National Centre for Computer Animation

Image Perception & 2D Images

Vision is a matter of perception.
Perception is a matter of vision.

ES Overview

- Introduction to ES
- **2D Graphics in Entertainment Systems**
- Sound, Speech & Music
- 3D Graphics in Entertainment Systems

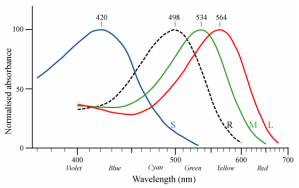
Image Perception & 2D Images

- Human Vision & Colour Perception
- Digital Images
- Image manipulation (*overview*)

Sensing Light

Colour perception does not function uniformly:

- more than half of the colour receptors of the human retina (*about 64%*) perceive red light only (*long waves*), while a third of receptors (*about 32%*) are attuned to green light and only a small fraction (*about 4%*) actually perceive blue light (*short waves*)
- a lot of blue light is also absorbed by the lens of the eye and never reaches the retina, making the human eye much more insensitive to blue coloured objects than red coloured objects
- the lens of the eye performs no colour correction – short waves focus on the retina in front of long waves and consequently only one kind of light wave can be perceived sharply at any one time



Normalised absorption spectra of human cone (S,M,L) and rod (R) cells (*source: Wikipedia*)

Only a small area of the retina can perceive sharp images – most image data is only sensed peripherally (*appears blurred*).

Understanding Perception

- light sensed by the receptors within the retina is processed by so called detectors in the part of the brain which handles colour information from the eyes.
- the three types of detectors are **light-dark** receptors, **red-green** receptors and **blue-yellow** receptors.
  - these detectors are responsible for sensing complementary colours.
  - however since each detector only handles one of the pair of complementary colours, the use of objects in close proximity that are coloured in complementary colours can make it impossible to distinguish between the two.

This explains many optical illusions.

Digital Images

Digital pictures
- Image representation
- Colour models

File formats
- Uncompressed
- Compressed

Digital video

Image Representation

Images are represented on-screen as pixels (*picture elements*) on a raster of rows and columns

- Image resolution (*measured in dpi – dots per inch or ppi – pixels per inch*) dictates the quality (*sharpness etc.*) of the image
- There are two kinds of digital image representation
  - As pixel image
  - As vector image

Image Representation

Pixel Images (*aka* ***bitmap images*** *aka* ***raster graphics***)

- Store images as a rectangular grid (*rows and columns*) of pixels.
- Often used for photographic images.

Resizing pixel images usually results in a loss of image quality.

Image Representation

Vector Images (*aka* ***vector graphics***)

- Store images as a mathematical representation of the shapes (*geometrical primitives*) shown in the picture. (*TrueType fonts are stored as vector images*)
- Less useful for photographic pictures.

Vectorised images can be resized without a loss in image quality.

Colour Models

Colour models are mathematical representations of colours as combinations of different components.

Common colour models are:

- RGB
- CMY(*K*)
- HSB
- YUV

Colour Models

RGB colour (*Red, Green and Blue*)

- Uses the three primary colours.
- Colour mixing is additive (*mixture results in white*).
- Most commonly used colour model in digital entertainment systems.

Colour Models

CMY(K) colour (*Cyan, Magenta, Yellow, Key - black*)

- Complementary to RGB.
  RGB = **{(**1-C**)(**1-K**),(**1-M**)(**1-K**),(**1-Y**)(**1-K**)}**
- Colour mixing is subtractive (*mixture results in black*).
- Used for print media (*with black as 4th colour for extra rich representation*).

Colour Models

HSB colour (*Hue, Saturation and Brightness*)

- Uses the three components:
  - Hue (*colour type/wavelength representation, i.e. red or blue*)
  - Saturation (*colour intensity*)
  - Brightness (*greatest spectral component*)
- Used mainly in computer graphics (*to create transitions*)
- Every HSB colour has an RGB equivalent

Colour Models

YUV

Uses a colour model based on three colour components:

- Luminance Y (*brightness in weighted terms of RGB sum*)
  `Y = 0.299 * R + 0.587 * G + 0.114 * B`
- Chrominance U (*scaled Y value subtracted from B value*)
  `U = 0.436 * (B - Y) / (1 - 0.114)`
- Chrominance V (*scaled Y value subtracted from R value*)
  `V = 0.615 * (R - Y) / (1 - 0.299)`

Used for broadcast images (*PAL – phase-alternating line*)

Bit Depth

the bit depth of an image describes how many data bits are used to encode each pixel

- measured in bpp (*bits per pixel*)
- 24 bpp image usually used for RGB images (*1 byte per colour channel, R, G, B*)

24 bit (*1 byte per channel*) 8 bit (*256 colour palette*) 8 bit greyscale

Image Storage

Storing images can be very problematic:

- image of 800 by 600 pixels (*3 bytes each*) ≈ 1.4 Mb
- 3 Megapixel camera image ≈ 8.5 Mb
- file sizes can be massive
  - moving images are worse: usually 25 frames per second (FPS)
  - 1 minute of video at 3 Megapixel image quality ≈ 12.5 Gb

Image compression should be used (*unless uncompressed image data is required*)

Image Compression

There are 2 kinds of image compression:

- lossless image compression
  - usually results in larger file sizes
  - image content can be fully restored to original state
  - example formats: BMP (*with RLE*), GIF (*LZW*), TIFF (*LZW*)
- lossy image compression
  - usually results in smaller file sizes
  - some image content is lost
  - example formats: JPEG, Adobe Illustrator Artwork

Digital Video

Video really must be compressed (*otherwise it would be unaffordable*).

most commonly used video file format is **MPEG**

- based on JPEG image compression
- only stores every n^{th} image (*key frame*) and only an approximation of changes for the frames inbetween

alternative would be **M-JPEG**

- every frame is stored as a JPEG image
- much larger file size than MPEG

compression methods are also known as codecs

Image Manipulation

Image Processing involves manipulation of pixel values.

Examples:

- converting image to grayscale
  - replace each pixels colour channel value with average pixel value: **(R+G+B)/3**
- adding glare to an image
  - calculate highlight of each colour channel and add to original colour channel value
- edge detection, blurring etc.
  - uses convolution filters → see Leigh‘s tutorials this week