Digital Image Processing

Image Compression

Background

Principal objective:

To minimize the number of bits required to represent an image.

Applications

Transmission:

Broadcast TV via satellite, military communications via aircraft, teleconferencing, computer communications etc.

Storage:

Educational and business documents, medical images (CT, MRI and digital radiology), motion pictures, satellite images, weather maps, geological surveys, ...

Overview

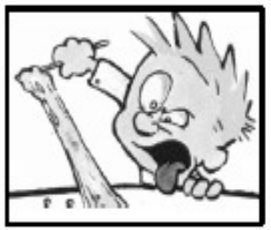
Image data compression methods fall into two common categories:

I. Information preserving compression

- Especial for image archiving (storage of legal or medical records)
- Compress and decompress images without losing information

II. Lossy image compression

- Provide higher levels of data reduction
- Result in a less than perfect reproduction of the original image
- Applications: *–broadcast television, videoconferencing*



compression

102900910-23155
70291-787418029
809578759187582
745187598475198
751878758457109
-58507905750905

original image

"compact information"
(for storage or transmission)

(loss less compression)

decompression

(lossy compression)

approximation of the original image

Data vs. information

- *Data* is not the same thing as *information*
- Data are the means to convey information; various amounts of data may be used to represent the same amount of information Part of data may provide no relevant information: data redundancy
- The amount of data can be much larger expressed than the amount of information.

Data Redundancy

- Data that provide no relevant information=*redundant data* or *redundancy*.

- Image compression techniques can be designed by

  reducing or eliminating the Data Redundancy

- Image coding or compression has a goal to reduce the amount of data by reducing the amount of redundancy.

Data Redundancy

Let n_1 and n_2 refer to amounts of data in two data sets that carry the same information

Compression ratio: $C_R = \frac{n_1}{n_2}$

Relative data redundancy: $R_D = 1 - \frac{1}{C_R} = 1 - \frac{n_2}{n_1}$
(of the first data set, n_1)

- if $n_1 = n_2$, $C_R = 1$ and $R_D = 0$, relative to the second data set, the first set contains no redundant data
- if $n_1 >> n_2$, $C_R \to \infty$, and $R_D \to 1$, relative to the second data set, the first set contains highly redundant data
- if $n_1 << n_2$, $C_R \to 0$, and $R_D \to -\infty$, relative to the second data set, the first set is highly compressed

$C_R = 10$ means 90% of the data in the first data set is redundant

Data Redundancy

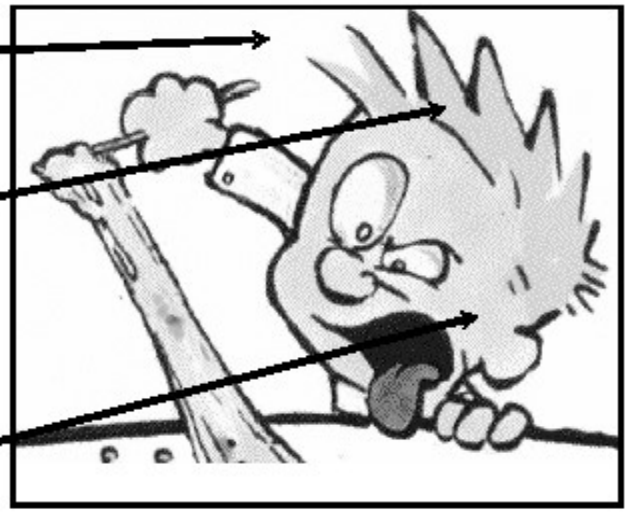
Three basic data redundancies

- Coding Redundancy
- Interpixel Redundancy
- Psychovisual Redundancy

CR: some graylevels are more common than others

IR: the same graylevel covers large areas

PVR: the eye can only resolve 32 graylevels locally



Coding Redundancy

A natural m-bit coding method assigns m-bit to each gray level without considering the probability that gray level occurs with: Very likely to contain coding redundancy

Basic concept:

- Utilize the probability of occurrence of each gray level (histogram) to determine length of code representing that particular gray level: variable-length coding.
- Assign shorter code words to the gray levels that occur most frequently or vice versa.

Coding Redundancy

Let $0 \le r_k \le 1$: gray levels (discrete random variable)

$p_r(r_k)$: probability of occurrence of r_k

n_k : number of pixels that r_k appears in the image

n : total number of pixels in an image

L : number of gray levels

$l(r_k)$: number of bits used to represent r_k

L_{avg} : average length of code words assigned to the grey levels

$$L_{avg} = \sum_{k=0}^{L-1} l(r_k) p_r(r_k) \quad \text{where} \quad p_r(r_k) = \frac{n_k}{n}, k = 0,1,....., L-1$$

Hence, total number of bits required to code an $M \times N$ image is MNL_{avg}

For a natural m-bit coding $L_{avg} = m$.

Coding Redundancy (Example)

Let us assume an 8-level image

8-levels: can be represented by 3-bits, m = 3.

Normalized histogram of the image is given by:

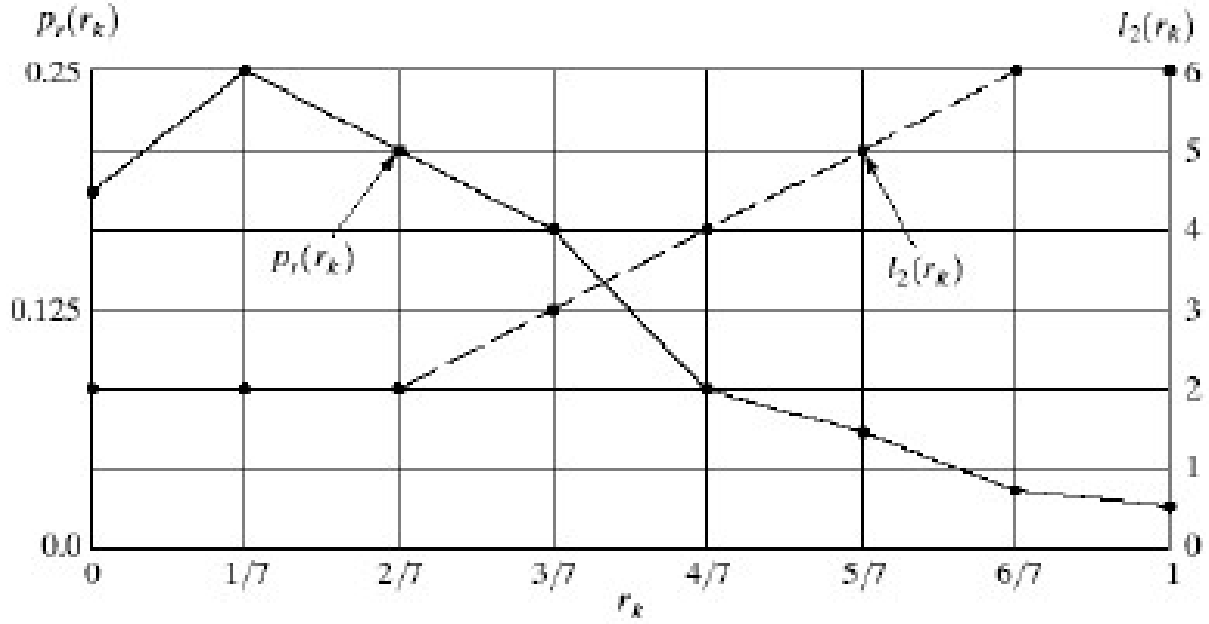


FIGURE 8.1 Graphic representation of the fundamental basis of data compression through variable-length coding.

Coding Redundancy (Example)

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.

$$L_{avg_1} = m = 3 \text{ bits}, \quad \text{and}$$

$$L_{avg_2} = \sum_{k=0}^{7} l_2(r_k) p_r(r_k)$$

$$= 2(0.19) + 2(0.25) + 2(0.31) + 3(0.16)$$

$$+ 4(0.08) + 5(0.06) + 6(0.03) + 6(0.02) = 2.7 \text{ bits}$$

Coding Redundancy (Example)

Example 3-bit image:

gray level r_k	probability $p(r_k)$	source	code
0	0.1	000	01
1	0.4	001	0
2	0.03	010	11
3	0.05	011	10
4	0.3	100	1
5	0.1	101	00
6	0.01	110	111
7	0.01	111	000

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

source : L_{avg}=(constant $l(r_k)$=3)=3*1=3

code: L_{avg}=0.1*2+0.4*1+…=1.32

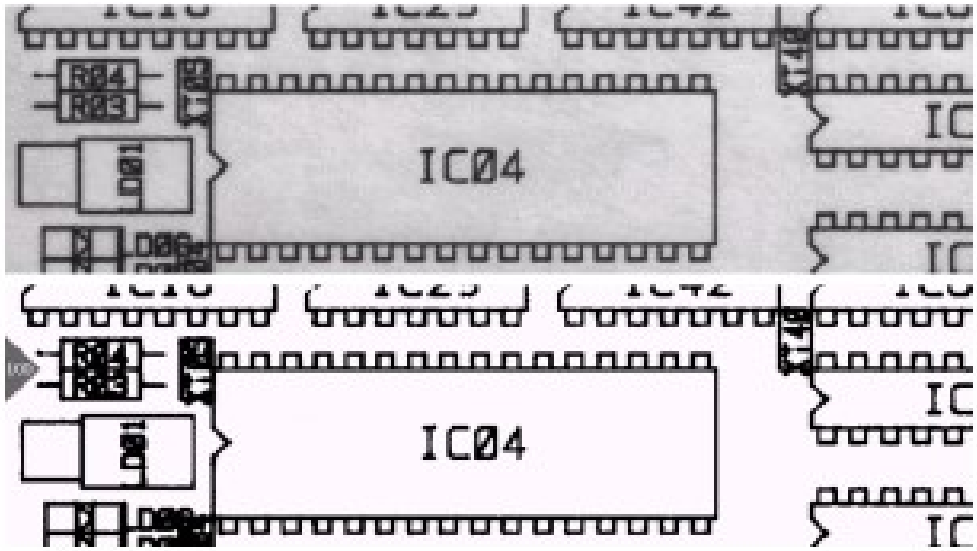
Interpixel Redundancy

- Caused by High Interpixel Correlations within an image, i.e., gray level of any given pixel can be reasonably predicted from the value of its neighbors (information carried by individual pixels is relatively small) spatial redundancy, geometric redundancy, interframe redundancy (in general, interpixel redundancy)
- To reduce the interpixel redundancy, mapping is used. The mapping scheme can be selected according to the properties of redundancy.
- An example of mapping can be to map pixels of an image: $f(x,y)$ to a sequence of pairs: (g_1,r_1), (g_2,r_2), ..., (g_i,r_i), ..

  g_i: ith gray level r_i: run length of the ith run

Interpixel Redundancy (Example)

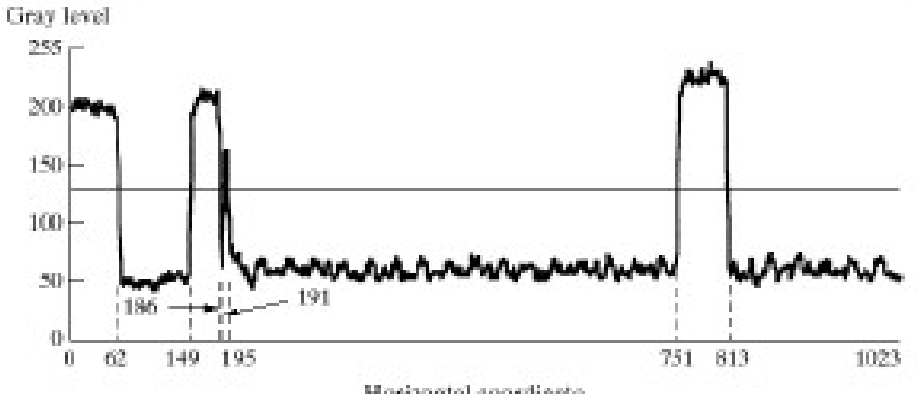
An image with size 1024(y)×343(x)



Consider the 1024 pixels along x=100 (horizontal direction)

To represent the binary image we only need 8 pairs (8 runs) because the gray level changes 7 times

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



Psychovisual Redundancy

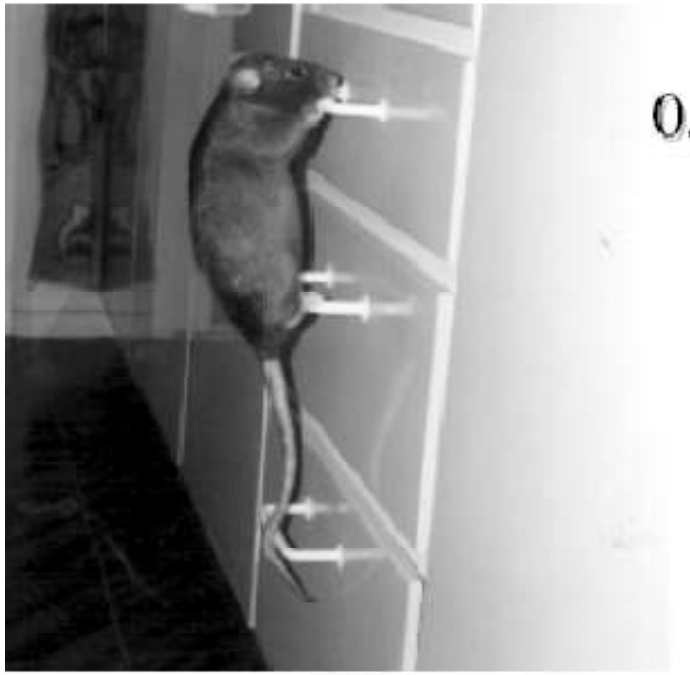
- The eye does not respond with equal sensitivity to all visual information.
- Certain information has less relative importance than other information in normal visual processing psychovisually redundant (which can be eliminated without significantly impairing the quality of image perception).
- The elimination of psychovisually redundant data results in a loss of quantitative information lossy data compression method.
- Image compression methods based on the elimination of psychovisually redundant data (usually called quantization) are usually applied to commercial broadcast TV and similar applications for human visualization.

Psychovisual Redundancy

Original image (256 levels) | Normal quantization (16 levels) | IGS quantization (16 levels)

Psychovisual Redundancy

If the image will only be used for visual observation (i.e. illustrations on the web etc), a lot of the information is usually psycho-visually redundant. It can be removed without changing the visual quality of the image. This kind of compression is usually irreversible.



0.5kB

0.05kB

Psychovisual Redundancy

Improved gray-scale (IGS) quantization:

If we use fewer bits/gray levels to represent an image it causes the false contouring. To reduce this effect, a level of randomness (graininess) is added to the artificial edges associated with false contouring by IGS quantization.

Method:

- *Add to each pixel a pseudo-random number generated from the low-order 4 bits of neighboring pixels.*
- *If the 4 most significant bits are 1111→ no action!*

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.

Fidelity Criteria

Quantify the nature and extent of information loss.

Objective fidelity criteria:

Level of information loss can be expressed as a function of the original (input) and compressed-decompressed (output) image.

Given an $M \times N$ image $f(x,y)$ (original image), its compressed-then-decompressed image: $\hat{f}(x,y)$, then the error between corresponding values are given as:

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

Total error is given by:

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

Fidelity Criteria

Normally the objective fidelity criterion parameters are as follows:

e_{rms} (root-mean-square error):

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

SNR_{ms} (mean-square signal-to-noise ratio):

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

Fidelity Criteria

Subjective fidelity criteria:

-Let a number of test persons grade the images as bad/OK/good etc.

Since most decompressed images ultimately are viewed by human beings, measuring image quality by the subjective evaluations of a human observer often is more appropriate.

Method: evaluations are made using an absolute rating scale, by means of side-by-side comparisons of $f(x,y)$ and $\hat{f}(x,y)$.

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.

Image Compression Models

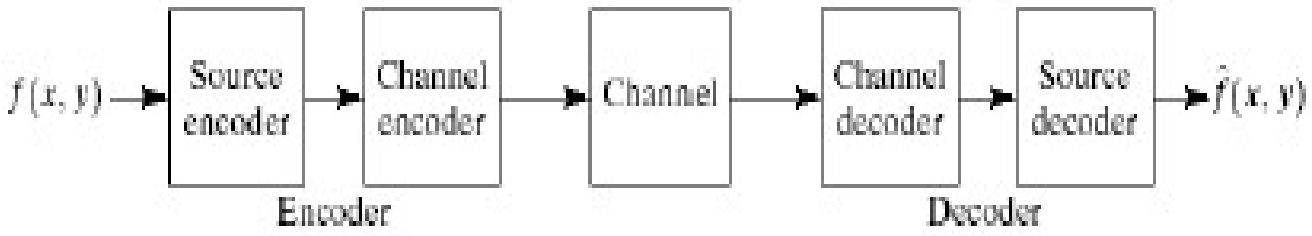
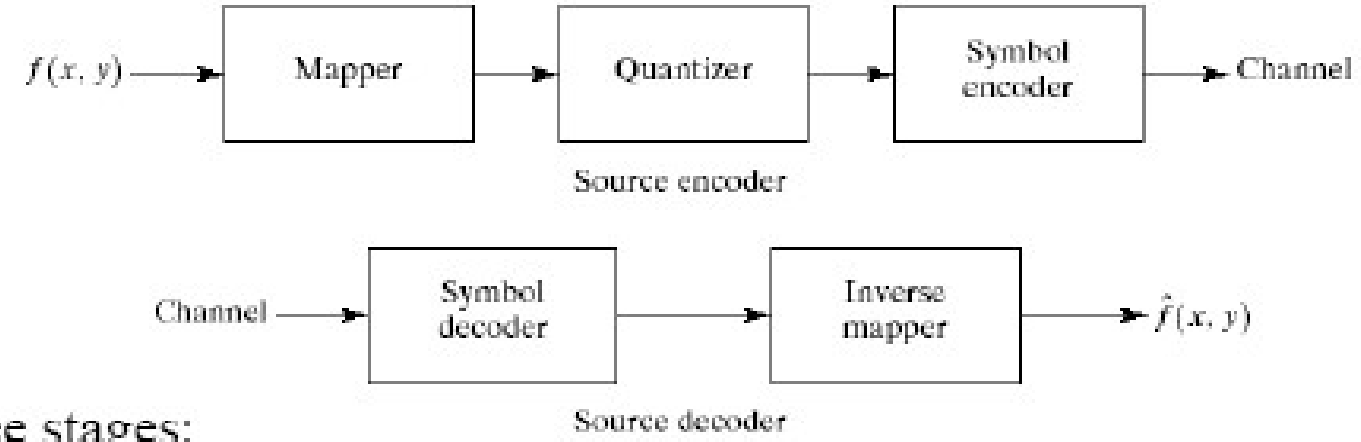


FIGURE 8.5 A general compression system model.

- The encoder creates a set of symbols (compressed) from the input data.
- The data is transmitted over the channel and is fed to decoder.
- The decoder reconstructs the output signal from the coded symbols.
- The source encoder removes the input redundancies, and the channel encoder increases the noise immunity.

Source Encoder and Decoder



Three stages:

Mapper: Transforms the input data into a (usually non-visual) format designed to reduce the interpixel redundancy in the image.

Quantizer: Reduces the accuracy of the mapper's output according to the predefined fidelity criterion. It is irreversible process and must be omitted in error-free compression scheme.

Symbol coder: Creates a fixed or variable length code to represent quantizer output, and maps the output in accordance with the code.

Error-Free Compression

Error-free compression involves:

- Variable length coding
- Mapping

Variable-Length Coding (error-free compression)

- Reduce only coding redundancy (1st kind) by minimizing the L_{avg} assign shorter code words to the most probable gray levels
- The source symbols may be
  - *gray levels of an image, or*
  - *output of a gray-level mapping operation (pixel differences, run-lengths, ...*

Variable-length Coding Methods: Huffman Coding

The most popular technique for removing coding redundancy; yields the smallest possible number of code symbols per source symbol

Huffman Coding Algorithm

- Arrange the symbol probabilities p_i in a decreasing order; consider them (p_i) as leaf nodes of a tree
- While there is more than one node:
  - *merge the two nodes with smallest probability to form a new node whose probability is the sum of the two merged nodes*
  - *Arrange the combined node according to its probability in the tree*
  - *Repeat until only two nodes are left*
- Starting from the top, arbitrarily assign 1 and 0 to each pair of branches merging into a node
- Continue sequentially from the root node to the leaf node where the symbol is located to complete the coding

Variable-length Coding Methods: Huffman Coding

Coding and decoding are done by simple look-up tables

Example:

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.

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				

FIGURE 8.12
Huffman code assignment procedure.

The Huffman code (continued)

- The Huffman code results in an unambiguous code, i.e. no code can be created by combining other codes.

- The code is reversible without loss.

- The table for the translation of the code has to be stored together with the coded image.

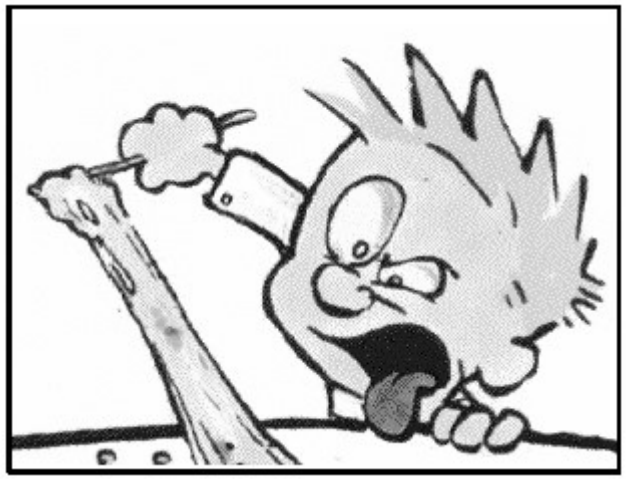
Mapping

•There is often correlation between adjacent pixels, i.e. the value of the neighbors of an observed pixel can often be predicted from the value of the observed pixel. (Interpixel Redundancy).

•Mapping is used to remove Interpixel Redundancy.

•Two mapping techniques are:

- Run length coding
- Difference coding.

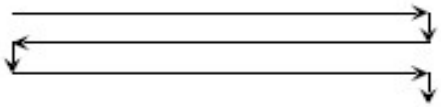


Run-length coding

Every code word is made upp of a pair **(g,l)** where **g** is the graylevel and **l** is the number of pixels with that graylevel (length, or ”run”).

Ex 56 56 56 82 82 82 83 80
56 56 56 56 56 80 80 80

creates the runlength code (56,3) (82,3) (83,1) (80,4) (56,5)



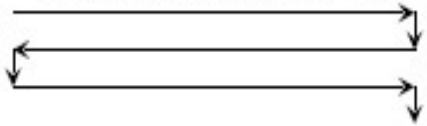
-The code is calculated row by row.
-Very efficient coding for binary data.
-Important to know position, and the image dimensions must be stored with the coded image.
-Used in most fax machines

Difference coding

$$f(x_i)= \begin{cases} x_i \text{ if } i=0, \\ x_i - x_{i-1} \text{ if } i>0 \end{cases}$$

Ex original:	56	56	56	82	82	82	83	80	80	80	80
code $f(x_i)$:	56	0	0	26	0	0	1	-3	0	0	0

- The code is calculated row by row.

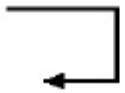


Both Run-length coding och Difference coding are reversible and can be combined with for example Huffman coding.

Example of combined Difference and Huffman coding

original image

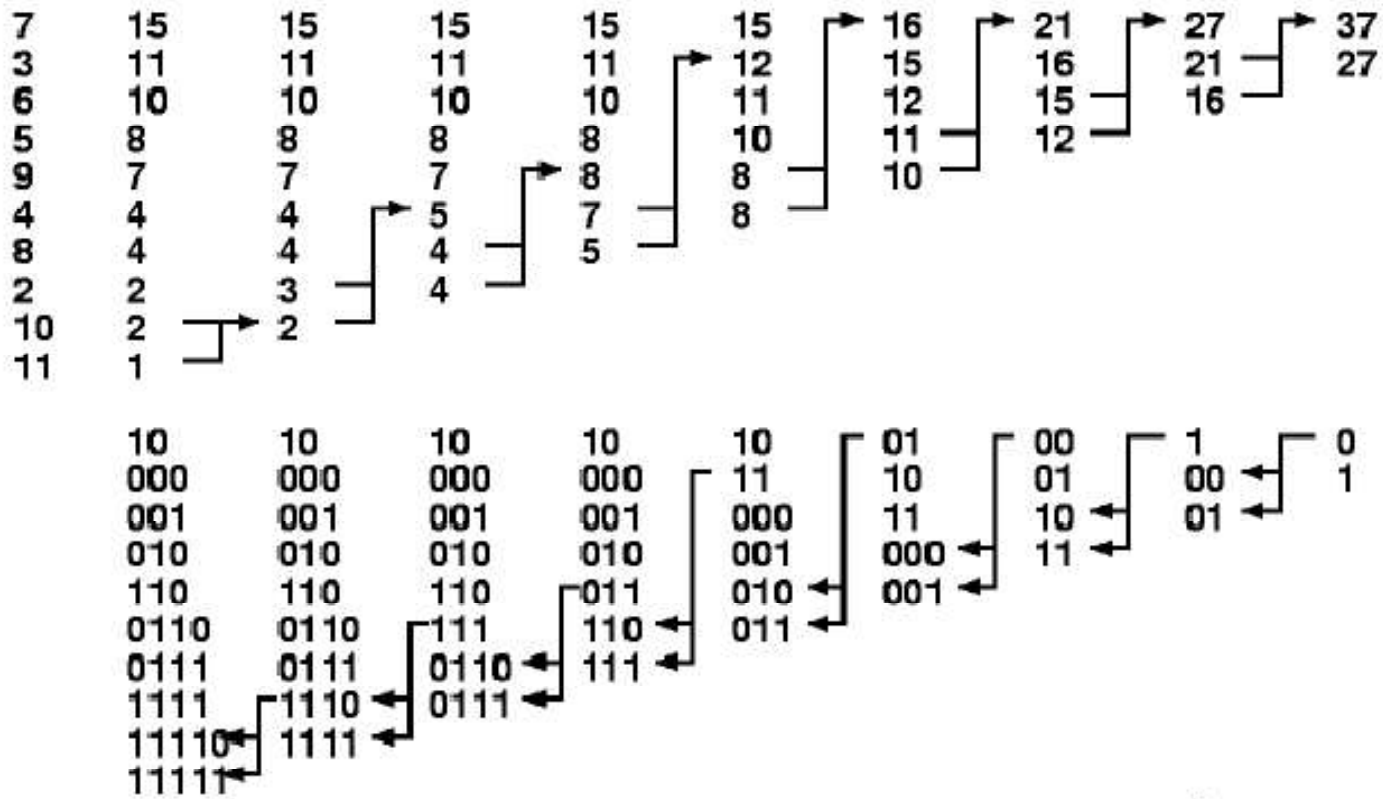
9	8	7	7	7	5	5	5
7	7	7	7	4	4	5	5
6	6	6	9	9	9	6	6
6	6	7	7	7	9	9	9
3	7	7	8	8	8	3	3
3	3	3	3	3	3	3	3
10	10	11	7	7	7	6	6
4	4	5	5	5	2	2	6



difference image

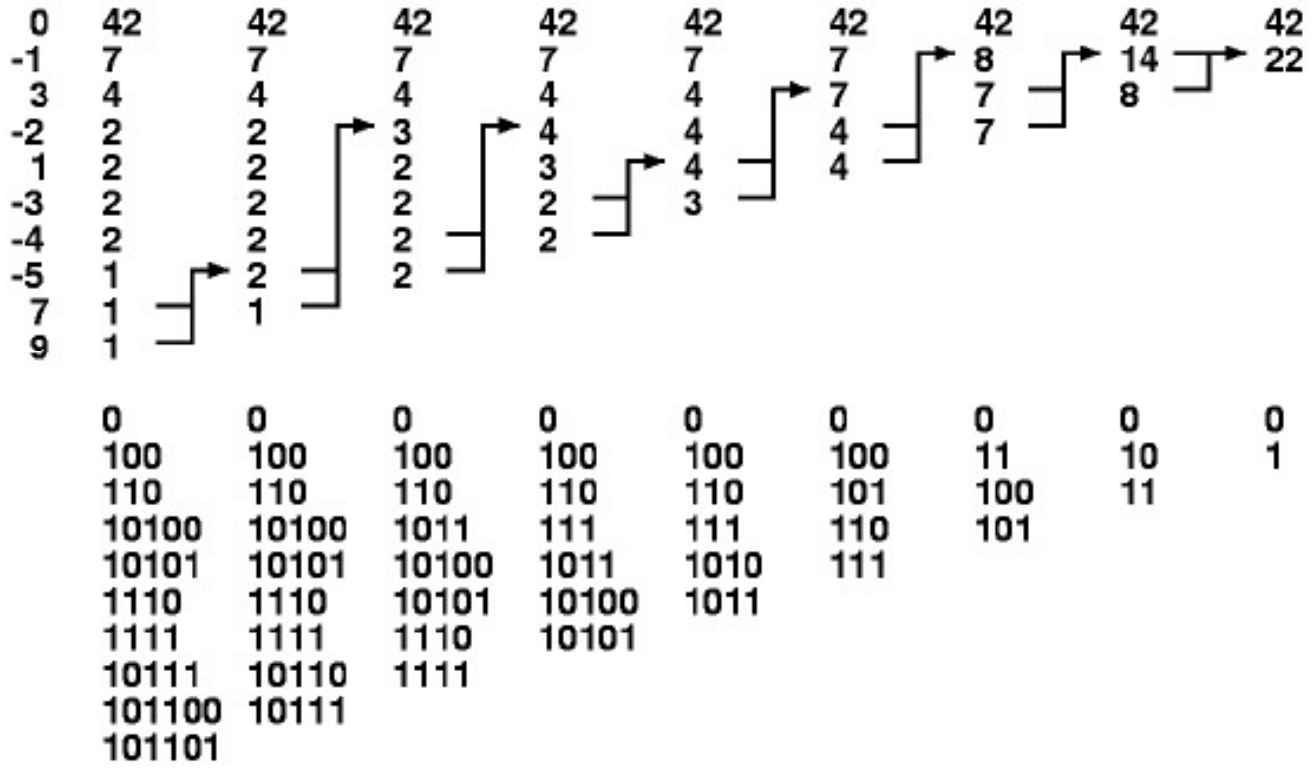
9	-1	-1	0	0	-2	0	0
0	0	0	3	0	-1	0	0
-1	0	0	3	0	0	-3	0
0	-1	0	0	-2	0	0	3
-3	4	0	1	0	0	-5	0
0	0	0	0	0	0	0	0
7	0	1	-4	0	0	-1	0
0	-1	0	0	3	0	-4	0

Huffman code of original image



$L_{avg}=3.1$

Huffman code of difference image



$L_{avg}=2$

LZW Coding

LZW coding assigns fixed length code words to variable length sequences of source symbols.

- It also takes care of the interpixel redundancies.
- Popular coding scheme used in formats like GIF, TIFF, and PDF.

1-D coding

Let us first consider the 1-D input: an alphabet of input "symbols"

Method

- Construct a "dictionary" of input symbol sequences (155's) and output (replacement) symbols
- Initially, the dictionary contains only the individual input symbols
  - *"sequences of length 1"*
- Scanning input from start to finish
  - *add each newly-encountered 155 to the dictionary*
  - *replace each 155 found in the dictionary with matching output symbol*
  - *then skip over the matched 155, and continue*

Bit-Plane Coding

Based on reduction of the image's interpixel redundancies

Concept:

- decompose a multilevel (monochrome or color) image into a series of binary (2-color) images,
- compress each binary image via one of binary compression methods

Bit-plane decomposition

primitive form: gray level r of an m-bit image:

$$r = a_{m-1}2^{m-1} + \cdots + a_1 2^1 + a_0 2^0 = \sum_{i=0}^{m-1} a_i 2^i$$

where $a_i = 0$ or 1

Bit-Plane Coding

- Separate the m coefficients of the polynomial into m 1-bit bit planes.
- Disadvantage: small changes in gray level causes a significant impact on the complexity of the bit planes;
- For example: 127(01111111) 128(10000000) (2 adjacent gray levels)
- For this case, every bit plane contains 0 1 (or 1 0) transition (similar to the false contouring effect)

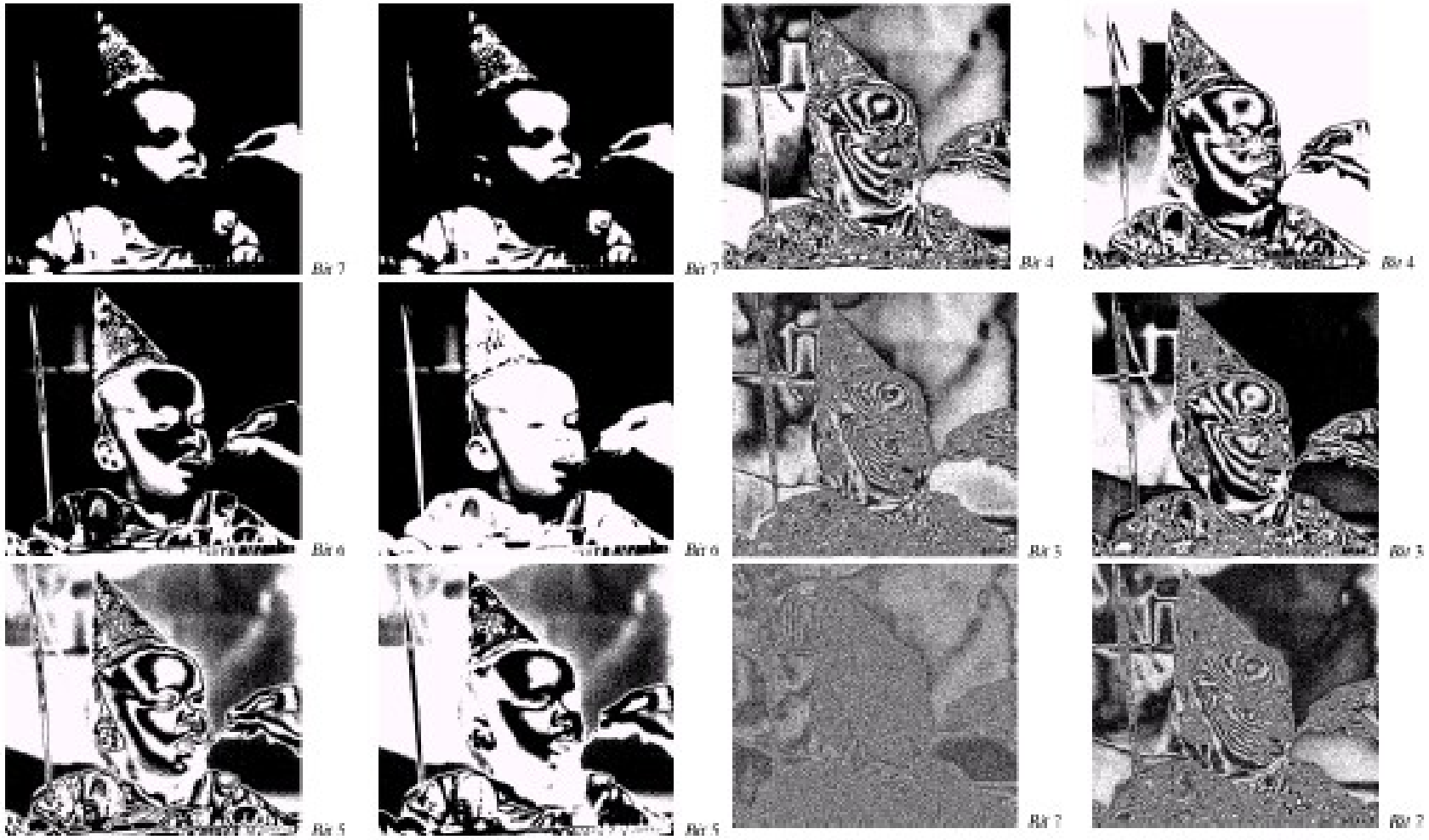
m-bit Gray code reduce the effect of small gray-level variations mentioned above

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

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

- e.g. 127=01111111 normal : 01000000 m-bit *Gray*
  128=10000000 normal : 11000000 m-bit *Gray*

Bit-Plane Decomposition (Example)



Bit-Plane Decomposition (Example)

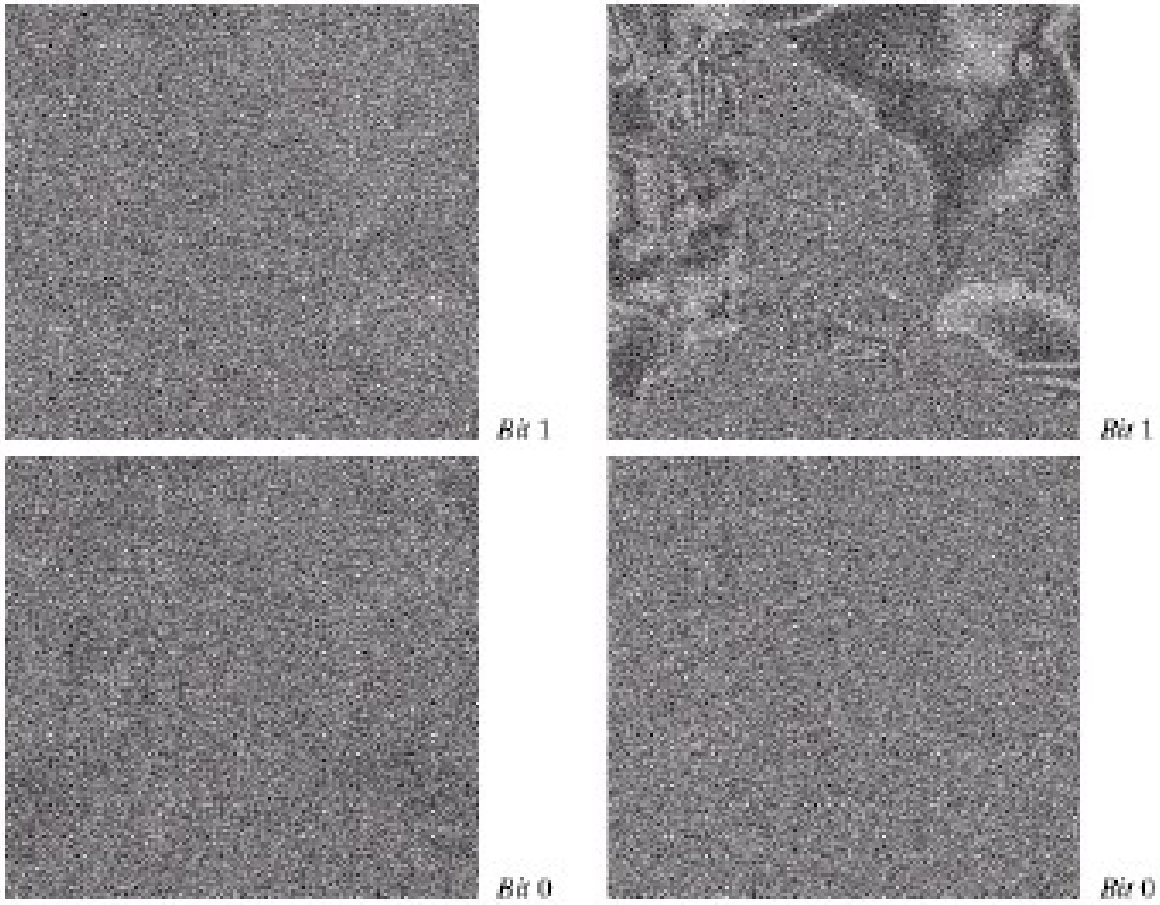


FIGURE 8.15 The four most significant binary (left column) and Gray-coded (right column) bit planes of the image in Fig. 8.14(a).

Binary Image Compression

Binary-image compression discuss 2 methods first:

1. Constant area coding (CAC) 2. 1-D run-length coding (RLC),

A. Constant area coding (CAC):

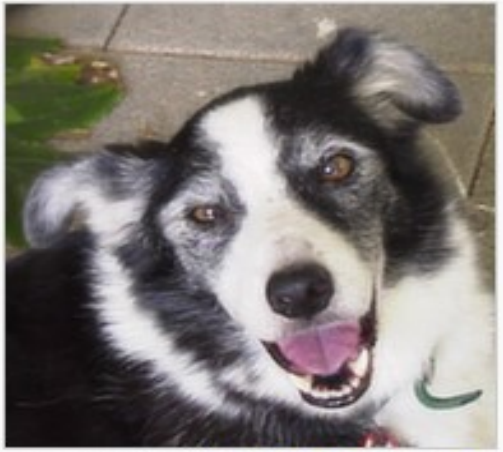
- *divide the image into blocks of size* m×n
- *each block is classified as all white, all black, mixed intensity, probability of occurrence: P(all white), P(mixed), P(all black)*
- *assign 1-bit code word '0' to the block category with largest P(.), and assign 2-bit code word '10' & '11' to the other 2 block categories*
- *the mixed-intensity block category is coded by (1)the code word (2)the mn-bit pattern of the block*

- Other alternative (especially for text documents white block skipping (WBS), code
  - *The (predominant) solid white areas as '0'; code all other blocks (including solid black blocks) by '1' followed by the bit pattern of the block*

Lossy Compression

- A **lossy compression** method is one where compressing data and then decompressing it retrieves data that may well be different from the original, but is close enough to be useful in some way.

- Lossy compression is most commonly used to compress multimedia data (audio, video, still images), especially in applications such as streaming media and internet telephony.



Original Image (lossless PNG, 60.1 KiB size) — uncompressed is 108.5 KiB

Medium compression (92% less information than uncompressed PNG, 4.82 KiB)

Low compression (84% less information than uncompressed PNG, 9.37 KiB)

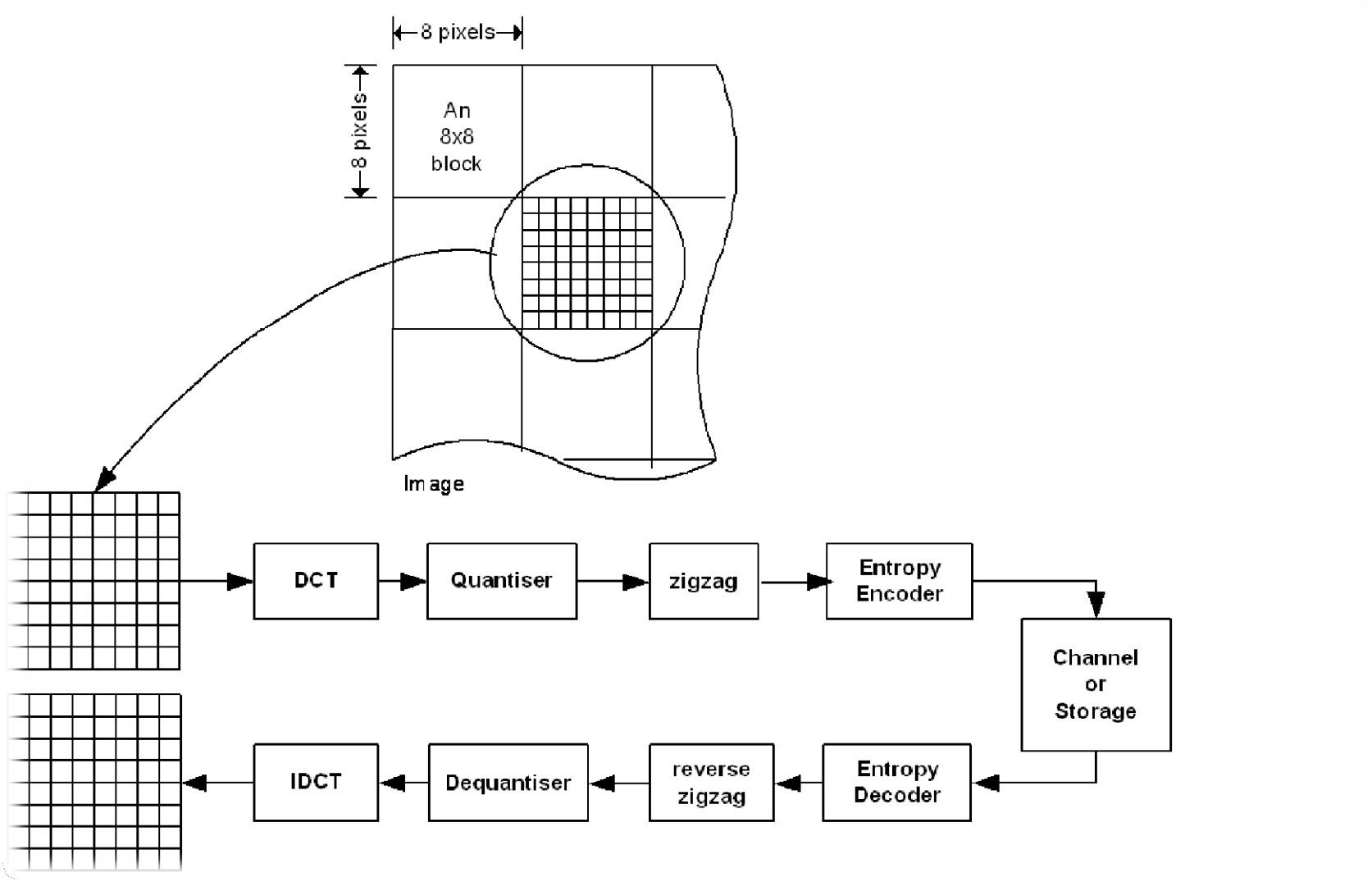
High compression (98% less information than uncompressed PNG, 1.14 KiB)

JPEG

- Lossy Compression Technique based on use of Discrete Cosine Transform (DCT)

- A DCT is similar to a Fourier transform in the sense that it produces a kind of spatial frequency spectrum

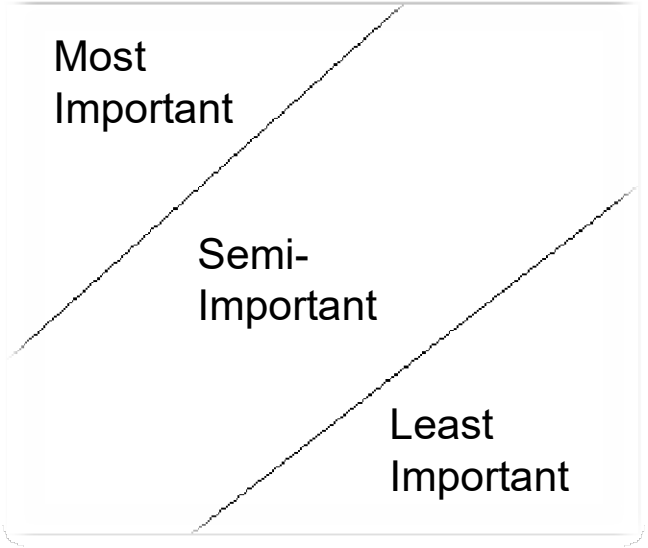
JPEG COMPRESSION



JPEG COMPRESSION

- The most important values to our eyes will be placed in the upper left corner of the matrix.

- The least important values will be mostly in the lower right corner of the matrix.



Some Common Image Formats

JPEG (Joint Photographic Experts Group) exists in many different versions but is always some kind of transform coding. JPEG is not reversible due to quantification.

LZW-Coding (Lempel-Ziv-Welch)

A "word-based" code. The data is represented by pointers to a library of symbols (see Huffman code). LZW compression is loss less and can often be choosen when **TIFF** (Tagged Image File Format) images are stored. The result is a smaller file which usually takes a bit longer to decode. An Image File Directory (set of symbols) is included in the header.

Some Common Image Formats

GIF (Graphics Interchange Format)

Creates a coding for color images where each color is coded by only a few bits (usually 3). GIF also uses LZW compression for storage and transfers. GIF is fully reversible (lossless) if less than 256 colors are present in the original image.