Week 12
Ch 11 (Multimedia)

Representing colour

What is colour? Colour = physiological & psychological response to the wavelengths of light entering our eye.

How is colour created? How are these sensations created?

- Young-Helmholtz trichromatic theory. Based on observations about colour mixing, Thomas Young (1802) launched the hypothesis that there are 3 types of cells in our retina, sensitive to
  - light in the red spectrum
  - -"- green -"-
  - -"- blue -"-

(we see light spectrum picture).

→ his theory was confirmed experimentally only recently.

(see fig. of spectral sensitivity of the 3 types of cells in human retina)

Representing colour

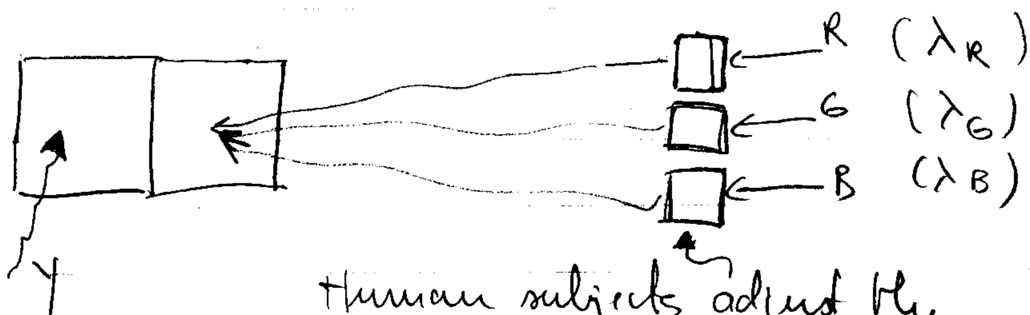
- there are many ways of representing colours.

• Typical = RGB. (mirror the properties of photosensitive cells in our retina).

How the RGB representation is obtained:

- numerous experiments with human subjects

(Ex) How to represent the color sensation made by light with a certain wavelength Y:



Human subjects adjust the intensities of RGB lights until the color sensation they perceive is same as that of light Y.

The wavelengths of the three sources of light R, G, B depend on the phosphorous materials used in the construction of monitors

Most direct representation of colour: using the intensities of the 3 primary colours determined earlier.

Problem: the intensity = continuous (analog) variable & can take any value between 0 ... Max.

Digitizing the intensities of primary colours:

$0 \longleftrightarrow Max$

000 (0)	8 possible values	111 (7)	3 bits
0000 (0)	16 possible values	1111 (15)	4 bits
⋮			⋮
0000 0000 (0)	256 possible values	1111 1111 (255)	8 bits
			⋮

the choice of # of bits used in representing the intensity of primary colors determines how many different colors we can represent.

ex) 3 bits / primary colour $\rightarrow \left\{ \begin{array}{l} 8 \text{ values R} \\ 8 \text{ -"- G} \\ 8 \text{ values B} \end{array} \right. = 8^3$ colours $=$ $= 512$ colours

8 bits / primary colour $\Rightarrow \left\{ \begin{array}{l} 256 \text{ values R} \\ 256 \quad \text{G} \\ 256 \quad \text{B} \end{array} \right. =$ approx 16 millions of colours.

Examples of colours coded using RGB representation with 8 bit / primary colour

R	G	B	
255	255	255	- white
0	0	0	- black
100	100	100	- gray
180	15	15	- probably a redish colour
	⋮		

Question: what is RGB representation of a dark shade of orange?
- difficult to answer.

For this reason, people have invented other representations for colour.
(ex) HSV (Hue, Saturation, Value) color model.

(see PowerPoint slides for HSV color model)

The HSV-model makes the mapping between our perception of colour and its representation (3 numbers) easier to see. Like for RGB, HSV color representations use 3 numbers, this time representing the H, V & S components).

Conclusions

- color = sensation
- color is not only determined by a physical phenomenon (eg light) but also by
  - physiological factors
  - psychological —"—

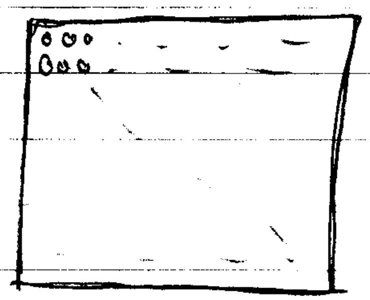
- goal in representing multimedia (eg for storing info) = to represent a phenomenon that can induce similar sensation as the original. The focus is on sensation not on replicating the phenomenon.

  the same idea present when digitizing images or sounds, and allows us to create very efficient encodings.

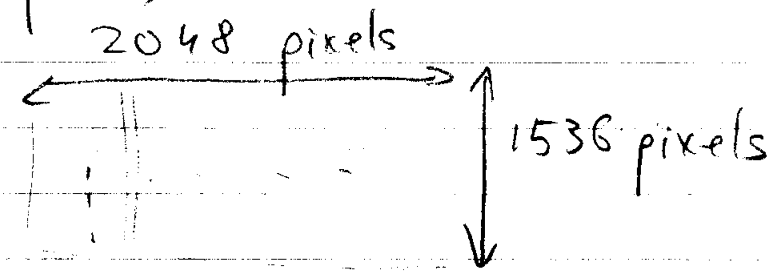
Week 12 day 2

- we've seen that color can be represented in various ways:
  - 3 integers representing R G B
  - 3 — " — " — " — Hue Saturation Value
  - there are many other ways

- An image (ex a digital photograph) - is a collection of pixels in a rectangular area. Pixels - have colours, so 1 pixel can be represented by 3 integers (R G B)



Example: a typical photograph made with old 3.2 M.P. (mega pixel) camera has size:



If each pixel ⇝ 3 integers of 8 bits each (1 byte)
Then the whole image requires $2048 \times 1536 \times 3 =$
$= \underbrace{3\,145\,728}_{\text{pixels}} \times 3 \cong$ almost 10 MB (mega bytes) to be represented.

10 MB of information

-> 80 sec to download with a high speed internet connection (1Mb/sec)

-> can store about 25 photos on a 256 MB memory card

Success of multi-media on Internet comes from representing a very similar image with significantly fewer bytes. This is called COMPRESSION.

(ex) Similar 10 MB image can take 128 KB if we want to use it for viewing on screen, in presentations, etc...

128 KB of info:

-> 0.2 sec to download (high speed Internet)

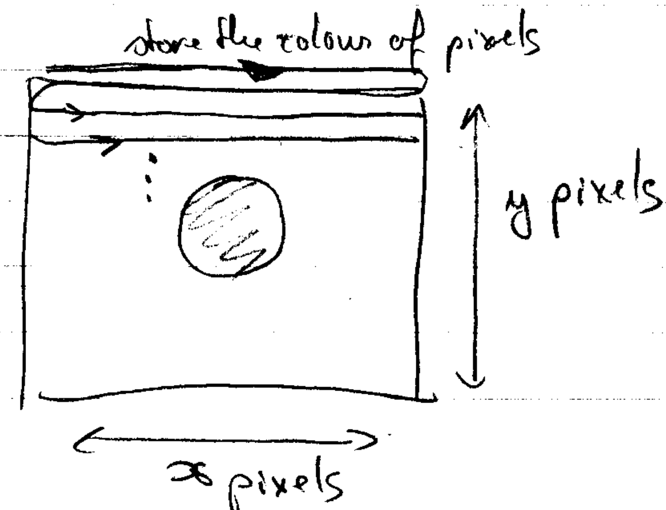
-> about 2000 photos on a 256 MB card.

Methods for COMPRESSION

- lossless (exact information is maintained)
- lossy (represent a different information / image which looks however the same)

LOSSLESS COMPRESSION

Simple image (a coloured disc on black background)



- to store in a file, we store the colour of each pixel, for example row by row.
  By convention, suppose we store in the file, as the first #, the value x. We need x to know how to display the image later on.

$$x \; \underbrace{0\;0\;0}_{\text{colour of first pixel, eg black}} \; \underbrace{0\;0\;0}_{\text{colour of 2nd pixel}} \; \ldots \text{ etc.}$$

$\downarrow$ we represent the image as a sequence of numbers.

How many numbers? How many bits in total?

- for value x – let's use 16 bits so that we can store a # as large as 65535
- for intensities of RGB, let's use 8 bits

Then, size of representation is (in multiple of 8 bits or 1 byte):

$$2 + 3 * y \quad \text{bytes}$$

However data looks like:

$$x \; \underbrace{000\,000 \ldots \ldots 000}_{\text{a sequence of } n \text{ zeros}} \; 25 \; 105 \; 34 \; 000 \ldots$$

replace the whole sequence by 2 numbers.

$(0 \;\; n)$

the value — how many times should be repeated

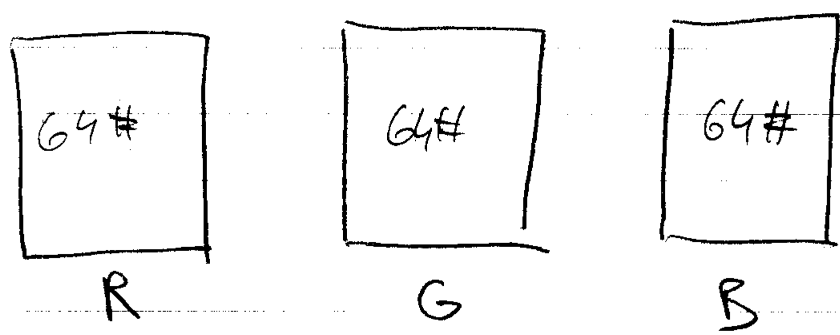
Instead of n bytes (original sequence) we use only 2 bytes to encode exactly same information.

This technique doesn't work very well on real images (photos).

- JPEG compression (lossy compression):

  ⇒ works with blocks of 8×8 pixels from the image:

  (in fact, 3 blocks)



(Obs 1) → the numbers in each of the bocks, do not usually differ by a lot

→ replace all numbers in each block by average, Store, of course only 1 # per block.

(we expect to save space 64 ×)!

The image is noticeable bad! (see picture on PPT slides)

→ We need a middle approach

For illustration, focus on 1 row of 8 #.

$$\underbrace{\underbrace{(2\ \ 3\ \ 3\ \ 4}_{\text{avg } 12/4=3}\ \ \underbrace{8\ \ 7\ \ 3\ \ 2)}_{20/4=5}}_{\text{average } = 32/8 = 4} \rightarrow \text{call the sequence } X.$$

Replacing X by 4444 4444 looks to the eye worse than replacing X by 3333 5555.

(This is like averaging over smaller grid)

We can store now for sequence X, 2 #.

4 (the average of whole sequence) and -1 the difference from this average, because with difference we can calculate the 2 averages over the smaller grid:

$4-1=3 \qquad 4+1=5$

Always, the two averages over smaller grid have the same difference from the total average, just the sign is changed.

We prefer to store the information as:
- total average (4), difference (-1)

and not
- average small grid (3), average small grid (5)

because:

• - differences are small #, most often 0, 1, 2 ... If we have a lot of occurrences of the same value, these can be compressed losslessly in efficient manner.

The same process can be applied further, on the smaller grids... For ex., for the first half:

2 3 3 4 ← original info

3 (avg

2.5 3.5 -> these 2 values now represent even a better image. We can represent both values by the difference again: -0.5

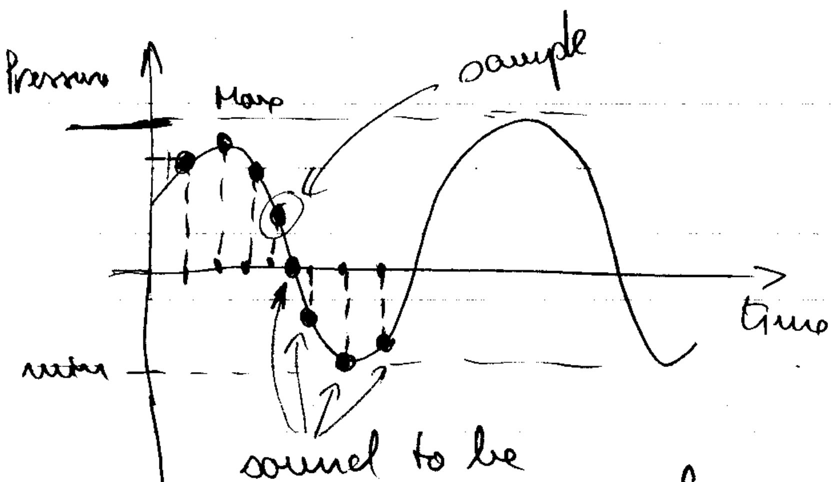
and so on...

Week 12 day 3

Digitizing sound

-> digitizing sound uses the same idea as in digitizing color intensities

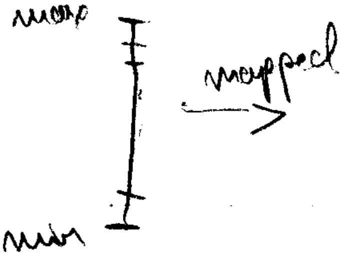
Sound waveform:



← continuous variable (pressure) (there is an infinite # of values for pressure)

To represent sound digitally, we need to choose discrete points (particular points in time) when we measure the pressure. This is called sampling

- measured value -> any number between min & max
  -> needs to be represented in binary form



integers < 3 bits, 8 bits, 16 bits }

max

min

the more the # of bits we use to represent a sample, the larger the # of integers representing different values of a sample and the better the accuracy of the representation

Sampling rate

- rule: Because sound perceived by humans ranges between 20 Hz ... 20 kHz in frequency, the sampling rate should be at least 2x as frequent as frequency of highest pitched sound.

- for CD-quality music, a typical sampling rate: 44 100 Hz (44 100 samples / sec.)
- if using 16 bit / sample => 1 minute of music can be represented with $16 \cdot 60 \cdot 44\,100$ bits, or about 4 MB.

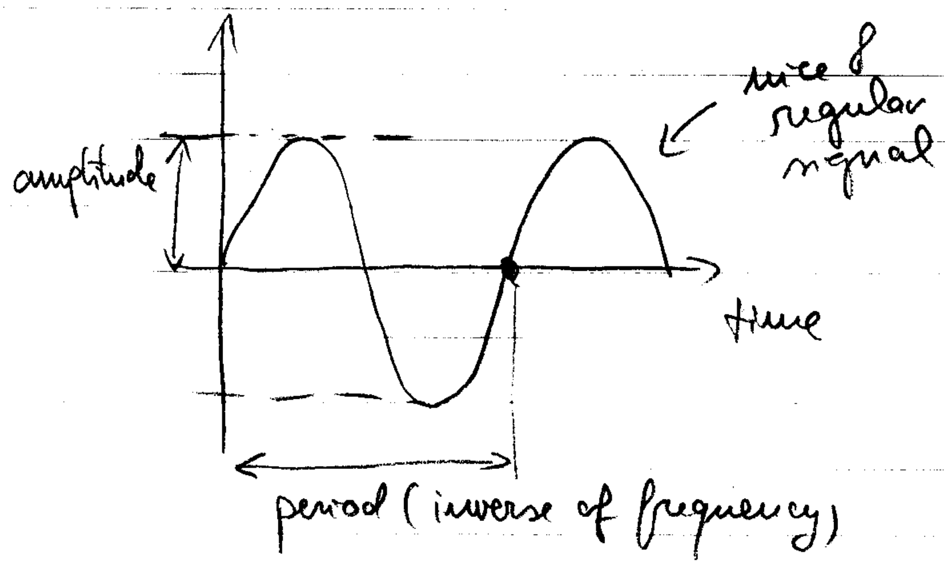
• we can reduce sampling rate → this affects the quality of recording significantly.
(hear 3 sample files (.WAV))

MP3 encoding (lossy)

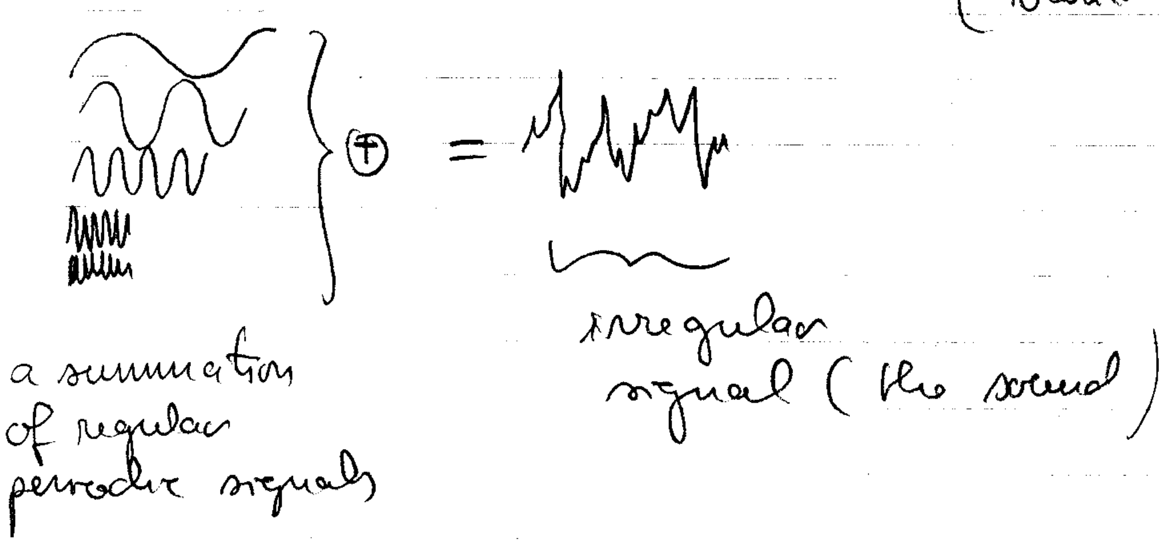
- Motion Picture Expert Group Audio Layer #3 encoding.

- the idea is similar to JPEG compression

(1) Any sound wave can be decomposed into many regular, sinusoidal signals of different frequencies & amplitudes.



in math, any function can be decomposed in a lot of sine & cosine signals of different frequencies & amplitudes (Fourier Transform)



Therefore, a sound can be represented

- time domain (value of samples & time when sample was taken)
- frequency domain : list of frequencies & amplitudes.

To encode a track in MP3 format, we use the frequency domain to determine what sounds cannot be heard by human ear. Then, simply we do not encode these sounds & we can reduce the size of the encoding significantly.

The rules that ~~tell~~ us what sounds cannot be heard by human ear = psychoacoustic model.

ex "acoustic masking" phenomenon

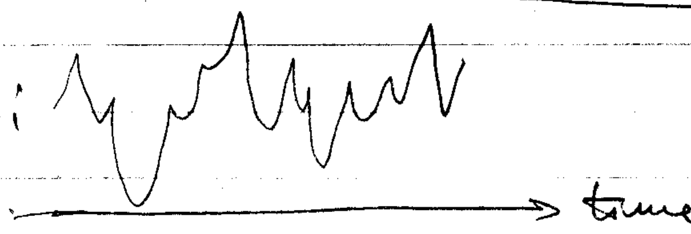
→ if 2 sounds of similar (but not identical) frequencies are played, the weaker sound will not be heard.

→ the detection of such situations can be easily performed in frequency domain..

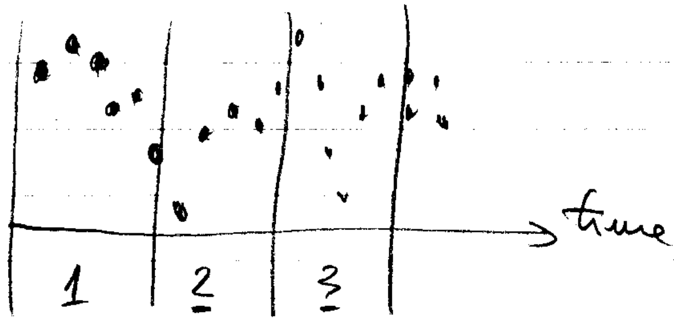
(eg: play file "masking-frequencies.wav".

How all this works together

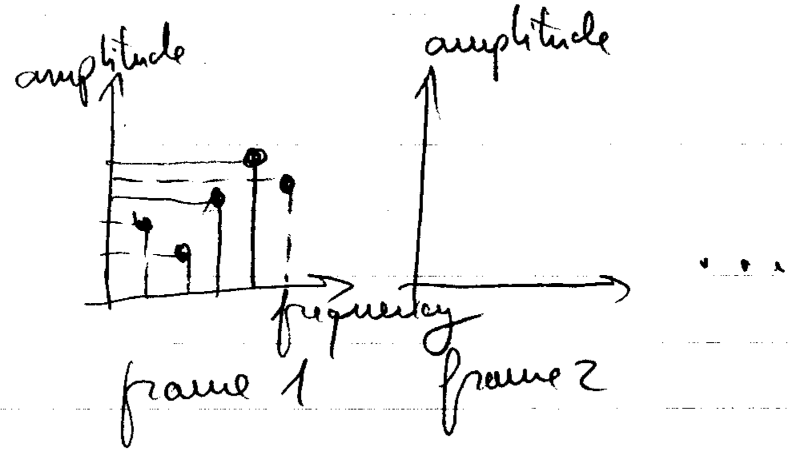
Sound:



Sampling



Samples are split in blocks (called frames) of same duration.
MP3 lossy compression is applied on each frame.



Samples in frames are converted to frequency domain
- redundant sounds are eliminated
- remaining data in frequency domain is further simplified (getting rid of finer details) & then represented digitally using bits.

The # of bits we want to use for the

data in a frame = bit rate.
Typical value for bit-rate = 128 kb/sec.
(CD-quality sound)

(ex) - test sounds for 8 kb/sec, 32 kb/sec & VBR

Recap of formats for multimedia

- images: - lossy : JPEG (files with jpg extension).
  - good for photographs of nature, portraits etc.
  - not so good for man made images like drawings, etc.
- lossless - GIF
  - TIF
  - PNG → quite popular.

Obs When sharing photos from digital cameras, if the purpose is viewing on screen, scale down the resolution (# of pixels) from several thousand pixels width to at most 1000 pixels width.
Keep original resolution only for printing.

Movies

MPEG files (lossy) — AVI / MPG / DVD format } all based on JPEG image compression & MP3 compression for sounds

Sound

(lossy) — MP3 – most popular format. (under patent)
— WMA – windows media player alternative
— M4A – iTunes
— OGG-VORBIS → completely free of patents, open source