JAIPUR ENGINEERING COLLEGE AND RESEARCH CENTRE

Department of Computer Science & Engineering

Digital Image Processing(6CS3-01)

Hangout Session Classes

COURSE OUTCOMES

CO4: Evaluate various coding algorithms used in image processing and compression

Topics Covered:

Introduction of image compression

Coding Redundancy

Inter pixel Redundancy

Psycho-visual Redundancy

Fidelity Criteria

Image Compression Model

LZW Coding

Lossy Compression

Image Compression Standards

Anju Rajput

VISION & MISSION OF DEPARTMENT

VISION: To become renowned Centre of excellence in computer science and engineering and make competent engineers & professionals with high ethical values prepared for lifelong learning.

MISSION:

- M1: To impart outcome based education for emerging technologies in the field of computer science and engineering.
- M2: To provide opportunities for interaction between academia and industry.
- M3: To provide platform for lifelong learning by accepting the change in technologies
- M4: To develop aptitude of fulfilling social responsibilities

Course Outcomes

- Understand the fundamental aspects of image processing
- Apply the mathematical foundations of coloring and image enhancement in spatial and frequency domains
- Compare and Implement filters for various types of noise.
- Evaluate various coding algorithms used in image processing and compression

IMAGE COMPRESSION

Image Compression?

- The process of reducing the amount of data required to representa digital image.

- From a mathematical viewpoint: transforming a 2-D pixel array into a statistically uncorrelated data set.

Why do We Need Compression?

- For data STORAGE and data TRANSMISSION
  - DVD
  - Remote Sensing
  - Video conference
  - FAX
  - Control of remotely piloted vehicle

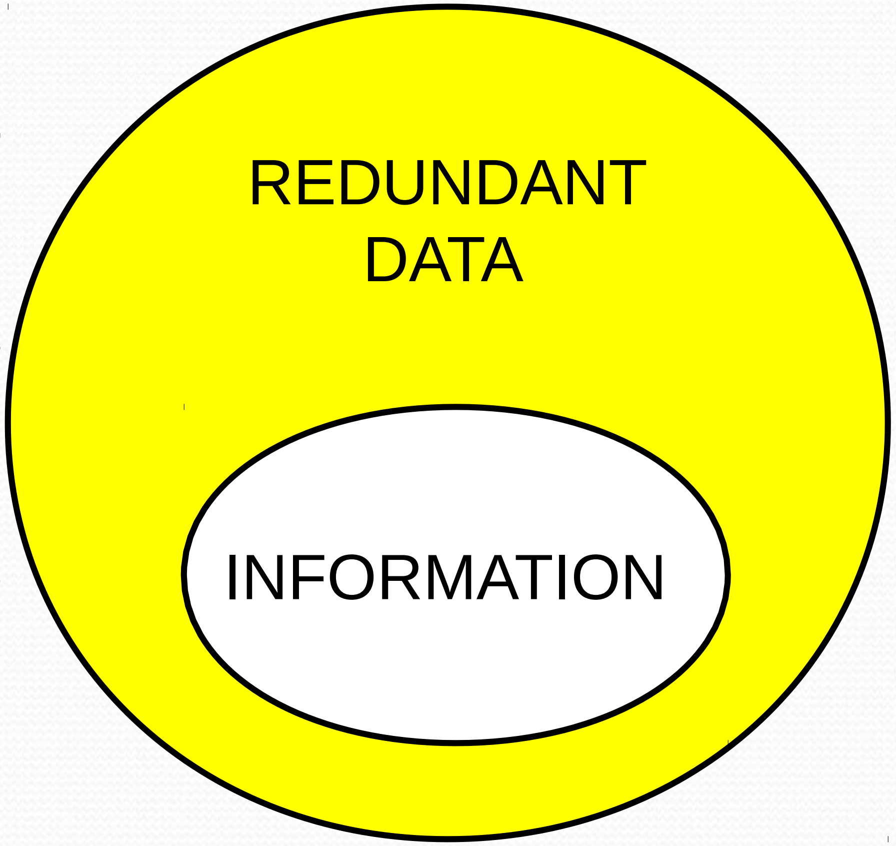
- The bit rate of uncompressed digital cinema data exceeds one Gbps.

Information vs Data

Data compression: Data compression refers to the process of reducing the amount of data required to represent a given quantity of information.

Data and Information are not the same thing ,**Data** is the means by which **information** is conveyed.

Data compression aims to **reduce** the amount of data required to represent a given quantity of information while **preserving** as much information as possible.



DATA = INFORMATION + REDUNDANT DATA

Fundamentals

Data Redundancy

It contains data (or words) that either provide no relevant information or simply restate that which is already known.

The Relative **data redundancy R_D** of the first data set ,is defined by

Fundamentals

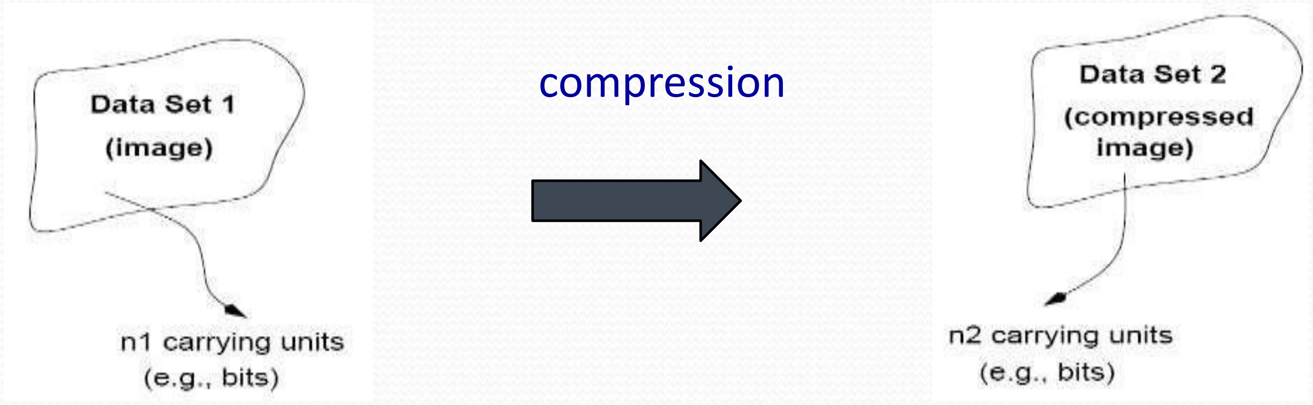
Data Redundancy

It contains data (or words) that either provide no relevant information or simply restate that which is already known.

The Relative **data redundancy R_D** of the first data set ,is defined by :

$$R_D = 1 - \frac{1}{C_R}$$

Where, **R_D** is **Relative Data Redundancy** ,**C_R** is **Compression ratio**.



Fundamentals

C_R refers to the compression ratio and is defined by:

$$C_R = \frac{n_1}{n_2}$$

where ,**n1** and **n2** denoting the **information-carrying units** in two data sets that represent the same information/image.

If **n1 = n2** , then **C_R =1** and **R_D=0**, indicating that the first representation of the information contains no redundant data.

When **n2<<n1** , **$C_R \rightarrow$ large value(∞)** and **$R_D \rightarrow$ 1**. Larger values of C indicate better compression .

Why Can We Compress?

- Spatial redundancy
  - Neighboring pixels are not independent but correlated

Temporal redundancy

Fundamentals

- Basic data redundancies:
  1. Coding redundancy
  2. Inter-pixel redundancy
  3. Psycho-visual redundancy

Coding Redundancy

Let us assume, that a discrete random variable r_k in the interval [0,1] represent the gray level of an image:

$$p_r(r_k) = \frac{n_k}{n} \qquad k = 0,1,2,3\ldots., L-1$$

If the number of bits used to represent each value of r_k is $l(r_k)$, then the average number of bits required to represent each pixel:

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

The total number bits required to code an *MxN* image:

$$M.N.L_{avg}$$

Coding Redundancy

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.

$$\begin{aligned} L_{avg} &= \sum_{k=0}^{7} l_2(r_k)\, p_r(r_k) \\ &= 2(0.19) + 2(0.25) + 2(0.21) + 3(0.16) + 4(0.08) \\ &\quad + 5(0.06) + 6(0.03) + 6(0.02) \\ &= 2.7\, bits \end{aligned}$$

Compression ratio:

$$C_R = \frac{n_1}{n_2}$$

Relative data redundancy:

$$R_d = 1 - \frac{1}{C_R}$$

$$C_R = \frac{3}{2.7} = 1.11 \qquad R_d = 1 - \frac{1}{1.11} = 0.099$$

Inter-pixel Redundancy

When pixels are strongly correlated such that they have same or almost same values, So this is redundancy known as Inter-pixel redundancy.

Here the two pictures have

Approximately the same

Histogram.

We must exploit Pixel

Dependencies.

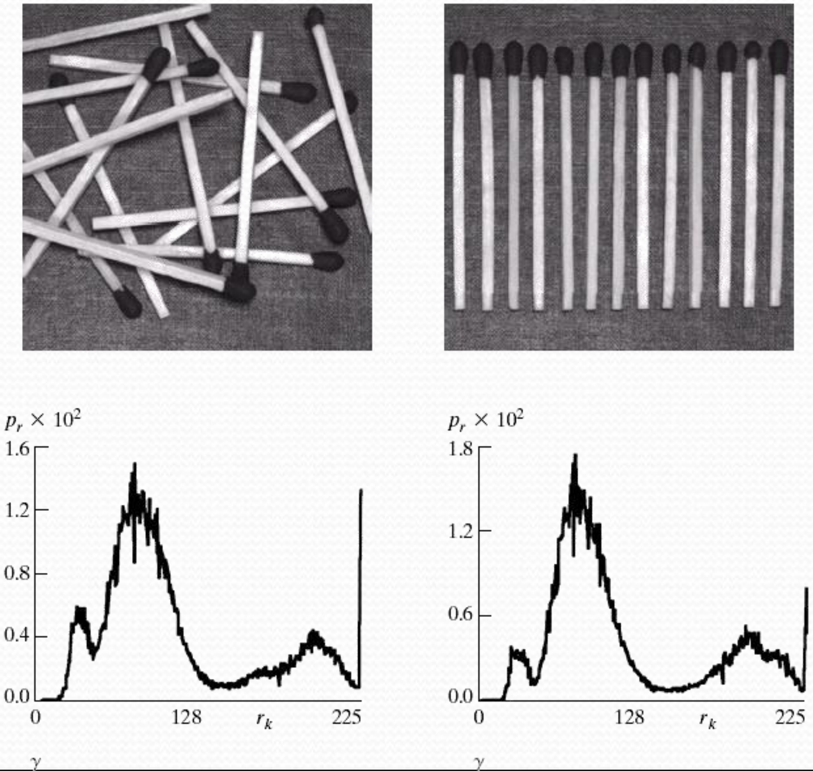
Spatial Redundancy

Each pixel can be estimated

Geometric Redundancy

From its neighbors

Inter-frame Redundancy



a b
c d
e f

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

Interpixel Redundancy

It is also called **Spatial & Temporal redundancy.**

Because the pixels of most 2D intensity arrays are correlated spatially(i.e. **Each pixel is similar to or dependent on neighbor pixel**), information is replicated unnecessarily .

This type of redundancy is related with the Interpixel correlations within an image.

In video sequence, temporally correlated pixels also duplicate information. The codes used to represent the gray levels of each image have nothing to do with correlation between pixel.

Psychovisual Redundancy

Certain information has relatively less importance for the quality of **image perception.** This information is said to be **psychovisually redundant.**

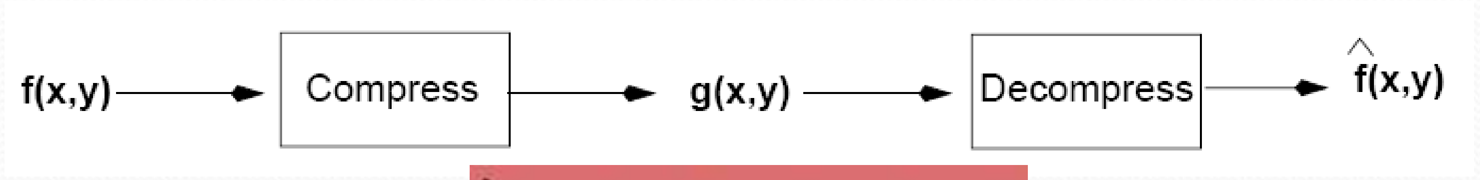
Unlike **coding** and **interpixel redundancies**, the Psychovisual redundancy is related with the **real/quantifiable** visual information. Its elimination results a loss of quantitative information. However psychovisually the loss is negligible.

Removing this type of redundancy is **a lossy** process and the lost information cannot be **recovered.**

The method used to remove this type of redundancy is called **quantization** which means the mapping of a broad range of input values to a limited number of output values.

Fidelity criteria

When lossy compression techniques are employed, the decompressed image will not be identical to the original image. In such cases , we can define **fidelity criteria** that measure the difference between this two images.



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

- **How close is** $f(x,y)$ **to** $\hat{f}(x,y)$ **?**

Two general classes of criteria are used :

(1) Objective fidelity criteria

(2) Subjective fidelity criteria

Objective Fidelity Criteria

When information loss can be expressed as a **mathematical function** of input & output of a compression process.

E.g.. **RMS error between 2 images.**

Error between two images

$$e(x, y) = f'(x, y) - f(x, y)$$

So, total error between two images

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

Where image are size of M x N

Root-mean-square error

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

Mean-square signal-to-noise ratio

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

Subjective Fidelity Criteria

A Decompressed image is presented to a cross section of viewers and averaging their evaluations.

It can be done by using an **absolute rating scale .**

Or

By means of side by side comparisons of **f(x, y)** & **f'(x, y).**

Side by Side comparison can be done with a scale such as **{-3, -2, -1, 0, 1, 2, 3}** to represent the subjective valuations **{much worse, worse, slightly worse, the same, slightly better, better, much better}** respectively.

Image Compression Model

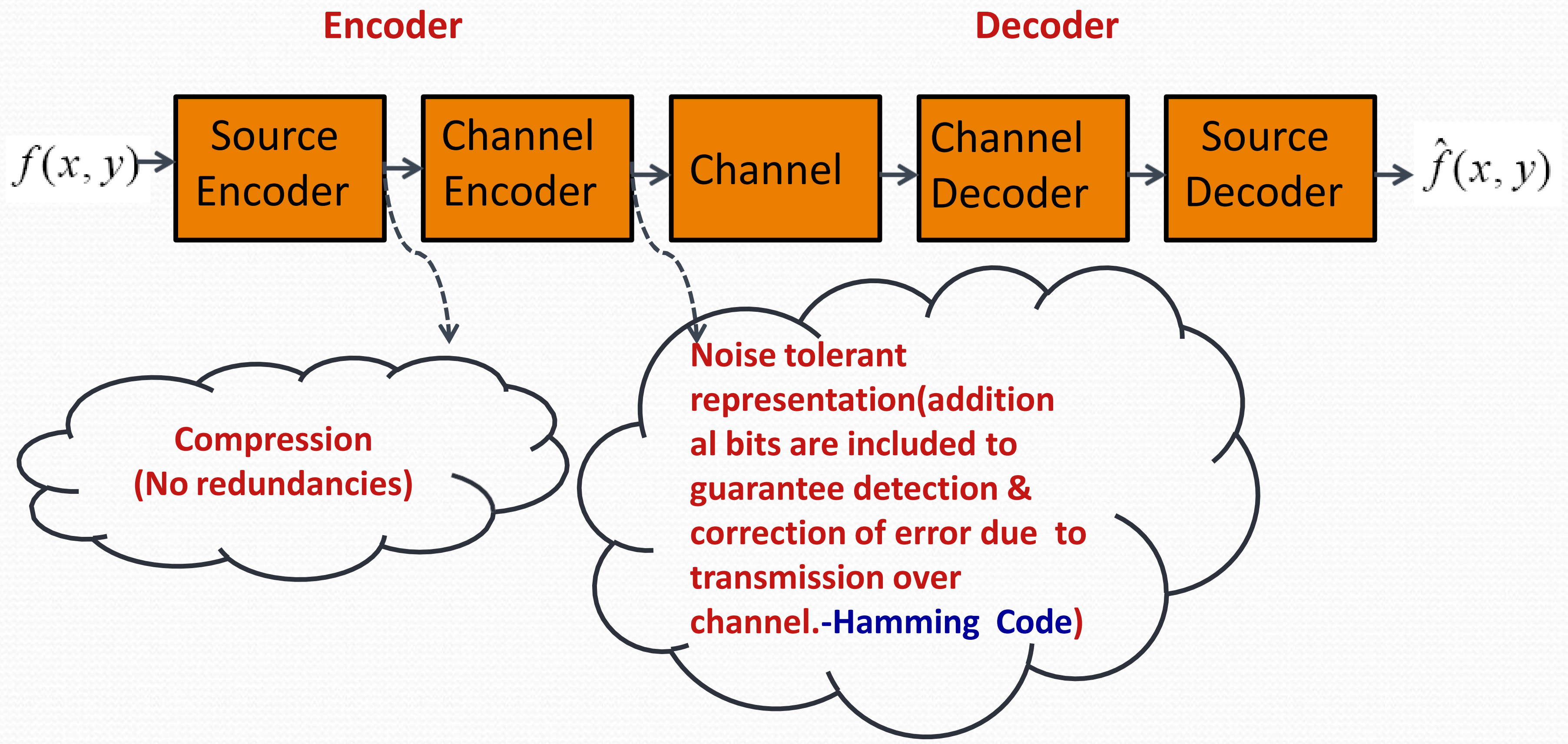

Encoder
Decoder
$f(x,y)$
Source Encoder
Channel Encoder
Channel
Channel Decoder
Source Decoder
$\hat{f}(x,y)$
Compression (No redundancies)
Noise tolerant representation(additional bits are included to guarantee detection & correction of error due to transmission over channel.-Hamming Code)

Image Compression Model

The image compression system is composed of 2 distinct functional component: an **encoder** & a **decoder**.

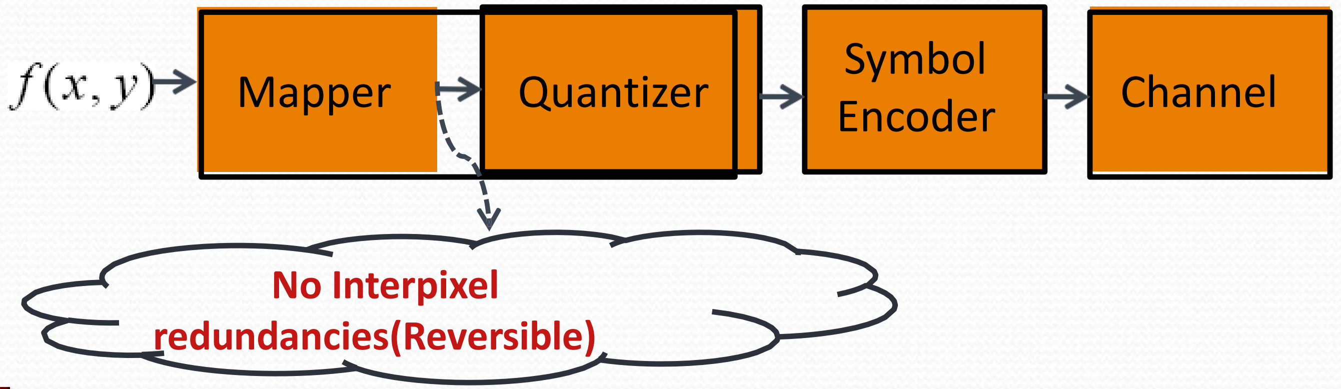
Encoder performs **Compression** while Decoder performs **Decompression**. Encoder is used to remove the redundancies through a **series of 3 independent operations**.

Both operations can be performed in Software, as in case of Web browsers & many commercial image editing programs. Or in a combination of hardware & firmware, as in DVD Players.

A **codec** is a device which performs coding & decoding.

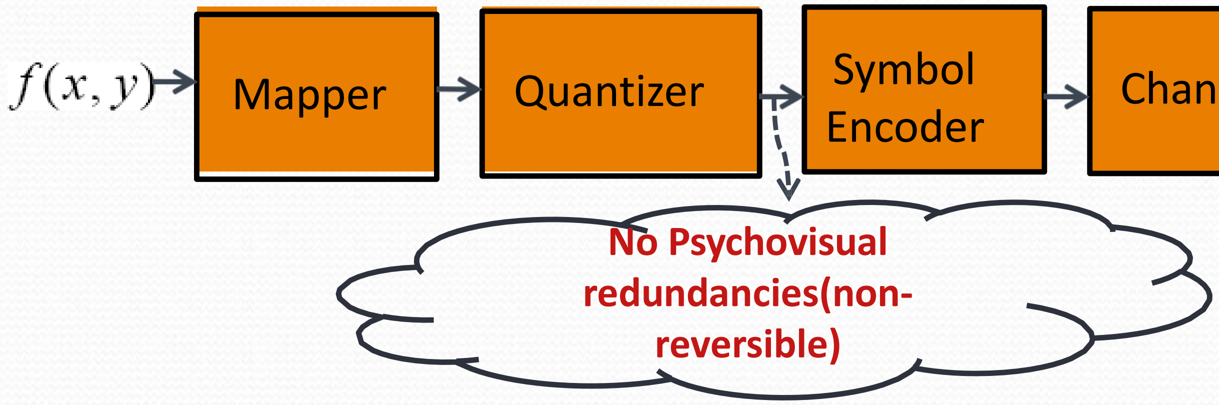
- Input image f(x,…...) is fed into the encoder, which creates a compressed representation of input.
- It is stored for future for later use or transmitted for storage and use at a remote location.
- When the compressed image is given to decoder, a reconstructed output image f’(x,…...) is generated.
- In still image applications, the encoded input and decoder output are f(x, y) & f’(x, y) resp.
- In video applications, they are f(x, y, t) & f’(x, y, t) where t is time.
- If both functions are equal then the system is called *lossless*, error free. If not then it is referred to as *lossy*

Mapper



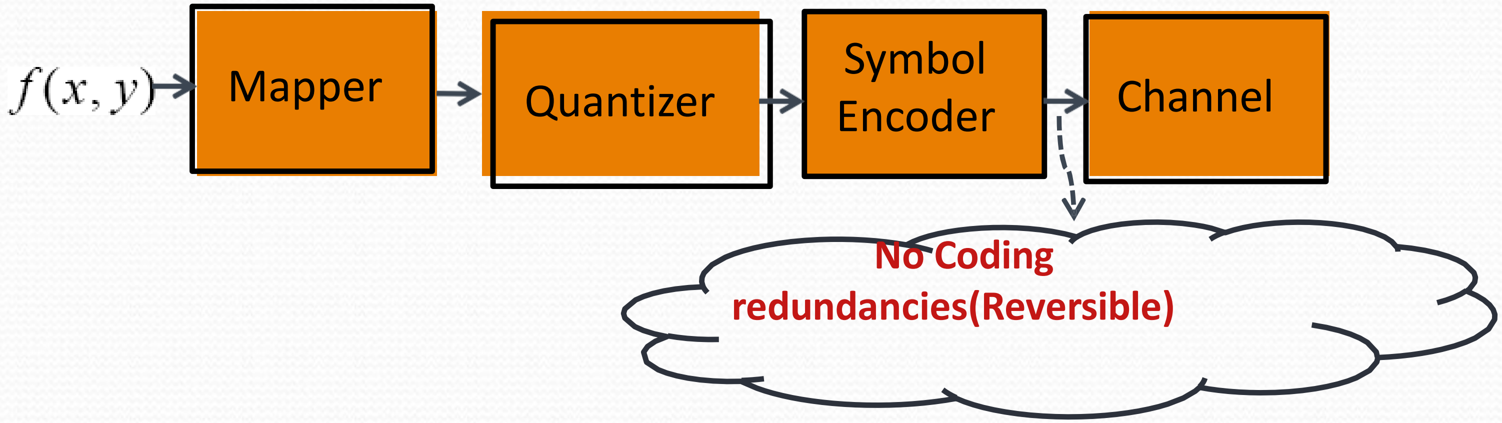
- It transforms input data in a way that facilitates **reduction of interpixel redundancies.**
- It is **reversible .**
- It may / may not reduce the amount of data to represent image. Ex. Run
- Length coding
- In video applications, mapper uses previous frames to remove temporal redundancies.

Quantizer



- It keeps irrelevant information out of compressed representations.
- This operation is **irreversible.**
- It must be **omitted when error free compression** is desired.
- The visual quality of the output can vary from frame to frame as a function of image content.

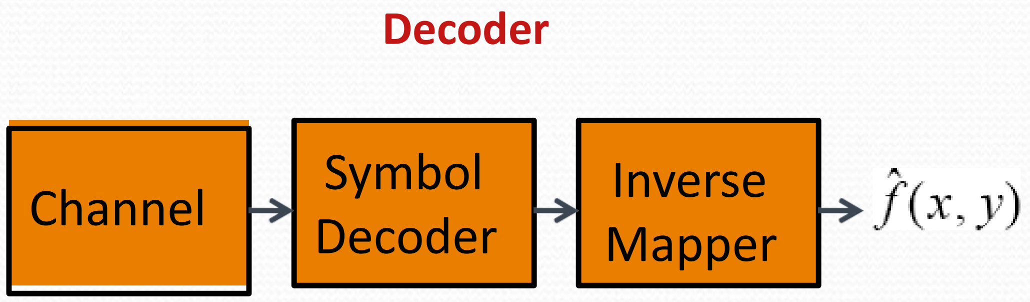
Symbol Encoder



- Generates a fixed or variable length code to represent the Quantizer output and maps the output in accordance with the code.
- Shortest code words are assigned to the most frequently occurring Quantizer output values. Thus **minimizing coding redundancy**.
- It is reversible. Upon its completion, the input image has been processed for the **removal of all 3 redundancies.**

Decoding or Decompression Process

- **Inverse steps** are performed .
- Quantization results in irreversible loss, an **inverse Quantizer block is not included** in the decoder block.



Classification

Lossless Compression

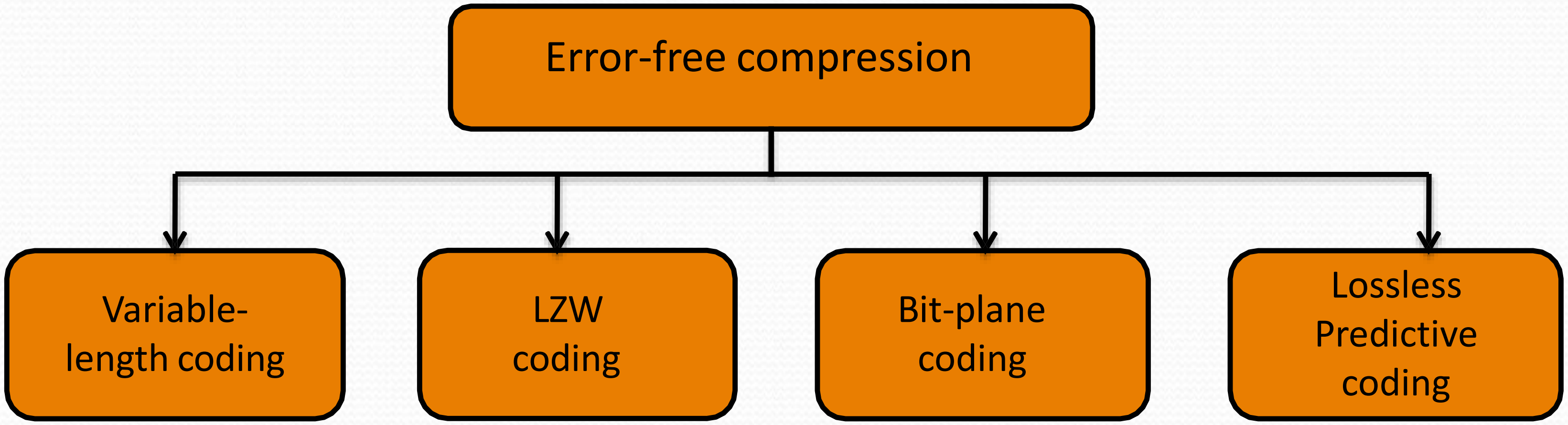
- Lossless compression for legal and medical documents, computer programs
- Exploit only code and inter pixel redundancy

Lossy Compression

- Digital image and video wheresome errors or loss can be tolerated.
- Exploit both code and interpixel redundancy and psycho visual perception properties.

Error-free compression

- **Error-free compression** is generally composed of two relatively independent operations: (1) reduce the interpixel redundancies and (2) introduce a coding method to reduce the coding redundancies.



Error-Free Compression

Variable-length Coding

Huffman coding

- The most popular technique for removing coding redundancy is due to Huffman (1952)
- Huffman Coding yields the smallest number of code symbols per source symbol
- The resulting code is *optimal*

Variable-length Coding

The coding redundancy can be minimized by using a variable- length coding method where the shortest codes are assigned to most probable gray levels.

The most popular variable-length coding method is the **Huffman Coding**.
Huffman Coding: The Huffman coding involves the following 2 steps.

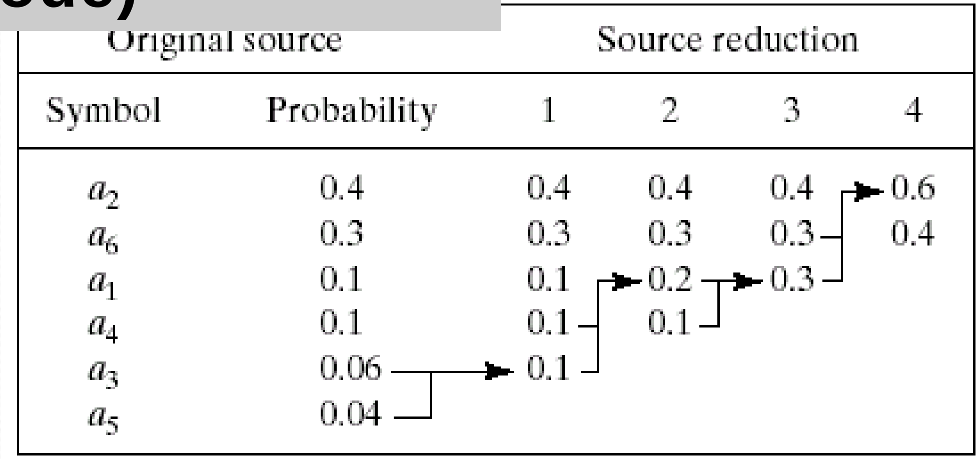
1)Create a series of source reductions by ordering the probabilities of the symbols and combining the lowest probability symbols into a single symbol and replace in the next source reduction.

2)each Code reduced source starting with the smallest source and working back to the original source.

Error-Free Compression

Variable-length Coding

Huffman coding (optimal code)



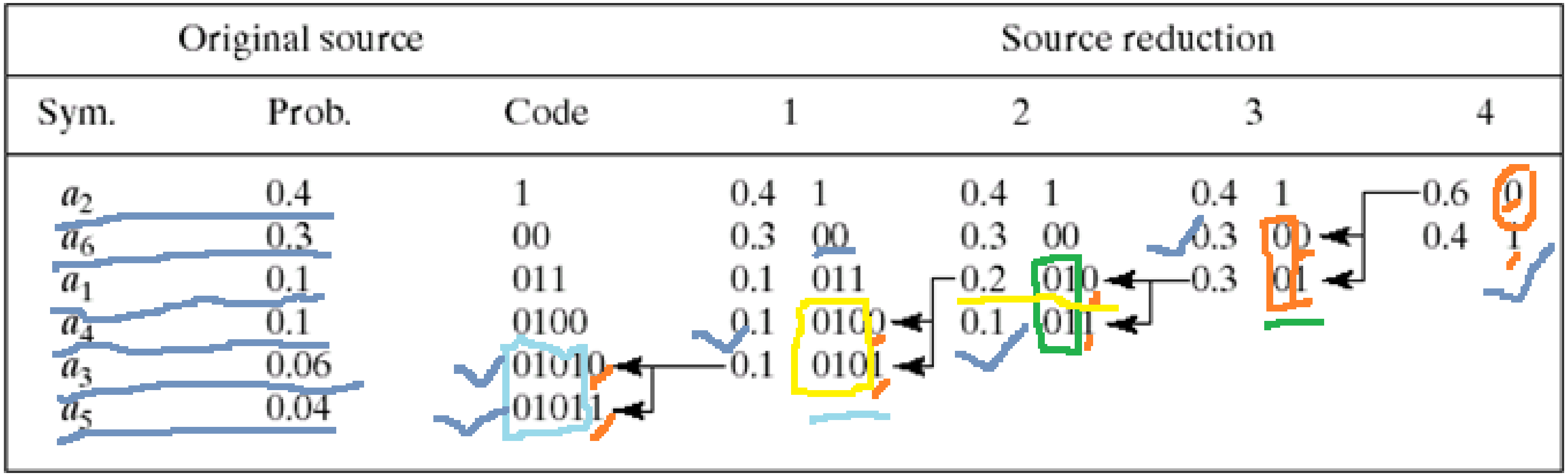
Original source		Source reduction			
Symbol	Probability	1	2	3	4
a_2	0.4	0.4	0.4	0.4	0.6
a_6	0.3	0.3	0.3	0.3	0.4
a_1	0.1	0.1	0.2	0.3	
a_4	0.1	0.1	0.1		
a_3	0.06	0.1			
a_5	0.04				

FIGURE 8.11 Huffman source reductions.

Error-Free Compression

Variable-length Coding

Huffman



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				

$= 2.2\ bits\ /\ symbol$

Arithmetic Coding

- Efficient code
- Lossless data compression
- Non block code
- Works very well for sequence with low entropy

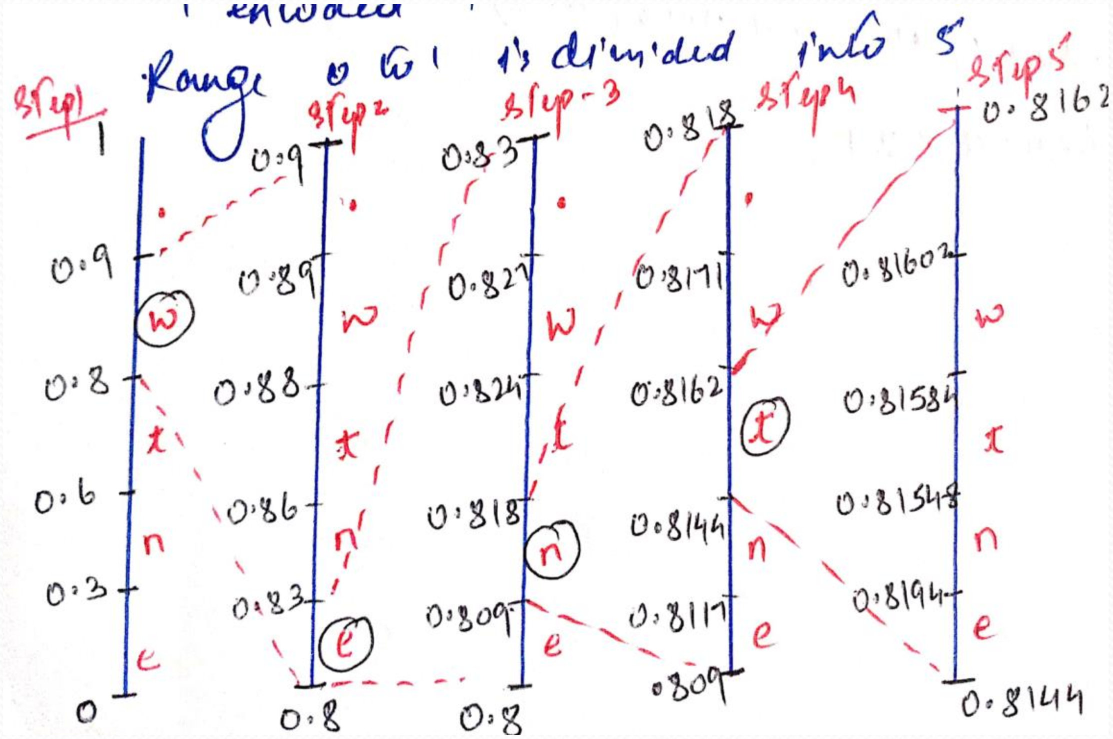
Procedure:

- Divide numeric range 0 to 1 into no. of different symbol present in image.
- Expand first letter to be coded along with range. Further subdivided this range into no. of symbols.
- Repeat the procedure until termination character is encoded.

Arithmetic Coding

Example: Consider transmission of a message "went." comprising string of characters with probability.

e=0.3, n=0.3, t=0.2, w=0.1 and .=0.1



d=upper bound-lower bound

Range of symbols= lower limit: lower limit + d(probability of symbol)

Arithmetic Code word:
0.81602<code word<0.8162

Tag=(upper limit of code word+lower limit of code word)/2
Here
(0.8162+0.81602)/2= 0.81611

Lempel–Ziv–Welch (LZW)

Lempel–Ziv–Welch (LZW)is a universal lossless data compression algorithm created by **Abraham Lempel, Jacob Ziv, and Terry Welch.**

The key to LZW is building a dictionary of sequences of symbols (strings) as the data is read and compressed.

Whenever a string is repeated, it is replaced with a single code word in the output.

At **decompression time**, the same dictionary is created and used to replace code words with the corresponding strings.

Lempel–Ziv–Welch (LZW)

A **codebook** (or **dictionary)** needs to be constructed. **LZW** compression has been integrated into a several images file formats, such as **GIF** and TIFF and **PDF**.

Initially, the **first 256** entries of the dictionary are assigned to the **gray levels 0,1,2,..,255 (i.e., assuming 8 bits/pixel)**

Initial Dictionary

Consider a 4x4, 8 bit image

39	39	126	126
39	39	126	126
39	39	126	126
39	39	126	126

Dictionary Location	Entry
0	0
1	1
.	.
255	255
256	-
511	-

Lempel–Ziv–Welch (LZW)

As the encoder examines image pixels, gray level sequences (i.e., blocks) that are not in the dictionary are assigned to a new entry.

39	**39**	**126**	**126**
39	**39**	**126**	**126**
39	**39**	**126**	**126**
39	**39**	**126**	**126**

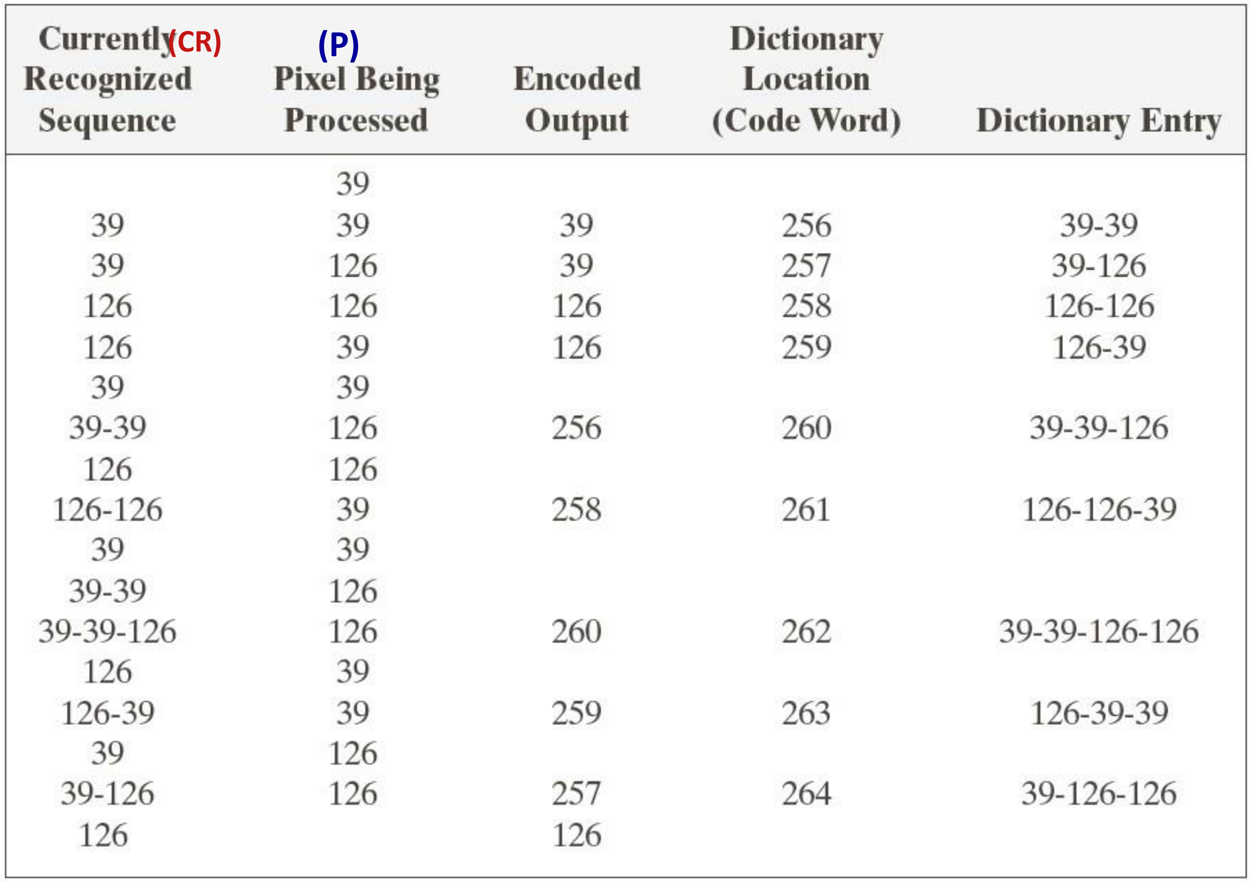
Dictionary Location	Entry
0	0
1	1
.	.
255	255
256	**39-39**
511	-

- Is **39** in the dictionary……..**Yes**
- What about **39-39**………….No
    * Add 39-39 at location 256

Lempel–Ziv–Welch (LZW)

39 39 126 126
39 39 126 126
39 39 126 126
39 39 126 126

Concatenated Sequence: CS = CR + P



Currently (CR) Recognized Sequence	(P) 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		

CR = empty

If CS is found:
(1) No Output
(2) CR=CS

else:
(1) Output D(CR)
(2) Add CS to D
(3) CR=P

LZW Example

LZW

in

ababbabcababba

a b a b b a b c a b a b b a

Initial LZW dictionary

index	entry
1	a
2	b
3	c

Encoded o/p	Dictionary	
	index	entry
-	1	a
-	2	b
-	3	c
1	4	ab
2	5	ba
4	6	abb
5	7	bab
2	8	bc
3	9	ca
4	10	aba
6	11	abba
1		

output is

1 2 4 5 2 3 4 6 1

Run length Coding

- Simple data compression technique.
- It is more efficient in case of 0 & 1.
- It is loss less compression technique.
- It is reversible.
- In this, runs of data are stored as a single data value and count rather than original sum.
- Run is sequence of same symbol/data value.

Example:

$$\underbrace{\text{BBBBBBBB}}_{\text{Long sequence}} \longrightarrow \underset{\text{Data value}}{\text{B}}\ \underset{\text{Count of data value}}{8}$$

Example:

$$\underbrace{\text{BBBBBBBBB}}\ \underbrace{\text{AAAAA}}\ \text{N}\ \underbrace{\text{GG}}\ \underbrace{\text{MMM}}$$

$$\Big\downarrow\ \text{RLC}$$

B09A05N01G02M03

$$\underbrace{00000000000000}_{14}\ \Big|\ \underbrace{0000}_{4}\ \Big|\ \underset{0}{\uparrow}\ \Big|\ \underbrace{000000000000}_{12}$$

Output is =1110 0100 0000 1100

Represent these value in binary form i.e
14=1110
4=0100
0=0000
12=1100

ex
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

we are having a 5x5 binary image which we want to compress.

1st Row → {5, 0] ✓

2nd Row → (3, 0), (2, 1)

3rd Row → [5, 1) ✓

4th Row → (5, 1) ✓

5th Row → (5, 1) ✓

Total vector = 6

Now we ~~select~~ max No i.e 5
we can represent 5 by using 3 bits
i.e Max bits = 3

$\therefore$ 6 (3 + 1)
⇒ 24

where 1 indicate intensity bit representation
means in above values are 0 or 1
so it can be represented by 1 bit

Bit plane Compression

- Another effective method to reduce interpixelredundancies.
- Image's bit planes are processedindividually.
- Based on decomposing a multilevel (monochrome / color) image into a series of binary images & compressing each binary using any binary compression method.

Bit plane compression

Step 1 consider a Matrix

180	4	80	33	201
120	27	11	160	28
224	1	133	67	144

Step 2 obtain binary equivalent

10110100	00000100	01010000	00100001	11001001
01111000	00011011	00001011	10100000	00011100
11100000	00000001	10000101	01000011	10010000

Step 3 Extract MSB for each value in matrix

1	0	0	0	1
0	0	0	1	0
1	0	1	0	1

In above we have 3×5 = 15 values
Rearrange above MSB values such that each row contains 8 columns or 8 bits

1	0	0	0	1	0	0	0
1	0	1	0	1	0	1	0

convert to decimal $\rightarrow$

136
170

Image Compression Standards

Why Do We Need International Standards?

- International standardization is conducted to achieve inter-operability .
  - Only syntax and decoder are specified.
  - Encoder is not standardized and its optimization is left to the manufacturer.
- Standards provide state-of-the-art technology that is developed by a group of experts in the field.
  - Not only solve current problems, but also anticipate the future application requirements.
- Most of the standards are sanction by the International Standardization Organization (ISO) and the Consultative Committee of the International Telephone and Telegraph (CCITT)

Image Compression Standards

Binary Image Compression Standards

CCITT Group 3 and 4

- They are designed as FAX coding methods.
- The Group 3 applies a non-adaptive 1-D run length coding and optionally 2-D manner.
- Both standards use the same non-adaptive 2-D coding approach, similar to RAC technique.
- They sometime result in data expansion. Therefore, the Joint Bilevel Imaging Group (JBIG), has adopted several other binary compression standards, JBIG1 and JBIG2.

Image Compression Standards

Continues Tone Still Image Comp.

What Is JPEG?

- "Joint Photographic Expert Group". Voted as international standard in 1992.
- Works with color and grayscale images, e.g., satellite, medical, ...
- Lossy and lossless

Image Compression Standards

Continues Tone Still Image Comp. - JPEG

- First generation JPEG uses DCT+Run length Huffman entropy coding.

- Second generation JPEG (JPEG2000) uses wavelet transform + bit plane coding + Arithmetic entropy coding.

Image Compression Standards

Continues Tone Still Image Comp. - JPEG

- Still-image compression standard
- Has 3 lossless modes and 1 lossy mode

    sequential baseline encoding

    encode in one scan

    input & output data precision is limited to 8 bits, while quantized DCT values are restricted to 11 bits

    - progressive encoding
    - hierarchical encoding
    - lossless encoding

Can achieve compression ratios of up-to 20 to 1 without noticeable reduction in image quality

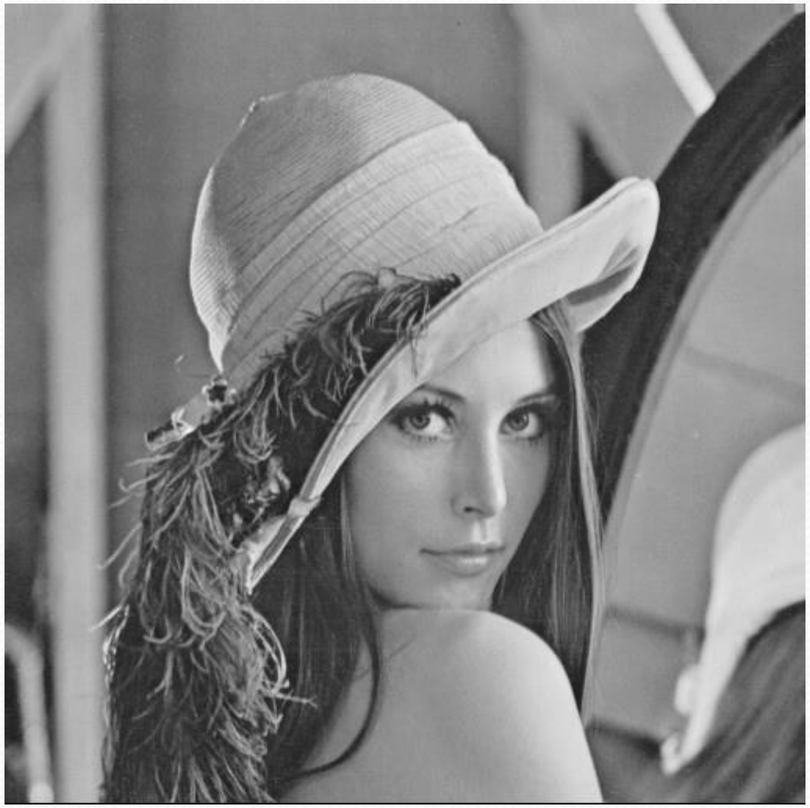
Image Compression Standards

Continues Tone Still Image Comp. - JPEG

- Work well for continuous tone images, but not good for cartoons or computer generated images.
- Tend to filter out high frequency data.
- Can specify a quality level (Q)
  - with too low Q, resulting images may contain blocky, contouring and ringing structures.
- 5 steps of sequential baseline encoding
  - transform image to luminance/chrominance space (YCbCr)
  - reduce the color components (optional)
  - partition image into 8x8 pixel blocks and perform DCT on each block
  - quantize resulting DCT coefficients
  - variable length code the quantized coefficients

Image Compression Standards

JPEG Encoding


Original

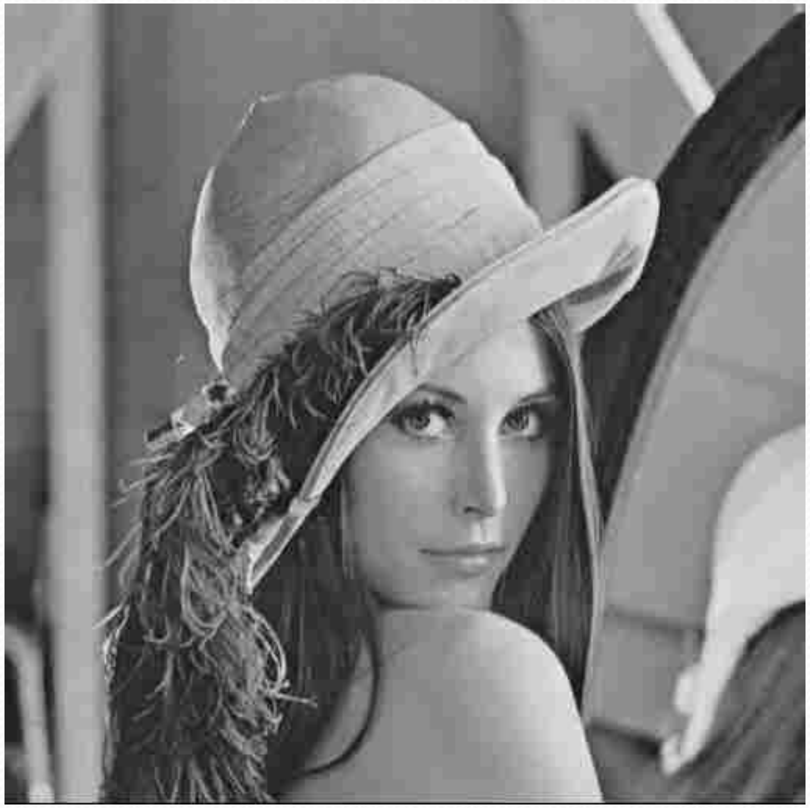

JPEG 27:1

Image Compression Standards

Video Compression Standards

Video compression standards:

1. Video teleconferencing standards
   - H.261 (Px64)
   - H.262
   - H.263 (10 to 30 kbit/s)
   - H.320 (ISDN bandwidth)
2. Multimedia standards
   - MPEG-1 (1.5 Mbit/s)
   - MPEG-2 (2-10 Mbit/s)
   - MPEG-4 (5 to 64 kbit/s for mobile and PSTN and uo to 4 Mbit/s for TV and film application)

THANK

YOU