MULTIMEDIA DATA COMPRESSION

Compression Principles

Introduction

- Uncompressed graphics, audio and video data require considerable storage capacity.
- Data transfer of uncompressed multimedia data also requires very high bandwidth to be provided for a single point-to-point communication.
- *Compression is the process of effectively reducing the total number of bits needed to represent multimedia information.*

Introduction

- In computer science, **data compression** or **source coding** is the process of encoding information using fewer bits (or other information-bearing units) than a more obvious representation would use, through use of specific encoding schemes.
- One popular instance of compression that many computer users are familiar with is the jar/ZIP file format, which, as well as providing compression, acts as an archiver, storing many files in a single output file.

Introduction

- As is the case with any form of communication, compressed data communication only works when both the sender and receiver of the information understand the encoding scheme.
- Similarly, compressed data can only be understood if the decoding method is known by the receiver.

Introduction

- Compression is possible because most real-world data have *statistical redundancy*.
- For example, the letter 'e' is much more common in English text than the letter 'z', and the probability that the letter 'q' will be followed by the letter 'z' is rather small.

Types of Redundancy

- *Spatial Redundancy*: The values of neighbouring pixels are strongly correlated in almost all natural images.
- *Redundancy in scale*: Important image features such as straight edges and constant regions are invariant under rescaling.
- *Redundancy in frequency*: In images composed of more than one spectral band, the spectral values for the same pixel location are often correlated and an audio signal can completely mask a sufficiently weaker signal in its frequency vicinity.

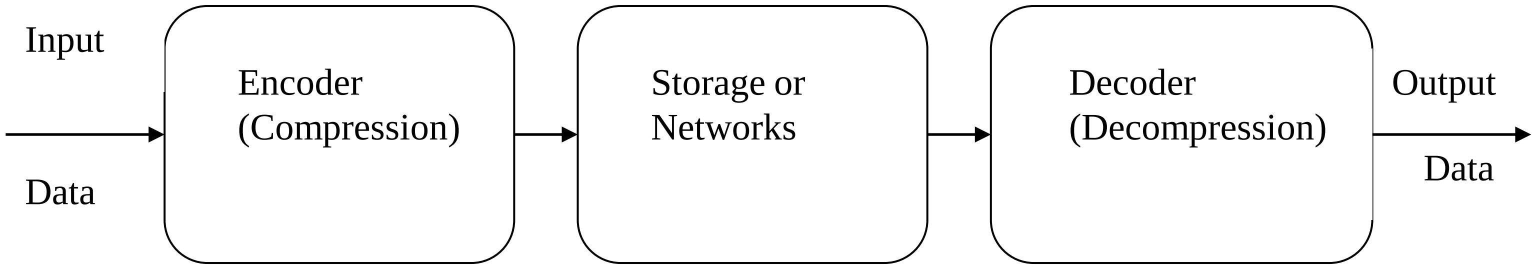
Types of Redundancy

- *Temporal Redundancy*: Adjacent frames in a video sequence often show very little change, and a strong audio signal in a given time block can mask an adequately lower distortion in a previous or future block.
- *Stereo Redundancy*: Audio coding methods can take advantage of the correlations between stereo channels.

Types of redundancy

- Spatial redundancy and redundancy in scale are typical for video streams, whereas stereo redundancy can be exploited only in the coding of audio streams.
- All other types of redundancy can be found in audio as well as in video steams.
- Compression is first applied to the source information to be transmitted to:
  - Reduce the volume of information to be transmitted (text, fax and images).
  - Reduce the bandwidth that is required for transmission (speech, audio and video).

General Compression Scheme



- The output of the encoder is called *codes* or *codewords*.
- The intermediate medium can either be data storage or a communication /computer network.
- If the compression and decompression processes induce no information loss, the compression scheme is called Lossless. Otherwise, it is lossy.

Compression Trade-offs

- Compression is important because it helps reduce the consumption of expensive resources, such as disk space or connection bandwidth.
- However, compression requires information processing power, which can also be expensive.
- The design of data compression schemes therefore involves trade-offs between various factors including:
  - Compression capability.
  - Any amount of introduced distortion.
  - Computational resource requirements e.g video conference.

Compression Ratio

- If the total number of bits required representing the data before compression is B0 and the total number of bits required to represent the data after compression is B1, then the compression ratio is defined as
  - Compression Ratio = B0 / B1
- Generally, any codec (encoder/decoder scheme) having a compression ratio larger than 1 is desired.
- The higher the compression ratio, the better the Lossless compression scheme, as long as it is computationally feasible.

Compression Principles

- Compression algorithms are based on certain principles.
- These principles can be discussed under the following headings:
  - Source encoders and destination decoders.
  - Lossless and lossy compression.
  - Entropy encoding.
  - Source encoding.

Source Encoders & Destination Decoders

- Prior to transmitting the source information relating to a particular multimedia application, a compression algorithm is applied to it.
- This implies that in order for the destination to reproduce the original source information, or in some instances, a nearly exact copy of it, a matching decompression algorithm must be applied to it.

Source Encoders & Destination Decoders

- The application of the compression algorithm is the main function carried out by the source encoder and the decompression algorithm is carried out by the destination decoder.
- In applications, which involve two computers communicating with each other, the time required to perform the compression and decompression algorithm is not always critical.
- So both algorithms are normally implemented in software within the two computers.

Lossless and Lossy Compression

- In the case of lossless compression algorithms, the aim is to reduce the amount of source information to be transmitted in such a way that, when the compressed information is decompressed, there is no loss of information.
- Therefore, Lossless compression is said to be reversible.

Lossless and Lossy Compression

- An example of application of Lossless compression is for the transfer over a network of text file since, in such applications, it is normally imperative that no part of the source information is lost during either the compression or decompression operations.

Lossless and Lossy Compression

- The aim of lossy compression algorithms is normally not to reproduce an exact copy of the source information after decompression but rather a version of the source information, which is perceived by the recipient as a true copy.
- In general, with such algorithms, the higher the level of compression being applied to the source information, the more approximate the received version becomes.

Lossless and Lossy Compression

- Examples of applications of lossy compression are for the transfer of digitized:
  - Images.
  - Audio and
  - Video streams.
- In such cases, the sensitivity of the human eye or ear is such that any fine details that may be missing from the original source signal after decompression are not detectable.

Lossy and Lossless Compression

- Lossy techniques usually achieve higher compression rates than lossless ones but the latter are more accurate.

Entropy Encoding

- **Entropy** is defined as the average information content of given data.
- It defines the minimum number of bits needed to represent the information content without information loss.
- Entropy coding tries to come as close as possible to this theoretical lower limit.
- The decompression process reconstructs the original data completely.

Entropy Encoding

- The data stream to be compressed is considered as a simple digital sequence, and the semantics of the data are ignored i.e. it is used for encoding regardless of their specific characteristics.
- This is a Lossless and independent of the type of information that is being compressed. It is concerned with solely how the **information is represented**.
- Examples of entropy encoding are:
  - **Run-length** encoding and **Statistical encoding**

Run Length encoding

- This is used when the source information comprises of long substrings of the same character or binary digits.
- Instead of independent codewords or bits, it is transmitted in the form of a different set of codewords, which indicate:
  - The particular character or bit being transmitted
  - An indication of the number of characters/bits in the substring.

Run Length Encoding

- Then provided the destination knows the set of codewords being used, it simply interprets each codeword received and outputs the appropriate number of characters/bits.
- For example, in an application that involves the transmission of long strings of binary bits that comprise a limited number of substrings, each substring can be assigned a separate codeword. The total number bit string is then transmitted in the form of a string of codewords selected from the codeword set.

Run length Encoding

- In many instances, for example, when scanning typed documents, the scanner produces long substrings of either binary 0s or 1s. Instead of transmitting these directly, they are sent in the form of a string of codewords, each indicating both the bit (1 or 0) and the number of bits in the substring.
- E.g. If the output of the scanner is 000000011111111110000011.. Then this could be represented as 0!7 1!10 0!5 11………………….
- It should be clear that it is useless to replace sequences of characters shorter than four by this run-length code because no compression can be achieved.

Statistical Encoding

- Many applications use a set of codewords to transmit the source information.
- E.g. a set of ASCII codewords are often used for the transmission of strings of characters.
- Normally all the codewords in the set comprise a fixed number of binary bits, for example 7 bits in the case of ASCII.
- However, in many applications, the symbols and hence codewords that are present in the source information do not occur with the same frequency of occurrence i.e. with equal probability.

Statistical Encoding

- E.g. in a string of text, the character A may occur more frequently than, say, the character P which occurs more frequently than the character Z.
- Statistical encoding exploits this property by using a set of variable-length codewords with the shortest codewords used to represent the most frequently occurring symbols. In practice, the use of variable-length codewords is not quite as straightforward.

Statistical Encoding

- As with run-length encoding, the destination must know the set of codewords being used by the source.
- However, with variable-length codewords, in order for decoding operation to be carried out correctly, it is necessary to ensure that a shorter codeword in the set does not form the start of a longer codeword otherwise the decoder will interpret the string on the wrong codeword boundaries.

Statistical Encoding

- A codeword set that avoids this happening is said to possess the **prefix property** and an example of an encoding scheme that generates codewords that have this property is the Huffman encoding algorithm.
- The theoretical minimum average number of bits that are required to transmit a particular source stream is called the **entropy** of the source.

Source Encoding

- This processes original data such that a distinction between relevant and irrelevant is possible. It takes into account the semantics of the data. Removal of the irrelevant data compresses the original data stream. In contrast to entropy encoding, source encoding is often a lossy process.
- This exploits a **particular property** of the source information in order to produce an alternative form of representation that is either a compressed version of the original form or it is more suitable to the application of compression.

Source Encoding

- Two types of source encoding algorithms are:
  - *Differential* encoding and
  - *Transform encoding*.

Differential Encoding

- Differential encoding is used extensively in applications where the amplitude of a value or signal covers a large range but the difference in amplitude between successive values or signals is relatively small.
- To exploit this property of the source information, instead of using a set of relatively large codewords to represent the amplitude of each value/signal, a set of smaller codewords can be used each of which indicates only the difference between the current value or signal being encoded and the immediately preceding value/signal.

Differential Encoding

- For example, if the digitization of an analog signal requires, say, 12 bits to obtain the required dynamic range but the maximum difference in amplitude between successive samples of the signal requires only 3 bits, then by using only the difference values, a saving of 75% on transmission bandwidth can be obtained.
- In practice, differential encoding can be either Lossless or lossy and depends on the number of bits used to encode the difference values.

Differential Encoding

- If the number of bits used is sufficient to cater for the maximum difference value, then it is Lossless.
- If this is not the case, then on those occasions when the difference value exceeds the maximum number of bits being used, temporary loss of information will result.

Transform Encoding

- Transform encoding involves transforming the source information from one form to another, the other form lending itself more readily to the application of compression.
- In general, there is no loss of information associated with transform operation and this technique is used in a number of applications involving both images and video.

Transform Encoding

- The most widely known transform coding is **Discrete Cosine Transform** (DCT).
- Other examples are:
  - Fourier Transform (FT).
  - Discrete wavelet transform (DWT)

The End

Next>> Text Compression