Compression principles Text and Image

- Compression principles Text and Image
  - Lossless and lossy compression
  - Entropy encoding, Source encoding
  - Differential encoding
- Text compression
  - Static Huffman coding
  - Arithmetic coding, Lempel-Ziv coding
- Image compression
  - GIF,TIFF,JPEG

I) Compression principles

Lossless and lossy compression

- Lossless compression algorithm
  - Reduce amount of source information
  - When the compressed information is decompressed, no loss of information
  - Reversible compression
- Lossy compression
  - Reduce amount of source information
  - When the compressed information is decompressed, (minor) loss of information

Entropy encoding

- Lossless and independent of the type of information that is compressed
- Two examples:
  - Run-length encoding
  - Statistical encoding

Run-length encoding

- Long binary “strings”
  - 000000011111111110000011
  - (0,7) (1,10) (0,5) (1,2)
  - Because we have a representation of 0 and 1 -
  - 7,10,5,2

Pointer coding

- Sparse code: binary string with more zeros then ones
- 0 1 0 0 0 1 1 0 0 0 0
- Pointer representation of ones
  - 2 6 7

Statistical encoding

- ASCII code words are often used for representation of strings
- Every character is represented by fixed number of bits (7 bits, 1 Byte)
- In many texts characters do not occur with the same frequency
  - "A" may occur more frequently than "X"
- Statistical encoding
  - Variable length of code words

- Variable-length code words
- For the decoding operation to work correctly
  - Shorter codeword in the set does not form a *start* of a longer code word
- A code word set with this property
  - **Prefix property**
- Example: Huffman encoding algorithm

- Theoretical minimum average numbers of bits that are required to transmit (represent) information is known is **entropy**
- Computed using Shannon's formula of Entropy
- Entropy, $H = -\sum_{i=1}^{n} P_i \log_2 P_i$
- n number of different symbols P_i the probability of of occurrence of the symbol i

- Efficiency of a particular encoding scheme is often computed as a ratio of entropy of the source
- To the *average number of bits per codeword* that are required for the scheme

$$= \sum_{i=1}^{n} N_i P_i$$

- *n* number of different symbols P_i the probability of of occurrence of the symbol *i*, N_i number of Bits to represent this symbol

Example:

A statistical encoding algorithm is being considered for the transmission of a large number of long text files over a public network. Analysis of the file contents has shown that each file comprises only the six different characters M, F, Y, N, 0, and 1 each of which occurs with a relative frequency of occurrence of 0.25, 0.25, 0.125, 0.125, 0.125, and 0.125 respectively. If the encoding algorithm under consideration uses the following set of codewords:

M = 10, F = 11, Y = 010, N = 011, 0 = 000, 1 = 001

compute:

(i) the average number of bits per codeword with the algorithm,

(ii) the entropy of the source,

(iii) the minimum number of bits required assuming fixed-length codewords.

Answer:

- N_i is either *2* or *3* bits...

(i) Average number of bits per codeword

$$= \sum_{i=1}^{6} N_i P_i = (2(2 \times 0.25) + 4(3 \times 0.125))$$

$$= 2 \times 0.5 + 4 \times 0.375 = 2.5$$

(ii) Entropy of source

$$= \sum_{i=1}^{6} P_i \log_2 P_i = -(2(0.25\log_2 0.25) + 4(0.125\log_2 0.125))$$

$$= 1 + 1.5 = 2.5$$

(iii) Since there are 6 different characters, using fixed-length codewords would require a minimum of 3 bits (8 combinations).

Source encoding

- Produce an alternative form of representation
  - Differential encoding
  - Transform encoding

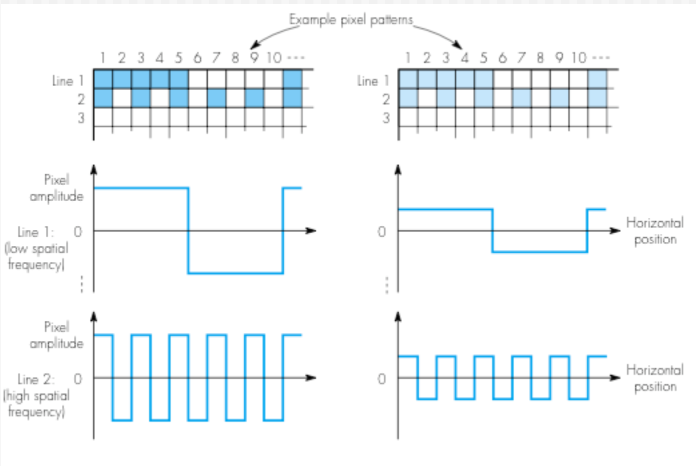
Differential encoding

- Amplitude of a value covers large range
- The difference in amplitude between successive values is relatively small
- Instead of representing amplitude by large code words, a set of smaller code words can be used each of which indicates only the **difference** in amplitude between current values
  - We need 12 bits to represent a signal, but the maximum difference in amplitude between successive samples can be represented by 3 bits

Transform encoding

- Transforming the information from one representation into another
- No loss of information associated with the transformation

Digital Image



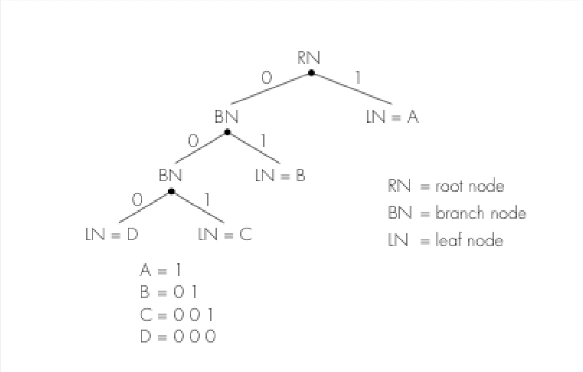
- The change of the magnitude can be represented by spatial frequency
- Human eye is less sensitive to higher spatial frequencies
  - If the amplitude of the higher frequency components falls below a certain amplitude threshold, they will be not detected by the eye
  - Eliminate these frequencies, no degrading the quality of the image
  - Lossy compression

II) Text compression

- Static Huffman coding
- The character string to be compressed is analyzed
- The *character types* and their relative *frequency* are determined

- Coding operation by a Huffman code tree
  - Binary tree with branches assigned the values 0 and 1
  - Base of the tree is the *root node*, point at which a branch divides is called a *branch node*
  - Termination point of a branch is the *leaf node*

- An example of the Huffmann code tree that corresponds to the string of characters AAAABBCD

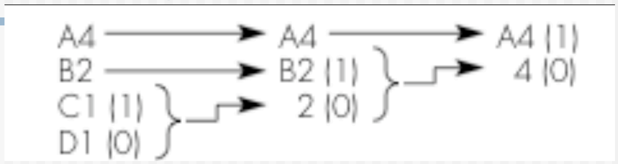


- Each branch divides, a binary value 0 or 1 is assigned for the new branch
- The the binary code words are determined by tracing the path from the root node out to each leaf

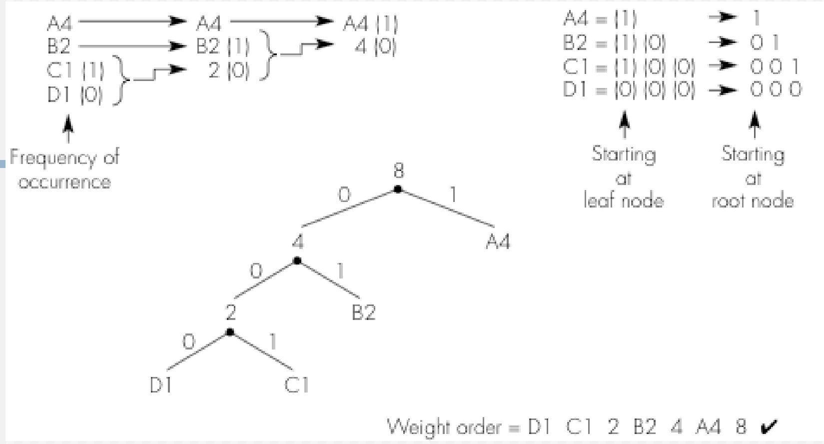
- Code has a **prefix property**
  - A shorter code word in the set does not form a start of a longer code word

- To code AAAABBCD by the Huffman code tree we need 14 bits
- *4*1+2*2+1*3+1*3=14* bits
- For 7-bits ASCII code words we need 8*7=56 bits
  - Which *56%* of the Huffman code tree
    - *56%=14/56*100*

Building a Huffman code tree



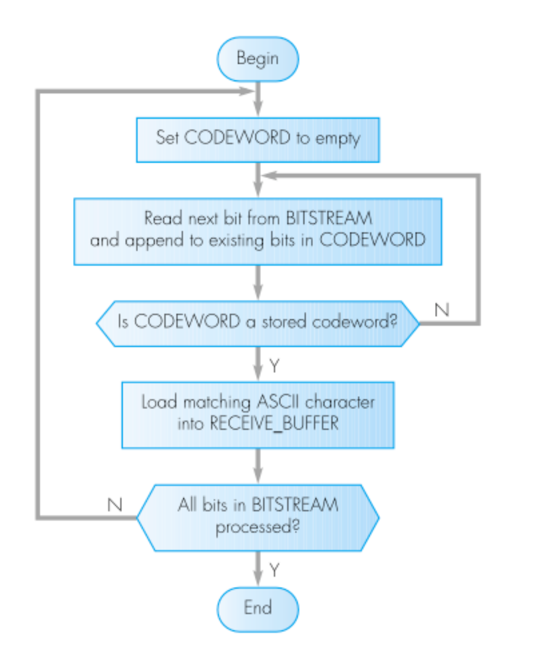
- The first two less frequent characters C and D with their frequency 1 (C1,D1) are assigned to the (1) and (0) branches
  - The two leaf nodes are then replaced by a branch node whose weight is the the sum of the weights of the two leaf nodes (sum is two)
- This procedure is repeated until two nodes remain



- We check that his is the optimum tree - and - hence the code words
- List the resulting weights
- The code words are optimum if the resulting tree increments in weight order

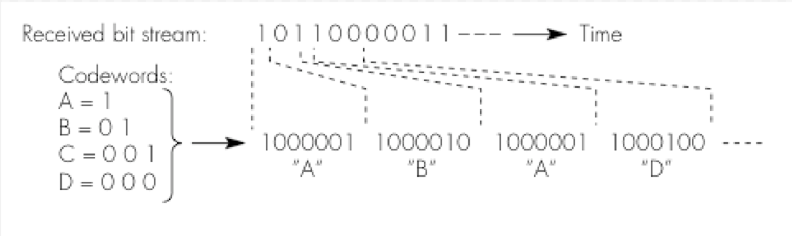
---

- Because of the order in which bits are assigned during the encoding procedure Huffman code words have the unique property that shorter code words will never form the start of a longer code word
- Prefix property



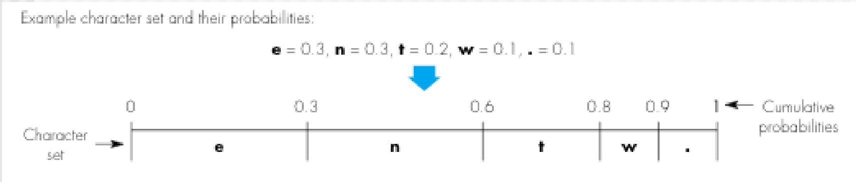
Example

- Decoding into ASCII



Arithmetic coding

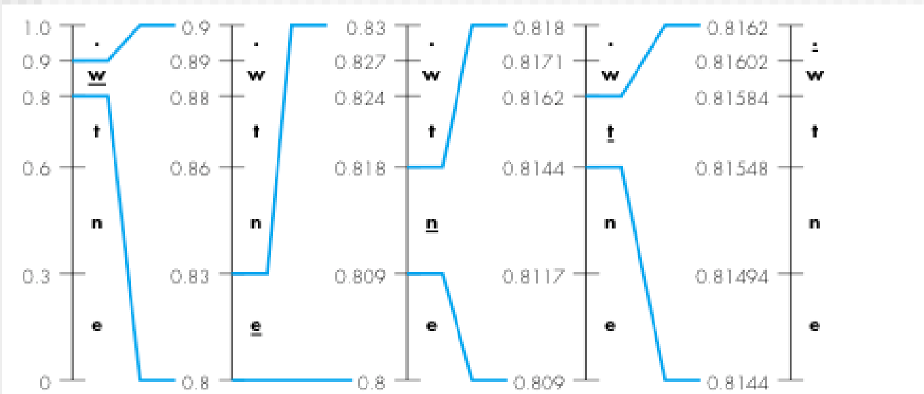
- Arithmetic coding achieve the Shannon value
  - Set of characters with the probabilities
  - At the end of each string a known character is represented, for example period „.“



  - Divide the numeric range from 0 to 1 into a number of different characters present
  - The size of each segment corresponds to the probability of each character

Encoding
of string “went.”

- Every string is represented by an unique number



Encoded version of the character string **went.** is a single codeword in the range 0.816 02 ≤ codeword < 0.8162

Decoding

- The decoder knows the set of characters that are present
- It knows the segment to which each character has been assigned

Example

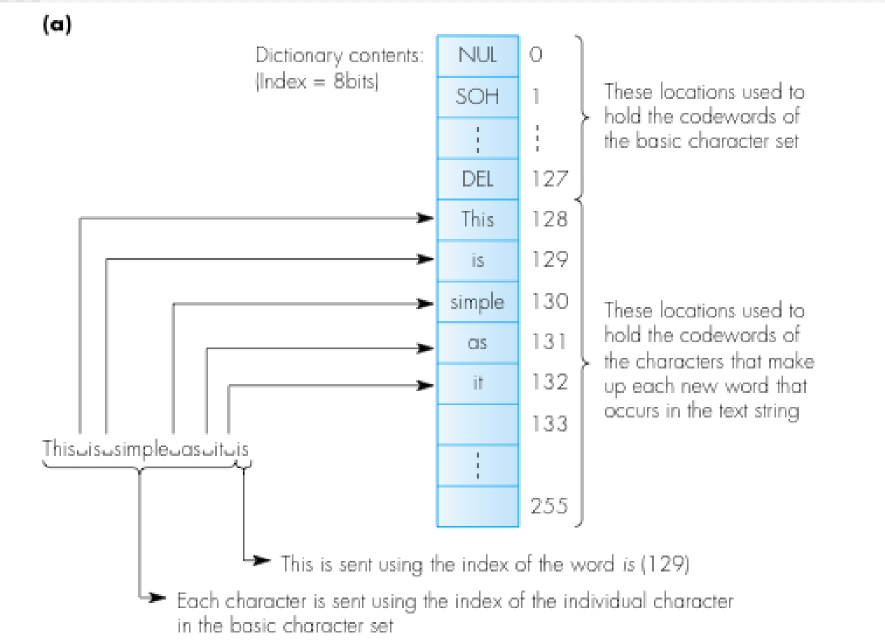
- Decoder receives *0.8161*
- It knows that the first character is **w** since it is the only character within the range *0.8* to *0.9*
- It expands the retrieval as before, the second character must be **e** since *0.861* is within the range *0.8* to *0.83*
- This procedure then repeats until it decodes the known termination character "."

- The number of decimal digits in the final code word increase linearly with the numbers of characters in the string to be encoded
- Maximum number of characters in a string is determined by the precision with which floating point numbers are represented
  - A complete message can be fragmented into smaller strings

Lempel-Ziv coding

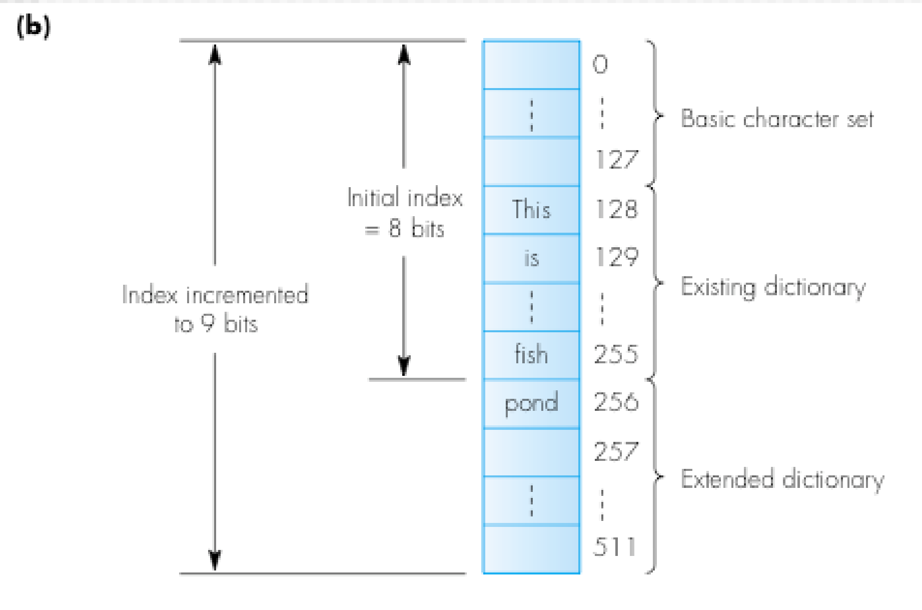
- The Lempel-Ziv (LZ) compressing algorithm uses whole strings as the basis of the coding operation
- For compression of a text, a table containing all the possible words that occur in the text is held by the encoder and decoder
- As each word occurs in the text the word is represented by a code
- Each word is represented by an unique code in a table (dictionary)

- Most word-processing packages have a dictionary associated with them
  - Used for spell checking
  - Used for compression
- Typically they contain 25 000 words
  - 15 bits are required



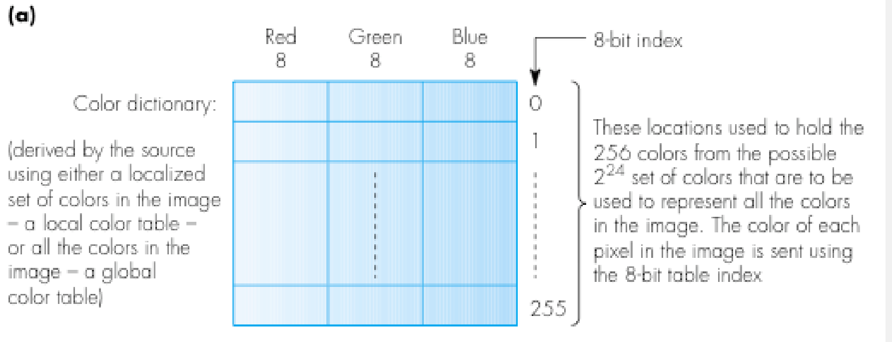
Lempel-Ziv-Welsh coding

- Lempel-Ziv-Welsh (LZW) coding algorithm is for the encoder and decoder to built the contents of the dictionary dynamically
- Initially the dictionary contains only the character code
- The remaining entries in the dictionary are then build dynamically



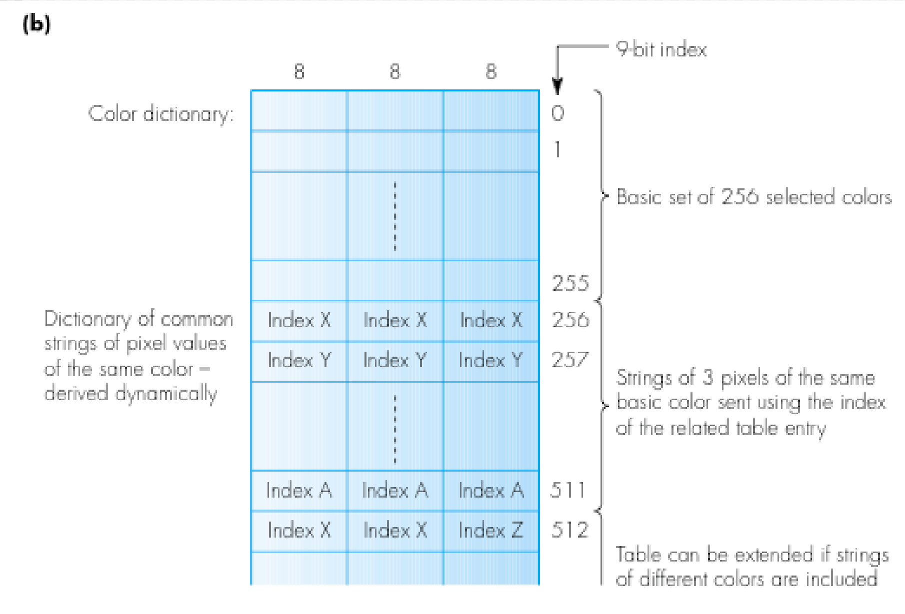
III) Image compression

- The graphic interchange format GIF
- Reduce the number of possible colors that are present by choosing the 256 colors from the original 2^{24} colors that match most closely
- The table of colors can refer to the whole image
  - global color table
- Portion of the image
  - Local color table



The color dictionary, screen size, and aspect ratio are sent with the set of indexes for the image.

- LZW coding can be used to obtain further levels of compression
- Extending the basic color table dynamically as the compressed image data is being encoded and decoded
- Occurrence of common pixel values (long strings of the same color) is detected and stored in the table

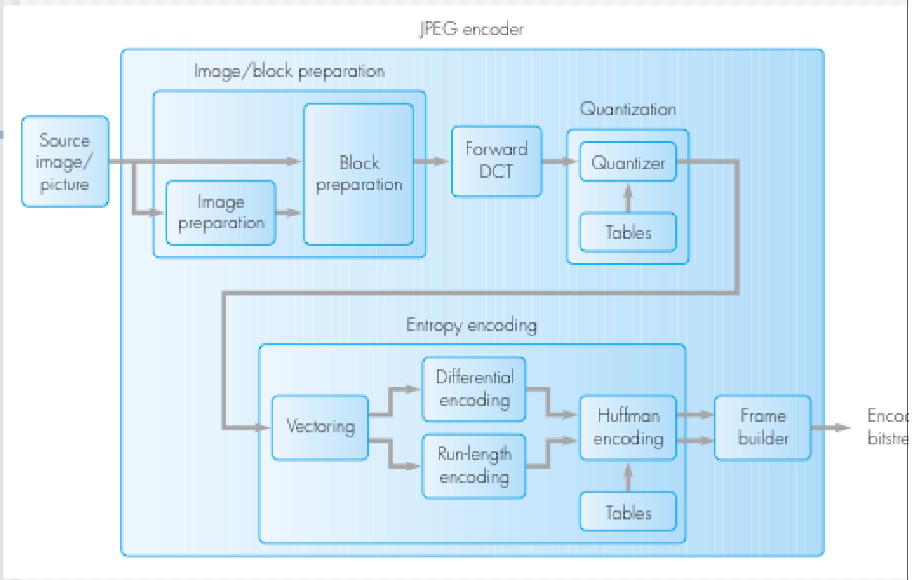


TIFF

- Tagged image file format (TIFF)
- Supports pixel resolution up to 48 bits (16 bits for R,G,B)
- Information can be stored in number of ways
- The particular format being used is indicated by a code
  - Uncompressed format code 1
  - LZW compressed code 5
  - Codes 2,3,4 are used for digitized documents

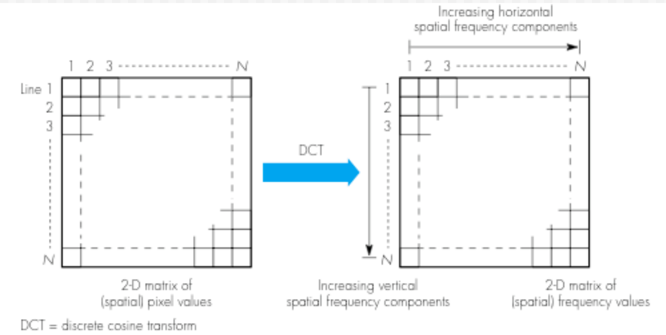
JPEG

- Defines a range of different compression methods
- We describe the lossy sequential mode also known as the basline method

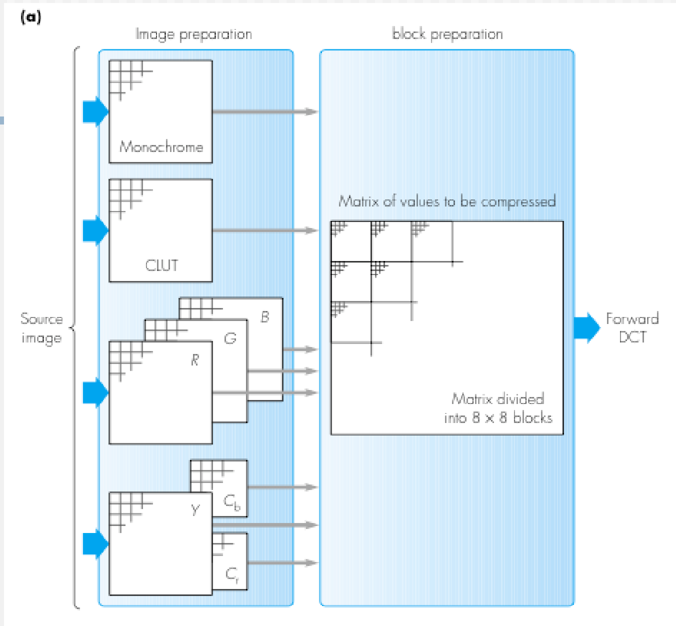


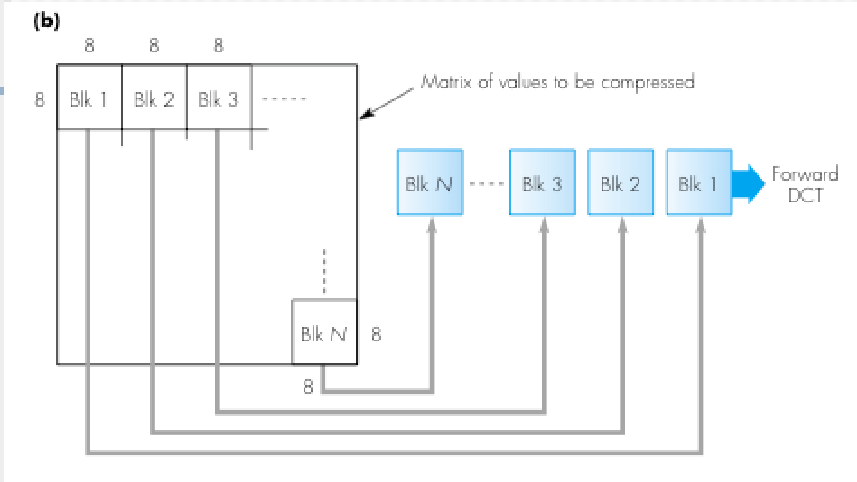
Discrete Cosinus Transformation

- Transformation of two-dimensional matrix of pixel values into an equivalent matrix of spatial frequency components



- It would be too time consuming to compute the transformed values of each position of the total matrix representing the image
- Matrix is divided into smaller 8*8 submatrices
- Each is known as block





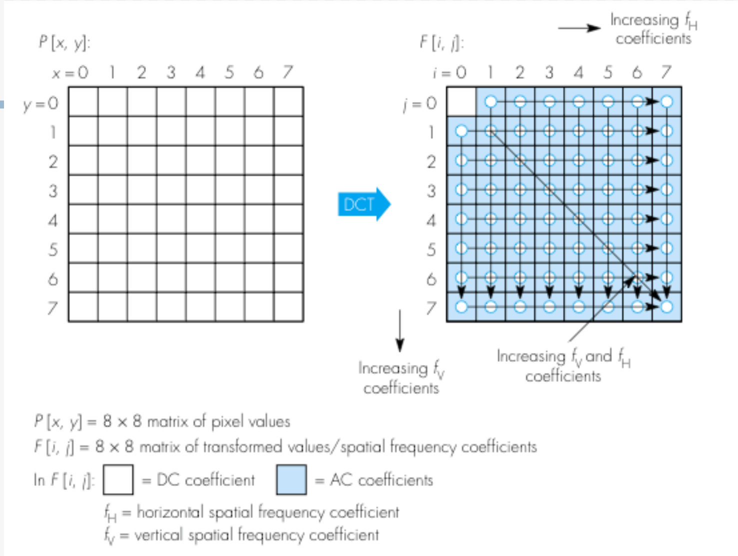
$$F[i,j] = \frac{1}{4} C(i)C(j) \sum_{x=0}^{7} \sum_{y=0}^{7} P[x,y] \cos\frac{(2x+1)i\pi}{16} \cos\frac{(2y+1)j\pi}{16}$$

- C(i) and C(j) = *1/sqrt(2)* for *i,j=0*
- C(i) and C(j) = 1 for all other values of *i,j*
- *x, y, i, j* all vary from *0* to *7*

- All 64 values in the input matrix *P[x,y]* contribute to each entry of the transformation matrix *F[i,j]*

- For *i=j=0* the two cosine terms are both *0*, since *cos(0)=1* the value in *F[0,0]* of the transformed matrix is simply a summation of all the values in the input matrix
- Essentially it is the mean of all *64* values in the matrix, it is known as the **DC coefficient**

- Since the values in all the other locations of the transformed matrix have a frequency coefficient associated with them, they a known as **AC coefficients**
- For *j=0* only horizontal frequency coefficients
- For *i=0* only vertical frequency coefficients



Quantization

- If the magnitude of a higher frequency coefficient is below a certain threshold the eye will not detect it
- Quantization: dropping, setting to zero spatial coefficients below a threshold
- Sensitivity of the eye varies with spatial frequency
  - Amplitude threshold below which eye will detect a particular spatial frequency also varies
  - The threshold values vary for each of the 64 DCT coefficients
  - Represented in the **quantization table**

Assuming a quantization threshold value of 16, derive the resulting quantization error for each of the following DCT coefficients:

127, 72, 64, 56, –56, –64, –72, –128

Answer:

Coefficient	*Quantized value*	*Rounded value*	*Dequantized value*	*Error*
127	127/16 = 7.9375	8	8 × 16 = 128	−1
72	4.5	5	80	+8
64	4	4	64	0
56	3.5	4	64	+8
−56	−3.5	−4	−64	−8
−64	−4	−4	−64	0
−72	−4.5	−5	−80	−8
−128	−8	−8	−128	0

As we can deduce from these figures, the maximum quantization error is plus or minus 50% of the threshold value used.

DCT coefficients

120	60	40	30	4	3	0	0
70	48	32	3	4	1	0	0
50	36	4	4	2	0	0	0
40	4	5	1	1	0	0	0
5	4	0	0	0	0	0	0
3	2	0	0	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantizer

Quantized coefficients

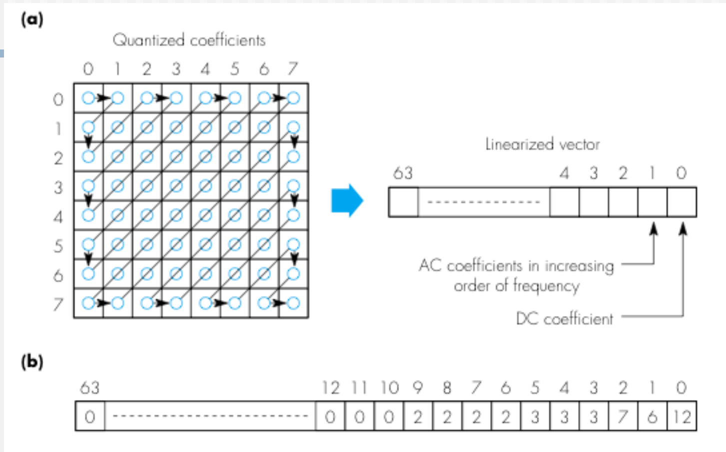
12	6	3	2	0	0	0	0
7	3	2	0	0	0	0	0
3	2	0	0	0	0	0	0
2	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

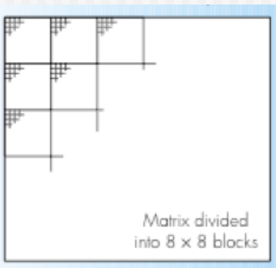
10	10	15	20	25	30	35	40
10	15	20	25	30	35	40	50
15	20	25	30	35	40	50	60
20	25	30	35	40	50	60	70
25	30	35	40	50	60	70	80
30	35	40	50	60	70	80	90
35	40	50	60	70	80	90	100
40	50	60	70	80	90	100	110

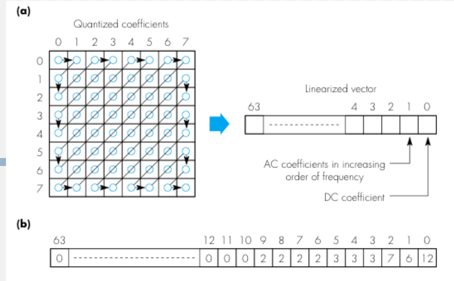
Quantization table

Entropy encoding

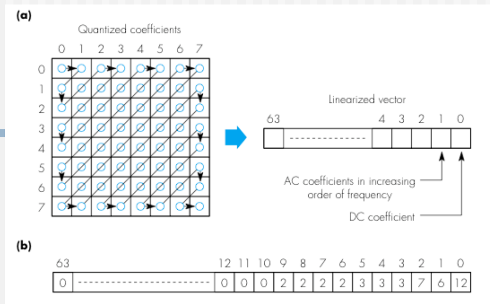
- The various entropy encoding algorithms operate on a vector
- We must represent the matrix as a vector
- If we simply scanned the matrix line by line approach then the resulting vector contain a mix of non-zero and zero values
- Long strings of zeros in the vector, **zig-zag scan**







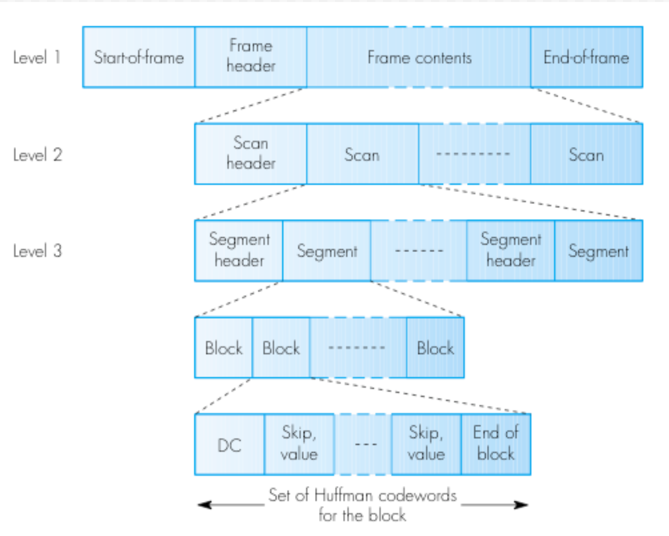
- Differential encoding of **all DC coefficients** of the image
    - *12,13,11,11,10*
- The corresponding difference values
    - *12,1,-2,0,-1*



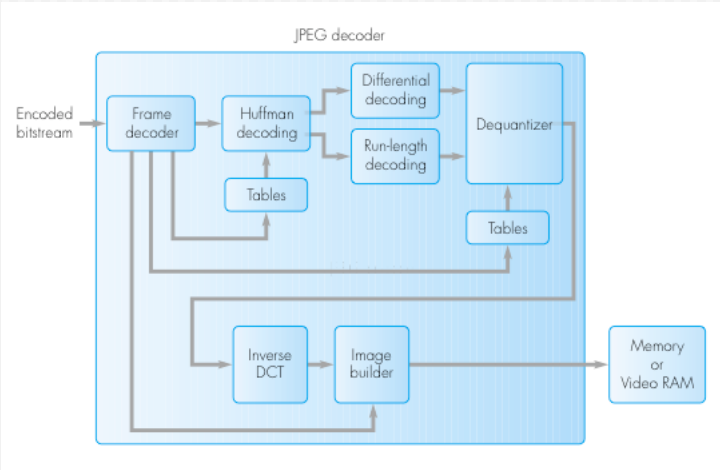
- The remaining 63 AC coefficients Run-length encoding
    - For each run-length encoded AC coefficients in the block, the bits are encoded using a default Huffman table

JPEG encoder output bitstream format

- Frame builder encapsulate all the information relating to encode image
- The structure of the frame is hierarchical



Decoder



- Compression principles Text and Image
  - Lossless and lossy compression
  - Entropy encoding, Source encoding
  - Differential encoding
- Text compression
  - Static Huffman coding
  - Arithmetic coding, Lempel-Ziv coding
- Image compression
  - GIF,TIFF,JPEG