Data compression

Lecture №5

Lecture’s Outline

The aim of topic: understand how data is compressed, data compression types and methods.

Agenda:

- Text Compression
- Lossy and Lossless Compression
- Audio Compression
- Image Compression
- Video Compression

Data Compression

Data compression is used everywhere. Many different file types use compressed data. Without data compression, a 3-minute song would be over 100Mb in size, while a 10-minute video would be over 1Gb in size. Data compression shrinks big files into much smaller ones. It does this by getting rid of unnecessary data while retaining the information in the file.

Data compression can be expressed as a decrease in the number of bits required to illustrate data. Compressing data can conserve storage capacity, accelerate file transfer, and minimize costs for hardware storage and network capacity.

Text Compression

“Ask not what your country can do for you - ask what you can do for your country.”

J.F. Kennedy

Ask not what your country can do for you - ask what you can do for your country. [79 symbols]

The quote has 17 words, made up of 61 letters, 16 spaces, one dash and one period. If each letter, space or punctuation mark takes up one unit of memory, we get a total file size of 79 units. To get the file size down, we need to look for redundancies.

Here we have words that are repeated two times

- ask
- what
- your
- country
- do
- for
- you

Text Compression

The system for arranging dictionaries varies, but it could be as simple as a numbered list. When we go through Kennedy's famous words, we pick out the words that are repeated and put them into the numbered index. Then, we simply write the number instead of writing out the whole word.

So, if this is our dictionary:

Instead of words that have occurred in text more that one time insert numbers:

- ask - 1
- what - 2
- your - 3
- country - 4
- can - 5
- do - 6
- for – 7
- you – 8

Text Compression

Replaces words by digits:

1 not 2 3 4 5 6 7 - 1 2 8 5 6 7 3 4 [37 symbols]

If you knew the system, you could easily reconstruct the original phrase using only this dictionary and number pattern.

Store words and its encodings:

1ask2what3your4country5can6do7for8you [37 symbols]

Total:

37+37 = 74 < 79 (5 symbol place was saved)

Text Compression

Not bad! But can do be better

No matter what specific method you use, this in-depth searching system lets you compress the file much more efficiently than you could by just picking out words. Using the patterns we picked out above, and adding "__" for spaces, we come up with this larger dictionary:

1. ask_
2. what_
3. you
4. r_country
5. _can_do_for_you

Text Compression

And this smaller sentence:

"1not_2345_-_12354“

The sentence now takes up 18 units of memory, and our dictionary takes up 41 units. So we've compressed the total file size from 79 units to 59 units!

Lossy and Lossless Compression

Lossy compression loses data, while lossless compression keeps all the data. With lossless compression, we don't get rid of any data. Instead, the technique is based on finding smarter ways to encode the data. With lossy compression, we get rid of data, which is why we need to distinguish data from information.

Lossless compression allows the potential for a file to return to its original size, without the loss of a single bit of data, when the file is uncompressed. Lossless compression is the usual approach taken with executables, as well as with text and spreadsheet files, where the loss of words or numbers would change the information. Lossless compression can compress the data whenever redundancy is present. Therefore, lossless compression takes advantage of data redundancy.

Lossy Compression

- Lossy compression is the one that does not promise that the data received is exactly the same as data send *i.e.* the data may be lost.
- This is because a lossy algorithm removes information that it cannot later restore.
- Lossy algorithms are used to compress still audio, images and video.
- Lossy algorithms typically achieve much better compression ratios than the lossless algorithms.

Audio Compression

Audio compression is used for speech or music.

For speech, we need to compress a 64-KHz digitized signal; For music, we need to compress a 1.411.MHz signal

Two types of techniques are used for audio compression:

- 1. Predictive encoding
- 2. Perceptual encoding

Audio Compression

Predictive encoding

- In predictive encoding, the differences between the samples are encoded instead of encoding all the sampled values.
- This type of compression is normally used for speech.

Perceptual encoding

- Perceptual encoding scheme is used to create a CD-quality audio that requires a transmission bandwidth of 1.411 Mbps.
- MP3 (MPEG audio layer 3), a part of MPEG standard uses this perceptual encoding.
- Perceptual encoding is based on the science of psychoacoustics, a study of how people perceive sound.
- The perceptual encoding exploits certain flaws in the human auditory system to encode a signal in such a way that it sounds the same to a human listener, even if it looks quite different on an oscilloscope.

Image Compression

There are two types of compression:

Lossless compression: do not lose image's quality

Lossy compression: in real-world photos some regions may contain pixels that doesn't differ for human's eyes, but different in RGB. E.g. #FF0001, #FF0000 , save them as one color.

Image Compression

Lossless compression

Most of graphics (logos, graphics) contain limited variety of colors. (E.g. flag of Japan contains only two colors (red and white))

So we can store information about color of pixel not in 3 bytes, but in one bit.
(E.g. that 0 is red and 1 is white)
Used for Icons and graphics

only 4 colors

Image Compression

Lossy compression

Real-world photos has many different colors used, but many of them are neighbouring colors, that can’t be differentiated by human’s eye. So we can use instead of **two** different color notation **one** notation.

cannot be used for graphics and icons since it tends to make it one color

Lossy compression is used for real-world photos

hundreds of colors

Image Compression

Image types: BMP

- developed by Microsoft
- large file size
- lossless compression

Image Compression

Image type: GIF

- supports animation
- limited to 256 colors
- transparency support
- lossless compression
- Usage: animation, logos

Image Compression

Image type: JPEG

- no transparency
- lossy compression
- small file size
- Usage: photos

Image Compression

Image type: PNG

- was created as improvement of GIF
- transparency
- Lossless compression
- not limited to 256 colors
- Usage: icons, web elements

Image Compression

Image type: WEBP

- New format of images developed by Google
- 26% smaller than PNG
- will support animation
- 25-34% smaller than JPEG
- supported by Chrome and Opera

Video Compression

Videos combine image compression with audio compression. There are usually separate codecs for each aspect of a video, which are then wrapped together as a single compression codec. Because of the high data rate required for uncompressed video, most video files are compressed using lossy compression. The most prevalent form of (lossy) video compression is MPEG.

Video Compression

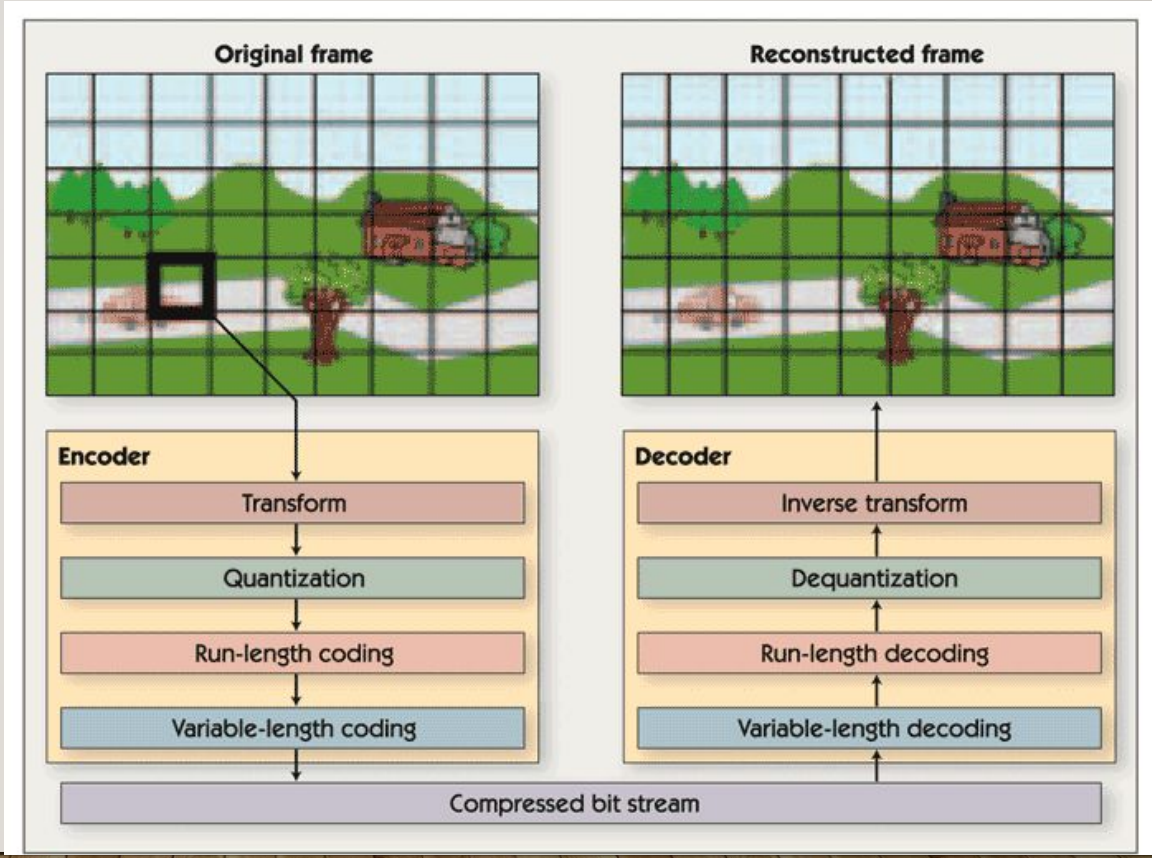
Video is a sequence of images

Most of video compression algorithms: tracks changes between two images, and stores only changed pixels

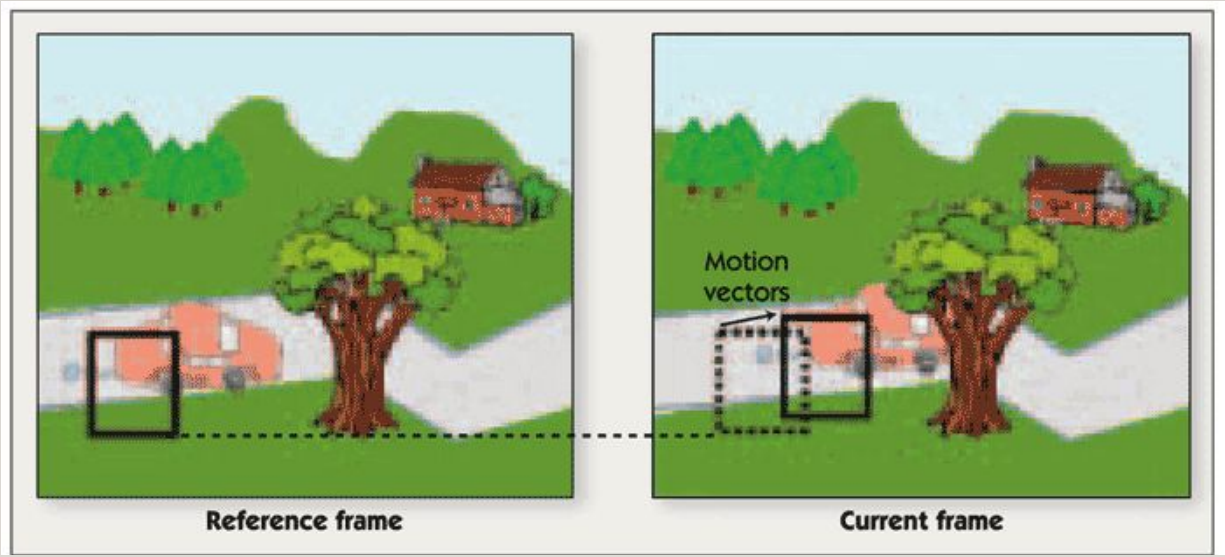
Video clips are made up of sequences of individual images, or “frames,” video compression algorithms share many concepts and techniques with still-image compression algorithms. Therefore, we begin our exploration of video compression by discussing the inner workings of transform-based still image compression algorithms such as JPEG

Video Compression

The image compression techniques used in JPEG and in most video compression algorithms are “lossy.” That is, the original uncompressed image can't be perfectly reconstructed from the compressed data, so some information from the original image is lost.



Video Compression



Lossless Compression Algorithms

The various algorithms used to implement lossless data compression are :

- 1. Run length encoding
- 2. Differential pulse code modulation
- 3. Dictionary based encoding

Lossless Compression

Run length encoding

This method replaces the consecutive occurrences of a given symbol with only one copy of the symbol along with a count of how many times that symbol occurs. Hence the names ‘run length’.

- For example, the string AAABBCDDDD would be encoded as 3A2B1C4D.
- A real life example where run-length encoding is quite effective is the fax machine. Most faxes are white sheets with the occasional black text. So, a run-length encoding scheme can take each line and transmit a code for while then the number of pixels, then the code for black and the number of pixels and so on.
- This method of compression must be used carefully. If there is not a lot of repetition in the data then it is possible the run length encoding scheme would actually increase the size of a file.

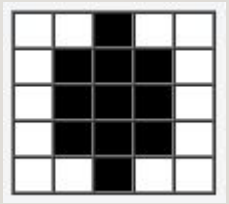
RUN-LENGTH ENCODING

RLE isn't widely used nowadays

E.g. you have image on right

Instead of bitmap we can save it as:

white,black
2,1,2
1,3,1,
1,3,1,
1,3,1,
2,1,2



Lossless Compression

Differential pulse code modulation

• In this method first a reference symbol is placed. Then for each symbol in the data, we place the difference between that symbol and the reference symbol used.

• For example, using symbol A as reference symbol, the string AAABBC DDDD would be encoded as A0001123333, since A is the same as reference symbol, B has a difference of 1 from the reference symbol and so on.

Lossless Compression

Dictionary based encoding

One of the best known dictionary based encoding algorithms is Lempel-Ziv (LZ) compression algorithm.

- This method is also known as substitution coder.
- In this method, a dictionary (table) of variable length strings (common phrases) is built.
- This dictionary contains almost every string that is expected to occur in data.
- When any of these strings occur in the data, then they are replaced with the corresponding index to the dictionary.
- In this method, instead of working with individual characters in text data, we treat each word as a string and output the index in the dictionary for that word.
- For example, let us say that the word “compression” has the index 4978 in one particular dictionary; it is the 4978^{th} word is usr/share/dict/words. To compress a body of text, each time the string “compression” appears, it would be replaced by 4978.

Lempel-Ziv Text Compression

While looking through document, if some part of text occurred before, insert some notation which shows distance to that text and length of repeated text

Do not have dictionary part

http://www.unicode.org/notes/tn31/tn31-2.html

LZ realization

Original Text:

ask not what your country can do for you - ask what you can do for your country

Compressed Text:

ask not what your country can do for [24,3] - [44,4][44,8][18,40][30,14][50,9]

Thank you for your attention!