

CPE409 Image Processing

Part 10

Image Compression

Assist. Prof. Dr. Caner ÖZCAN

But life is short and information endless ... Abbreviation is a necessary evil and the abbreviator's business is to make the best of a job which, although intrinsically bad, is still better than nothing. ~Aldous Huxley

Outline

8. Image Compression

- ▶ Fundamentals
 - ▶ Some Basic Compression Methods (Huffman Coding)
- Digital Image Watermarking

Relative Data Redundancy

- ▶ Let b and b' denote the number of bits in two representations of the same information, the relative data redundancy R is

$$R = 1 - 1/C$$

C is called the compression ratio, defined as

$$C = b/b'$$

e.g., $C = 10$, the corresponding relative data redundancy of the larger representation is 0.9, indicating that 90% of its data is redundant

Why do we need compression?

- ▶ Data storage
- ▶ Data transmission

How can we implement compression?

▶ **Coding redundancy**

Most 2-D intensity arrays contain more bits than are needed to represent the intensities

▶ **Spatial and temporal redundancy**

Pixels of most 2-D intensity arrays are correlated spatially and video sequences are temporally correlated

▶ **Irrelevant information**

Most 2-D intensity arrays contain information that is ignored by the human visual system

Examples of Redundancy



FIGURE 8.1 Computer generated $256 \times 256 \times 8$ bit images with (a) coding redundancy, (b) spatial redundancy, and (c) irrelevant information. (Each was designed to demonstrate one principal redundancy but may exhibit others as well.)

Image Compression Standards

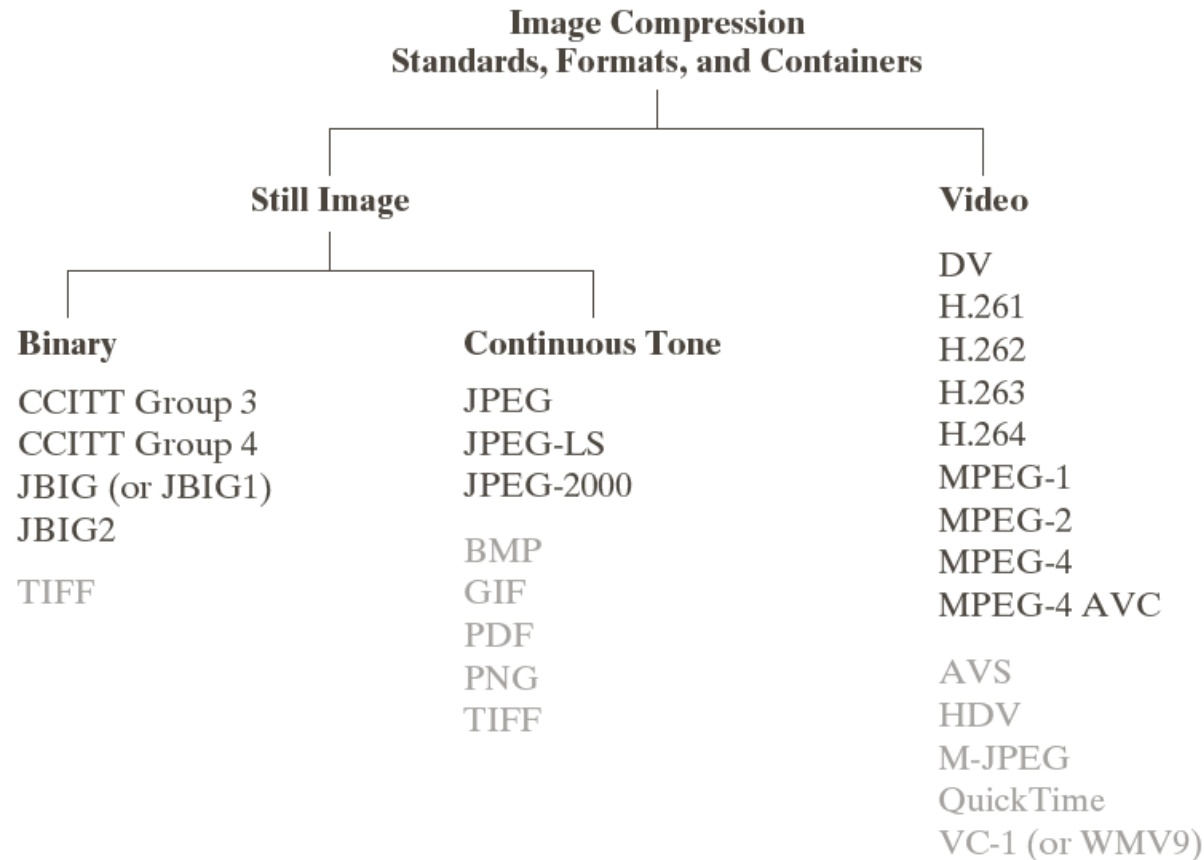


FIGURE 8.6 Some popular image compression standards, file formats, and containers. Internationally sanctioned entries are shown in black; all others are grayed.

Name	Organization	Description
<i>Bi-Level Still Images</i>		
CCITT Group 3	ITU-T	Designed as a facsimile (FAX) method for transmitting binary documents over telephone lines. Supports 1-D and 2-D run-length [8.2.5] and Huffman [8.2.1] coding.
CCITT Group 4	ITU-T	A simplified and streamlined version of the CCITT Group 3 standard supporting 2-D run-length coding only.
JBIG or JBIG1	ISO/IEC/ ITU-T	A <i>Joint Bi-level Image Experts Group</i> standard for progressive, lossless compression of bi-level images. Continuous-tone images of up to 6 bits/pixel can be coded on a bit-plane basis [8.2.7]. Context sensitive arithmetic coding [8.2.3] is used and an initial low resolution version of the image can be gradually enhanced with additional compressed data.
JBIG2	ISO/IEC/ ITU-T	A follow-on to JBIG1 for bi-level images in desktop, Internet, and FAX applications. The compression method used is content based, with dictionary based methods [8.2.6] for text and halftone regions, and Huffman [8.2.1] or arithmetic coding [8.2.3] for other image content. It can be lossy or lossless.
<i>Continuous-Tone Still Images</i>		
JPEG	ISO/IEC/ ITU-T	A <i>Joint Photographic Experts Group</i> standard for images of photographic quality. Its lossy <i>baseline coding system</i> (most commonly implemented) uses quantized discrete cosine transforms (DCT) on 8×8 image blocks [8.2.8], Huffman [8.2.1], and run-length [8.2.5] coding. It is one of the most popular methods for compressing images on the Internet.
JPEG-LS	ISO/IEC/ ITU-T	A lossless to near-lossless standard for continuous tone images based on adaptive prediction [8.2.9], context modeling [8.2.3], and Golomb coding [8.2.2].
JPEG-2000	ISO/IEC/ ITU-T	A follow-on to JPEG for increased compression of photographic quality images. Arithmetic coding [8.2.3] and quantized discrete wavelet transforms (DWT) [8.2.10] are used. The compression can be lossy or lossless.

TABLE 8.3
Internationally sanctioned image compression standards. The numbers in brackets refer to sections in this chapter.

(Continues)

Name	Organization	Description
<i>Video</i>		
DV	IEC	<i>Digital Video</i> . A video standard tailored to home and semiprofessional video production applications and equipment—like electronic news gathering and camcorders. Frames are compressed independently for uncomplicated editing using a DCT-based approach [8.2.8] similar to JPEG.
H.261	ITU-T	A two-way videoconferencing standard for ISDN (<i>integrated services digital network</i>) lines. It supports non-interlaced 352×288 and 176×144 resolution images, called CIF (<i>Common Intermediate Format</i>) and QCIF (<i>Quarter CIF</i>), respectively. A DCT-based compression approach [8.2.8] similar to JPEG is used, with frame-to-frame prediction differencing [8.2.9] to reduce temporal redundancy. A block-based technique is used to compensate for motion between frames.
H.262	ITU-T	See MPEG-2 below.
H.263	ITU-T	An enhanced version of H.261 designed for ordinary telephone modems (i.e., 28.8 Kb/s) with additional resolutions: SQCIF (<i>Sub-Quarter CIF</i> 128×96), 4CIF (704×576), and 16CIF (1408×512).
H.264	ITU-T	An extension of H.261–H.263 for videoconferencing, Internet streaming, and television broadcasting. It supports prediction differences within frames [8.2.9], variable block size integer transforms (rather than the DCT), and context adaptive arithmetic coding [8.2.3].
MPEG-1	ISO/IEC	A <i>Motion Pictures Expert Group</i> standard for CD-ROM applications with non-interlaced video at up to 1.5 Mb/s. It is similar to H.261 but frame predictions can be based on the previous frame, next frame, or an interpolation of both. It is supported by almost all computers and DVD players.
MPEG-2	ISO/IEC	An extension of MPEG-1 designed for DVDs with transfer rates to 15 Mb/s. Supports interlaced video and HDTV. It is the most successful video standard to date.
MPEG-4	ISO/IEC	An extension of MPEG-2 that supports variable block sizes and prediction differencing [8.2.9] within frames.
MPEG-4 AVC	ISO/IEC	MPEG-4 Part 10 <i>Advanced Video Coding</i> (AVC). Identical to H.264 above.

Name	Organization	Description
<i>Continuous-Tone Still Images</i>		
BMP	Microsoft	<i>Windows Bitmap</i> . A file format used mainly for simple uncompressed images.
GIF	CompuServe	<i>Graphic Interchange Format</i> . A file format that uses lossless LZW coding [8.2.4] for 1- through 8-bit images. It is frequently used to make small animations and short low resolution films for the World Wide Web.
PDF	Adobe Systems	<i>Portable Document Format</i> . A format for representing 2-D documents in a device and resolution independent way. It can function as a container for JPEG, JPEG 2000, CCITT, and other compressed images. Some PDF versions have become ISO standards.
PNG	World Wide Web Consortium (W3C)	<i>Portable Network Graphics</i> . A file format that losslessly compresses full color images with transparency (up to 48 bits/pixel) by coding the difference between each pixel's value and a predicted value based on past pixels [8.2.9].
TIFF	Aldus	<i>Tagged Image File Format</i> . A flexible file format supporting a variety of image compression standards, including JPEG, JPEG-LS, JPEG-2000, JBIG2, and others.
<i>Video</i>		
AVS	MII	<i>Audio-Video Standard</i> . Similar to H.264 but uses exponential Golomb coding [8.2.2]. Developed in China.
HDV	Company consortium	<i>High Definition Video</i> . An extension of DV for HD television that uses MPEG-2 like compression, including temporal redundancy removal by prediction differencing [8.2.9].
M-JPEG	Various companies	<i>Motion JPEG</i> . A compression format in which each frame is compressed independently using JPEG.
Quick-Time	Apple Computer	A media container supporting DV, H.261, H.262, H.264, MPEG-1, MPEG-2, MPEG-4, and other video compression formats.
VC-1 WMV9	SMPTE Microsoft	The most used video format on the Internet. Adopted for HD and <i>Blu-ray</i> high-definition DVDs. It is similar to H.264/AVC, using an integer DCT with varying block sizes [8.2.8 and 8.2.9] and context dependent variable-length code tables [8.2.1]—but no predictions within frames.

Some Basic Compression Methods: Huffman Coding

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	
a_1	0.1	0.1	0.2	0.3	0.4
a_4	0.1	0.1			
a_3	0.06	0.1	0.1	0.1	0.1
a_5	0.04				

FIGURE 8.7
Huffman source reductions.

Some Basic Compression Methods: Huffman Coding

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.

The average length of this code is

$$\begin{aligned}
 L_{avg} &= 0.4 * 1 + 0.3 * 2 + 0.1 * 3 + 0.1 * 4 + 0.06 * 5 + 0.04 * 5 \\
 &= 2.2 \text{ bits/pixel}
 \end{aligned}$$

Huffman Example

- Suppose you have a message consisting of 6 symbols.
- Each symbol has a width of 8 bits.
- The length of the message is 100 symbols. $(8 \times 100) = 800$ bits.
- The following table gives the frequency of symbols.

Symbol (Sembol)	frequency	Probability (Olasılık)
a_1	8	0.08
a_2	7	0.07
a_3	19	0.19
a_4	45	0.45
a_5	3	0.03
a_6	18	0.18

Huffman Example

Sembol	Code	Probability Olasılık	1	2	3	4
a ₄	1	0.45	0.45	0.45	0.45	<div>+</div> 0.55
a ₃	01	0.19	0.19	0.19	<div>+</div> 0.36	0.45
a ₆	000	0.18	0.18	0.18	0.19	
a ₁	0011	0.08	<div>+</div> 0.1	<div>+</div> 0.18		
a ₂	00100	0.07	0.08			
a ₅	00101	0.03				

Huffman Example

Notice that in the first case the probability of 0.55 is expressed as 0 and the probability of 0.45 is 1.

Symbol Sembol	Code	Probabilty Olasılık	1	2	3	4
a ₄	1	0.45	0.45	0.45	0.45	0.55 (0)
a ₃	01	0.19	0.19	0.19	0.36 (00)	0.45 (1)
a ₆	000	0.18	0.18	0.18 (000)	0.19 (01)	
a ₁	0011	0.08	0.1 (0010)	0.18 (001)		
a ₂	00100	0.07 (00100)	0.08 (0011)			
a ₅	00101	0.03 (00101)				

Huffman Example

- ▶ Original message length $b = 8 \text{ bits} * 100 = 800 \text{ bits}$
- ▶ Huffman encoded message length

$$b' = \text{length}(a_1) * \text{freq}(a_1) + \text{length}(a_2) * \text{freq}(a_2) + \text{length}(a_3) * \text{freq}(a_3) + \text{length}(a_4) * \text{freq}(a_4) + \text{length}(a_5) * \text{freq}(a_5) + \text{length}(a_6) * \text{freq}(a_6)$$

$$b' = 4*8 + 5*7 + 2*19 + 1*45 + 5*3 + 3*18 = 219$$

$$\text{compression ratio } c = \frac{b}{b'}$$

$$c = \frac{800}{219} = 3.65$$

$$\text{Redundancy } R = 1 - \frac{1}{c} = 0.72$$

If you use fixed length encoding, the message length is $3 \text{ bits} * 100 = 300 \text{ bits}$.

This means that variable-length code is better.

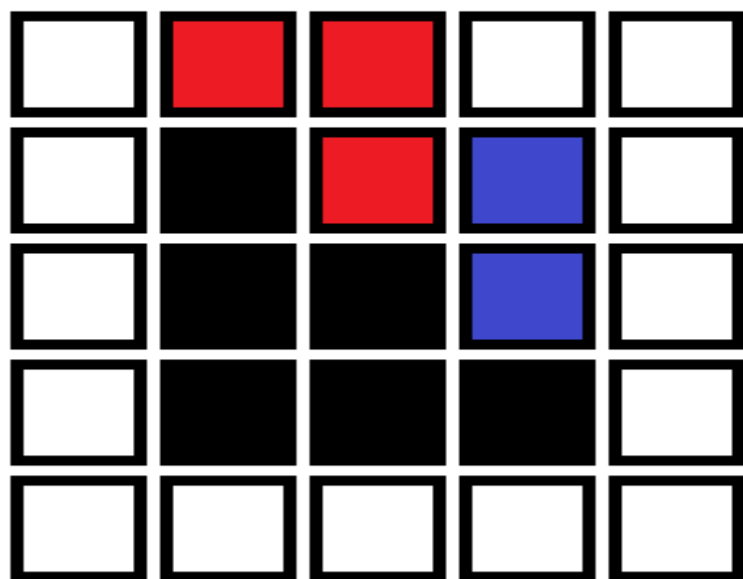
Symbol Sembol	code	Length In bits	freq
a_4	1	1	45
a_3	01	2	19
a_6	000	3	18
a_1	0011	4	8
a_2	00100	5	7
a_5	00101	5	3

Huffman Coding – Base of JPEG Image Compression

In 1952 David Huffman, a graduate student at the famous Massachusetts Institute of Technology developed an elegant algorithm for lossless compression as part of his schoolwork. The algorithm is now known as Huffman coding.

Huffman coding can be used to compress all sorts of data. It is an entropy-based algorithm that relies on an analysis of the frequency of symbols in an array.

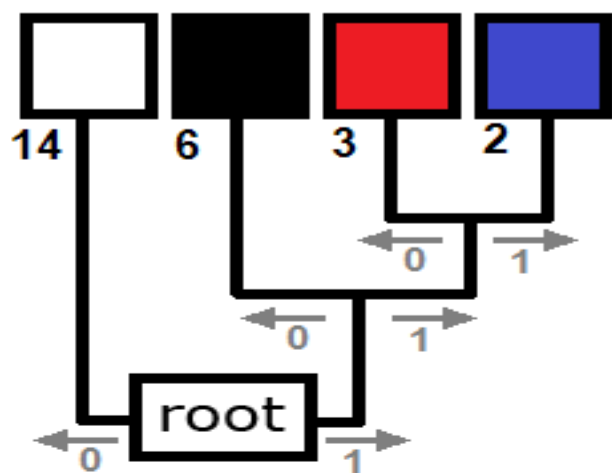
Huffman coding can be demonstrated most vividly by compressing a raster image. Suppose we have a 5×5 raster image with 8-bit color, i.e. 256 different colors. The uncompressed image will take $5 \times 5 \times 8 = 200$ bits of storage.






First, we count up how many times each color occurs in the image. Then we sort the colors in order of decreasing frequency. We end up with a row that looks like this:



Now we put the colors together by building a tree such that the colors farthest from the root are the least frequent. The colors are joined in pairs, with a node forming the connection. A node can connect either to another node or to a color. In our example, the tree might look like this:

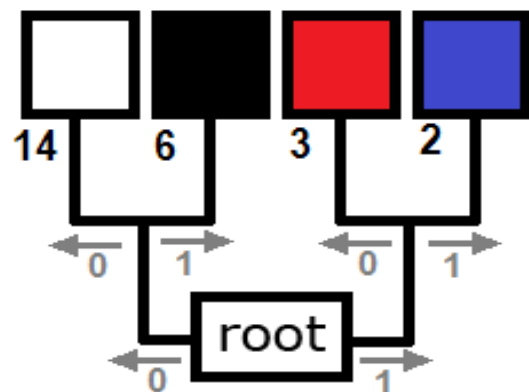


Our result is known as a Huffman tree. It can be used for encoding and decoding. Each color is encoded as follows. We create codes by moving from the root of the tree to each color. If we turn right at a node, we write a 1, and if we turn left – 0. This process yields a Huffman code table in which each symbol is assigned a bit code such that the most frequently occurring symbol has the shortest code, while the least common symbol is given the longest code.

color	freq.	bit code
	14	0
	6	10
	3	110
	2	111

Huffman Coding – Base of JPEG Image Compression

The Huffman tree and code table we created are not the only ones possible. An alternative Huffman tree that looks like this could be created for our image:



The corresponding code table would then be:

color	freq.	bit code
	14	00
	6	01
	3	10
	2	11

Using the variant is preferable in our example. This is because it provides better compression for our specific image.

Huffman Coding – Base of JPEG Image Compression

Using the variant is preferable in our example. This is because it provides better compression for our specific image.

Because each color has a unique bit code that is not a prefix of any other, the colors can be replaced by their bit codes in the image file. The most frequently occurring color, white, will be represented with just a single bit rather than 8 bits. Black will take two bits. Red and blue will take three. After these replacements are made, the 200-bit image will be compressed to $14 \times 1 + 6 \times 2 + 3 \times 3 + 2 \times 3 = 41$ bits, which is about 5 bytes compared to 25 bytes in the original image.

References

- ▶ Sayısal Görüntü İşleme, Palme Publishing, Third Press Trans. (Orj: R.C. Gonzalez and R.E. Woods: "Digital Image Processing", Prentice Hall, 3rd edition, 2008).
- ▶ "Digital Image Processing Using Matlab", Gonzalez & Richard E. Woods, Steven L. Eddins, Gatesmark Publishing, 2009
- ▶ Lecture Notes, CS589-04 Digital Image Processing, Frank (Qingzhong) Liu, <http://www.cs.nmt.edu/~ip>
- ▶ Lecture Notes, BIL717-Image Processing, Erkut Erdem
- ▶ Lecture Notes, EBM537-Image Processing, F.Karabiber
- ▶ Huffman Coding – Base of JPEG Image Compression, <https://www.print-driver.com/stories/huffman-coding-jpeg>
- ▶ <https://docs.opencv.org/>
- ▶ Bekir Aksoy, Python ile İmgeden Veriye Görüntü İşleme ve Uygulamaları, Nobel Akademik Yayıncılık