Today's Agenda

- Image Compression

Image Compression

Why?

Standard definition (SD) television movie (raw data)

$$30\frac{\text{frames}}{\text{sec}}\times(720\times 480)\frac{\text{pixels}}{\text{frame}}\times 3\frac{\text{bytes}}{\text{pixel}}=31{,}104{,}000\text{bytes/sec}$$

A two-hour movie

$$31{,}104{,}000\frac{\text{bytes}}{\text{sec}}\times\left(60^2\right)\frac{\text{sec}}{\text{hour}}\times 2\text{hours}\approx 224\text{GB}$$

Need 27 8.5GB dual-layer DVDs!

High-definition (HD) television 1920x1080x24 bits/image!

Image Compression (Cont'd)

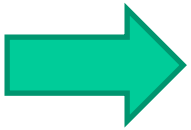
Standard definition (SD) television movie (raw data)

$$30\frac{\text{frames}}{\text{sec}}\times(720\times480)\frac{\text{pixels}}{\text{frame}}\times\frac{24\text{bits}}{\text{pixel}}=248{,}832{,}000\text{bit/sec}>200\text{Mbit/sec}$$

WAN modems	Ethernet LAN	WiFi WLAN	Mobile data

• 1972: Acoustic coupler 300 baud
• 1977: 1200 baud Vadic and Bell 212A
• 1986: ISDN introduced with two 64 kbit/s channels (160 kbit/s gross bit rate)
• 1990: v.32bis modems: 2400 / 4800 / 9600 / 19200 bit/s
• 1994: v.34 modems with 28.8 kbit/s
• 1995: v.90 modems with 56 kbit/s downstreams, 33.6 kbit/s upstreams
• 1999: v.92 modems with 56 kbit/s downstreams, 48 kbit/s upstreams
• 1998: ADSL up to 8 Mbit/s,
• 2003: ADSL2 up to 12 Mbit/s
• 2005: ADSL2+ up to 24 Mbit/s
	• 1972: IEEE 802.3 Ethernet 2.94 Mbit/s
	• 1985: 10b2 10 Mbit/s coax thinwire
	• 1990: 10bT 10 Mbit/s
	• 1995: 100bT 100 Mbit/s (125 Mbit/s gross bit rate)
	• 1999: 1000bT (Gigabit) 1 Gbit/s (1.25 Gbit/s gross bit rate)
	• 2003: 10GBASE 10 Gbit/s
		WiFi WLANs
		• 1997: 802.11 2 Mbit/s
		• 1999: 802.11b 11 Mbit/s
		• 1999: 802.11a 54 Mbit/s (72 Mbit/s gross bit rate)
		• 2003: 802.11g 54 Mbit/s (72 Mbit/s gross bit rate)
		• 2005: 802.11g (proprietary) 108 Mbit/s
		• 2007: 802.11n 600 Mbit/s
			• 1G:
			  • 1981: NMT 1200 bit/s
			• 2G:
			  • 1991: GSM CSD and D-AMPS 14.4 kbit/s
			  • 2003: GSM EDGE 296 kbit/s down, 118.4 kbit/s up
			• 3G:
			  • 2001: UMTS-FDD (WCDMA) 384 kbit/s
			  • 2007: UMTS HSDPA 14.4 Mbit/s
			  • 2008: UMTS HSPA 14.4 Mbit/s down, 5.76 Mbit/s up
			  • 2009: HSPA+ (Without MIMO) 28 Mbit/s downstreams (56 Mbit/s with 2x2 MIMO), 22 Mbit/s upstreams
			  • 2010: CDMA2000 EV-DO Rev. B 14.7 Mbit/s downstreams
			• Pre-4G:
			  • 2007: Mobile WiMAX (IEEE 802.16e) 144 Mbit/s down, 35 Mbit/s up.
			  • 2009: LTE 100 Mbit/s downstreams (360 Mbit/s with MIMO 2x2), 50 Mbit/s upstreams
			See also Comparison of mobile phone standards

http://en.wikipedia.org/wiki/Bit_rate



3.5Mbits/sec with MPEG-2 compression

Fundamentals

Data represent information – different ways

Representations that contain irrelevant or repeated information → contain redundant data

Two representations of the same information: b and b' bits, then the relative data redundancy

$$R = 1 - \frac{1}{C}, \quad \text{where}$$

$$C = \frac{b}{b'} \text{ is called the compression ratio}$$

In digital image processing, b is the # bits for the 2D array representation and b' is the compressed representation

Three Types of Data Redundancies

Coding redundancy

- Code/code book is a system to represent information
- Code length is the number of symbols in each code word
- Do we really need 8 bits to represent a gray-level pixel?

Spatial and temporal redundancy

- Neighboring (spatially or temporally) pixels usually have similar intensities!
- Do we need to represent every pixel?

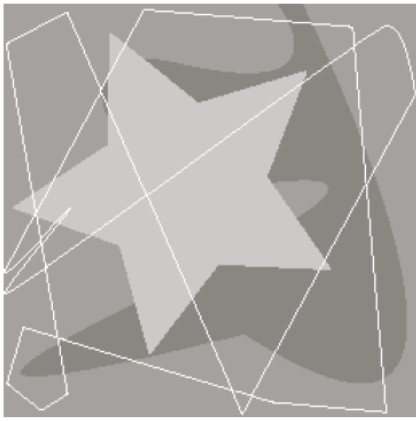
Irrelevant information

- Some image information can be ignored.

Coding Redundancy

Histogram

$$p_r(r_k) = \frac{n_k}{MN}, \quad k = 0,1,2,\ldots, L-1$$



Average # bits required to represent each pixel

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

↓

Number of bits representing each intensity level

Total bits MNL_{avg}

r_k	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_{87} = 87$	0.25	01010111	8	01	2
$r_{128} = 128$	0.47	10000000	8	1	1
$r_{186} = 186$	0.25	11000100	8	000	3
$r_{255} = 255$	0.03	11111111	8	001	3
r_k for $k \neq 87, 128, 186, 255$	0	—	8	—	0
		Fixed length		Variable length	

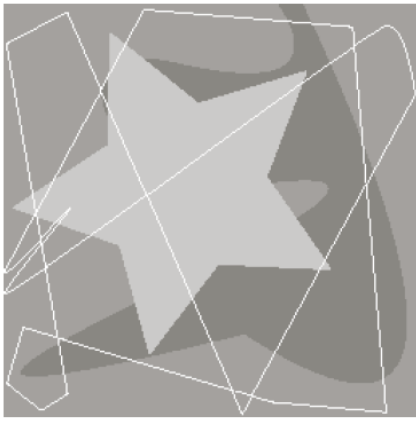
TABLE 8.1 Example of variable-length coding.

$$L_{avg} = 2 * 0.25 + 0.47 * 1 + 0.25 * 3 + 0.03 * 3 = 1.8$$

Coding Redundancy

TABLE 8.1
Example of variable-length coding.

r_k	$p_r(r_k)$	Code 1	$l_1(r_k)$	Code 2	$l_2(r_k)$
$r_{87} = 87$	0.25	01010111	8	01	2
$r_{128} = 128$	0.47	10000000	8	1	1
$r_{186} = 186$	0.25	11000100	8	000	3
$r_{255} = 255$	0.03	11111111	8	001	3
r_k for $k \neq 87, 128, 186, 255$	0	—	8	—	0

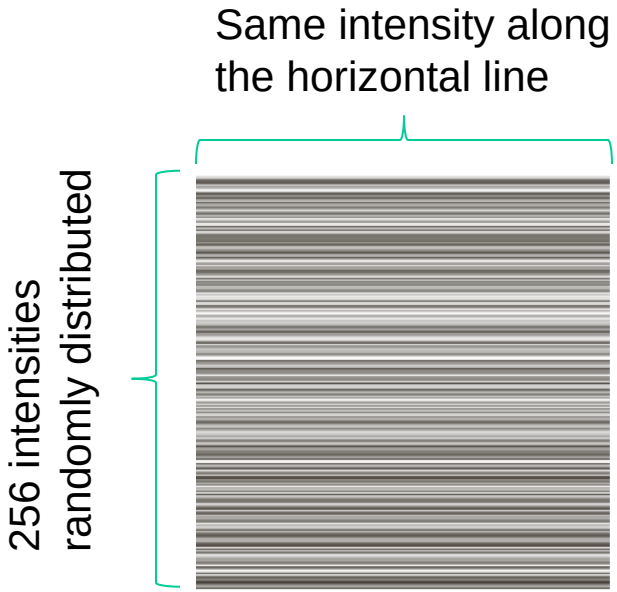


C and R with variable length coding?

$$C = \frac{8}{1.81} = 4.42$$

$$R = 1 - \frac{1}{C} = 0.77$$

Spatial and Temporal Redundancy



The histogram is uniform.

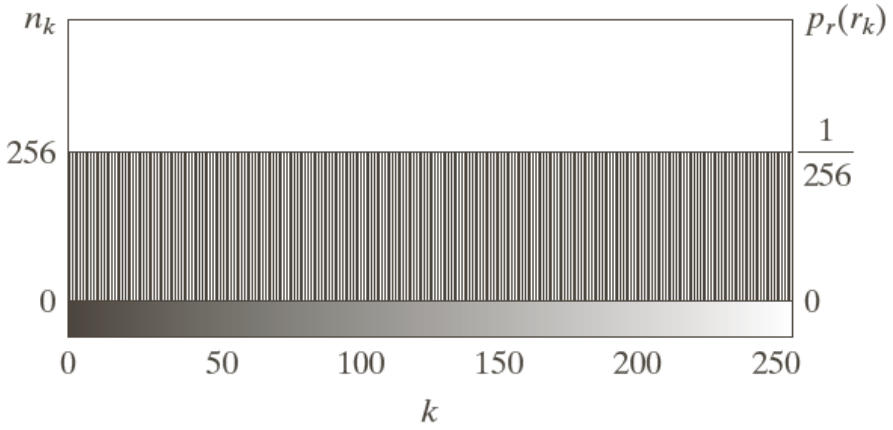
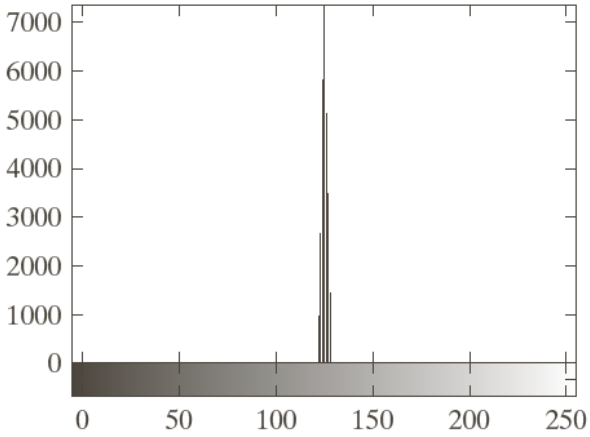


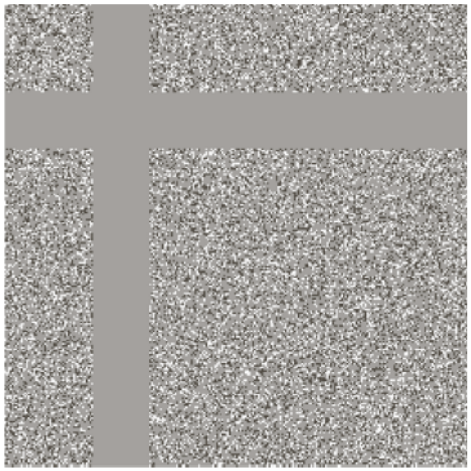
FIGURE 8.2 The intensity histogram of the image in Fig. 8.1(b).

Compression by mapping:

- Run-length coding:
  - one word representing the intensity, and one word representing the length
  - C and R?
- Difference between two neighboring pixels

Irrelevant Information





a b

FIGURE 8.3
(a) Histogram of the image in Fig. 8.1(c) and (b) a histogram equalized version of the image.

Represented by a single byte → More details

Quantization – loss of quantitative information: irreversible operation

Measuring Image Information

Minimum amount of data without losing information?

A random event E with probability $P(E)$ contains information

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

Entropy (average information per image intensity)

$$H = -\sum_{k=0}^{L-1} p_r(r_k) \log_2 p_r(r_k)$$

Shannon's first theorem (noiseless coding theorem)

$$L_{\text{avg}} \geq H$$

and the low-bound *H* can be achieved by a coding method

Fidelity Criteria – Quantify the Loss

objective fidelity criteria

Root mean square error

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

Mean-square signal to noise ratio

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

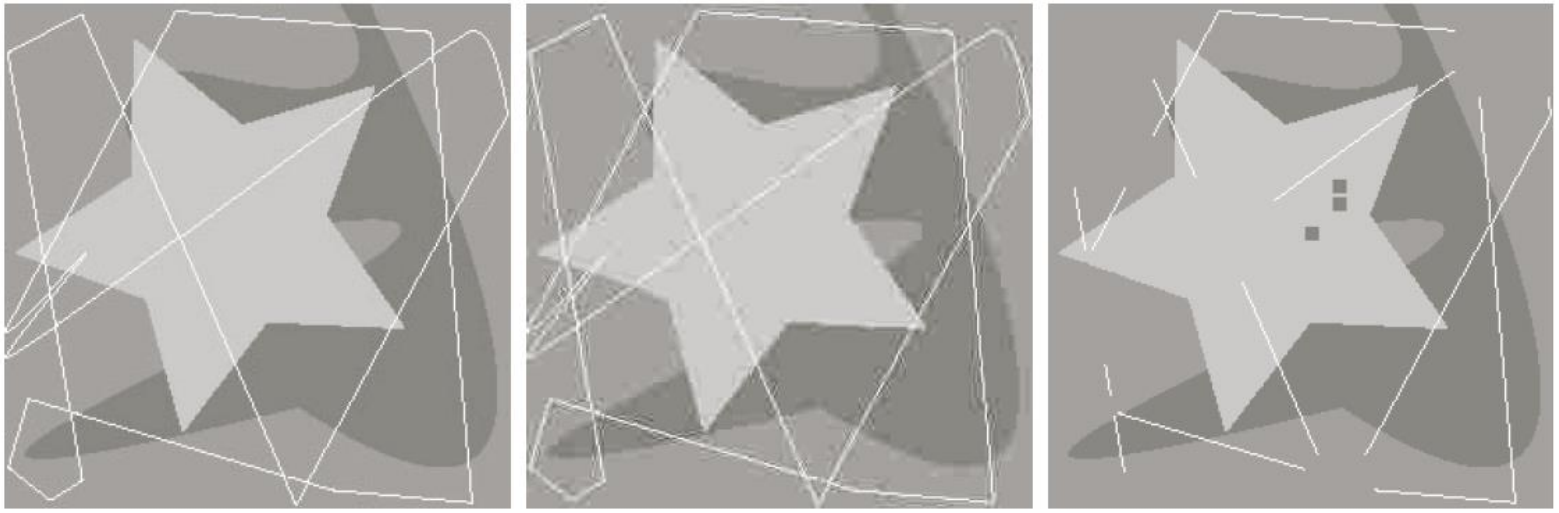
Subjective Fidelity Criteria

TABLE 8.2
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.

Inconsistency between Objective and Subjective Fidelity Criteria

Objective: rms = 5.17 rms = 15.67 rms = 14.17



Subjective: Excellent Passable unusable

a b c

FIGURE 8.4 Three approximations of the image in Fig. 8.1(a).

Image-Compression Models

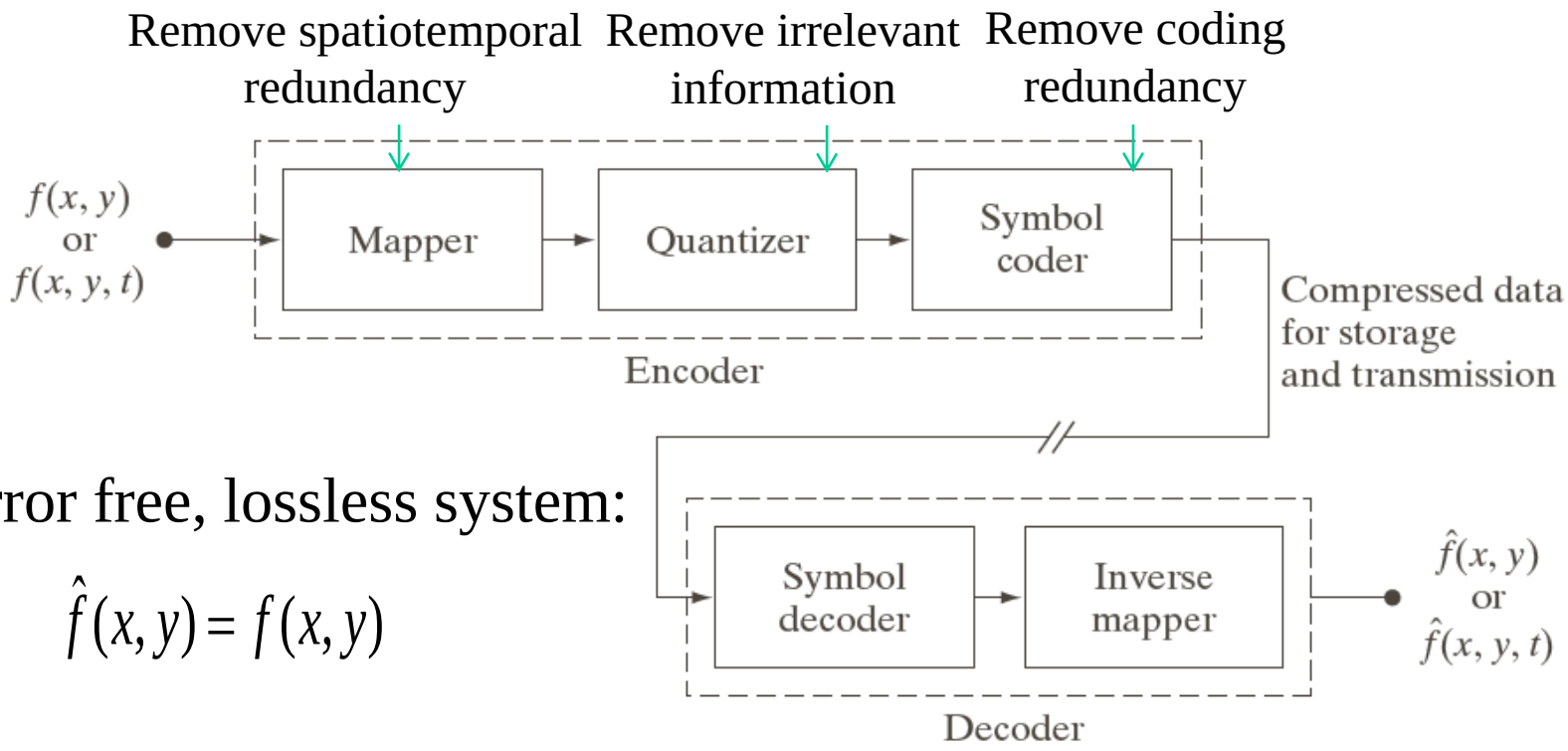


FIGURE 8.5 Functional block diagram of a general image compression system.

Error free, lossless system:

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

Lossy system:

$$\hat{f}(x,y) \neq f(x,y)$$

Image Formats, Containers and Compression Standards

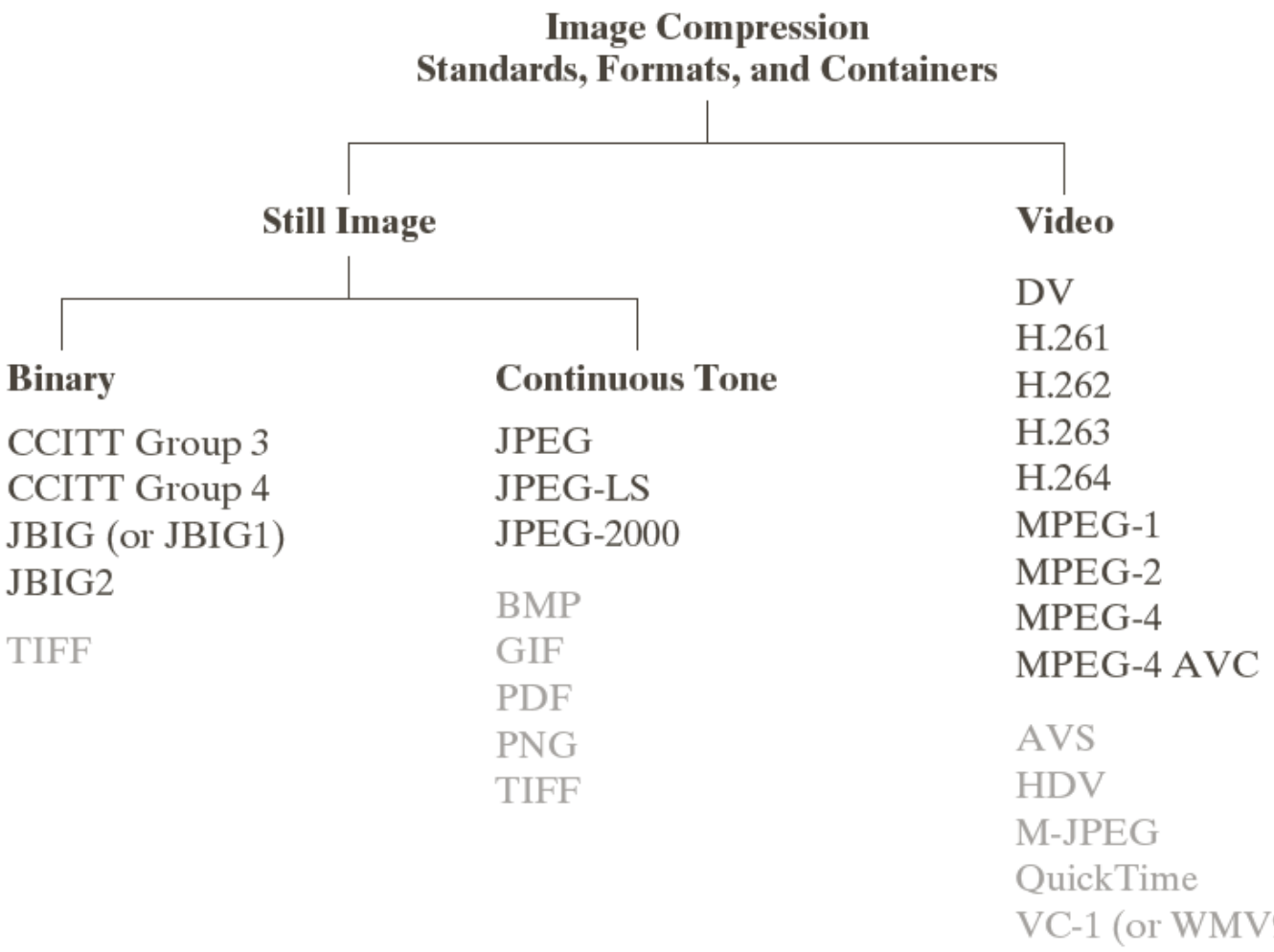


FIGURE 8.6 Some popular image compression standards, file formats, and containers. Internationally sanctioned entries are shown in black; all others are grayed.

Some Basic Compression Methods – Huffman Coding (Block Code)

- Remove coding redundancy
- Used widely in CCITT group3, JBIG2, JPEG, and MPEG
- Create the optimal code for a set of symbols

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 8.7 Huffman source reductions.

Huffman Coding – Cont.

Original source			Source reduction							
Symbol	Probability	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								

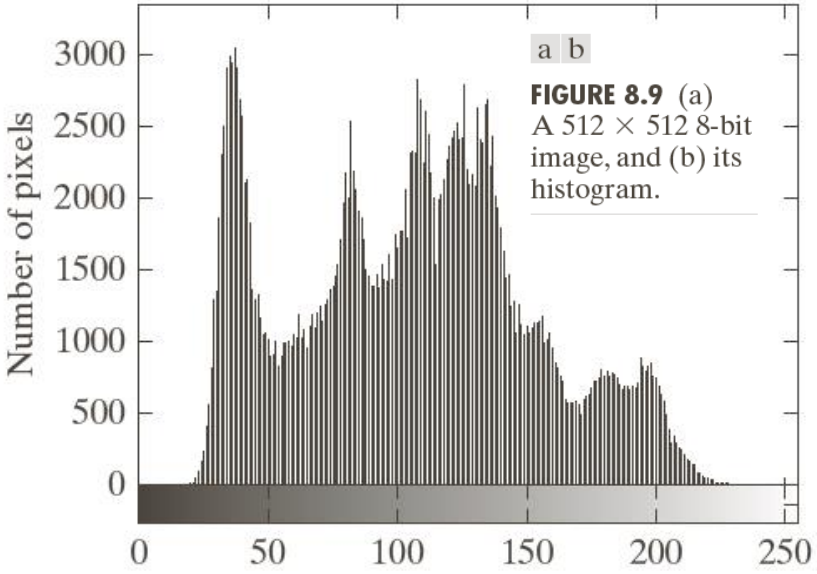
FIGURE 8.8 Huffman code assignment procedure.

$$L_{\text{avg}} = (0.4)(1) + (0.3)(2) + (0.1)(3) + (0.1)(4) + (0.06)(5) + (0.04)(5) = 2.2\text{bits/pixel}$$

$$H = 2.14\text{bits/pixel}$$

- **Block code**: each source symbol is represented by a fixed code symbol
- **Instantaneous**: lookup table
- **Uniquely decodable**: extract symbols in a left-to-right manner

Example



7.428 bits/pixel

In practice, a pre-computed Huffman coding table is used (e.g., JPEG and MPEG)

Arithmetic Coding -- Nonblock Code

- Used in JBIG, JBIG2, JPEG2000, and MPEG4
- Non-block: the whole message is encoded into a single code word (real value in [0, 1])

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.

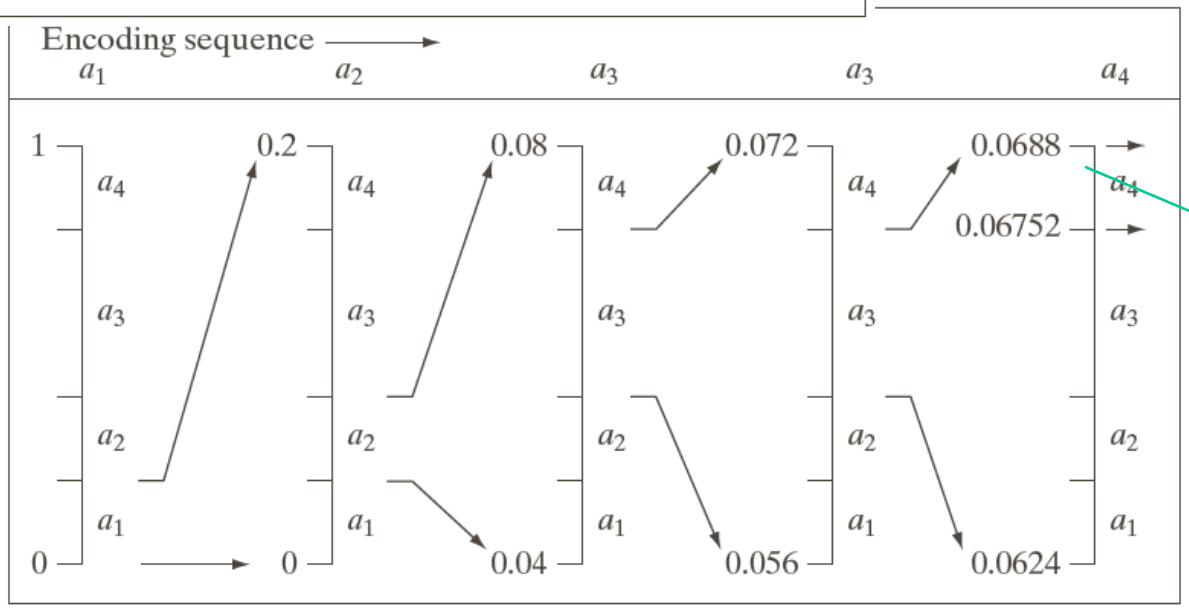


FIGURE 8.12
Arithmetic coding procedure.

Any number in the range can be used

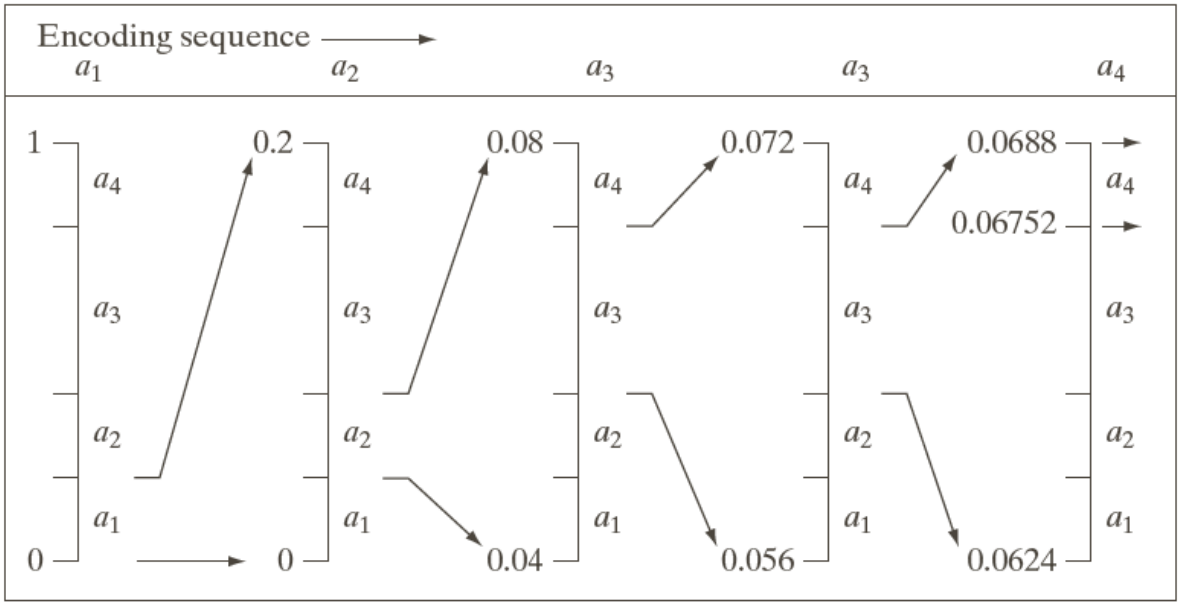
Arithmetic Coding

Decoding

- final value
- probabilities of the input symbols

Two decoding methods:

- Straightforward decoding



Arithmetic Coding

Decoding

- final value
- probabilities of the input symbols

Two decoding methods:

- Straightforward decoding
- An efficient method

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)

Step0: $v_t = v_0$

Repeat:

step1: find symbol s_t satisfying $low(s_t) \le v_t \le up(s_t)$

step2: $$v_{t+1} = \frac{v_t - low(s_t)}{p(s_t)}$$

Until: s_t is the end symbol

Arithmetic Coding

Require an end-of-message indicator

Potential issues:

- Decoding starts when all the message is received
- Sensitive to the noise during transmission
- Limited by the precision – solved by scaling

Run-Length Encoding (RLE)

- Eliminate spatial redundancy – groups of identical symbols
- Used in CCITT, JPEG, and fax machines

Encoded Mode: two bytes

# of consecutive pixels, the color index

Absolute Mode: two bytes

0, conditional signals

Second Byte Value	Condition
0	End of line
1	End of image
2	Move to a new position
3–255	Specify pixels individually

TABLE 8.8
BMP absolute coding mode options. In this mode, the first byte of the BMP pair is 0.

Run-Length Encoding (RLE)

C=0.98

C=1.35

C=128

- Effective on binary images (black and white)
- Not effective on natural images - increase the file size if there are few runs of identical symbols
- Combine with other variable-length coding

Motion Compensation

Eliminate temporal redundancy

- Step1 : Capture differences between a reference frame and the target frame
- Step2 : Model the difference in terms of transformation function
- Step3 : Video/image sequences are synthesized by the reference frame according to the estimate transformation

Global motion compensation

- Few parameters; no partition;
- Static objects; not applicable for moving objects

Local motion compensation

- Block motion compensation
- Overlapped block motion compensation

Reading Assignments

Other methods covered in Chapter 8.2