IVPL

Chapter 8

Image Compression

Introduction

- *Basis* - redundancy may exist in image data and can be removed
- Applications of image compression
  - video-conferencing
  - remote sensing
  - document and medical imaging
  - facsimile transmission (FAX)

Introduction

- *Categories of image compression*:
  - *information preserving* - loseless image compression, useful in image archiving (as in the storage of medical records)
  - *information lossy* - lossy image compression, provides higher level of data reduction (e.g. broadcast television, videoconferencing, and facsimile transmission, etc.)

8.1 Fundamentals

- *Data redundancy* may be quantified as follows:
  - given two data sets:
    D_1 with n_1 information units (input)
    D_2 with n_2 information units (output)
  - the *compression ratio* C_R (of D_2 w.r.t. D_1) is
    $C_R = n_1/n_2$
  - the *relative data redundancy* R_D (of D_1) is
    $R_D = 1 - (1/C_R) = 1 - n_2/n_1$
  - *when* $n_2 << n_1$,then
    $C_R \longrightarrow \infty$ (D_2 with good compression ratio)
    $R_D \longrightarrow 1$ (D_1 with high redundancy)

8.1 Fundamentals

- Types of image data redundancy:
  - coding redundancy (Huffman code)
  - interpixel redundancy
  - psychovisual redundancy

8.1 Fundamentals

- Coding Redundancy
  - ❑Gray levels are usually coded with equal-length binary codes (8 bits per pixel for gray levels 0 through 255), and redundancy exists in such codes.
  - ❑More efficient coding can be achieved if variable-length coding is employed.
  - ❑Variable-length coding assigns less bits the gray levels with higher occurrence probabilities in an image.

8.1 Fundamentals

- Average length required to represent a pixel:

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

- where $l(r_k)$ is the code length for gray level r_k, and $p_k(r_k)$ is the probability of r_k
- The average length L_{avg} for equal-length coding is 8 and less than 8 for variable-length coding.
- See the example in page 412 (including Table 8.1 and Fig. 8.1)

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

TABLE 8.1
Example of variable-length coding.

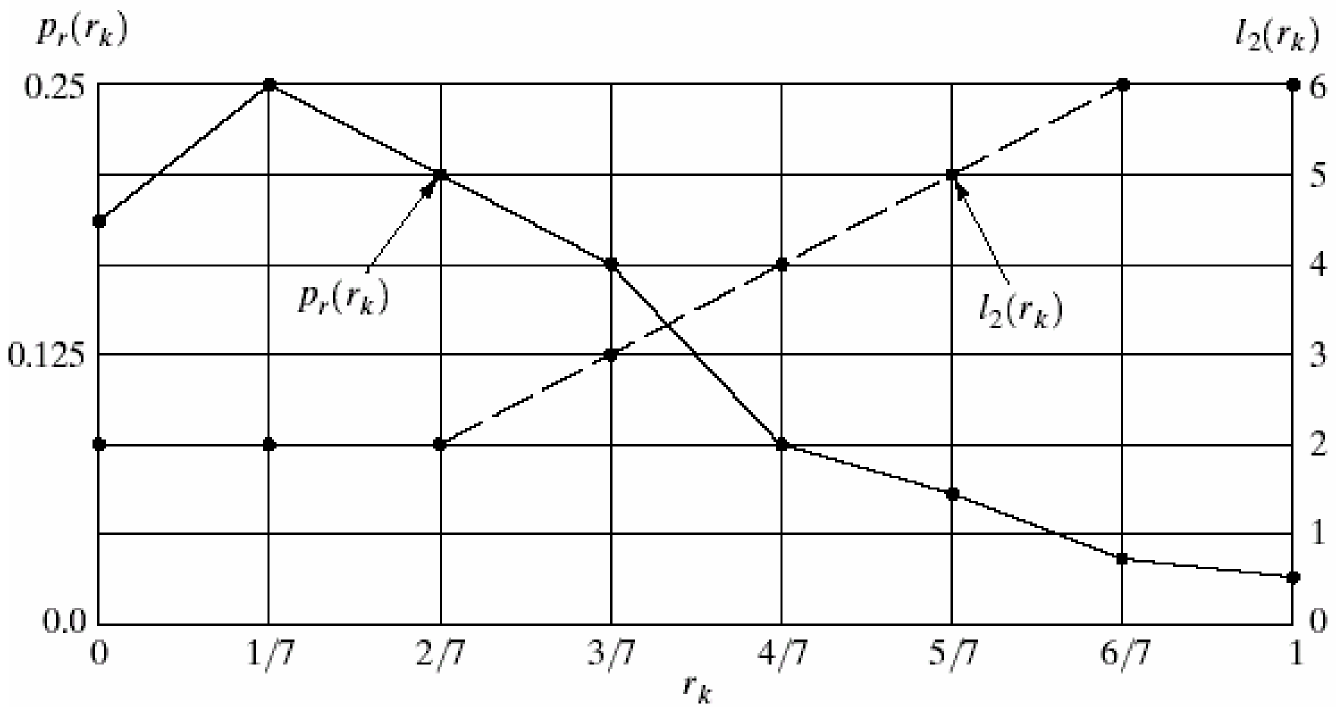


FIGURE 8.1 Graphic representation of the fundamental basis of data compression through variable-length coding.

8.1 Fundamentals

- Interpixel Redundancy
  - The gray levels of adjacent pixels in normal images are highly correlated, resulting in interpixel redundancy.
  - See Fig. 8.2 for an example. Note that when pixel distance = 45 pels, the normalized autocorrelated coefficients are very high.
  - The gray level of a given pixel can be predicted by its neighbors and the difference is used to represent the image; this type of transformation is called *mapping*.
  - *Run-length coding* can also be employed to utilize interpixel redundancy in image compression.

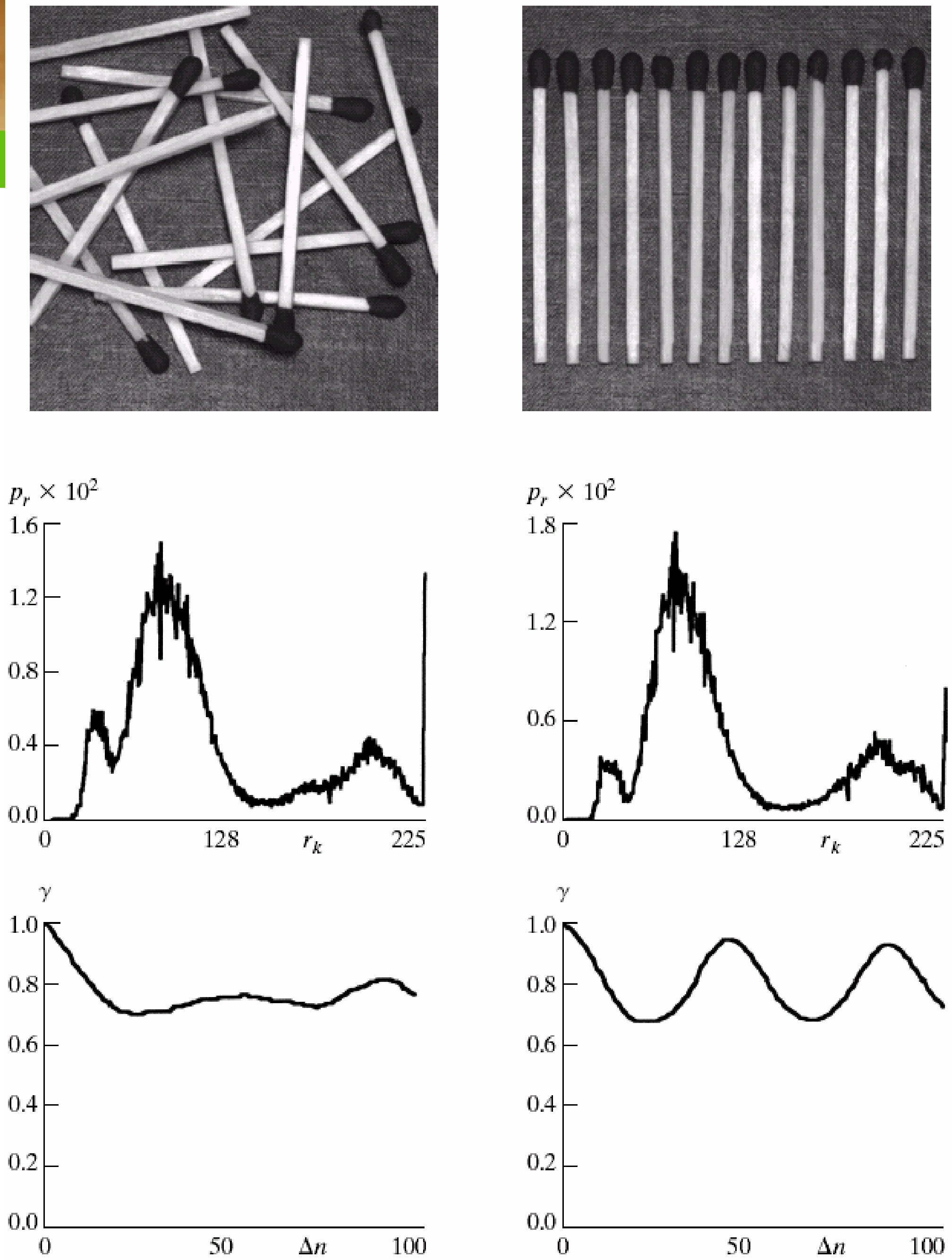
8.1 Fundamentals

- See Fig. 8.3 for an example of run-length coding. Each run R_i is represented by a pair (g_i , r_i) with

  g_i = gray level of R_i

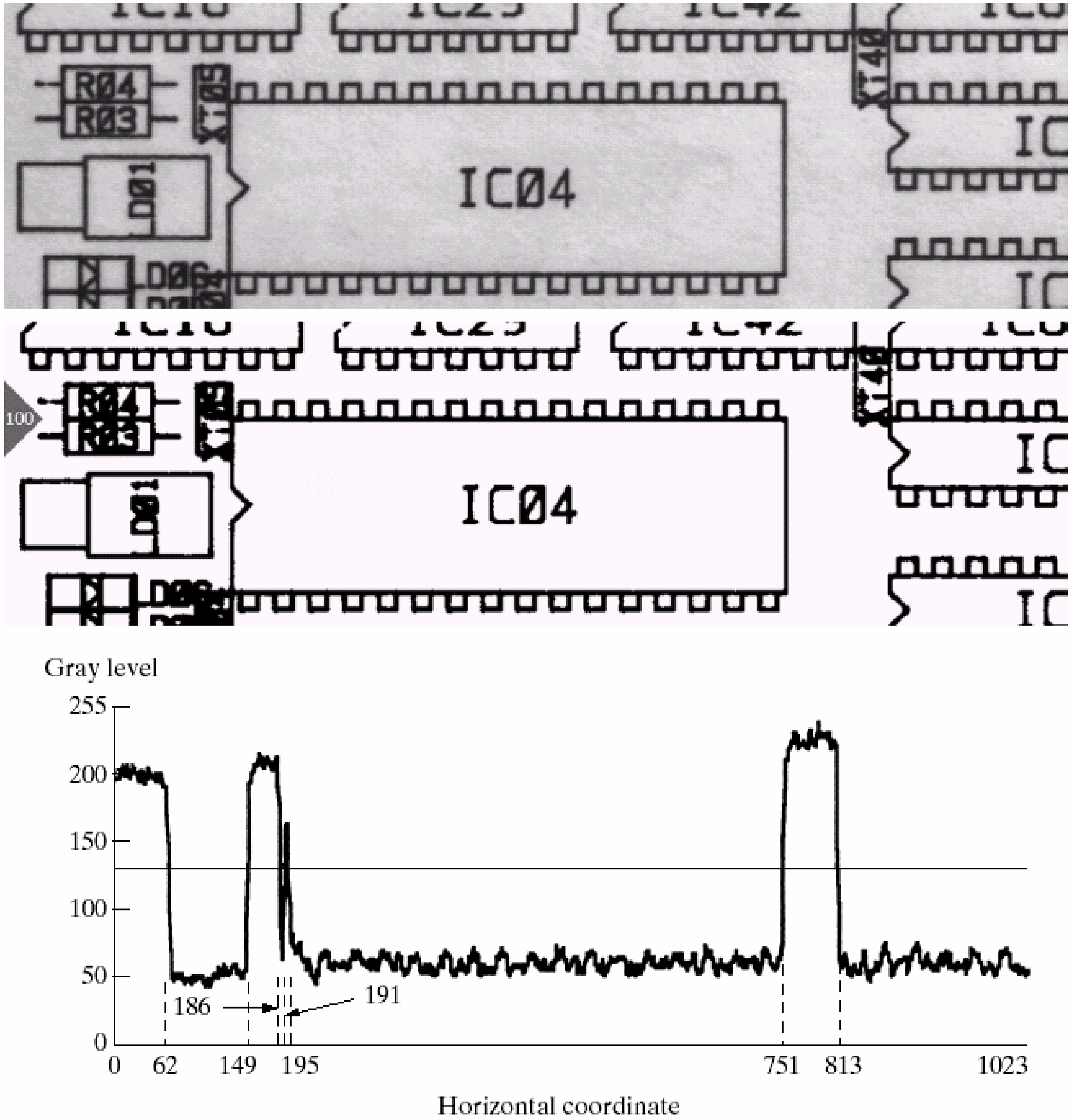
  r_i = length of R_i

  (see Fig. 8.3(d) for an example)
- See the value of C_R of the example in page 416.
- Removing the interpixel redundancy is loseless.



a b
c d
e f

FIGURE 8.2 Two images and their gray-level histograms and normalized autocorrelation coefficients along one line.



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

a
b
c
d

FIGURE 8.3 Illustration of run-length coding: (a) original image. (b) Binary image with line 100 marked. (c) Line profile and binarization threshold. (d) Run-length code.

8.1 Fundamentals

- Psychovisual Redundancy
  - Certain visual information are less important than other information in normal visual processing.
  - *Requantization* using less bits can be used to eliminate psychovisual redundancy, but *false contour* will appear (see Fig. 8.4 for an example).
  - A better way is to use *improved gray-scale* (IGS) *quantization*, which breaks up edges by adding to each pixel a pseudo-random number generated from low-order bits of neighboring pixels before quantizing the result.

8.1 Fundamentals

❑ Details of IGS quantization:

- ❖ Set initial SUM = 0000 0000
- ❖ If most significant 4 bits of current pixel **A** = 1111

  new_SUM = **A** + 0000

  else

  new_SUM = **A** + least significant 4 bits of old SUM

❑ IGS Quantization Procedure

- Table 8.2

❑ Removing psychovisual redundancy is information lossy (see Fig. 8.4).

a b c

FIGURE 8.4
(a) Original image.
(b) Uniform quantization to 16 levels. (c) IGS quantization to 16 levels.

Pixel	Gray Level	Sum	IGS Code
$i-1$	N/A	0000 0000	N/A
i	0110 1100	0110 1100	0110
$i+1$	1000 1011	1001 0111	1001
$i+2$	1000 0111	1000 1110	1000
$i+3$	1111 0100	1111 0100	1111

TABLE 8.2
IGS quantization procedure.

8.1 Fundamentals

- Fidelity Criteria
  - Fidelity criteria is used to measure information loss.
  - Two classes:
    - objective fidelity criteria - measured mathematically
    - subjective fidelity criteria - measured by human observation
  - Objective criteria
    - **root-mean-square (rms) error** between input and output images of *f* and *f'* (both of size *M* x *N*)

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

- **mean-square signal-to noise ratio** of *f'*

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

❑ Subjective fidelity criteria - see Table 8.3.

TABLE 8.3
Rating scale of the Television Allocations Study Organization. (Frendendall and Behrend.)

Value	Rating	Description
1	Excellent	An image of extremely high quality, as good as you could desire.
2	Fine	An image of high quality, providing enjoyable viewing. Interference is not objectionable.
3	Passable	An image of acceptable quality. Interference is not objectionable.
4	Marginal	An image of poor quality; you wish you could improve it. Interference is somewhat objectionable.
5	Inferior	A very poor image, but you could watch it. Objectionable interference is definitely present.
6	Unusable	An image so bad that you could not watch it.

8.2 Image Compression Models

- A general compression system model consists of two distinct structural blocks: an *encoder* and a *decoder*.

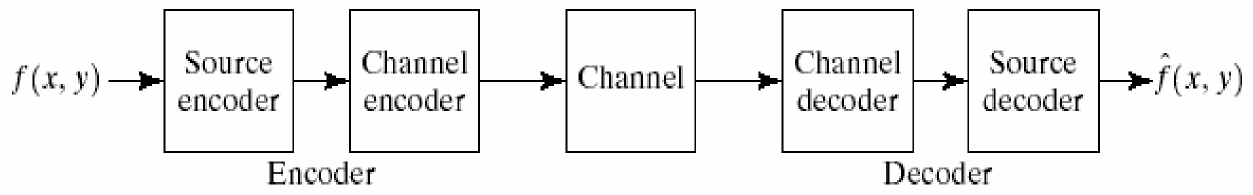


FIGURE 8.5 A general compression system model.

- The source encoder <u>removes input redundancies</u> (interpixel, psychovisual, and coding redundancy).
- The channel encoder increases the <u>noise immunity</u> of the source encoder output.

8.2 Image Compression Models

- *mapper* - reduce interpixel redundancy, reversible (run-length coding, Transform coding) *quantizer* - reduce psychovisual redundancy, irreversible *symbol encoder* - reduce coding redundancy, reversible (Huffman coding)

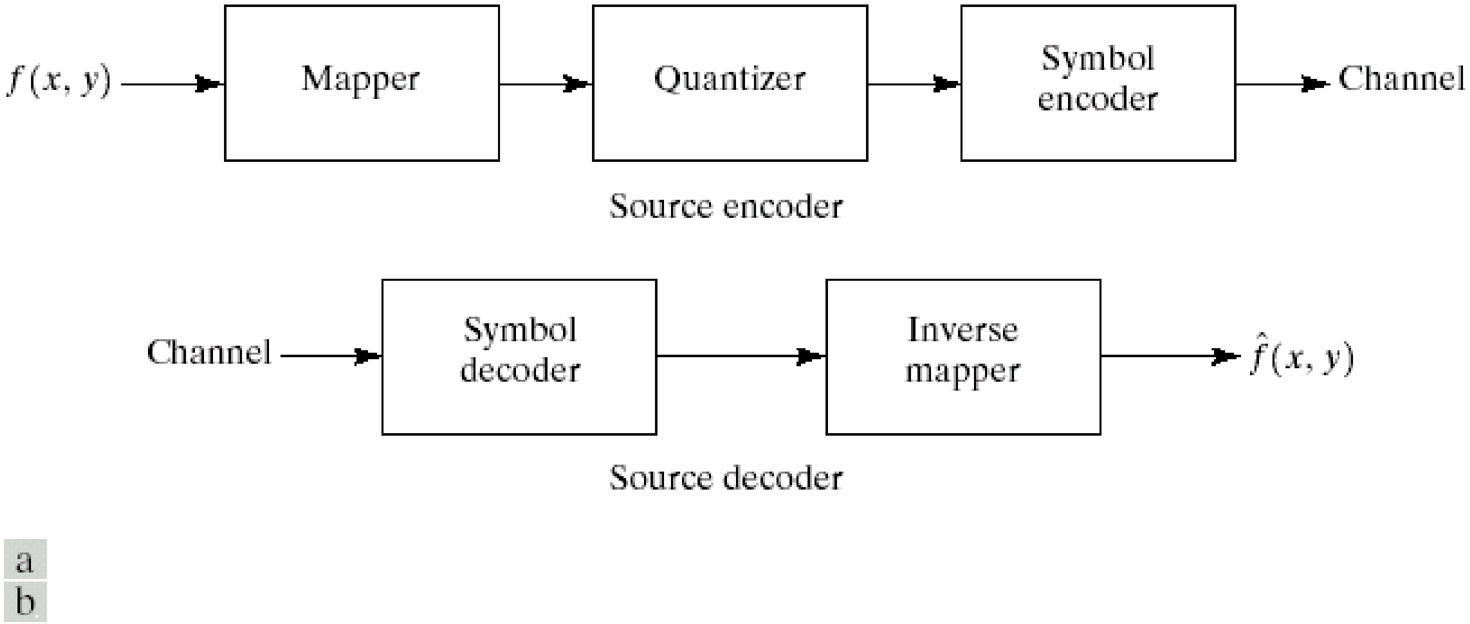


FIGURE 8.6 (a) Source encoder and (b) source decoder model.

- The Channel Encoder and Decoder
  - The channel encoder adds "control redundancy" into the output of the source encoder to reduce the impact of channel noise at the expense of reducing the compression ratio.
  - One way of channel encoding is to use Hamming codes
  - Example: 3 bits are added to a 4-bit word so that the distance (number of different bits) between any two valid code words is 3, all single-bit errors can be corrected.
  - Hamming(7,4) code word $h_1h_2h_3h_4h_5h_6h_7$ associated with a 4-bit binary number $b_3b_2b_1b_0$ is

$h_1 = b_3 \oplus b_2 \oplus b_0$ $\quad h_3 = b_3 \quad h_5 = b_2 \quad h_6 = b_1 \quad h_7 = b_0$

$h_2 = b_3 \oplus b_1 \oplus b_0$ $\quad h_4 = b_2 \oplus b_1 \oplus b_0$

where $\oplus$ denotes XOR operation, $h_1 h_2 h_4$ are even-parity bits

- Hamming(7,4) code

  (b) to decode a Hamming encoded result: a single-bit error is indicated by a nonzero parity word , where

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

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

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

  (c) If a nonzero value is found, the decoder complements the code word bit position indicated by the parity word .

8.3 Elements of Information Theory

- Entropy: (first-order estimate)

$$H = -\Sigma p(a_i) \log p(a_i)$$

- If the source symbols are equally probable, the entropy is maximized
- If base is 2, H means the amount of bits needed per symbol

8.4 Error-Free Compression

- Compression ratios are approximately 2 to 10.
  - An error-free compression method is composed of two operations:
  - mapping - reduce interpixel redundancy
  - symbol encoding - reduce coding redundancy
  - Assigning the shortest possible code words to the most probable gray levels.
  - The source symbols may be the gray levels of an image or a gray-level mapping operation (pixel differences, run-lengths, etc.)

8.4 Error-Free Compression

- Huffman coding
  - Yielding the smallest possible number of code symbols per source symbol when coding source symbols individually.
- Steps – Fig. 8-11~8.12
  - The Huffman encoded symbols can be decoded by examining the individual symbols of the string in a left to right manner since Huffman coding is
    (1)***instantaneous*** - each code word can be decoded without referencing succeeding symbols
    (2)***uniquely decodable*** - any string of code symbols can be decoded in only one way.

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.

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								

8.4 Error-Free Compression

- Variable-length codes
  - Table 8.5
- Arithmetic coding
  - Fig. 8.13
  - Table 8.6
- LZW Coding
  - GIF, TIFF, and PDF
  - Example 8.12 (pp. 447)

TABLE 8.5 Variable-length codes.

Source symbol	Probability	Binary Code	Huffman	Truncated Huffman	B_2-Code	Binary Shift	Huffman Shift
Block 1							
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
Block 2							
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	10 0000	C10C00	111 101	00 1110
a_{14}	0.03	01101	01111	10 0001	C10C01	111 110	00 1111
Block 3							
a_{15}	0.03	01110	01100	10 0010	C10C10	111 111 000	00 00 10
a_{16}	0.02	01111	010000	10 0011	C10C11	111 111 001	00 00 11
a_{17}	0.02	10000	010001	10 0100	C11C00	111 111 010	00 00 110
a_{18}	0.02	10001	001010	10 0101	C11C01	111 111 011	00 00 100
a_{19}	0.02	10010	001011	10 0110	C11C10	111 111 100	00 00 101
a_{20}	0.02	10011	011010	10 0111	C11C11	111 111 101	00 00 1110
a_{21}	0.01	10100	011011	10 1000	C00C00C00	111 111 110	00 00 1111
Entropy	4.0						
Average length		5.0	4.05	4.24	4.65	4.59	4.13

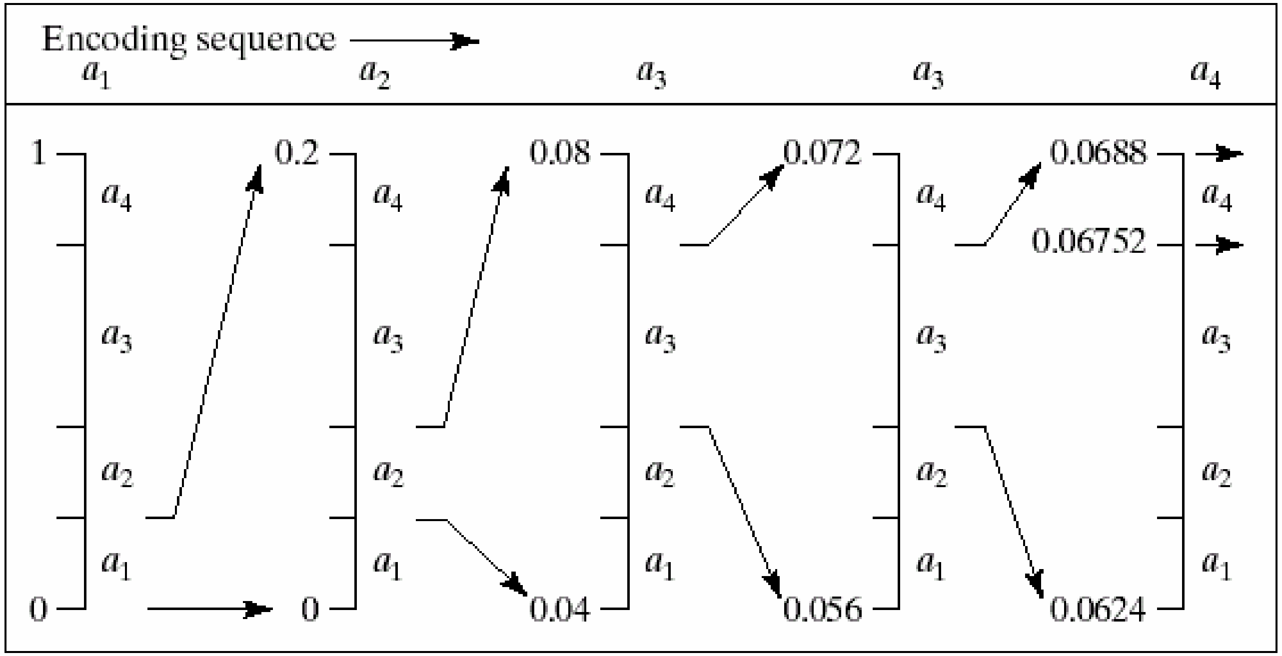


FIGURE 8.13 Arithmetic coding procedure.

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	$[0.8, 1.0)$

TABLE 8.6
Arithmetic coding example.

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		

TABLE 8.7
LZW coding example.

- Bit-Plane Coding
  - attack an image's interpixel redundancies
  - Bit-plane coding decomposes a multilevel image into a series of binary images and compresses each binary image via a binary image compression method.
  - ***Bit-plane decomposition***
    - Normal way:

      (1)Represent each m-bit gray level by a base 2 polynomial

      $$a_{m-1}2^{m-1} + a_{m-2}2^{m-2} + \ldots + a_1 2^1 + a_0 2^0$$

      (2)The ith-order bit-plane collects the a_i bits of each pixel

      (3)Disadvantages - small changes in gray level can have a significant impact on the complexity of the bit planes (e.g. 127 = 01111111 and 128 = 10000000).

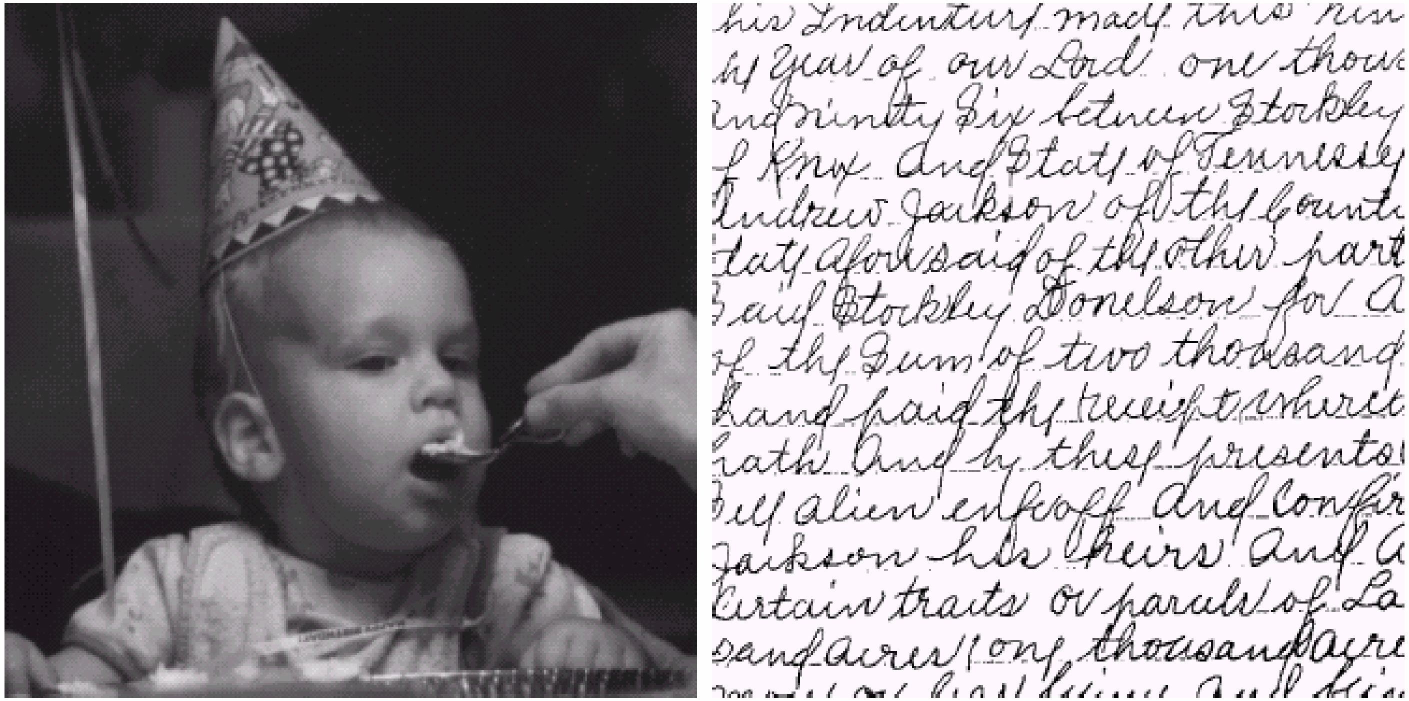
8.4 Error-Free Compression

- ***Gray code representation***:
  (1)The m-bit Gray code $g_{m-1}\, g_{m-2} \ldots g_1\, g_0$ can be computed by

$$g_i = a_i \oplus a_{i+1} \qquad 0 \le i \le m-2$$

- $$g_{m-1} = a_{m-1}$$

  (2)Each Gray coded bit plane collects a Gray code bit g_i.
  (3)Successive code words differ in only one bit position. Thus, small changes in original gray level will not causing great changes in code words (127 = 01000000, 128 = 11000000).
  - See Fig. 8.14 to 8.16 for normal and Gray coded bit-planes.



a b

FIGURE 8.14 A 1024 × 1024 (a) 8-bit monochrome image and (b) binary image.

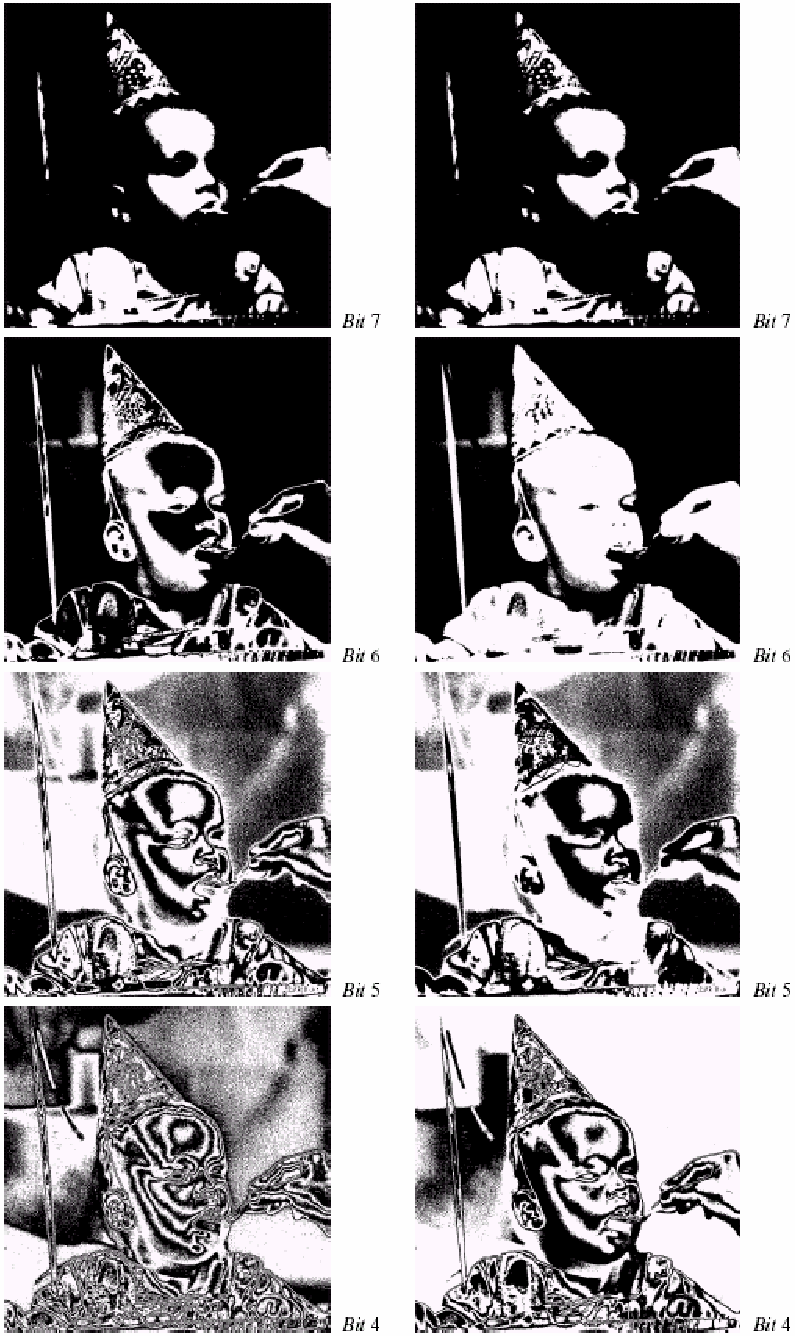


FIGURE 8.15 The four most significant binary (left column) and Gray-coded (right column) bit planes of the image in Fig. 8.14(a).

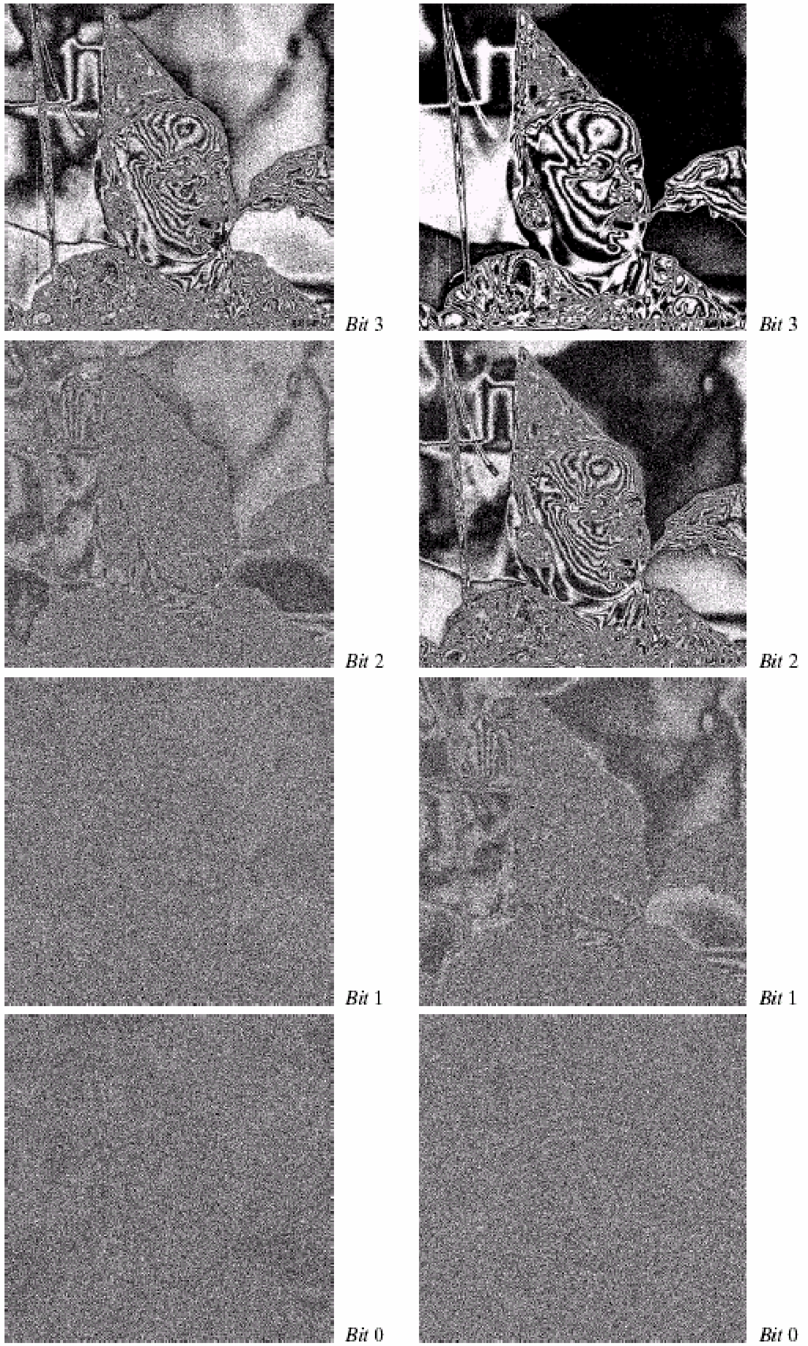


FIGURE 8.16 The four least significant binary (left column) and Gray-coded (right column) bit planes of the image in Fig. 8.14(a).

8.4 Error-Free Compression

- ***Constant area coding* (*CAC*)**

- Idea: using special code words to identify large areas of contiguous 0's and 1's to compress a binary image.
- Method:

  (1)divide the input image into blocks of size m x n pixels.

  (2)classify each block into the categories of all white, all black, and mixed intensity.

  (3)assign 1-bit code word 0 to the most frequently occurring category; and assign 2-bit code words 10 and 11 to the other two categories.

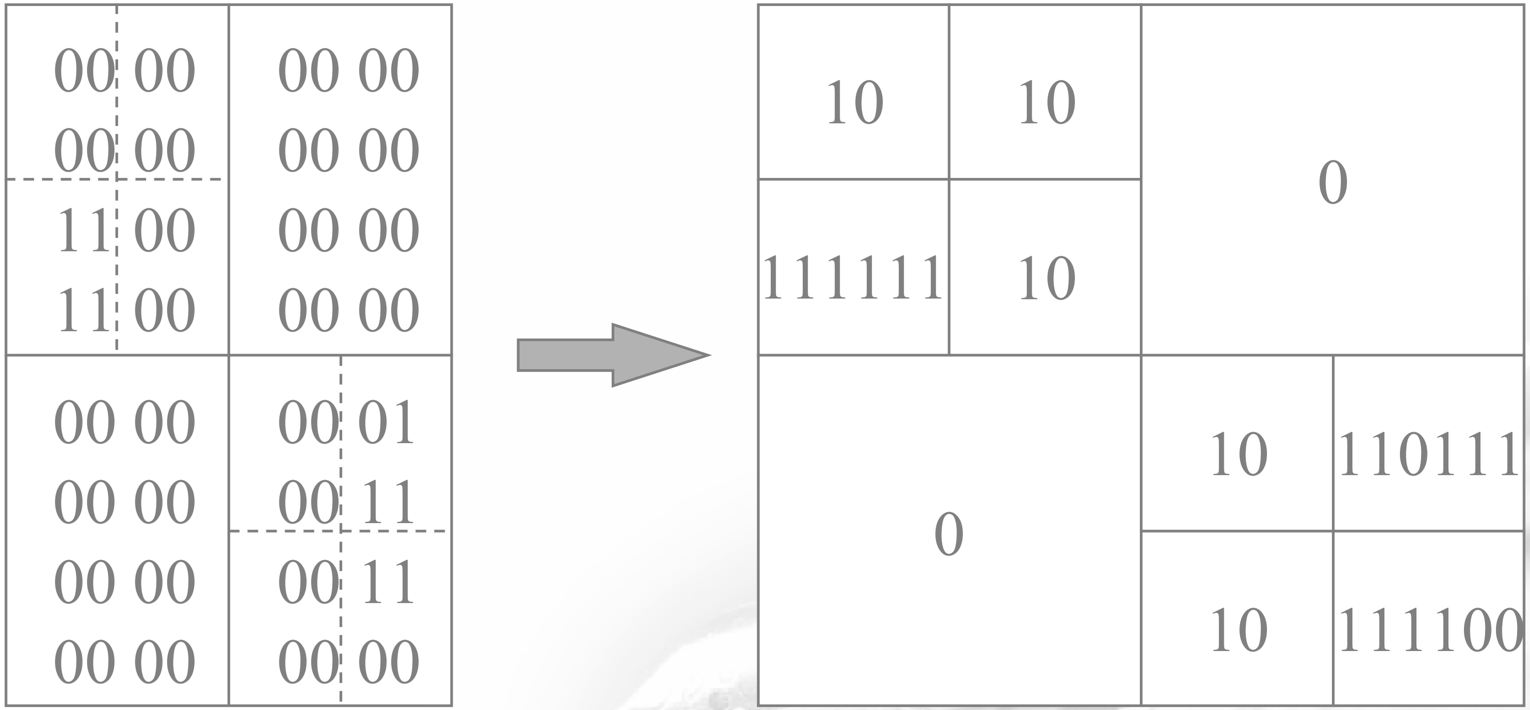
  (4)the code assigned to the mixed intensity category is just used as a prefix, which is followed by the *mn*-bit pattern of the block.

- ***White block skipping*** (***WBS***)
- Useful for predominantly white text documents
- Assign code word 0 to solid white areas; and 1 to other blocks (including the solid black blocks) followed by the bit pattern of the block.
- Reason: few black areas occur in texts.
- A modification of *WBS* is to code solid white lines as 0's and other lines with a 1 followed by the line pattern.
- A second modification of of *WBS*:

  (1)a solid white block is coded as a 0 and all other blocks are divided into subblocks that are assigned a prefix of 1 and coded successively.

- A second modification of of *WBS*: (cont.)

  (2)An example



❑ One-dimensional run-length coding

- Run-length coding is also a constant area coding method.
- A standard compression method for facsimile (FAX) coding.
- Basic concept - to code each contiguous group of 0's or 1's in a row from left to right by run lengths and run values (0 or 1).
- Approaches for deciding values of runs:
  (1)specify the value of the first run of each row;
  (2)assume each row begins with a white run, whose run length may be zero.
- The run-lengths may be further compressed by variable-length coding method.

8.4 Error-Free Compression

- An example: given a row contents:

  1111001001111

  The code are:

  for case (a) - 1, 4, 2, 1, 2, 4

  for case (b) - 0, 4, 2, 1, 2, 4

- Another example: given a row contents:

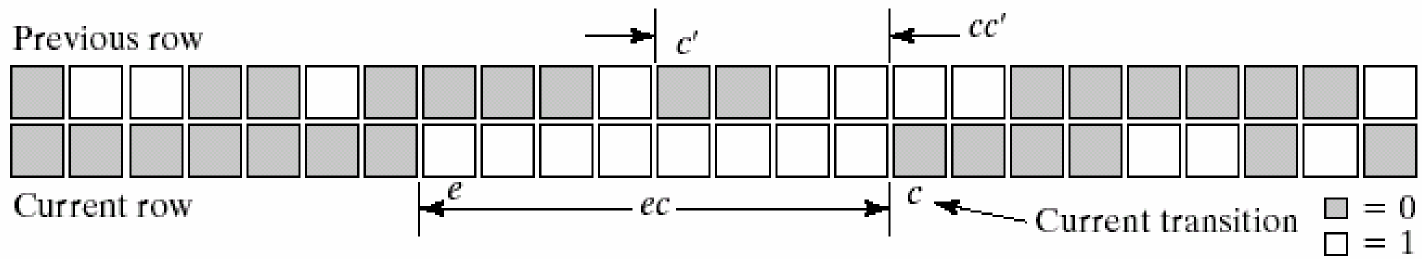
  0001100011100

  The code are:

  for case (a) - 0, 3, 2, 3, 3, 2

  for case (b) - 3, 2, 3, 3, 2

- Two-dimensional run-length coding
  - Fig. 8-17



Distance measured	Distance	Code	Distance range	Code $h(d)$
cc'	0	0	1 – 4	0 xx
ec or cc' (left)	1	100	5 – 20	10 xxxx
cc' (right)	1	101	21 – 84	110 xxxxxx
ec	$d(d>1)$	111 $h(d)$	85 – 340	1110 xxxxxxxx
cc' (c' to left)	$d(d>1)$	1100 $h(d)$	341 – 364	11110 xxxxxxxxxx
cc' (c' to right)	$d(d>1)$	1101 $h(d)$	1365 – 5460	111110 xxxxxxxxxxxx

a
b

FIGURE 8.17 A relative address coding (RAC) illustration.

❑ ***Contour tracing and coding***

- Useful for coding regions
- Methods:
  (1) represent each contour by a set of boundary points
  (2) *direct contour tracing* - represent a contour by a single boundary point and a set of directions.
  (3) *predictive differential quantizing* (PDQ)
  (4) *double delta coding* (DDC)

8.4 Error-Free Compression

- ***predictive differential quantizing*** (PDQ)
  (1) A scan line-oriented contour tracing procedure.
  (2)Represent each object boundary by the following items:
    (a)a “new start” message (indicate start of a new contour)
    (b)a sequence of pairs (Δ', Δ'')
    (c)a “merge” message (indicate end of a old contour)
  (3)Definition of (Δ', Δ'') - (see Fig. 8.18)
    (a) Δ' :the difference between the starting coordinate of the front contours on adjacent lines.
    (b) Δ'' :the difference between the front-to-back contour lengths.

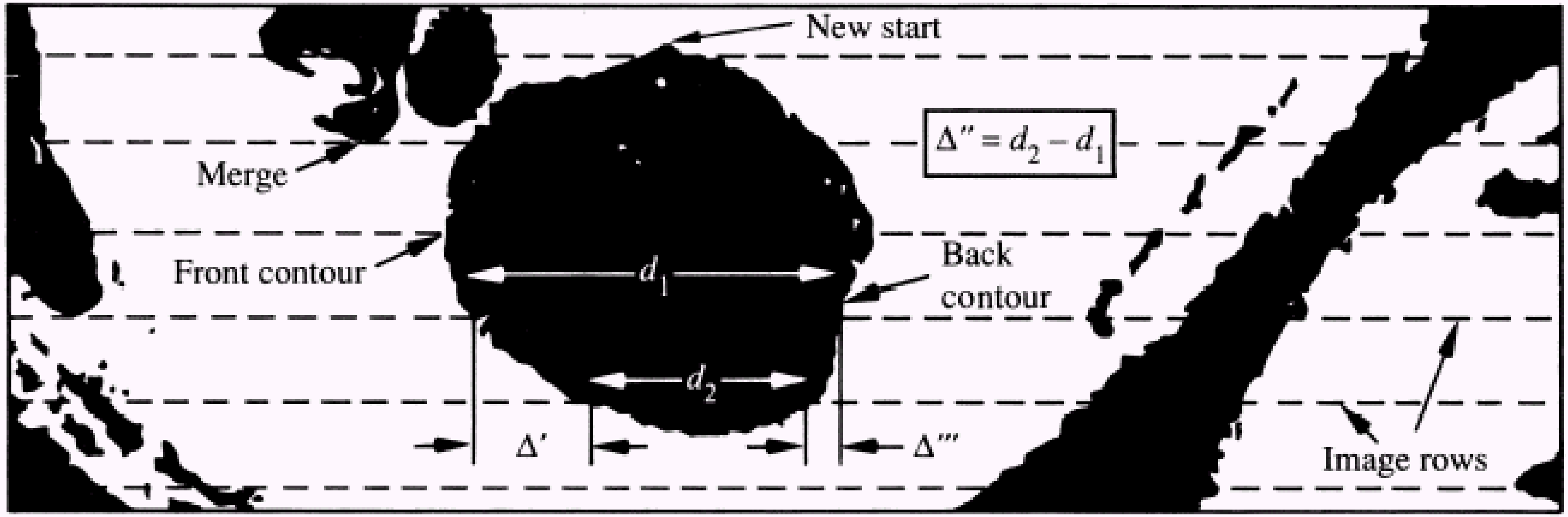


FIGURE 8.18 Parameters of the PDQ algorithm.

8.4 Error-Free Compression

- D***ouble delta coding*** (DDC)
  (1)The value D'' in PDQ is replaced by D'''.
  (2)Definition of D''' : the difference between back contour coordinates of adjacent lines.
- The values of Δ' , Δ'', Δ''' and coordinates of new starts and merges can be coded with a variable-length code.
- example 8.14 (pp. 455)
  - Comparison of binary compression techniques.
  - Table 8.8 and 8.9

Method	Bit-plane code rate (bits/pixel) 7	6	5	4	3	2	1	0	Code Rate	Compression Ratio
Binary Bit-Plane Coding										
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
Gray Bit-Plane Coding										
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

TABLE 8.8
Error-free bit-plane coding results for Fig. 8.14(a): $H \approx 6.82$ bits/pixel

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

TABLE 8.9
Error-free binary image compression results for Fig. 8.14(b): $H \approx 0.55$ bits/pixel.

8.4 Error-Free Compression

- Loseless Predictive Coding
  - Eliminating the interpixel redundancies of closely spaced pixels by extracting and coding only the new information in each pixel.
  - The new information of a pixel is defined as the difference between the actual and predicted value of that pixel.
  - The predictive coding system consists of an encoder and a decoder, each containing an identical *predictor*.
  - Notations:

    f_n : the input pixel

    f'_n : the output of the predictor and rounded to nearest integer

    e_n : prediction error

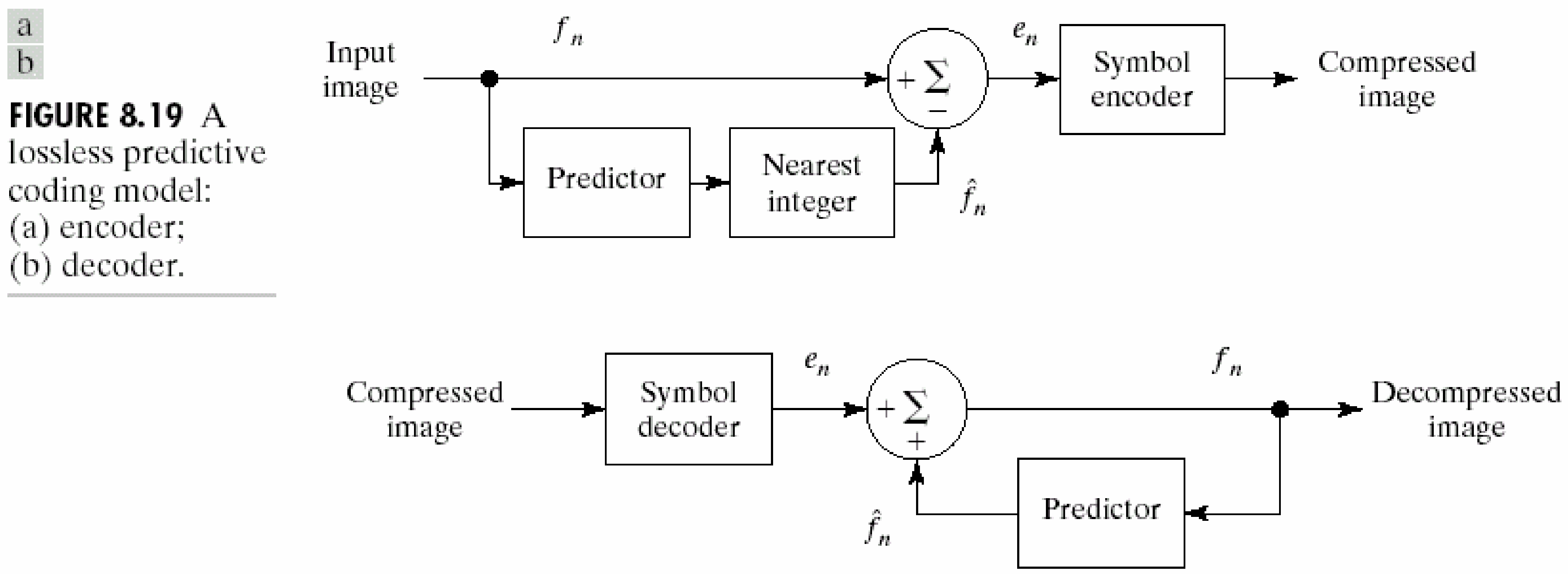


FIGURE 8.19 A lossless predictive coding model: (a) encoder; (b) decoder.

Encoder : $e_n = f_n - f'_n$

Decoder : $f_n = e_n + f'_n$

8.4 Error-Free Compression

- In general, the prediction is formed by a linear combination of *m* previous samples, that is,

$$f_n' = round[\sum_{i=1}^{m} \alpha_i f_{n-i}],$$

where *m* is the order of the linear predictor, *round* is a function to denote the rounding or nearest integer operation, and the α_i are prediction coefficients.

- In 2-D linear predictive coding, the index on the spatial domain is used:

$$f_n'(x, y) = round[\sum_{i=1}^{m} \alpha_i f(x, y - i)],$$

8.4 Error-Free Compression

- In 2-D linear predictive coding, the prediction is a function of the previous pixels in a left-to-right, top-to-bottom scan of an image.

$f\,(i\text{-}1\,,j\text{-}1)$	$f\,(i\text{-}1,j)$	$f\,(i\,\text{-}1\,,j\text{+}1)$
$f\,(i,j\text{-}1)$	$f\,(i,j)$	

- In 3-D case, the prediction is based on the above pixels and the previous pixels of preceding frames.
- See Fig. 8.20 for a previous pixel predictor.

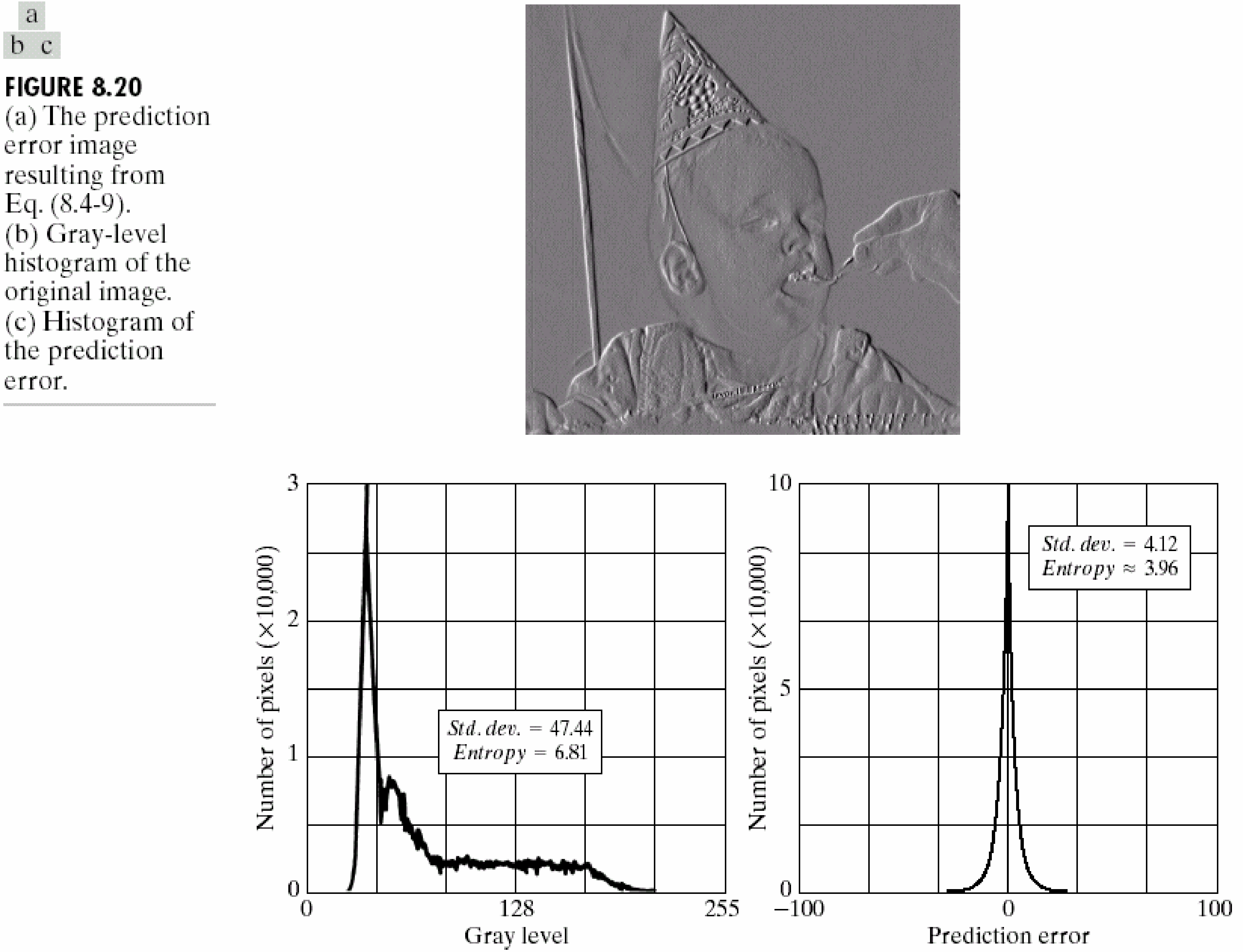
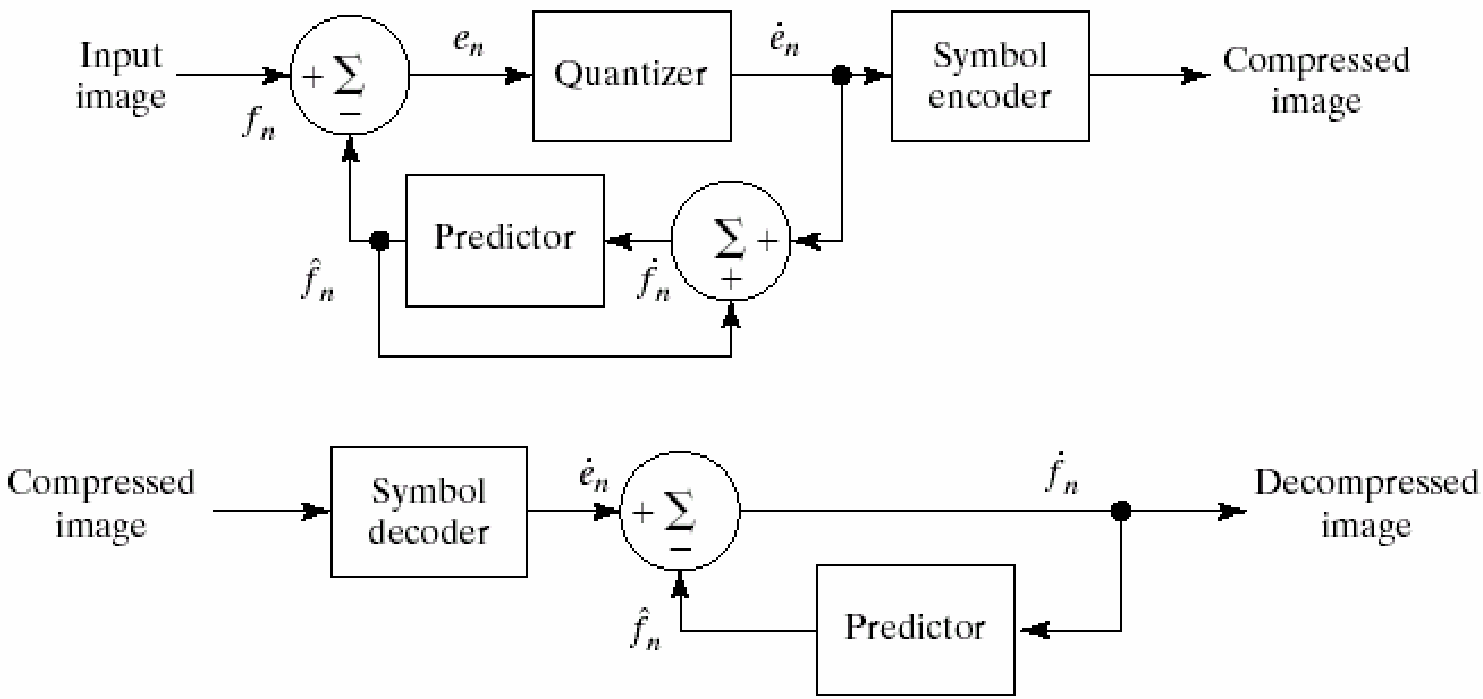


FIGURE 8.20 (a) The prediction error image resulting from Eq. (8.4-9). (b) Gray-level histogram of the original image. (c) Histogram of the prediction error.

8.5 Lossy Compression

- Lossy compression is based on the concept of compromising the accuracy of the reconstructed image in exchange for compression ratio.
- The compression ratio of lossy compression can be obtained by more than 30:1, and images that are virtually indistinguishable from the original at 10:1 to 20:1.
- The difference between error-free compression and lossy compression is due to the presence or absence of the quantizer.



a
b

FIGURE 8.21 A lossy predictive coding model: (a) encoder and (b) decoder.

8.5 Lossy Compression

- Example 8.16 (pp. 460) - *Delta modulation* (DM) is a simple lossy predictive coding method in which the predictor and quantizer are $\hat{f}_n = \alpha \hat{f}_{n-1}$

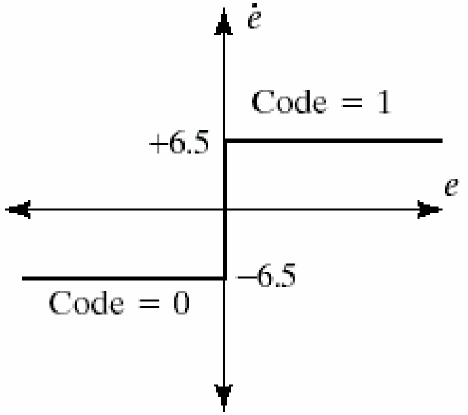
  and

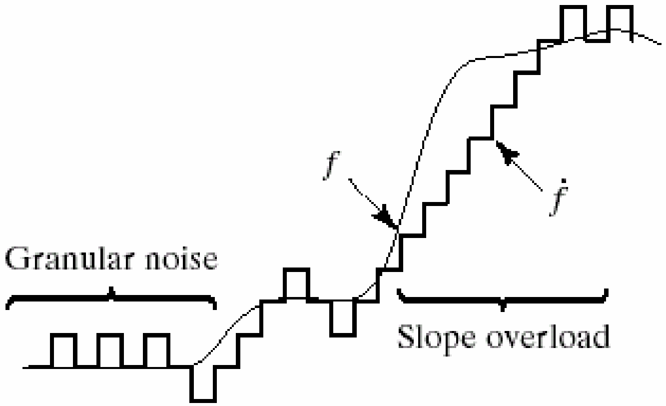
  $$\hat{e}_n = \begin{cases} +\xi & for\, e_n > 0 \\ -\xi & otherwise \end{cases}$$

  where α is a prediction coefficient (normally less than 1) and ξ is a positive constant.
- The output of the quantizer can be represented by a single bit.
- DM code rate is 1 bit/pixel.

8.5 Lossy Compression

- See Fig. 8.22 for a delta modulation example.
- When ξ is too small to represent the input's largest changes, a *slope overload* distortion occurs.
  - blurred object edges
- When ξ is too large to represent the input's smallest changes, usually in the relatively smooth region, *granular noise* appears.
  - grainy or noisy surfaces (that is, distorted smooth areas)





Input		Encoder				Decoder		Error
n	f	$\hat{f}$	e	$\dot{e}$	$\dot{f}$	$\hat{f}$	$\dot{f}$	$[f - \dot{f}]$
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
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.

a b
c

FIGURE 8.22 An example of delta modulation.

8.5 Lossy Compression

- *Differential pulse code modulation* (DPCM)
  - Having an optimal predictor that minimizes the encoder's mean-square prediction error

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

    subject to the constraint

    $$f'_n = \hat{e}_n + \hat{f}_n \approx e_n + \hat{f}_n \approx f_n$$

    and

    $$\hat{f}_n = \sum_{i=1}^{m} \alpha_i f_{n-i}$$

  - The optimal predictor assumes the quantization error is negligible and the prediction is constrained to a linear combination of *m* previous pixels.

8.5 Lossy Compression

- The sum of the prediction coefficients must satisfy

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

- the predictor's output falls within the allowed range of gray levels and to reduce the impact of transmission noise, which is generally seen as horizontal streaks in the reconstructed image.

- The problem is to select the m α_i that minimizes

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

8.5 Lossy Compression

- Differentiating the above Eq. with respect to α_i, i = 1, 2, . . ., m, and set the derivatives to 0, yields

  $\alpha = \mathbf{R}^{-1}\mathbf{r}$,

  where

$$\boldsymbol{\alpha} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_m \end{bmatrix} \qquad \mathbf{r} = \begin{bmatrix} E\{f_n f_{n-1}\} \\ E\{f_n f_{n-2}\} \\ \vdots \\ E\{f_n f_{n-m}\} \end{bmatrix}$$

$$\mathbf{R} = \begin{bmatrix} E\{f_{n-1} f_{n-1}\} & E\{f_{n-1} f_{n-2}\} & \cdots & E\{f_{n-1} f_{n-m}\} \\ E\{f_{n-2} f_{n-1}\} & \vdots & \vdots & \vdots \\ \vdots & \vdots & \vdots & \vdots \\ E\{f_{n-m} f_{n-1}\} & E\{f_{n-m} f_{n-2}\} & \cdots & E\{f_{n-m} f_{n-m}\} \end{bmatrix}$$

8.5 Lossy Compression

- In general, optimal α_i values must be computed image by image (named *local prediction*), which is impractical.
- In practice, a set of global coefficients is computed by assuming a simple image model, for example

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

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

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

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

where $\Delta h = |f(x-1, y) - f(x-1, y-1)|$, $\Delta v = |f(x, y-1) - f(x-1, y-1)|$ denote the horizontal and vertical gradients at points (x, y).

- See Figs. 8.23 and 6.24.

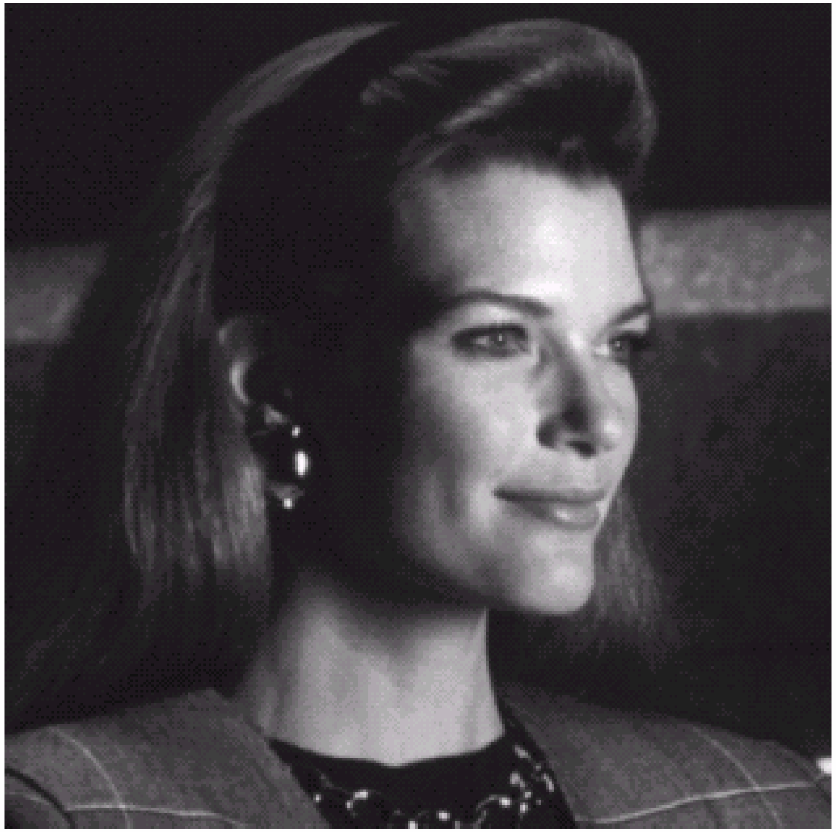
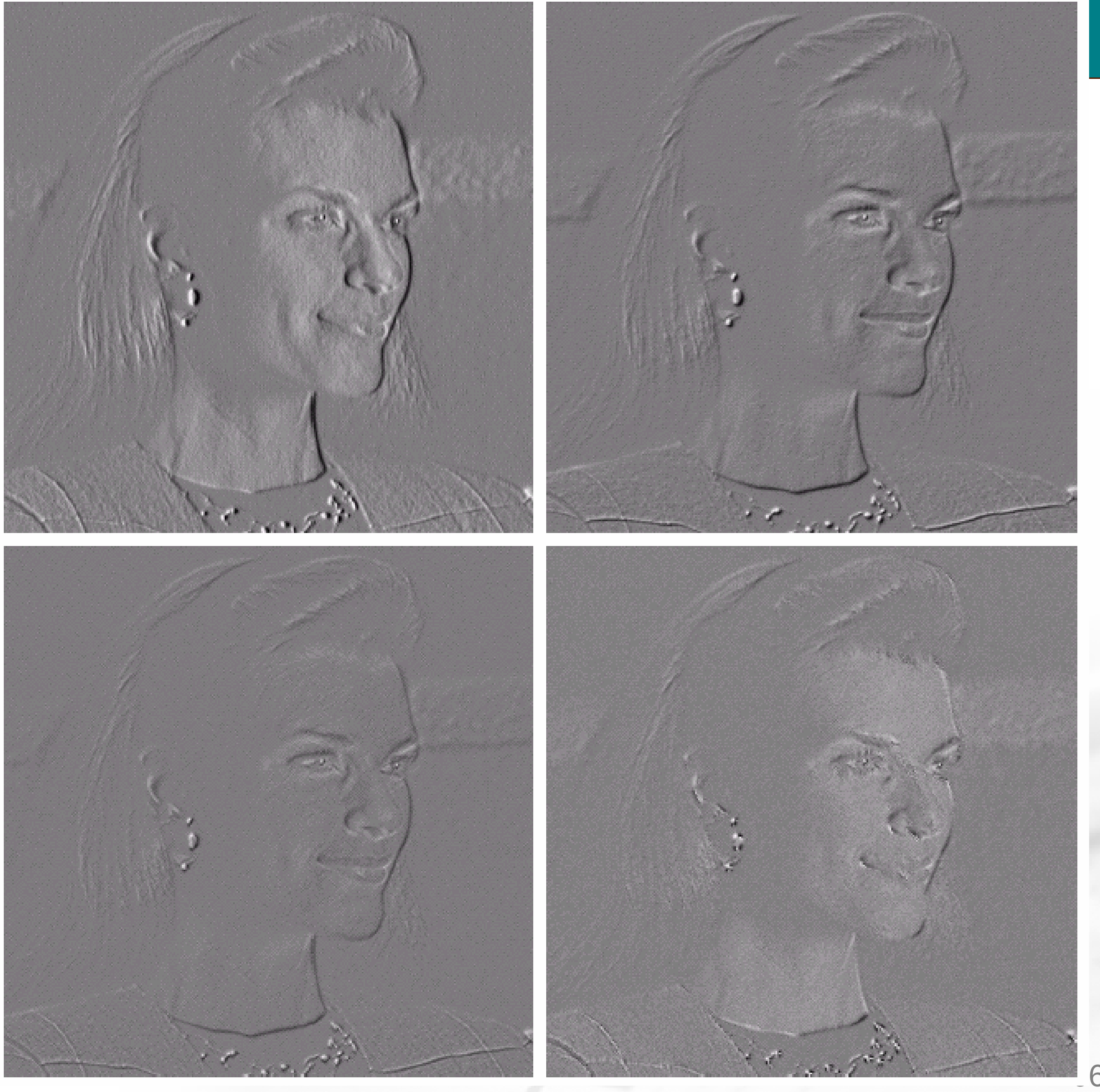


FIGURE 8.23 A 512×512 8-bit monochrome image.



a b
c d

FIGURE 8.24 A comparison of four linear prediction techniques.

8.5 Lossy Compression

- Transform Coding
  - ❑The predictive coding methods which operate on the pixels of an image may be called spatial domain method.
  - ❑In transform coding, a reversible, linear transform is used to map the image into a set of transform coefficients, which are then quantized and coded.

8.5 Lossy Compression

- Encoding steps:
  - subimage decomposition - dividing images into small blocks
  - transformation - decorrelate the pixels of each block or to pack most of the information into the smallest number of transform coefficients.
  - quantization - eliminating or more coarsely quantizing the coefficients that carry the least information.
  - symbol encoder - coding the quantized coefficients by variable length coding method.
- The decoder performs the inverse steps of the encoder except the quantization function.

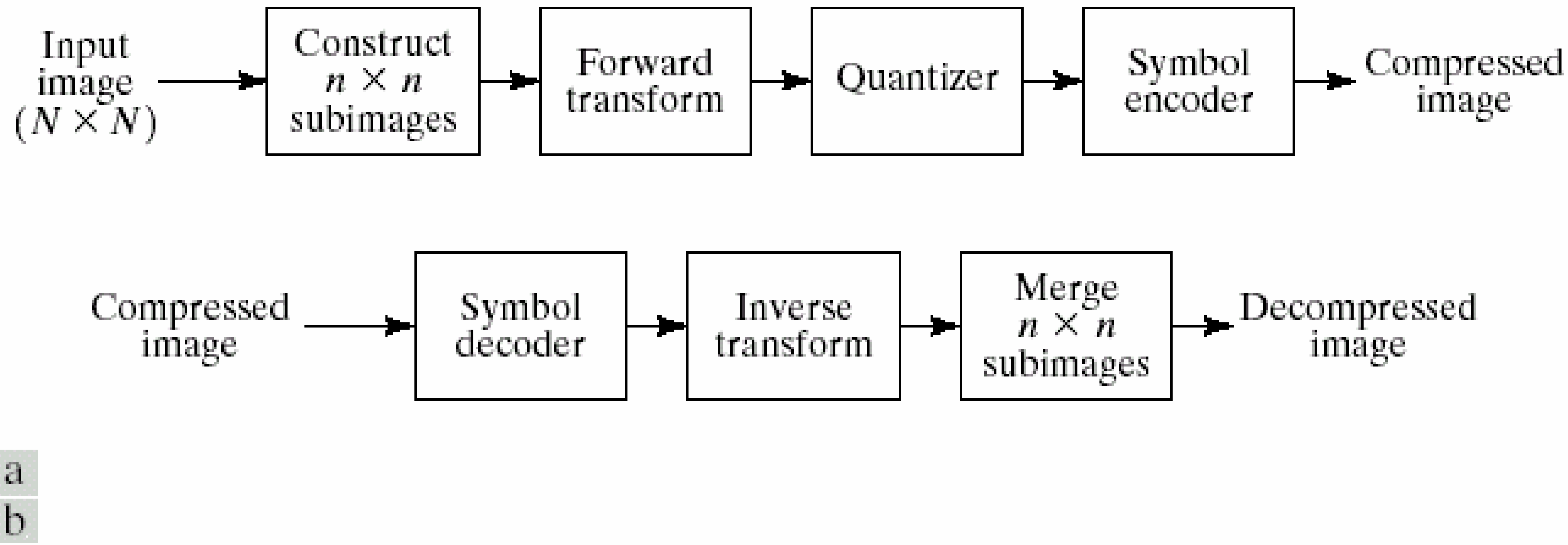


FIGURE 8.28 A transform coding system: (a) encoder; (b) decoder.

8.5 Lossy Compression

- **Transform selection - Fig. 8-31**
  - **Karhunen-Loeve transform (KLT)**
    - the optimal transform in information packing in minimizing the mean-square error for any image and number of retained coefficients but is data dependent (having to compute the bases for each subimage).
  - **Discrete Fourier transform (DFT)**
    - has fixed bases (input independent) and closely approximates the information packing ability of KLT, but creates Gibbs phenomenon causing boundary discontinuity and so blocking artifact.
  - **Discrete cosine transform (DCT)**
    - has fixed bases (input independent), closely approximates the information packing ability of KLT, and minimizing blocking artifact (Fig. 8.30 and Fig.8.32).
  - **Walsh-Hadamard transform (WHT)**
    - simplest to implement (Fig. 8.29).

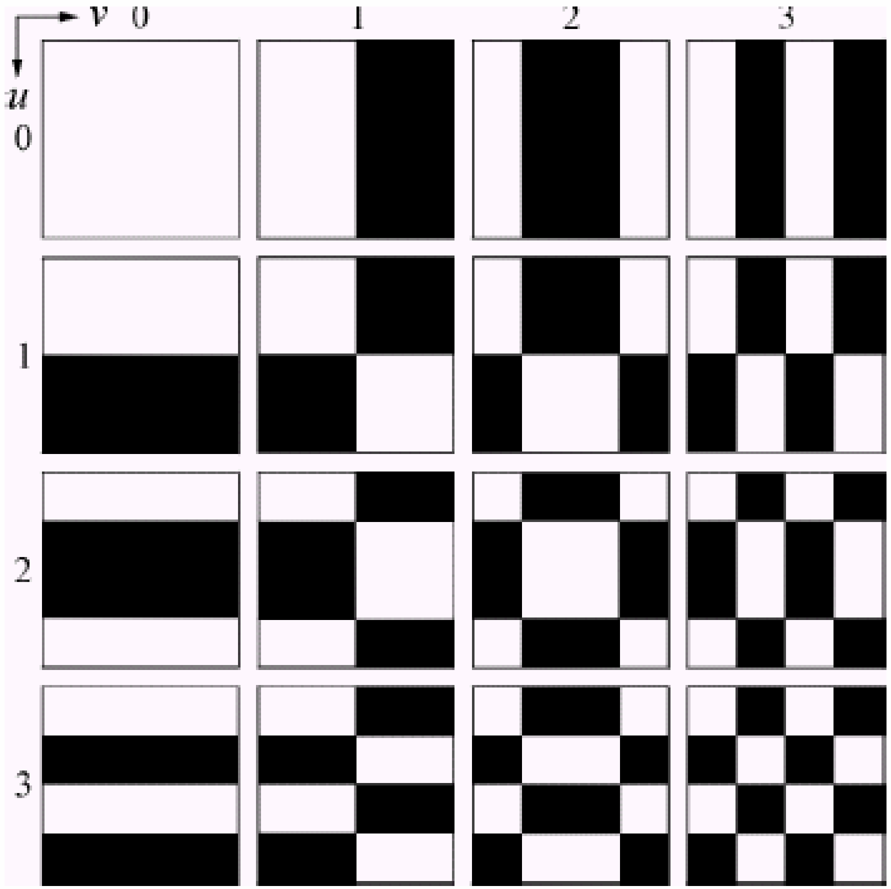


FIGURE 8.29 Walsh-Hadamard basis functions for $N = 4$. The origin of each block is at its top left.

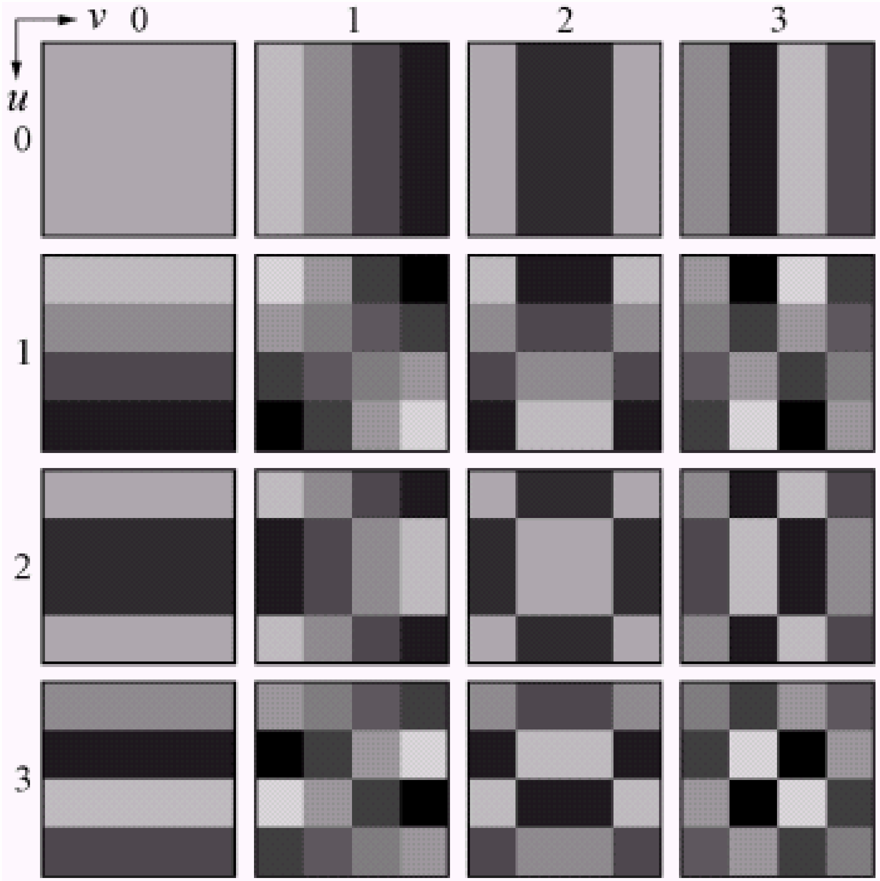
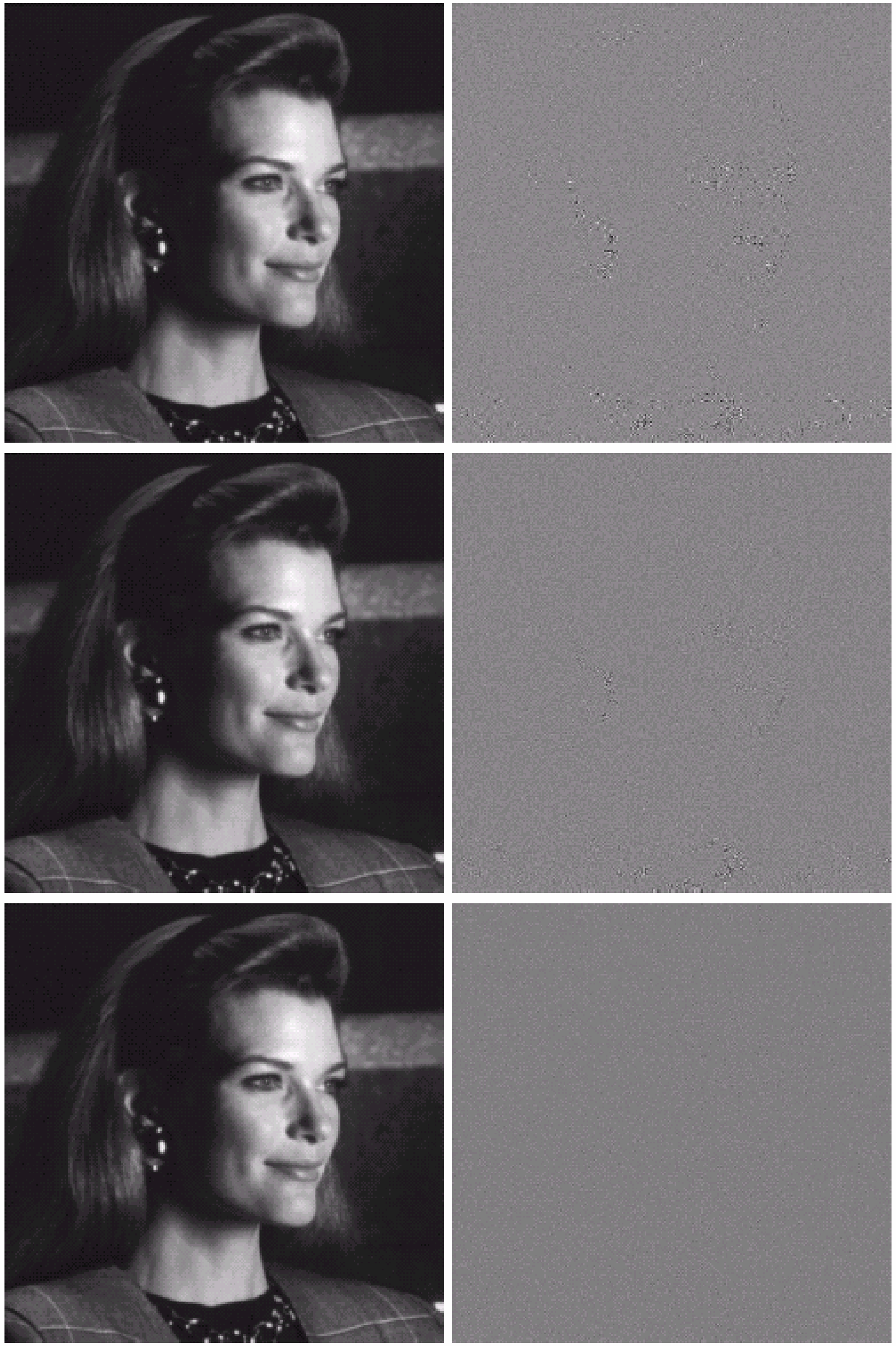


FIGURE 8.30 Discrete-cosine basis functions for $N = 4$. The origin of each block is at its top left.

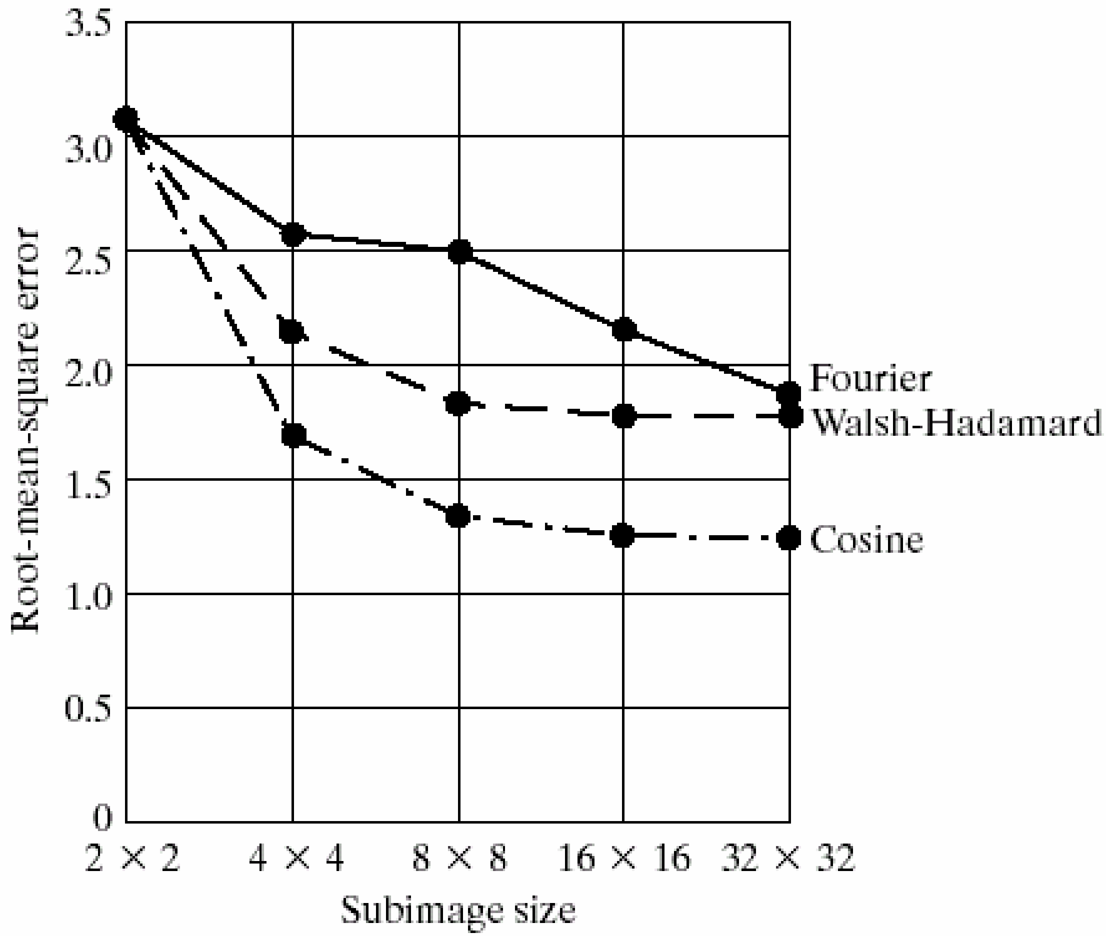


Example 8-19
pp.473

a b
c d
e f

FIGURE 8.31 Approximations of Fig. 8.23 using the (a) Fourier, (c) Hadamard, and (e) cosine transforms, together with the corresponding scaled error images.

FIGURE 8.33 Reconstruction error versus subimage size.



8.5 Lossy Compression

- Most practical transform coding systems are based on the DCT which is a compromise between information packing ability and computational complexity and has the following advantages:
  - having been implemented in a single IC
  - packing the most information into the fewest coefficients for most natural images
  - minimizing the blocklike appearance (blocking artifact).

8.5 Lossy Compression

- Review of the forward and inverse 2-D DCT:

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

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

where

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

8.5 Lossy Compression

❑ Subimage size selection -

- Purposes - the correlation (redundancy) between adjacent subimages is reduced and make the subimage dimension to be power of 2 to simplify the computation of the transform.
- In general, the compression and computational complexity increases as the subimage size increases.
- The most popular subimage size are 8×8 and 16×16.
- See Figs. 8.34 and 8.35.

a b
c d
e f

FIGURE 8.34 Approximations of Fig. 8.23 using 25% of the DCT coefficients: (a) and (b) 8×8 subimage results; (c) zoomed original; (d) 2×2 result; (e) 4×4 result; and (f) 8×8 result.

❑ Bit allocation -

- The overall process of truncating, quantizing, and coding the coefficients of a transformed subimage is commonly called *bit allocation*.

- Two ways of truncating coefficients
  (1)**zonal coding** - the retained coefficients are selected on the basis of maximum variance.
  (2)**threshold coding** - selecting the coefficients according to
  the maximum magnitude of coefficients.

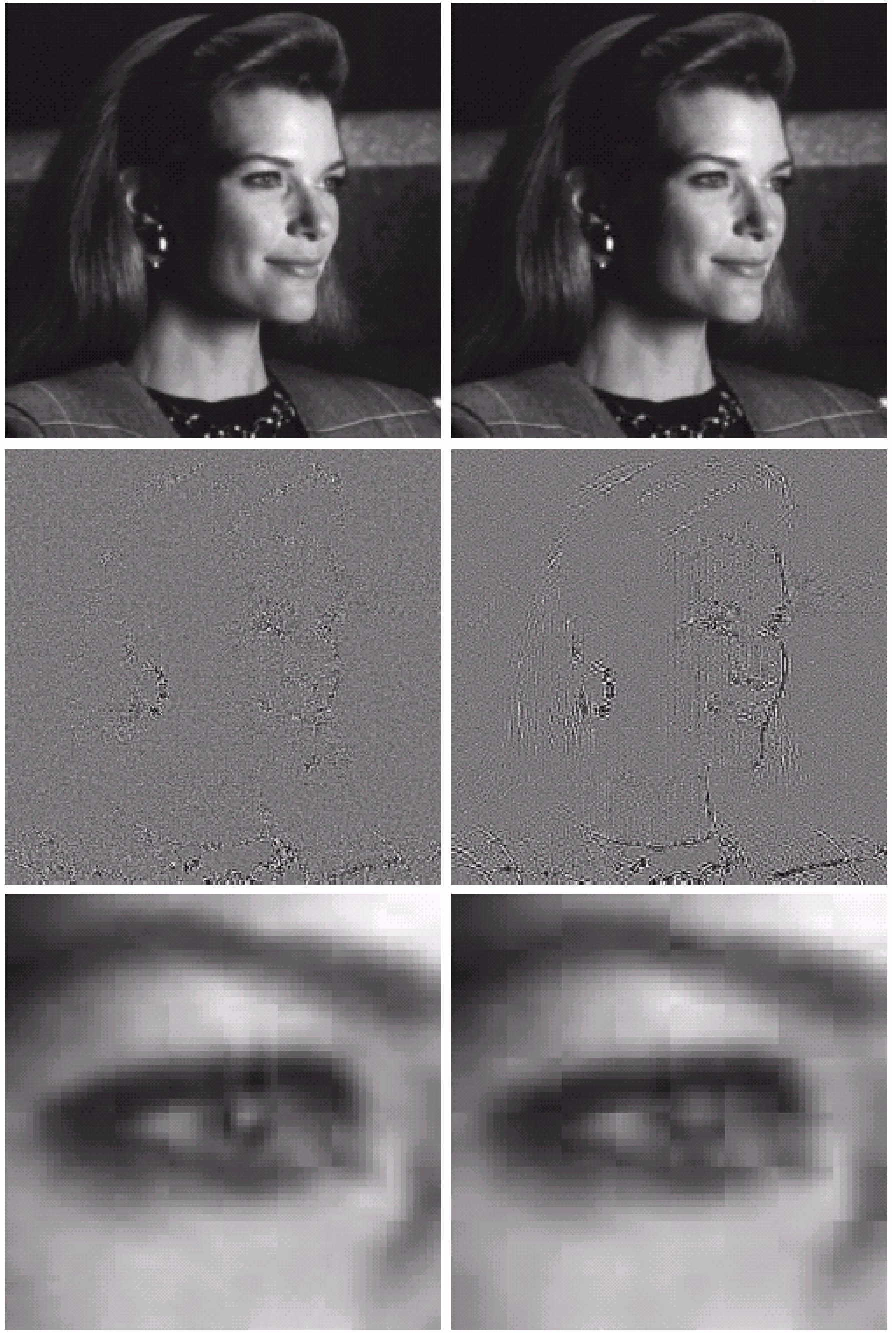


FIGURE 8.35 Approximations of Fig. 8.23 using 12.5% of the 8×8 DCT coefficients: (a), (c), and (e) threshold coding results; (b), (d), and (f) zonal coding results.

❑ Zonal coding -

- Principle - according to the information theory, the transform coefficients with maximum variance carry most picture information and should be retained in the coding process.
- The variance can be calculated directly by the set of transformed subimage of assumed image model.
- The process can be viewed as multiplying each transformed coefficient by the corresponding element in the **zonal mask** (Fig. 8.36(a), and then coding the selected coefficients according to the bit allocation table (Fig. 8.36(b))
- In general, the coefficients with maximum variance are located around the origin of an image transform.

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.

(a)

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

(b)

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

(c)

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

(d)

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

8.5 Lossy Compression

❑ Threshold coding -

- Principle - the transform coefficients with largest magnitude contribute most to reconstructed subimage quality.
- The most often used adaptive transform coding approach due to its simplicity.
- **Threshold mask** - showing the locations of retained coefficients which are usually reordered by a zigzag scan method to produce 1-D sequence; then the long runs of 0's can be coded by run-length coding method.
- Three ways of threshold coding:
  (1)a single global threshold for all subimages - the
  compression ratio differs from image to image.

8.5 Lossy Compression

2)a different threshold for each subimage - called *N-largest coding*, the same number of coefficients is discarded and so the coding rate is fixed.

(3)the threshold varies as a function of the location of each coefficient within the subimage - results in a variable code rate and offers the possibility to combine thresholding and quantization by replacing $T(u, v)$ ($C(u, v)$) with

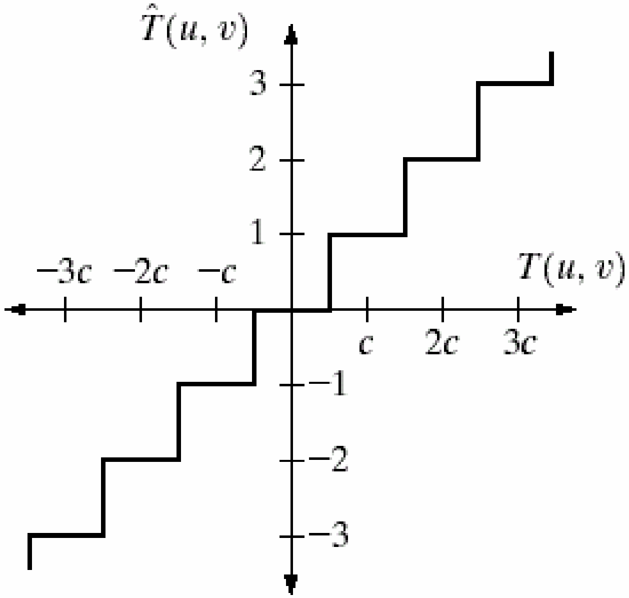
$$\hat{T}(u,v) = \mathbf{round}\left[\frac{T(u,v)}{Z(u,v)}\right]$$

where $Z(u, v)$ is an element of a normalization array $\boldsymbol{Z}$.

- The elements of $\boldsymbol{Z}$ can be scaled to achieve various compression levels.
- See Fig. 8.37 and 8.38.

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

a	b
c	d
e	f

FIGURE 8.38 Left column: Approximations of Fig. 8.23 using the DCT and normalization array of Fig. 8.37(b). Right column: Similar results for 4**Z**.

8.5 Lossy Compression

❑ Wavelet coding

a
b

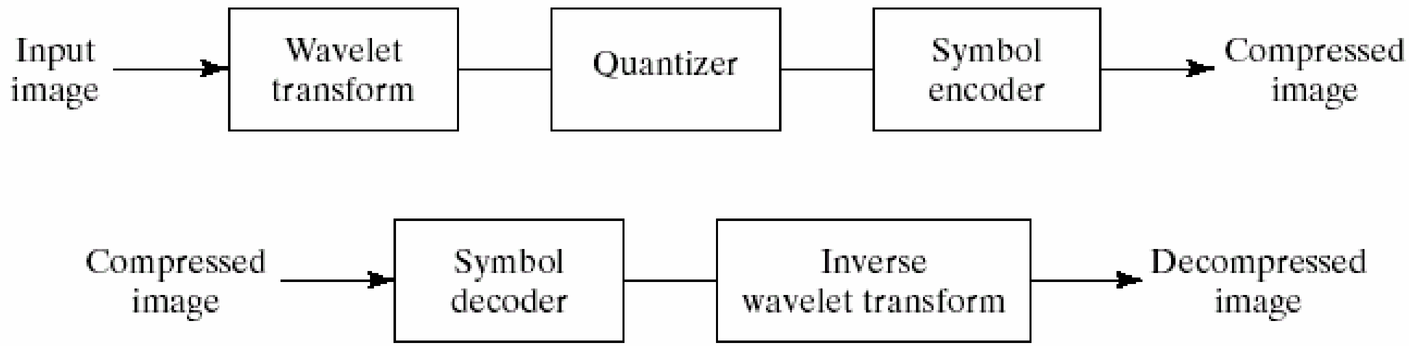


FIGURE 8.39 A wavelet coding system: (a) encoder; (b) decoder.

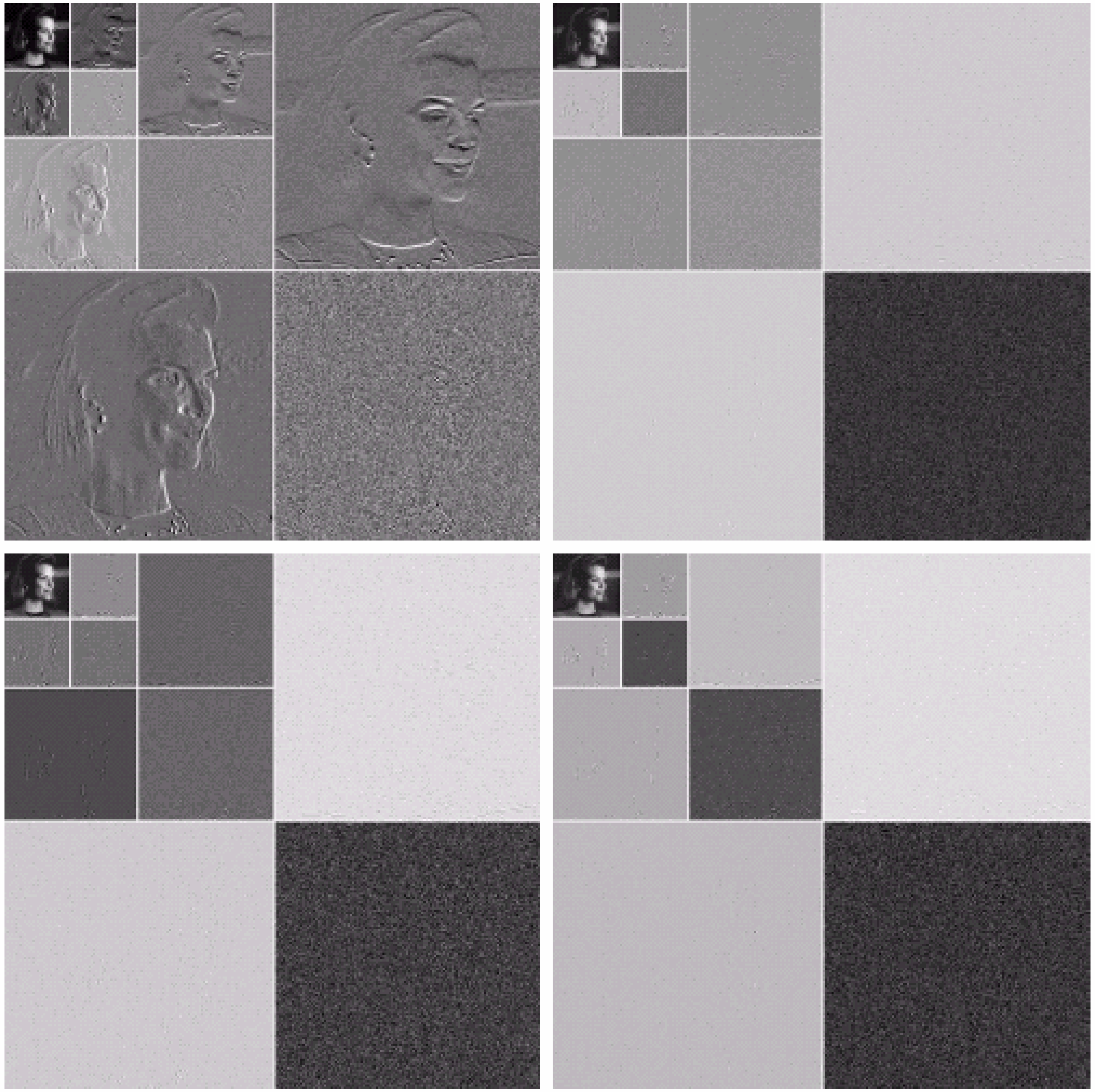
- ❑ Results
  - ❑ Fig. 8.40 and 8.41
- ❑ Example 8.24 and 8. 25(pp. 490~492)
  - ❑ Wavelet bases and decomposition level impact
  - ❑ Fig. 8.42 and 8.43
  - ❑ Table 8.12

a b
c d
e f

FIGURE 8.40 (a), (c), and (e) Wavelet coding results comparable to the transform-based results in Figs. 8.38(a), (c), and (e); (b), (d), and (f) similar results for Figs. 8.38(b), (d), and (f).

a	b
c	d
e	f

FIGURE 8.41 (a), (c), and (e) Wavelet coding results with a compression ratio of 108 to 1; (b), (d), and (f) similar results for a compression of 167 to 1.



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.

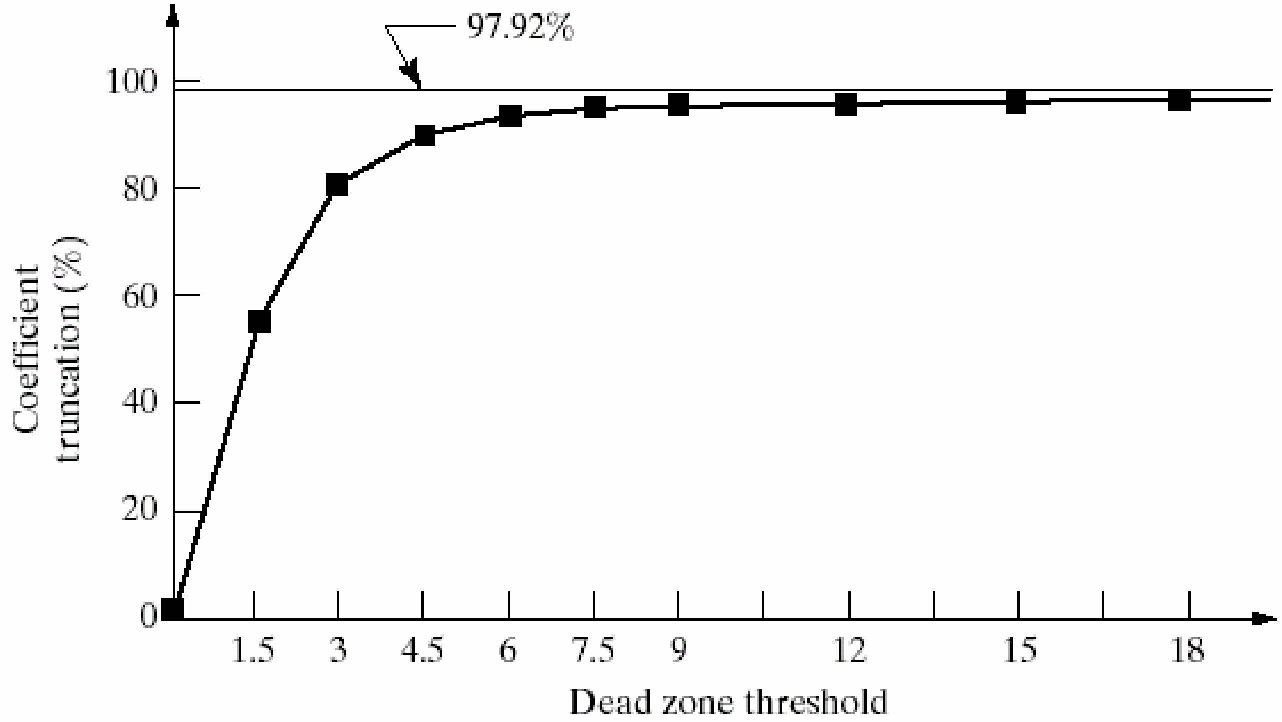
Wavelet	Filter Taps (Scaling + Wavelet)	Zeroed Coefficients
Haar (see Ex. 7.10)	2 + 2	46%
Daubechies (see Fig. 7.6)	8 + 8	51%
Symlet (see Fig. 7.24)	8 + 8	51%
Biorthogonal (see Fig. 7.37)	17 + 11	55%

TABLE 8.12
Wavelet transform filter taps and zeroed coefficients when truncating the transforms in Fig. 8.42 below 1.5.

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.

FIGURE 8.43 The impact of dead zone interval selection on wavelet coding.



8.6 Image Compression Standard

- ***Bilevel (binary) image compression standard***
  - CCITT Group 3 standard - facsimile (FAX) coding
  - CCITT Group 4 standard - facsimile (FAX) coding
  - JBIG - facsimile (FAX) coding

- ***Still-frame monochrome and color image compression***
  - JPEG - still-frame image compression (DCT)
  - JPEG 2000 – using Wavelet transform

- ***Sequential-frame monochrome and color compression***
  - H.261 - video teleconferencing
  - MPEG 1 - “entertainment quality” video compression (e.g., CD-ROM)
  - MPEG 2 - cable TV, narrow-channel satellite broadcasting

TABLE 8.14
CCITT terminating codes.

Run Length	White Code Word	Black Code Word	Run Length	White Code Word	Black Code Word
0	00110101	0000110111	32	00011011	000001101010
1	000111	010	33	00010010	000001101011
2	0111	11	34	00010011	000011010010
3	1000	10	35	00010100	000011010011
4	1011	011	36	00010101	000011010100
5	1100	0011	37	00010110	000011010101
6	1110	0010	38	00010111	000011010110
7	1111	00011	39	00101000	000011010111
8	10011	000101	40	00101001	000001101100
9	10100	000100	41	00101010	000001101101
10	00111	0000100	42	00101011	000011011010
11	01000	0000101	43	00101100	000011011011
12	001000	0000111	44	00101101	000001010100
13	000011	00000100	45	00000100	000001010101
14	110100	00000111	46	00000101	000001010110
15	110101	000011000	47	00001010	000001010111

Table 8.14 (Cont')

16	101010	0000010111	48	00001011	000001100100
17	101011	0000011000	49	01010010	000001100101
18	0100111	0000001000	50	01010011	000001010010
19	0001100	00001100111	51	01010100	000001010011
20	0001000	00001101000	52	01010101	000000100100
21	0010111	00001101100	53	00100100	000000110111
22	0000011	00000110111	54	00100101	000000111000
23	0000100	00000101000	55	01011000	000000100111
24	0101000	00000010111	56	01011001	000000101000
25	0101011	00000011000	57	01011010	000001011000
26	0010011	000011001010	58	01011011	000001011001
27	0100100	000011001011	59	01001010	000000101011
28	0011000	000011001100	60	01001011	000000101100
29	00000010	000011001101	61	00110010	000001011010
30	00000011	000001101000	62	00110011	000001100110
31	00011010	000001101001	63	00110100	000001100111

TABLE 8.15
CCITT makeup codes.

Run Length	White Code Word	Black Code Word	Run Length	White Code Word	Black Code Word
64	11011	0000001111	960	011010100	0000001110011
128	10010	000011001000	1024	011010101	0000001110100
192	010111	000011001001	1088	011010110	0000001110101
256	0110111	000001011011	1152	011010111	0000001110110
320	00110110	000000110011	1216	011011000	0000001110111
384	00110111	000000110100	1280	011011001	0000001010010
448	01100100	000000110101	1344	011011010	0000001010011
512	01100101	0000001101100	1408	011011011	0000001010100
576	01101000	0000001101101	1472	010011000	0000001010101
640	01100111	0000001001010	1536	010011001	0000001011010
704	011001100	0000001001011	1600	010011010	0000001011011
768	011001101	0000001001100	1664	011000	0000001100100
832	011010010	0000001001101	1728	010011011	0000001100101
896	011010011	0000001110010			

	Code Word		Code Word
1792	00000001000	2240	000000010110
1856	00000001100	2304	000000010111
1920	00000001101	2368	000000011100
1984	000000010010	2432	000000011101
2048	000000010011	2496	000000011110
2112	000000010100	2560	000000011111
2176	000000010101		

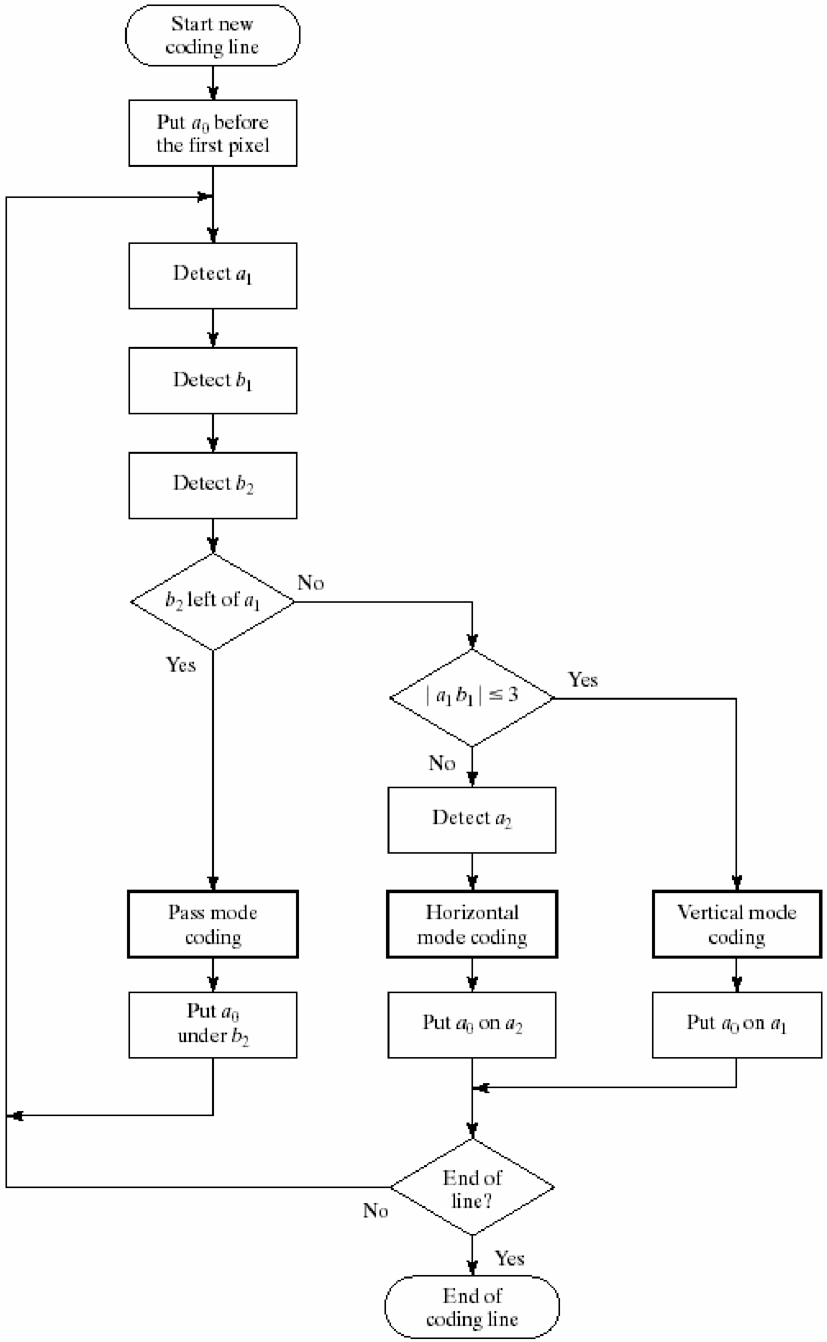
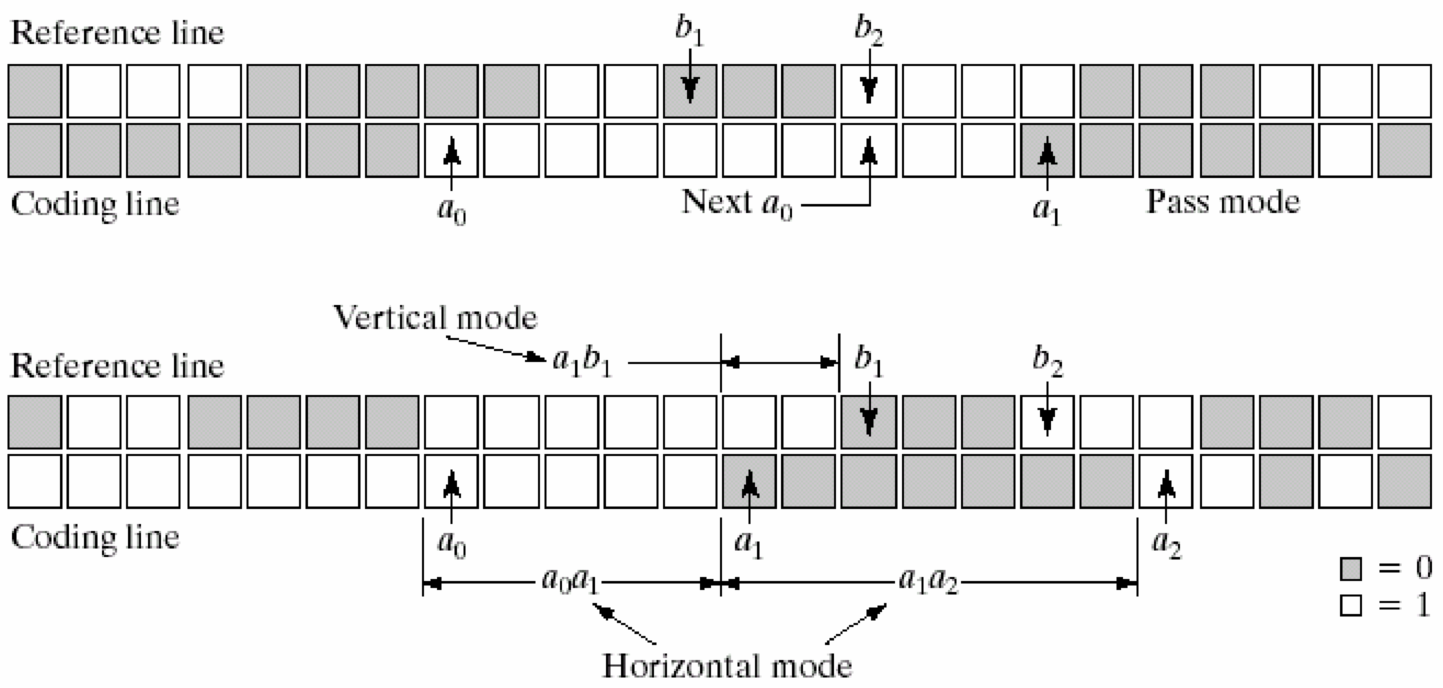


FIGURE 8.44 CCITT 2-D coding procedure. The notation $|a_1 b_1|$ denotes the absolute value of the distance between changing elements a_1 and b_1.



a
b

FIGURE 8.45
CCITT (a) pass mode and (b) horizontal and vertical mode coding parameters.

TABLE 8.16
CCITT two-dimensional code table.

Mode	Code Word
Pass	0001
Horizontal	$001 + M(a_0a_1) + M(a_1a_2)$
Vertical	
a_1 below b_1	1
a_1 one to the right of b_1	011
a_1 two to the right of b_1	000011
a_1 three to the right of b_1	0000011
a_1 one to the left of b_1	010
a_1 two to the left of b_1	000010
a_1 three to the left of b_1	0000010
Extension	0000001xxx

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
$-7, \ldots, -4, 4, \ldots, 7$	3	3
$-15, \ldots, -8, 8, \ldots, 15$	4	4
$-31, \ldots, -16, 16, \ldots, 31$	5	5
$-63, \ldots, -32, 32, \ldots, 63$	6	6
$-127, \ldots, -64, 64, \ldots, 127$	7	7
$-255, \ldots, -128, 128, \ldots, 255$	8	8
$-511, \ldots, -256, 256, \ldots, 511$	9	9
$-1023, \ldots, -512, 512, \ldots, 1023$	A	A
$-2047, \ldots, -1024, 1024, \ldots, 2047$	B	B
$-4095, \ldots, -2048, 2048, \ldots, 4095$	C	C
$-8191, \ldots, -4096, 4096, \ldots, 8191$	D	D
$-16383, \ldots, -8192, 8192, \ldots, 16383$	E	E
$-32767, \ldots, -16384, 16384, \ldots, 32767$	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

TABLE 8.19 JPEG default AC code (luminance) *(continues on next page).*

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	111111111000000	17
0/3	100	6	8/3	1111111110110111	19
0/4	1011	8	8/4	1111111110111000	20
0/5	11010	10	8/5	1111111110111001	21
0/6	111000	12	8/6	1111111110111010	22
0/7	1111000	14	8/7	1111111110111011	23
0/8	1111110110	18	8/8	1111111110111100	24
0/9	1111111110000010	25	8/9	1111111110111101	25
0/A	1111111110000011	26	8/A	1111111110111110	26
1/1	1100	5	9/1	111111000	10
1/2	111001	8	9/2	1111111110111111	18
1/3	1111001	10	9/3	1111111111000000	19
1/4	111110110	13	9/4	1111111111000001	20
1/5	11111110110	16	9/5	1111111111000010	21
1/6	1111111110000100	22	9/6	1111111111000011	22
1/7	1111111110000101	23	9/7	1111111111000100	23
1/8	1111111110000110	24	9/8	1111111111000101	24
1/9	1111111110000111	25	9/9	1111111111000110	25
1/A	1111111110001000	26	9/A	1111111111000111	26
2/1	11011	6	A/1	111111001	10
2/2	11111000	10	A/2	1111111111001000	18
2/3	1111110111	13	A/3	1111111111001001	19
2/4	1111111110001001	20	A/4	1111111111001010	20
2/5	1111111110001010	21	A/5	1111111111001011	21
2/6	1111111110001011	22	A/6	1111111111001100	22
2/7	1111111110001100	23	A/7	1111111111001101	23

Table 8.19 (Con't)

2/8	1111111110001101	24	A/8	1111111111001110	24
2/9	1111111110001110	25	A/9	1111111111001111	25
2/A	1111111110001111	26	A/A	1111111111010000	26
3/1	111010	7	B/1	111111010	10
3/2	111110111	11	B/2	1111111111010001	18
3/3	11111110111	14	B/3	1111111111010010	19
3/4	1111111110010000	20	B/4	1111111111010011	20
3/5	1111111110010001	21	B/5	1111111111010100	21
3/6	1111111110010010	22	B/6	1111111111010101	22
3/7	1111111110010011	23	B/7	1111111111010110	23
3/8	1111111110010100	24	B/8	1111111111010111	24
3/9	1111111110010101	25	B/9	1111111111011000	25
3/A	1111111110010110	26	B/A	1111111111011001	26
4/1	111011	7	C/1	1111111010	11
4/2	1111111000	12	C/2	1111111111011010	18
4/3	1111111110010111	19	C/3	1111111111011011	19
4/4	1111111110011000	20	C/4	1111111111011100	20
4/5	1111111110011001	21	C/5	1111111111011101	21
4/6	1111111110011010	22	C/6	1111111111011110	22
4/7	1111111110011011	23	C/7	1111111111011111	23
4/8	1111111110011100	24	C/8	1111111111100000	24
4/9	1111111110011101	25	C/9	1111111111100001	25
4/A	1111111110011110	26	C/A	1111111111100010	26

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.

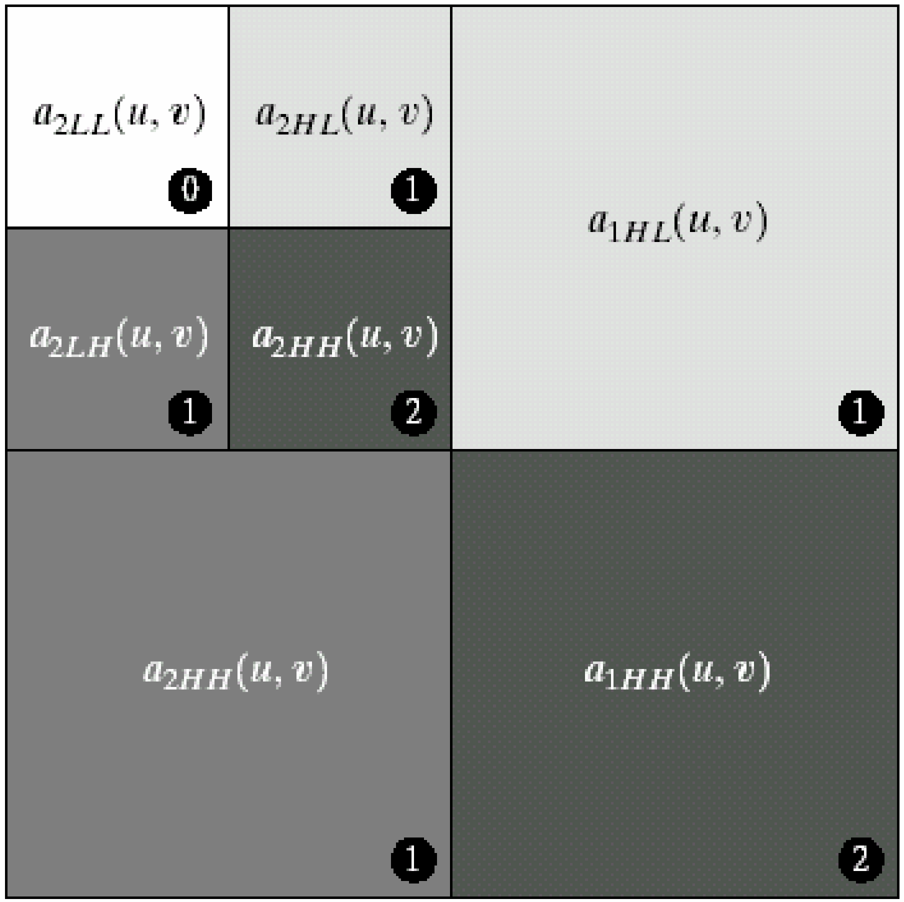


FIGURE 8.46 JPEG 2000 two-scale wavelet transform tile-component coefficient notation and analysis gain.

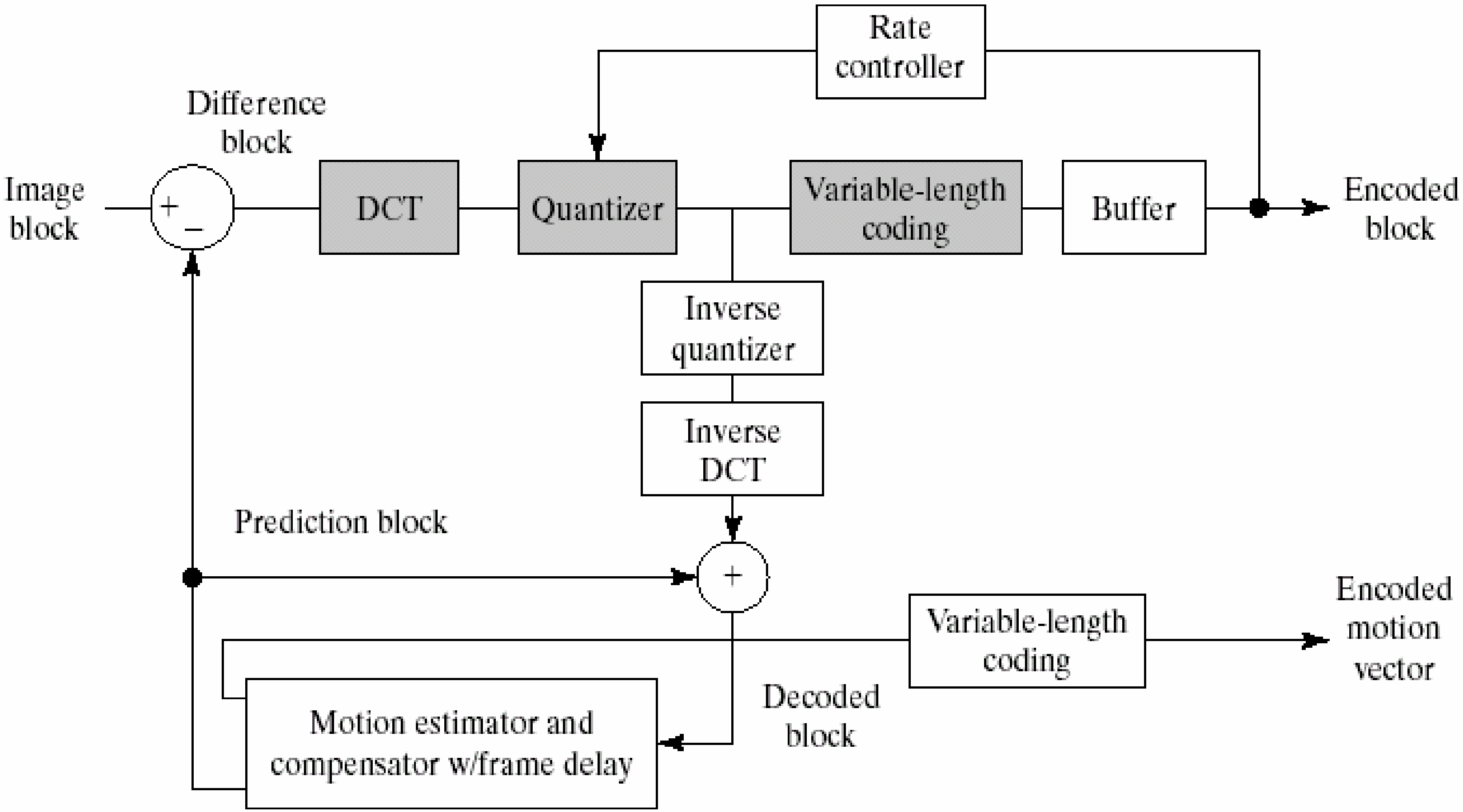


FIGURE 8.47 A basic DPCM/DCT encoder for motion compensated video compression.