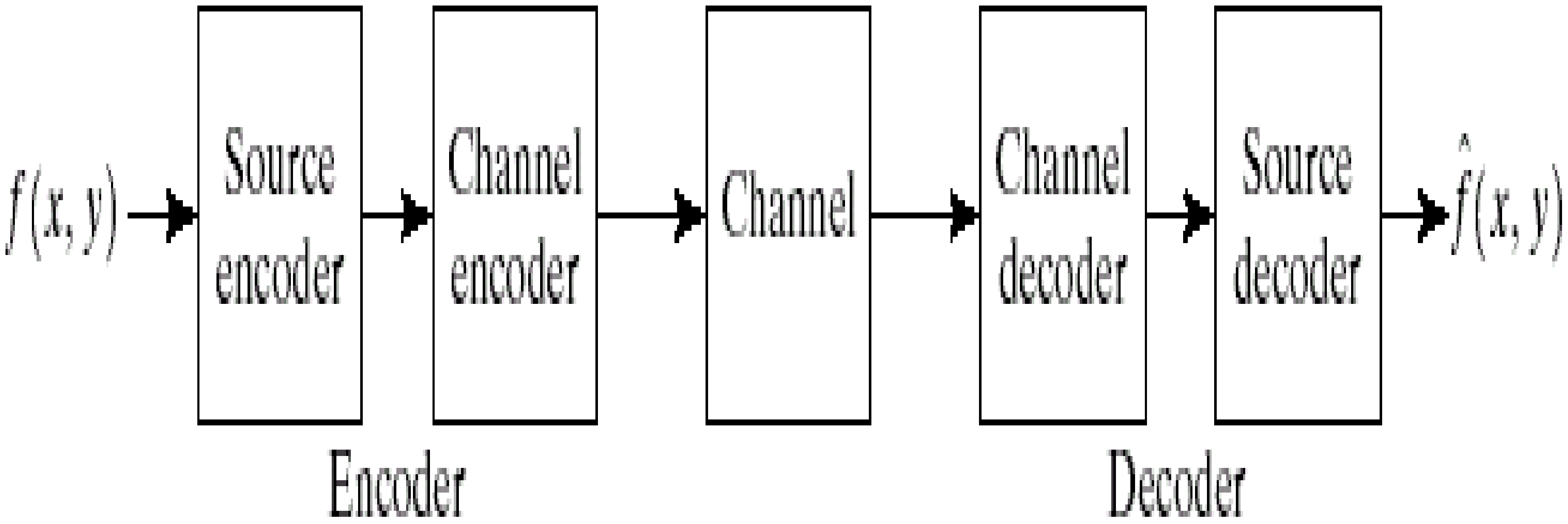
4202 Digital Multimedia

Lecture 3

Dr. Shaimaa Othman

Encoding/Decoding

Overall Encoding decoding Process



Basics or fundamentals of encoding

Encoding schemes:-

Tries to improve or remove one or more of:-

- Coding Redundancy
- Interpixel Redundancy
- Psychovisual Redundancy

Encoding/Decoding Classifications perspectives

- ***Information Loss***
  - Lossy / non-exact/ non-Error free
  - Exact/ lossless/error free
- ***Data unit Length***
  - Fixed
  - Variable
- ***Domain***
  - Spatial
  - Non spatial

Encoding Algorithms Metrics

- Performance Metrics of compression process such as:-
  - Mean Squared Error (MSE), Root Mean Square Error (RMSE)
  - Peak Signal to Noise Ratio (PSNR), (SNR)
  - Compression Ratio (c_R).

- $c_R = (sizebe\ before\ compression)/(size\ after)$
  - Bigger CR is better

- The relative redundancy R_D

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

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

Encoding Algorithms Metrics

For example:

An mxn image $f(i,j)$ and the reconstructed image $g(i,j)$ where $i = 1,2, \dots m$ and $j = 1,2, \dots n$ and defined as:-

- $MSE = \frac{1}{mxn} \sum_{i=1}^{m} \sum_{j=1}^{n} |f(i,j) - g(i,j)|^2$
  - $RMSE = \sqrt{\frac{1}{mxn} \sum_{i=1}^{m} \sum_{j=1}^{n} |f(i,j) - g(i,j)|^2}$
  - SNR $= \frac{\sum_{i=1}^{m} \sum_{j=1}^{n} f(i,j)^2}{\sum_{i=1}^{m} \sum_{j=1}^{n} |f(i,j) - g(i,j)|^2}$
  - $SNR_{db} = 20 \, log_{10}(SNR)$
  - $PSNR = 20 \, log_{10} \left[\frac{2^b - 1}{\sqrt{MSE}} \right]$
    - Where b is the pixel depth in bits.

- Low RMSE and high PSNR means better compression scheme.

Good Encoding scheme

- High CR
- Low MSE, RMSE
- High PSNR, SNR
- Low order complexity of encoding /decoding algorithms
- Ease of realization
- Parallelization Ability
- Hardware realization

Multimedia files

- Header
  - Meta Data which include:-
    - File type
    - Type of encoding
    - Sampling frequency
    - Sample size
    - Encoding information
  - Encoding Tables
- Multimedia data
- Error Correction and or Detection

1- Pixel Packing

- Basic concept :
  - An image of 24 bit sample size could contain 2^{24} $colos$. However
  - Real images contains too much less colors than this number

The Basic Idea

- keep list of colors that exist in the image, in a table and write their colors indices instead of colors
- Example
  - A 16 bit audio file examined and found to have only the following audio codes:-

0,5,17,80,170,200,500,1024,5000,6000,20400,30300,40500,50000,60000

Design a pixel packing encoding to compress the file, what will be data written to replaces

1024,5000,6000,20400,30300,0,5,17,80,170,200,500,1024, 0,5,17,80,170,200,500,1024, 30300,0,5,17,80,170

What will be the compression algorithm?

7,8,9,10,……. Encoding file (need to complete)

1- Pixel Packing

- Encoding
  - Read multimedia file to find out distinct values of codes =➔ construct the table
  - Write header data and save table to the encoded file
  - Seek data start
  - While (NOT EOF)
    - Read multimedia codes find its index in table *say (i)*
    - *Write (i) to the encoded file*
  - *Close file*

1- Pixel Packing

- *Decoding*
  - *Read table from encoded file*
  - Seek data start
  - While (NOT EOF)
    - Read read index *(i)*
    - *Write (table(i)) to the decoded/Uncompressed file*
  - *Close file*

Evaluate? Pixel packing

2- Run Length Encoding

- Basic concept :-
  - In an image file : we could have long sequences of typical codes
  - In an audio file : we could also have in silence interval typical
    repeated long sequences of codes
  - Practical example **<u>Faxing a paper</u>** could include a typical very long white codes
- Key Idea
  - Repeated Data rewritten in terms of <run code, run length>

Example

5,5,5,5,5,5,5,5,3,3,3,3,4,4,3,3,3,3,3,3   original file

<5,8>,<3,4>,<4,2>,<3,6>   encoded file

2- Run Length Encoding

- Basic concept :-
  - In an image file : we could have long sequences of typical codes
  - In an audio file : we could also have in silence interval typical repeated long sequences of codes
  - Practical example **<u>Faxing a paper</u>** could include a typical very long white codes
- Key Idea
  - Repeated Data rewritten in terms of <run code, run length>

  Example
  5,5,5,5,5,5,5,5,3,3,3,3,4,4,3,3,3,3,3,3,7,8
  <5,8>,<3,4>,<4,2>,<3,6>
- If two bits run length <5,4>,<5,4>,<3,4>,<4,2>,<3,4>,<3,2>,<7,1>,<8,1>

2- Run Length Encoding

- Basic concept :-
  - In an image file : we could have long sequences of typical codes
  - In an audio file : we could also have in silence interval typical repeated long sequences of codes
  - Practical example **<u>Faxing a paper</u>** could include a typical very long white codes
- Key Idea
  - Repeated Data rewritten in terms of <run code, run length>

  Example
  5,5,5,5,5,5,5,5,3,3,3,3,4,4,3,3,3,3,3,3,7,8
  <5,8>,<3,4>,<4,2>,<3,6>

- If ten bits run length <5,8>,<3,4>,<4,2>,<3,6>,<7,1>,<8,1>

2- Run Length Encoding

- **Questions need answers**
  - Run code number of bits?
  - Run length number of bits?

- Encoding
  - Scan the file for determination to suitable size for the run length ***rb***
  - ***Write rb size and code size to header as well as type of encoding***
  - ***Seek Data start***
  - ***Read code into cc***
  - ***While (NEOF)***
    - ***Rl=0***
    - ***While (read code into tc == cc and Rl <2^{rb})***
      - ***Rl++;***
    - ***Write <cc , Rl>***
    - ***cc=tc***
  - ***Close file***

2- Run Length Encoding

- Decoding
  - Read from header size of the run length ***rb and size of he code***
  - ***Seek Data start***
  - ***While (NEOF)***
    - ***Read pair <cc , Rl>***
    - ***While (Rl not negative)***
      - ***Write to decoded file (cc)***
      - ***Rl--;***
  - ***Close file***

Example

Run Length Encoding?