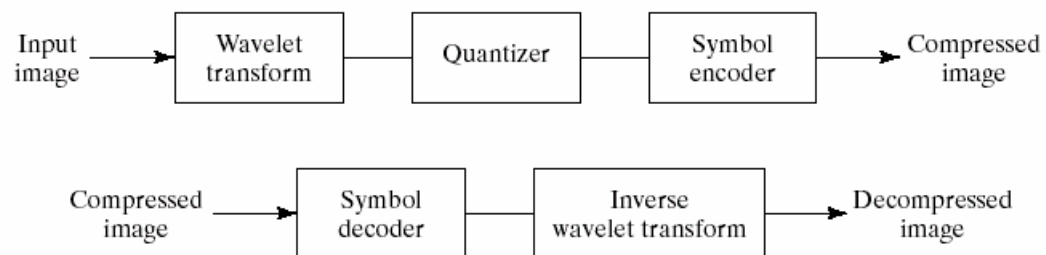
16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Wavelet coding

- Requirements in encoding an image
 - An analyzing wavelet
 - Number of decomposition levels

a
b

FIGURE 8.39 A wavelet coding system:
(a) encoder;
(b) decoder.



- The digital wavelets transform (DWT) converts a large portion of the original image to horizontal, vertical and diagonal decomposition coefficients with zero mean and Laplacian-like distributions.

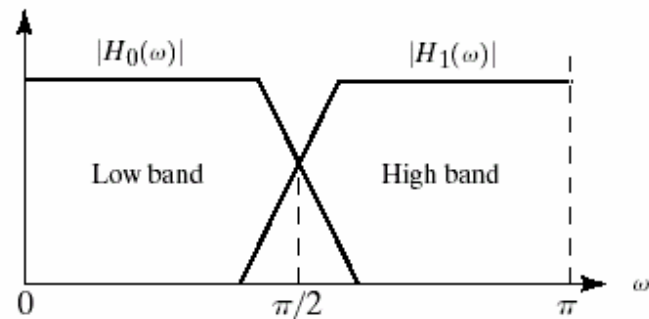
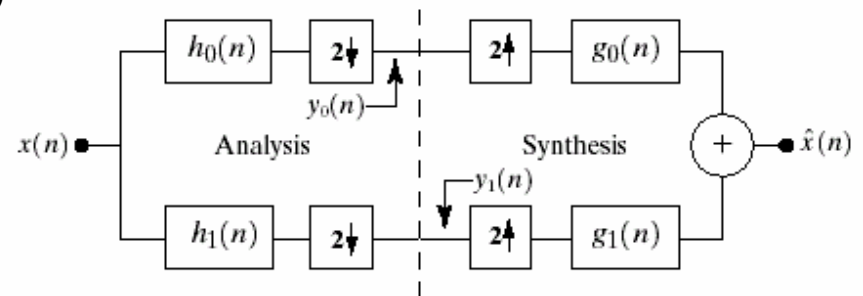


Wavelet coding (cont)

- Many computed coefficients can be quantized and coded to minimize intercoefficient and coding redundancy
- The quantization can be adapted to exploit any positional correlation across different decomposition levels
- Subdivision of the original image is unnecessary
 - Eliminating the blocking artifact that characterizes DCT-based approximations at high compression ratios

1-D Subband decomposition

- Analysis filter :
 - Low-pass : approximation $h_0(n)$
 - High-pass : detail $h_1(n)$
- Synthesis filter :
 - Low-pass : $g_0(n)$
 - High-pass : $g_1(n)$



2-D Subband image coding – 4 filter bank

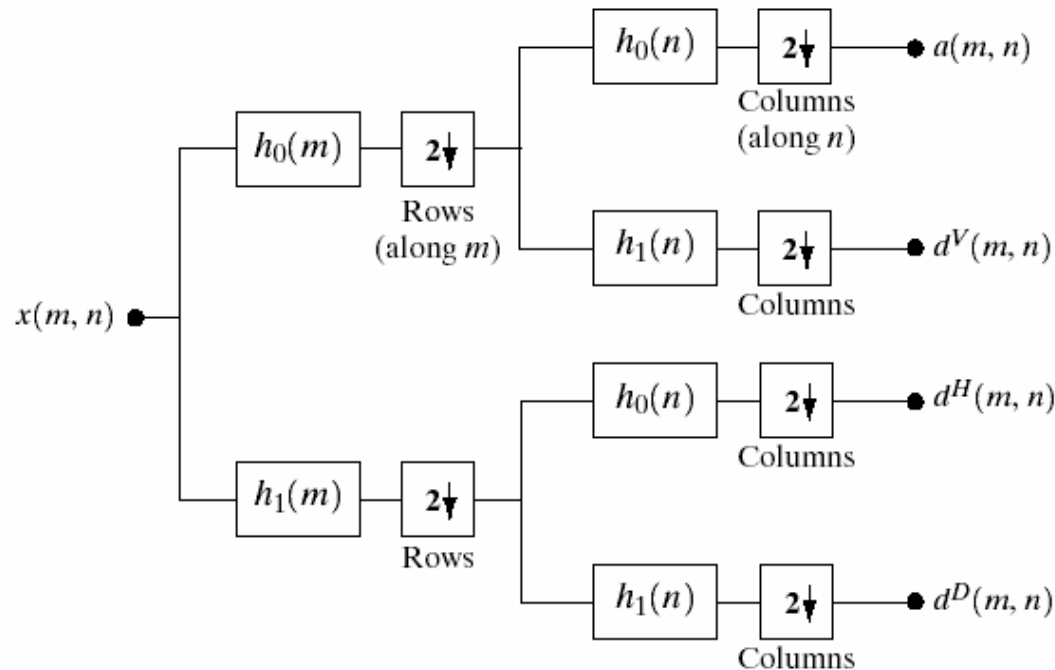
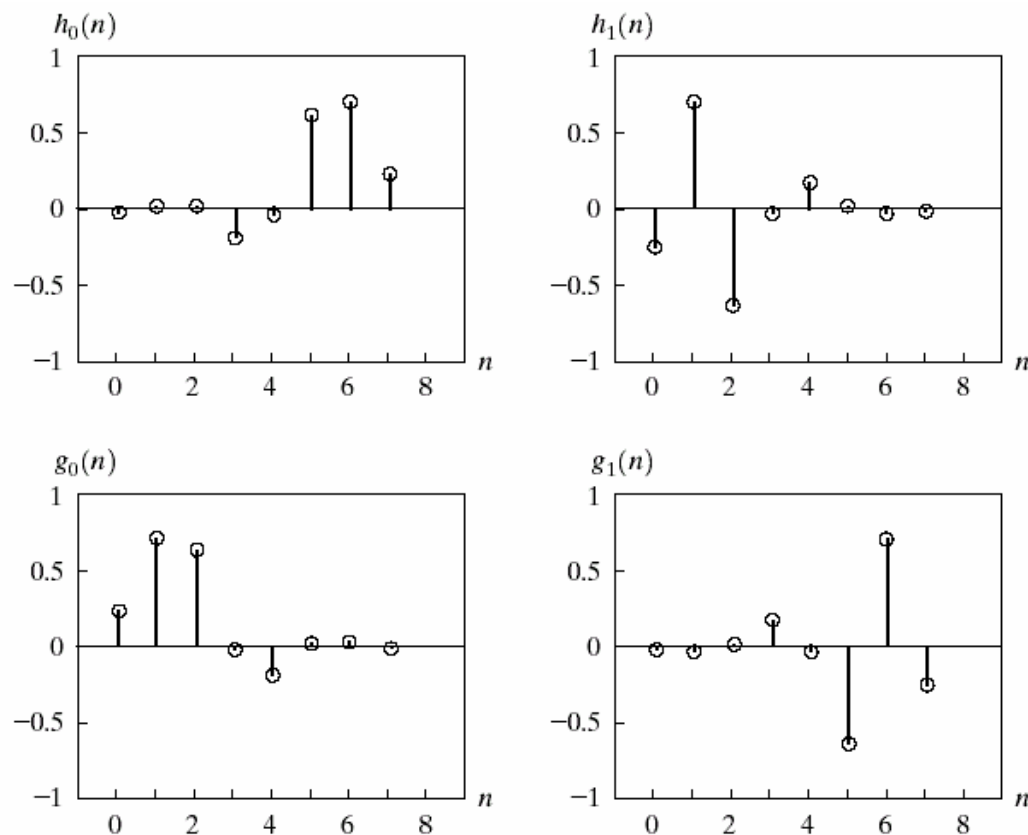


FIGURE 7.5 A two-dimensional, four-band filter bank for subband image coding.

An example of Daubechies orthonormal filters

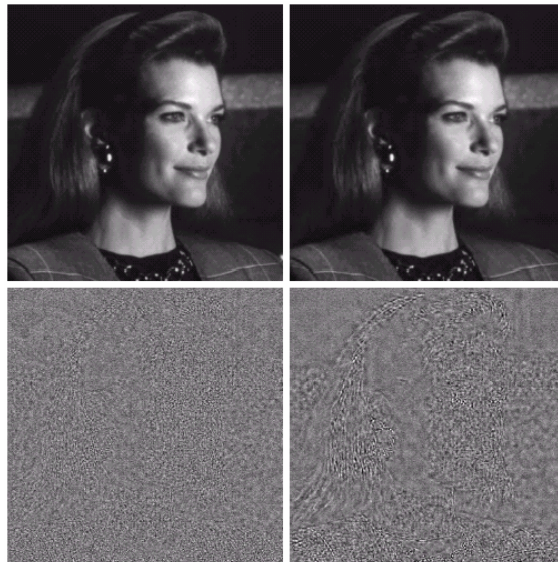
FIGURE 7.6 The impulse responses of four 8-tap Daubechies orthonormal filters.



Comparison between DCT-based and DWT-based coding

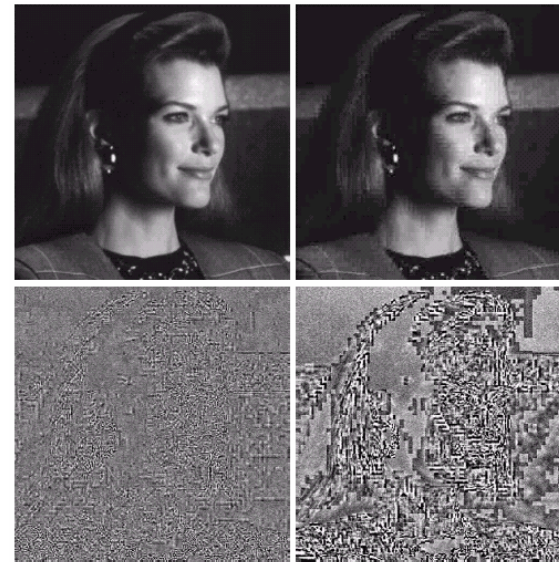
- A noticeable decrease of error in the wavelet coding results (based on compression ratios of 34:1 and 67:1)

DWT-based

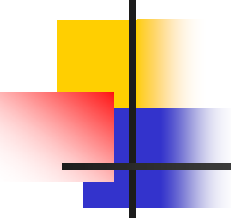


Rms error :
2.29, 2.96

DCT-based



Rms error :
3.42, 6.33

- 
- Based on compression ratios of 108:1 and 167:1
 - Even the 167:1 DWT result (rms=4.73) is better than 67:1 DCT result (rms=6.33)
 - At more than twice of compression ratio, the wavelet-based reconstruction has only 75% of the error of the DCT-based result



Rms error :
3.72, 4.73



Wavelet selection

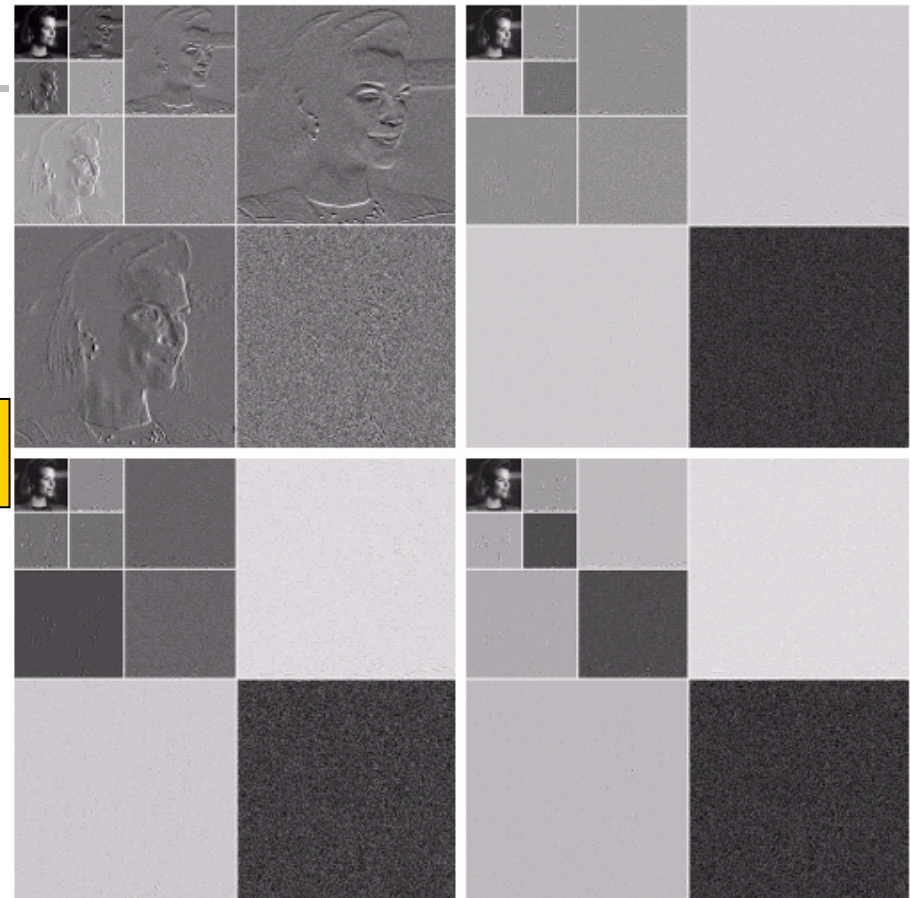
- The wavelet selection affects all aspects of wavelet coding system design and performance
 - Computational complexity of the transform
 - System's ability to compress and reconstruct images of acceptable error
 - Include the decomposition filters and reconstruction filters
 - Useful analysis property : number of zero moments
 - Important synthesis property : smoothness of reconstruction
- The most widely used expansion functions
 - Daubechies wavelets
 - Bi-orthogonal wavelets

Allows filters with binary coefficients (numbers of the form $k/2^a$)

Comparison between different wavelets

- (1) Harr wavelets : the simplest
- (2) Daubechies wavelets : the most popular imaging wavelets
- (3) Symlets : an extension of Daubechies with increased symmetry
- (4) Cohen-Daubechies-Feauveau wavelets : biorthogonal wavelets

1,2
3,4



a b
c d

FIGURE 8.42 Wavelet transforms of Fig. 8.23 with respect to (a) Haar wavelets, (b) Daubechies wavelets, (c) symlets, and (d) Cohen-Daubechies-Feauveau biorthogonal wavelets.

Comparison between different wavelets (cont)

- Comparisons in number of operations required
 - As the computational complexity increases, the information packing ability does as well

Wavelet	Filter Taps (Scaling + Wavelet)	Zeroed Coefficients
Haar (see Ex. 7.10)	2 + 2	46%
Daubechies (see Fig. 7.6)	8 + 8	51%
Symlet (see Fig. 7.24)	8 + 8	51%
Biorthogonal (see Fig. 7.37)	17 + 11	55%

TABLE 8.12

Wavelet transform filter taps and zeroed coefficients when truncating the transforms in Fig. 8.42 below 1.5.

For analysis and reconstruction filters, respectively



Other issues

- The number of decomposition levels required
 - The initial decompositions are responsible for the majority of the data compression (3 levels is enough)

Scales and Filter Bank Iterations	Approximation Coefficient Image	Truncated Coefficients (%)	Reconstruction Error (rms)
1	256 × 256	75%	1.93
2	128 × 128	93%	2.69
3	64 × 64	97%	3.12
4	32 × 32	98%	3.25
5	16 × 16	98%	3.27

TABLE 8.13
Decomposition level impact on wavelet coding the 512 × 512 image of Fig. 8.23.

- Quantizer design
 - Introduce an enlarged quantization interval around zero (i.e., the dead zone)
 - Adapting the size of the quantization interval from scale to scale
 - Quantization of coefficients at more decomposition levels impacts larger areas of the reconstructed image



Binary image compression standards

- Most of the standards are issued by
 - International Standardization Organization (ISO)
 - Consultative Committee of the International Telephone and Telegraph (CCITT)
- CCITT Group 3 and 4 are for binary image compression
 - Originally designed as facsimile (FAX) coding method
 - G3 : Nonadaptive, 1-D run-length coding
 - G4 : a simplified or streamlined version of G3, only 2-D coding
 - The coding approach is quite similar to the RAC method
- Joint Bilevel Imaging Group (JBIG)
 - A joint committee of CCITT and ISO
 - Proposed JBIG1: adaptive arithmetic compression technique (the best average and worst-case available)
 - Proposed JBIG2 : achieve compressions 2 to 4 times greater than JBIG1



Continuous-tone image compression -- JPEG

- JPEG, wavelet-based JPEG-2000, JPEG-LS
- JPEG – define three coding system
 - Lossy baseline coding system
 - Extended coding system – for greater compression, higher precision, or progressive reconstruction applications
 - Lossless independent coding system
 - A product or system must include support for the baseline system
- Baseline system
 - Input, output : 8 bits
 - Quantized DCT values : 11 bits
 - Subdivided into blocks of 8x8 pixels for encoding
 - Pixels are level shifted by subtracting 128 graylevels



JPEG (cont)

- DCT-transformed
- Quantized by using the quantization or normalization matrix
- The DC DCT coefficients is difference coded relative to the DC coefficient of the previous block
- The non-zero AC coefficients are coded by using a VLC that defines the coefficient's value and number of preceding zeros
 - The user is free to construct custom tables and/or arrays, which may be adapted to the characteristics of the image being compressed
- Add a special EOB (end of block) code behind the last non-zero AC coefficient



Huffman Coding for the DC and AC coefficients

- VLC = base code (category code) + value code
- An example to encode DC = -9
 - Category 4 : -15,...,-8,8,...,15 (base code = 101)
 - Total length = 7 \rightarrow value code length = 4
 - For a category K , additional K bits are needed as the value code and computed as either the K LSBs (positive value) or the K LSBs of 1's complement of its absolute value (negative value) : $K=4$, value code = 0110
 - Complete code : 101-0110
- An example to encode AC = (0,-3)
 - length of "0" run : 0, AC coefficient value : -3
 - AC = -3 \rightarrow category = 2 \rightarrow 0/2 (Table 8.19) \rightarrow base code = 01
 - Value code = 00
 - Complete code = 0100

A JPEG coding example

original

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

Level
shifted

-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

DCT
transformed

-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



quantized

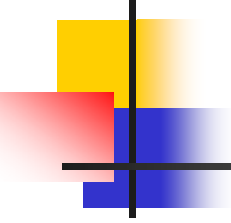
-26	-3	-6	2	2	0	0	0
1	-2	-4	0	0	0	0	0
-3	1	5	-1	-1	0	0	0
-4	1	2	-1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

1-D Zigzag scanning : [-26 -3 1 -3 -2 -6 2 -4 1 -4 1 1 5 0 2 0 0 -1 2 0 0 0 0 0 -1 -1 EOB]

Symbols to be coded : (DPCM =-9, assumed),(0,-3),(0,1),(0,-3),(0,-2),(0,-6),(0,2),(0,-4),(0,1),(0,-4),(0,1),(0,1),(0,5),(1,2),(2,-1),(0,2),(5,-1),(0,-1),EOB

Final codes :

1010110,0100,001,0100,0101,100001,0110,100011,001,100011,001,001,100101,1110011
0,110110,0110,11110100,000,1010



Reconstructed after VLD
and requantization

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

residual

-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

TABLE 8.17
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

TABLE 8.18
JPEG default DC
code (luminance).

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

Run/ Category	Base Code	Length	Run/ Category	Base Code	Length
0/0	1010 (= EOB)	4			
0/1	00	3	8/1	11111010	9
0/2	01	4	8/2	11111111000000	17
0/3	100	6	8/3	111111110110111	19
0/4	1011	8	8/4	111111110111000	20
0/5	11010	10	8/5	111111110111001	21
0/6	111000	12	8/6	111111110111010	22
0/7	1111000	14	8/7	111111110111011	23
0/8	111110110	18	8/8	111111110111100	24
0/9	111111110000010	25	8/9	111111110111101	25
0/A	111111110000011	26	8/A	111111110111110	26
1/1	1100	5	9/1	11111000	10
1/2	111001	8	9/2	111111110111111	18
1/3	1111001	10	9/3	111111111000000	19
1/4	111110110	13	9/4	111111111000001	20
1/5	11111110110	16	9/5	111111111000010	21
1/6	111111110000100	22	9/6	111111111000011	22
1/7	111111110000101	23	9/7	111111111000100	23
1/8	111111110000110	24	9/8	111111111000101	24
1/9	111111110000111	25	9/9	111111111000110	25
1/A	111111110001000	26	9/A	111111111000111	26
2/1	11011	6	A/1	111111001	10
2/2	11111000	10	A/2	111111111001000	18
2/3	1111110111	13	A/3	111111111001001	19
2/4	111111110001001	20	A/4	111111111001010	20
2/5	111111110001010	21	A/5	111111111001011	21
2/6	111111110001011	22	A/6	111111111001100	22
2/7	111111110001100	23	A/7	111111111001101	23

TABLE 8.19

JPEG default AC
code (luminance)
(continues on next
page).

2/8	111111110001101	24	A/8	111111111001110	24
2/9	111111110001110	25	A/9	111111111001111	25
2/A	111111110001111	26	A/A	111111111010000	26
3/1	111010	7	B/1	111111010	10
3/2	111110111	11	B/2	111111111010001	18
3/3	1111110111	14	B/3	111111111010010	19
3/4	111111110010000	20	B/4	111111111010011	20
3/5	111111110010001	21	B/5	111111111010100	21
3/6	111111110010010	22	B/6	111111111010101	22
3/7	111111110010011	23	B/7	111111111010110	23
3/8	111111110010100	24	B/8	111111111010111	24
3/9	111111110010101	25	B/9	111111111011000	25
3/A	111111110010110	26	B/A	111111111011001	26
4/1	111011	7	C/1	111111010	11
4/2	1111111000	12	C/2	111111111011010	18
4/3	111111110010111	19	C/3	111111111011011	19
4/4	111111110011000	20	C/4	111111111011100	20
4/5	111111110011001	21	C/5	111111111011101	21
4/6	111111110011010	22	C/6	111111111011110	22
4/7	111111110011011	23	C/7	111111111011111	23
4/8	111111110011100	24	C/8	111111111100000	24
4/9	111111110011101	25	C/9	111111111100001	25
4/A	111111110011110	26	C/A	111111111100010	26



JPEG-2000

- Increased flexibility to the access of compressed data
 - Portions of a JPEG 2000 compressed image can be extracted for re-transmission, storage, display, and editing
- Procedures
 - DC level shift by subtracting 128
 - Optionally decorrelated by using reversible or non-reversible linear combination of components (e.g., RGB to YCrCb component transformation) (i.e., RGB-based or YCrCb-based JPEG-2000 coding are allowed)

$$Y_0(x, y) = 0.299I_0(x, y) + 0.587I_1(x, y) + 0.114I_2(x, y)$$

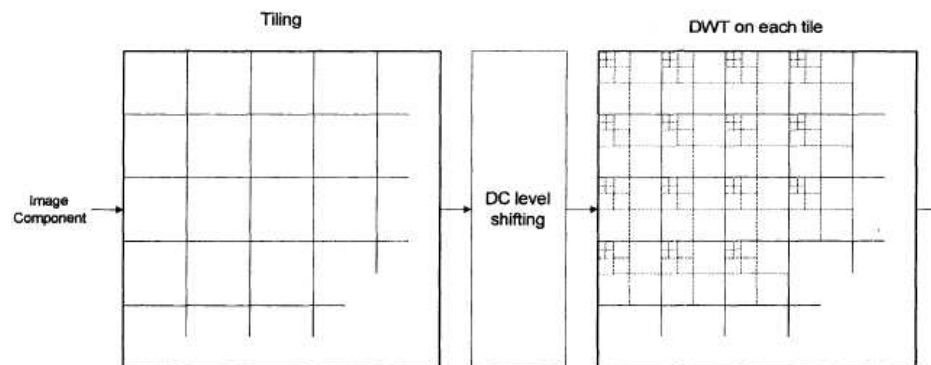
$$Y_1(x, y) = -0.16875I_0(x, y) - 0.33126I_1(x, y) + 0.5I_2(x, y)$$

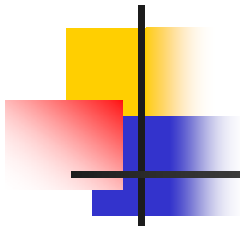
$$Y_2(x, y) = 0.5I_0(x, y) - 0.41869I_1(x, y) - 0.08131I_2(x, y)$$

Y1 and Y2 are different images highly peaked around zero

JPEG-2000 (cont)

- Optionally divided into **tiles** (rectangular arrays of pixels which can be extracted and reconstructed independently), providing a mechanism for accessing a limited region of a coded image
- 2-D DWT by using a biorthogonal **5-3** coefficient filter (lossless) or a **9-7** coefficient filter (lossy) produces a low-resolution approximation and horizontal, vertical, and diagonal frequency components (LL, HL, LH, HH bands)





Filter Tap	Highpass Wavelet Coefficient	Lowpass Scaling Coefficient
0	-1.115087052456994	0.6029490182363579
± 1	0.5912717631142470	0.2668641184428723
± 2	0.05754352622849957	-0.07822326652898785
± 3	-0.09127176311424948	-0.01686411844287495
± 4	0	0.02674875741080976

TABLE 8.20

Impulse responses of the low and highpass analysis filters for an irreversible 9-7 wavelet transform.

9-7 filter for lossy DWT

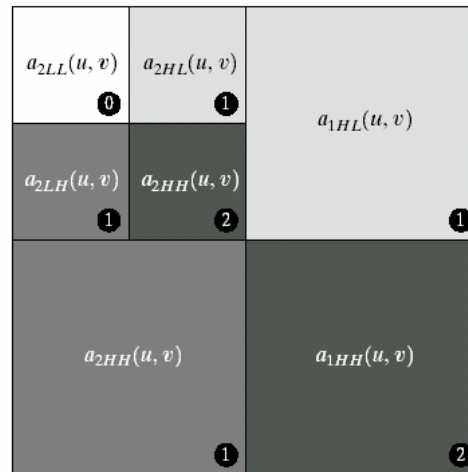
i	低通濾波器 H_0	高通濾波器 H_1
0	6/8	1
1	2/8	-1/2
2	-1/8	

5-3 filter for lossless DWT

JPEG-2000 (cont)

- A fast DWT algorithm or a **lifting-based** approach is often used
- Iterative decomposition of the approximation part to obtain :

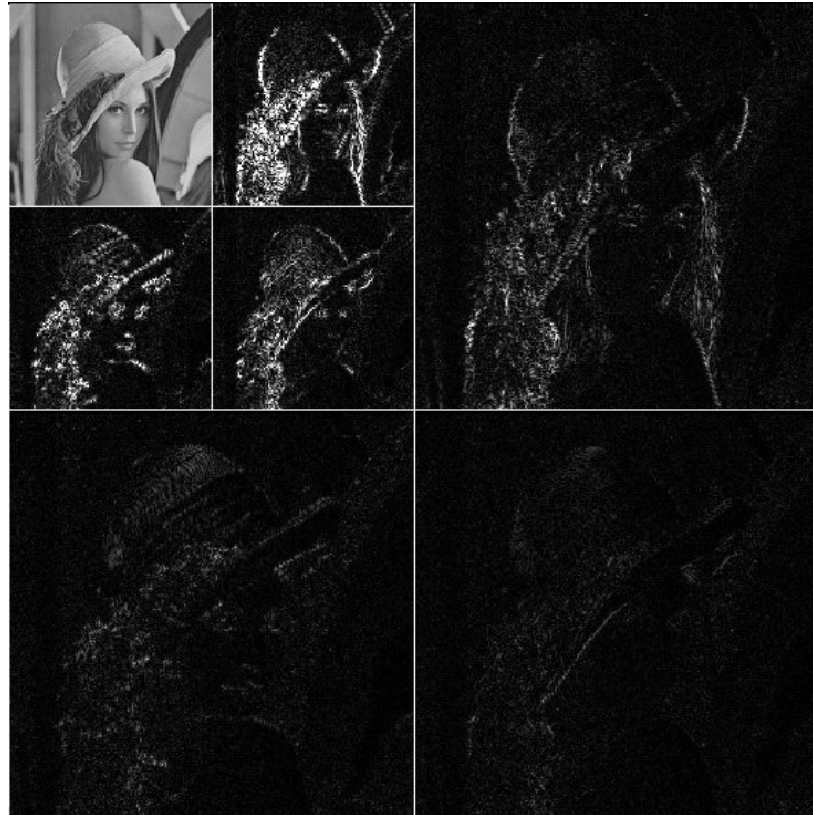
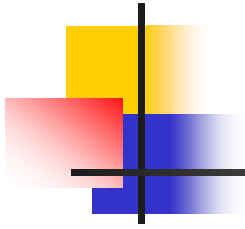
Difference from subband decomposition



The standard does not specify the number of scales (i.e., the number of decomposition levels) to be computed

FIGURE 8.46 JPEG 2000 two-scale wavelet transform tile-component coefficient notation and analysis gain.

- Important visual information is concentrated in a few coefficients





JPEG-2000 (cont)

- Coefficient quantization adapted to individual scales and subbands

$$q_b(u, v) = \text{sign}[a_b(u, v)] \cdot \text{floor}\left[\frac{|a_b(u, v)|}{\Delta_b}\right]$$

$$\text{quantization step size: } \Delta_b = 2^{R_b - \varepsilon_b} \left(1 + \frac{\mu_b}{2^{11}}\right)$$

R_b is the nominal dynamic range (in bits) of subband b , ε_b and μ_b are the number of bits allocated to the exponent and mantissa of subband's coefficients

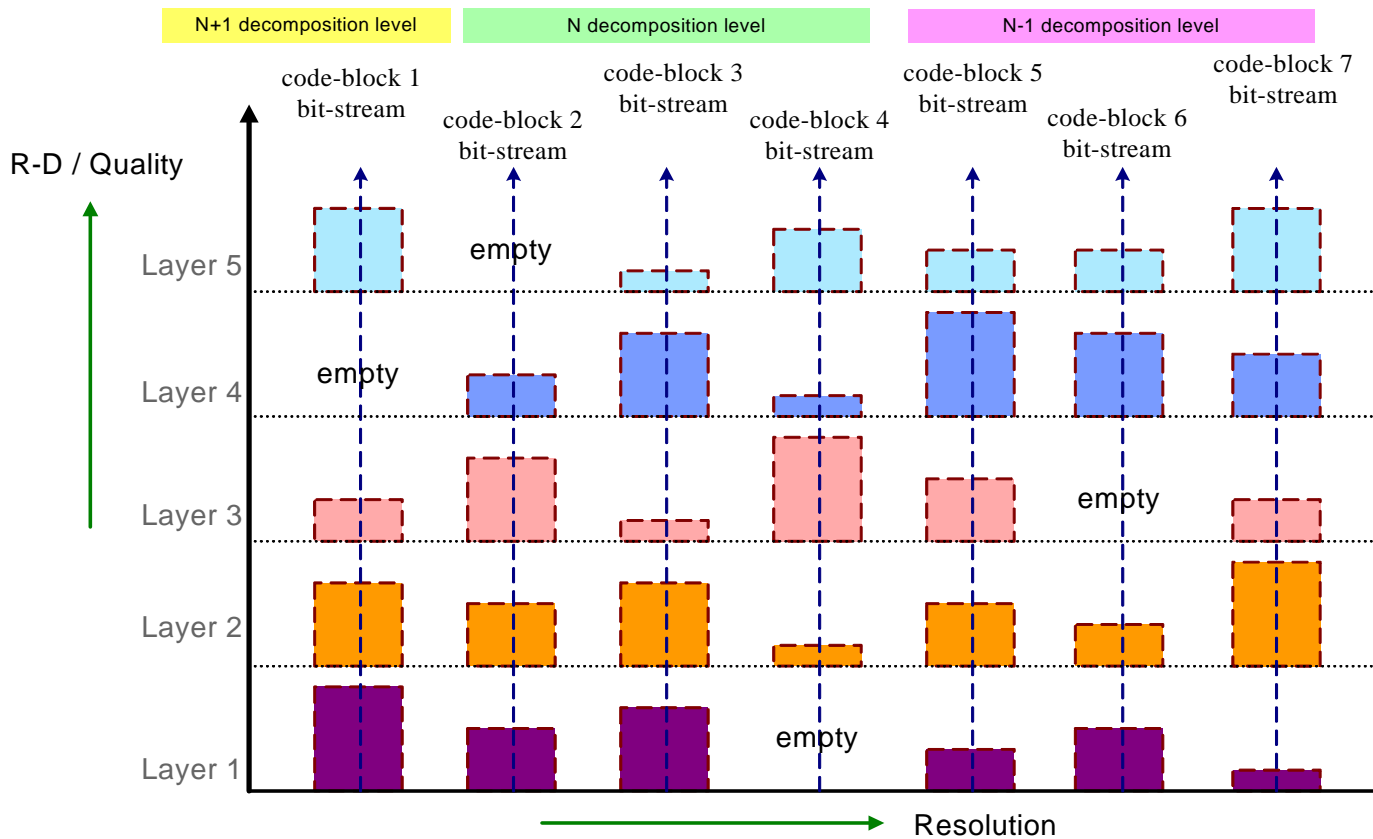
For error-free compression, $\mu_b = 0$, $R_b = \varepsilon_b$, and $\Delta_b = 1$

- The number of exponent and mantissa bits should be provided to the decoder on a subband basis, called *explicit quantization*, or for the LL subbands only, called *implicit quantization* (the remaining subbands are quantized using extrapolated LL subband parameters)



JPEG-2000 (cont)

- Quantized coefficients are arithmetically coded on a bit-plane basis
 - The coefficients are arranged into rectangular blocks called *code blocks*, which are individually coded a bit plane at a time
 - Starting from the MSB bit plane with nonzero elements to the LSB bit plane
 - Encoding by using the *context-based arithmetic coding* method
- The coding outputs from each code block are grouped to form *layers*
- The resulting layers are finally partitioned into *packets*
- The encoder can encode M_b bitplanes for a particular subband, the decoder can decode only N_b bitplanes, due to the embedded nature of the code stream





Video compression -- motion-compensated transform coding

- Standards
 - Video teleconferencing standard (H.261, H.263, H.263+, H.320, H.264)
 - Multimedia standard (MPEG-1/2/4)
- A combination of predictive coding (in temporal domain) and transform coding (in spatial domain)
- Estimate (predict) the image by simply propagating pixels in the previous frame along their motion trajectory – **motion compensation**
- The success depends on accuracy, speed, and robustness of the displacement estimator -- **motion estimation**
- Motion estimation is based on each image block (16×16 or 8×8 pixels)

Motion-compensated transform coding - encoder

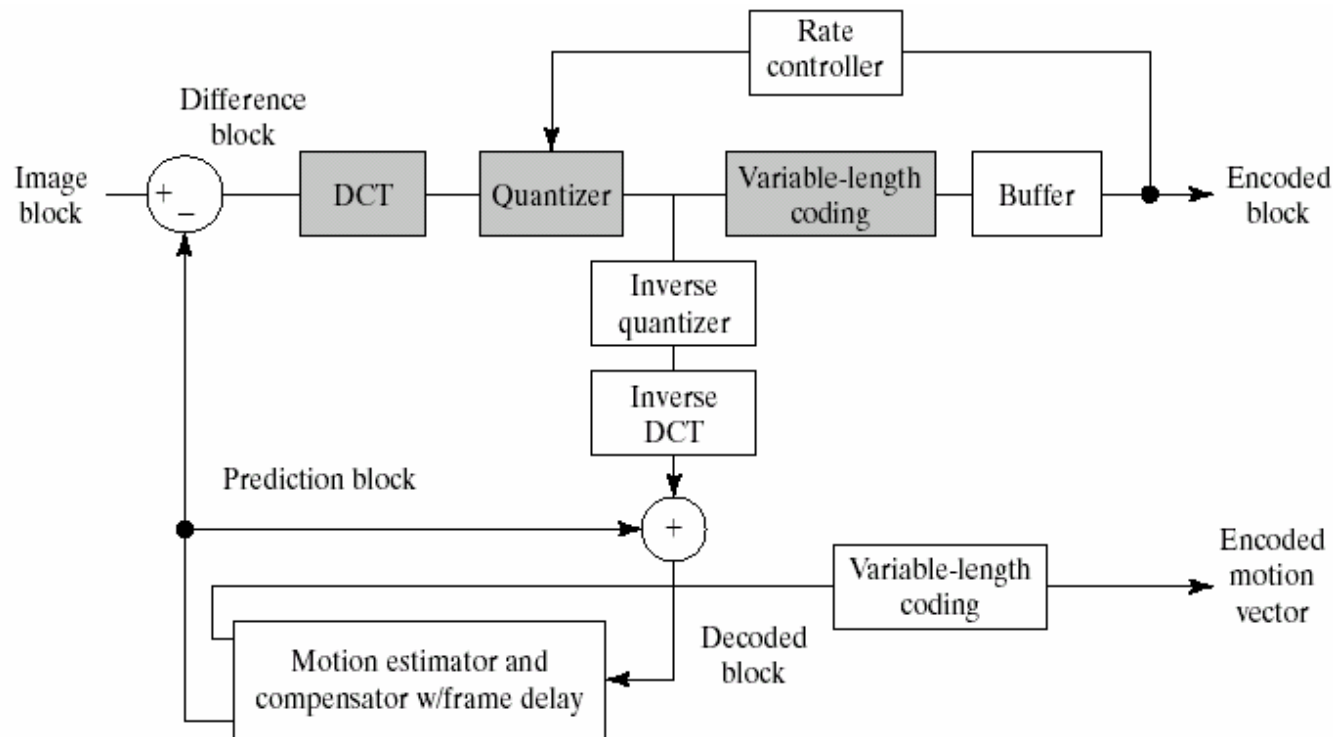
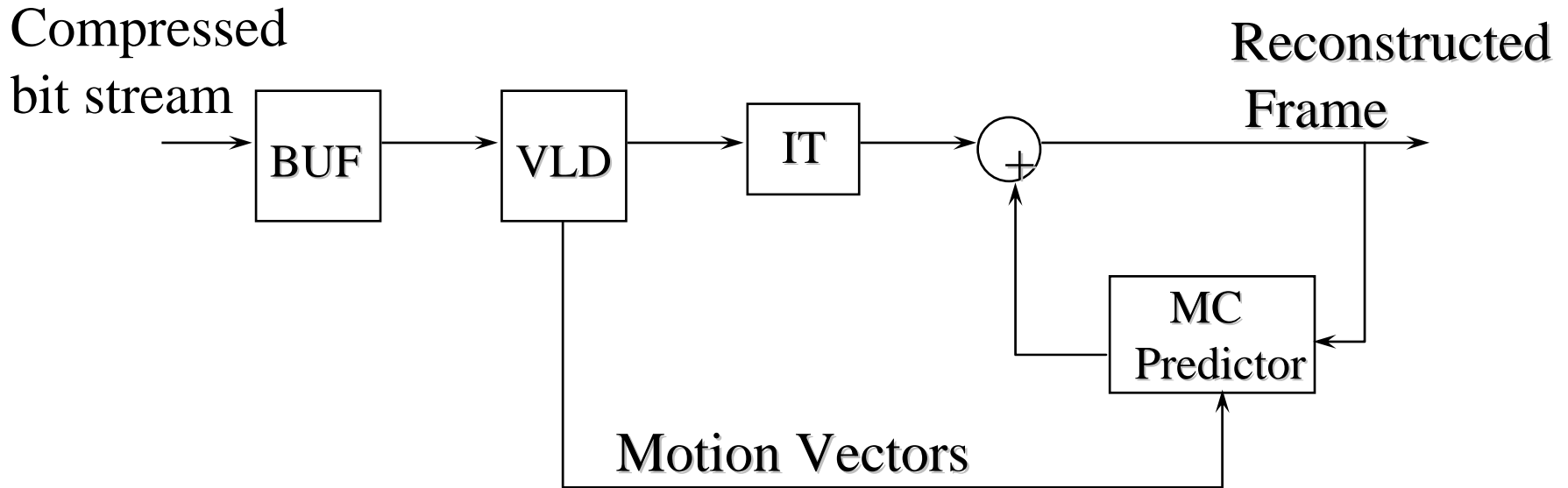


FIGURE 8.47 A basic DPCM/DCT encoder for motion compensated video compression.

Motion-compensated transform coding -- decoder



IT : inverse DCT transform

VLD : variable length decoding



2-D Motion estimation techniques

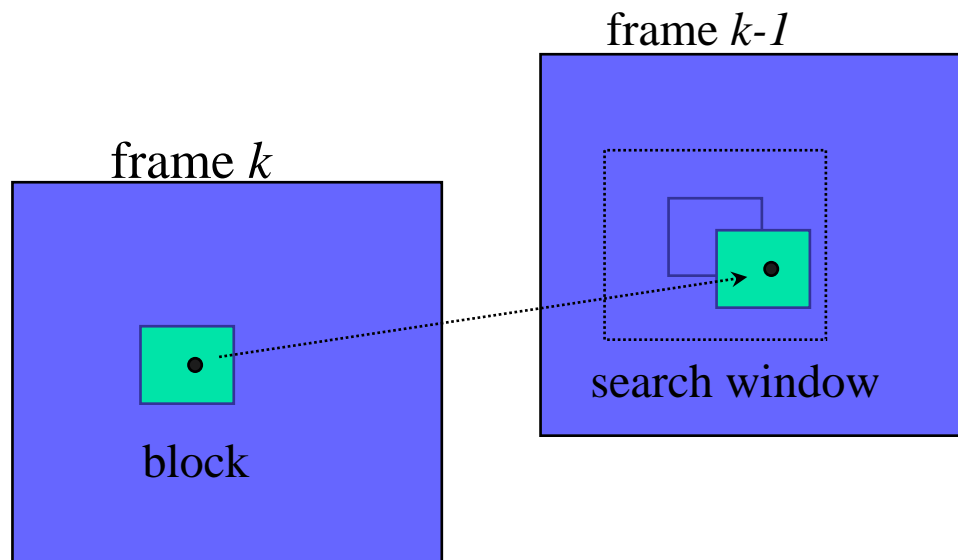
- Assumptions :
 - Rigid translational movements only
 - Uniform illumination in time and space
- Methods :
 - Optical flow approach
 - Pel-recursive approach
 - Block-matching approach
- Different from motion estimation in target tracking
 - Causal operations (forward prediction, no future frames at decoder)
 - Goal: reproduce pictures with minimum distortion, not necessarily estimate accurate motion parameters

Block-Matching (BM) Motion Estimation

- Find the best match (mv_x, mv_y) between current image block and candidates in the previous frame by minimizing the following cost

$$\sum_{(x,y) \in \text{block}} |x_n(x, y) - x_{n-1}(x + mv_x, y + mv_y)|$$

- One motion vector (MV) for each block between successive frames



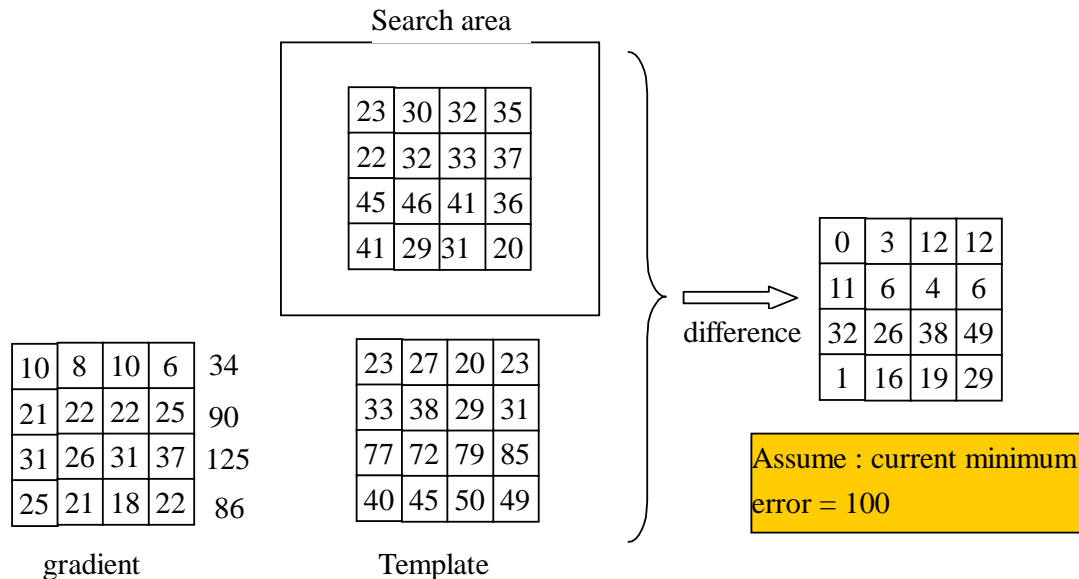


BM motion estimation techniques

- Full search method
- Fast Search (Reduced Search)
 - Usually a multistage search procedure that calculates fewer search points
 - Local-(or Sub-)optima in general
 - Evaluation – number of search points, noise immunity
- Existing Algorithms :
 - Three-step method
 - Four-step method
 - Log-search method
 - Hierarchical matching method
 - Prediction search method
 - Kalman prediction method

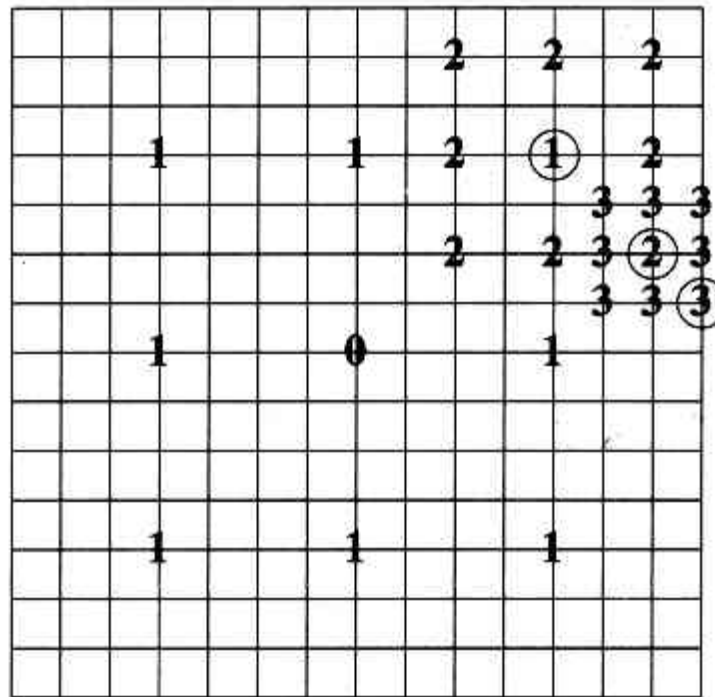
Fast full search

- Strategy of early quit to next search point in computing partial errors
- Re-ordering in computing errors
 - Conventional : raster-scanning order
 - Improvement : row or column sorting by gradient magnitudes in decreasing order
- A 3x~4x speedup with respect to traditional full search



Three steps search

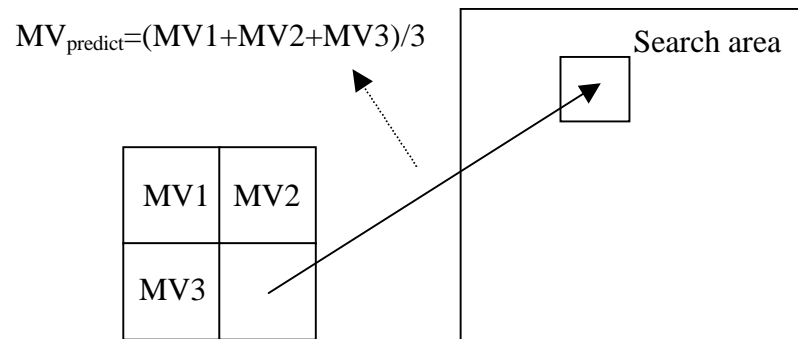
- The full search examines $15 \times 15 = 225$ positions, the three step search examines $9 + 8 + 8 = 25$ positions (assume MV range is -7 to $+7$)



Search range

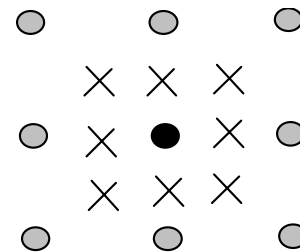
Predictive search

- Explore the similarity of MVs between adjacent macroblocks
- Obtain initially predicted MV from neighboring macroblocks by averaging
- Prediction followed by a small-area detailed search
- Significantly reduce the CPU time by 4~10X



Fractional pixel motion estimation

- Question : what happens if motion vector (mv_x, mv_y) are real numbers instead of integer numbers ?
- Answer : Find integer motion vector first and then do interpolation from surrounding pixels for refined search. Most popular case is the **half-pixel** accuracy (i.e., 1 out of 8 positions for refinement). **1/3 pixel** accuracy will be proposed in H.263L standard.
- Half-pixel accuracy often leads to a matching with less residue (high reconstruction quality), hence decreases the bit rate required in following encoding.



● : integer MV position

× : candidates of half-pixel MV positions



Evaluation of BM motion estimation

■ Advantages :

- Straightforward, regular operation, and promising for VLSI chip design
- Robustness (no differential operations as in optical flow approach)

■ Disadvantages :

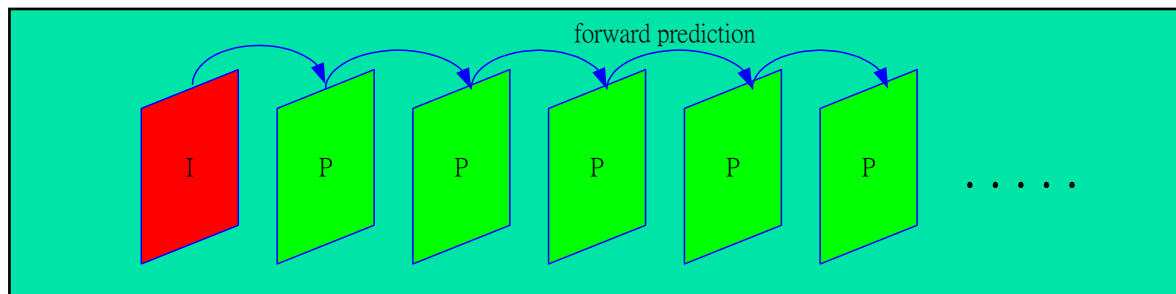
- Can not process several moving objects in a block (e.g., around occlusion boundaries)

■ Improvements :

- Post-processing of images to eliminate blocking effect
- Interpolation of images or averaging of motion vectors to subpixel accuracy

Image Prediction structure

- I/ P frame only (H.261, H.263)



Intra-frame

Predictive
frame

Bi-directional
frame

- I/ P/ B frame (MPEG-x)

