Images and Colour

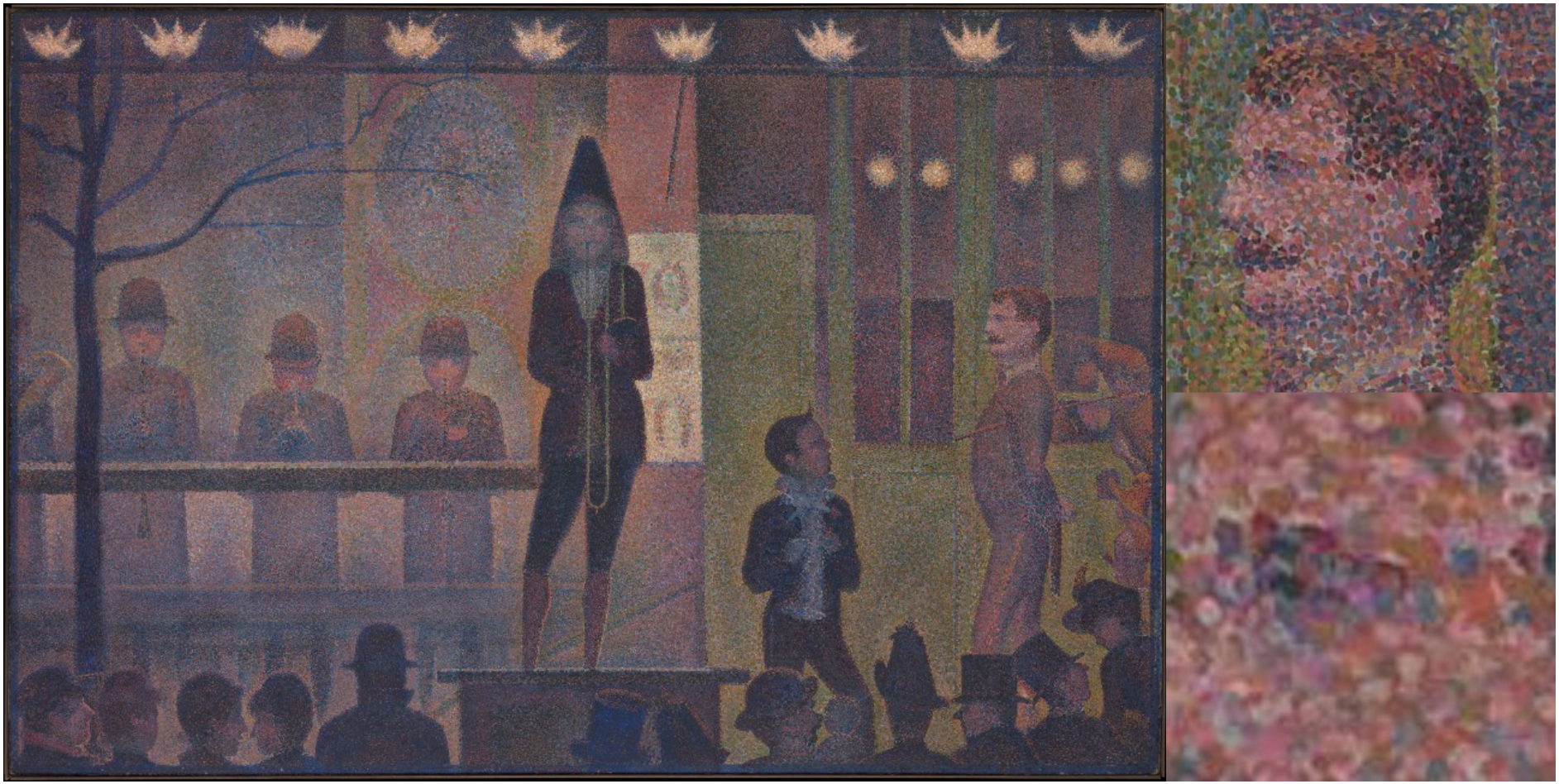
COSC342

Lecture 2
2 March 2015

In this Lecture

- Images and image formats
  - Digital images in the computer
  - Image compression and formats
- Colour representation
  - Colour perception
  - Colour spaces

Georges Seurat – Parade de Cirque



Metropolitan Museum of Art, New York

Images in the Computer

- A image, for our purposes, is a 2D array of pixels (*pic*ture *ele*ments)
- Each pixel may be a single value, or a list of values
  - A single value gives a monochrome greyscale image
  - Colour images typically have three values
  - Usually these are red, green, and blue (but see later)
- These values may be integers, floating point numbers, etc.
  - Most commonly integers in the range $[0, 255]$
  - Monochrome images are sometimes called 8-bit images
  - Colour images are 24-bit
  - Floats in the range $[0, 1]$ are also common during processing

Images in a Computer

$(189, 51, 71)$

$(176, 229, 234)$

$(116, 129, 89)$

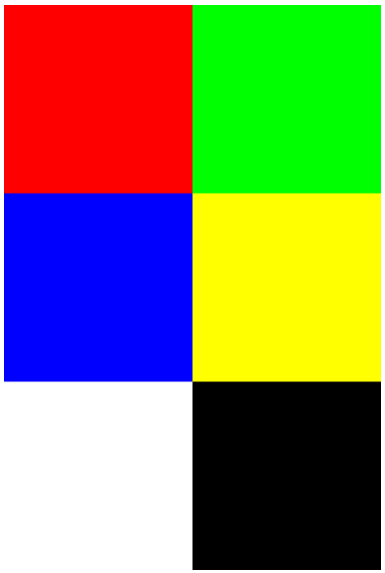
Images File Formats

- There are many different image file formats
- Simple file formats use a lot of space
- Compression is used to reduce this
  - *Lossless* compression means you can get the original data back
  - *Lossy* compression doesn't, but can give smaller files
- Common file formats:

Format	Compression	Colours	Features
JPEG	Lossy	Full colour	Small file sizes
GIF	Lossless	256 colours	Animation, transparency
PNG	Lossless	Full colour	Transparency
BMP	Lossless	Full colour	Simple format
PNM	None	Full colour	Very simple format

Sample PPM File

- The Portable aNyMap format has many versions
- This is an example of a small ASCII Portable PixMap (PPM) file.



```
P5
2 3 255
255   0   0     0 255   0
  0   0 255   255 255   0
255 255 255     0   0   0
```

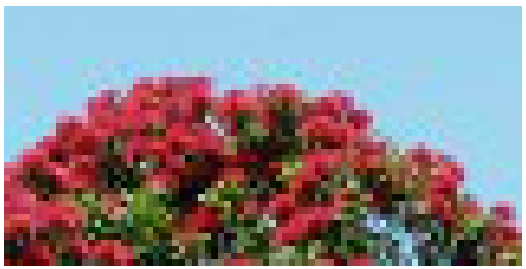
Compression

- Picture files get big fast – 8-megapixel images are 24MB
- Simple method – Runlength encoding
  - Suppose we have a sequence of 10 blue pixels
  - Raw data would be `0 0 255`, repeated 10 times
  - Instead, store `0 0 255 10`
  - When will this work? When is it a bad idea?
- More complex methods:
  - Algorithms like LZW create dictionaries of repeating strings
  - The change from one pixel to the next is often small
  - Frequency based methods - Discrete Cosine Transform in JPEG

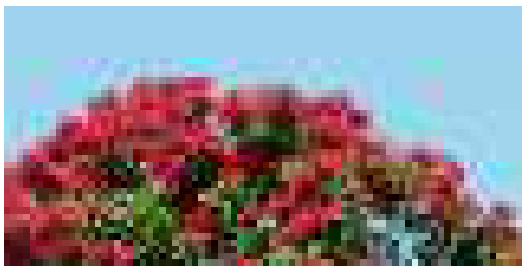
Compression Artefacts

- JPEG's lossy compression works on 8×8 blocks of pixels
- It has a parameter to trade off quality and file size

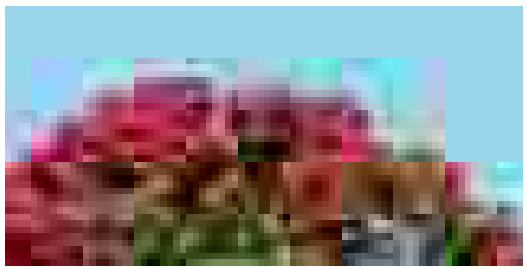
Quality = 90



Quality = 50

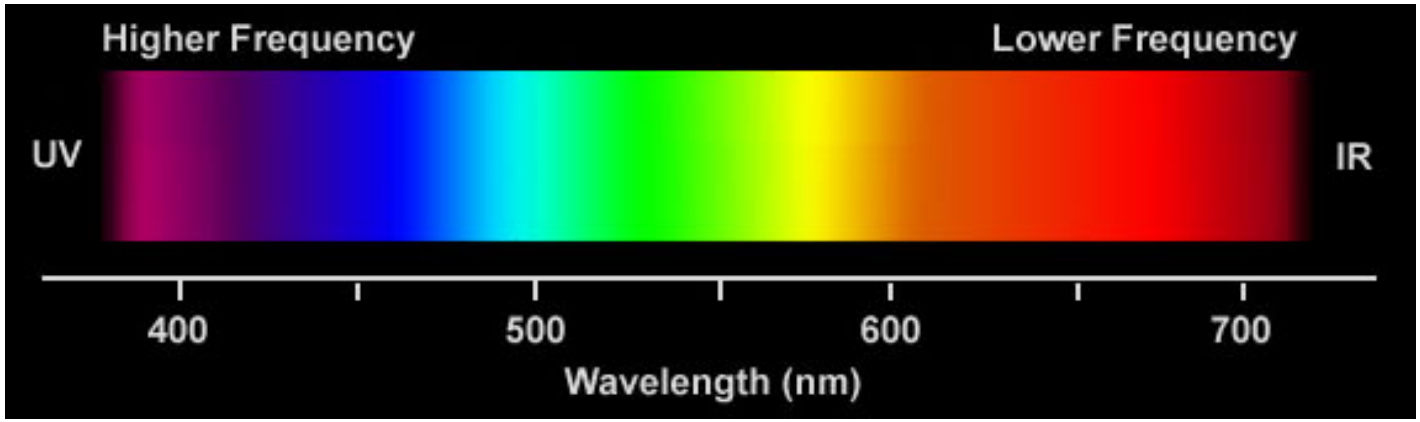


Quality = 10



Colour is Complicated

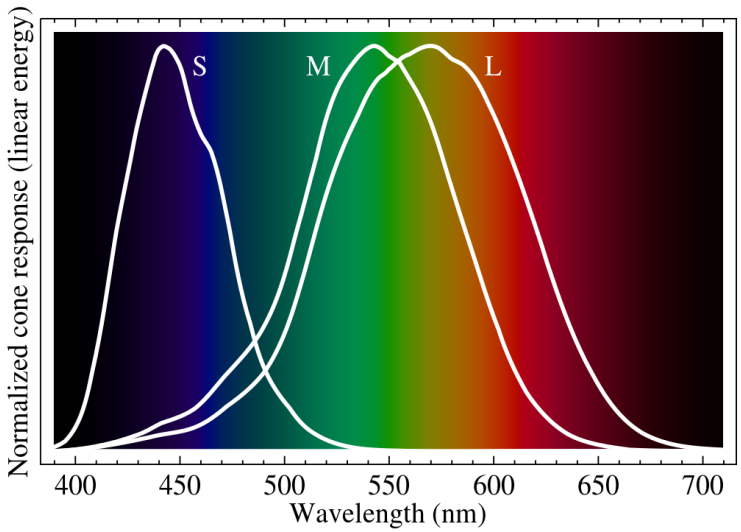
- Colour is often thought of as frequency or wavelength of light



- Blue + green light = cyan, a bright 'bluey-green'
- But red + green = yellow, not 'reddish-green'
- Even weirder, red plus blue light gives magenta (sort of purple)
- Blue + yellow paint gives green, but blue + yellow light gives white
- What's going on?

Human Colour Perception

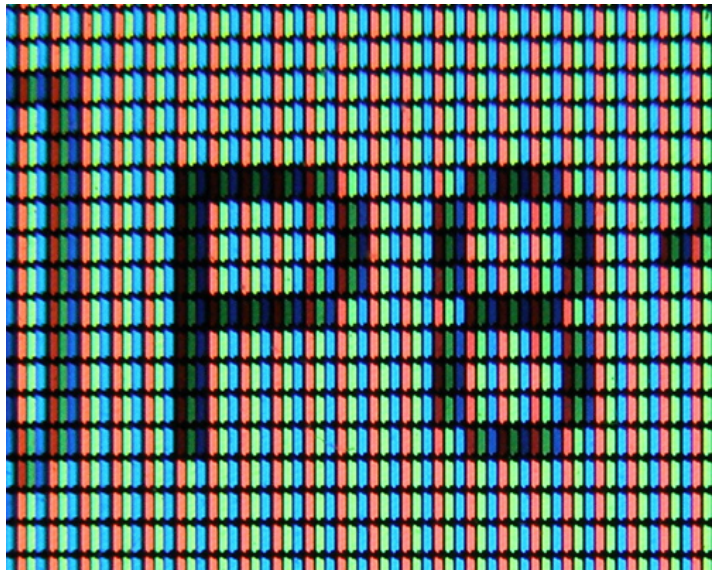
- Most people have four types of light receptors in the retina
  - Rods, which are just sensitive to brightness
  - Three types of cone, sensitive to short, medium, and long wavelengths



  - These are often called ‘blue’, ‘green’ and ‘red’ cones
- Before the signal reaches the brain, some processing happens
  - The brain receives *differences* between cone responses
  - One axis is $L - M$ (red–green)
  - The other axis is $S - (L + M)$ (blue–yellow)

Red-Green-Blue

- RGB is the most common colour space in computing
- It is used in displays – most monitors have red+green+blue pixels



- Red + Green + Blue = White
- Mixing the 'primary' colours gives secondary colours
- R + G = Yellow, G + B = Cyan, B + R = Magenta

RGB $\leftrightarrow$ Greyscale

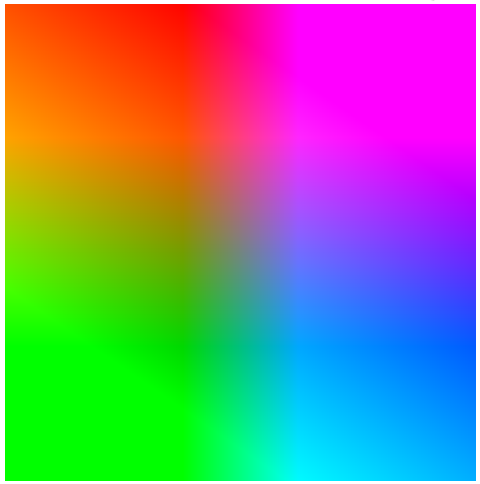
- Converting RGB to greyscale isn't just $(R + G + B)/3$
- Our eyes are more sensitive to green light, and less to blue
- The rough weighting is $0.3R + 0.6G + 0.1B$

$(R + G + B)/3$

$0.3R + 0.6G + 0.1B$

YUV / YCrCb

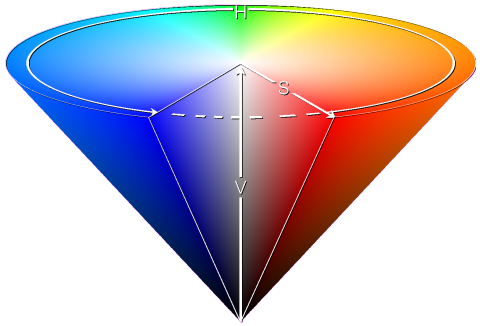
- While we have 3 types of cone, we see colour on two axes



  - Y is intensity (what our rods see)
  - U (Cr) is the blue–yellow axis, right to left
  - V (Cb) is the red–green axis, top to bottom
- Our eyes are much more sensitive to changes in intensity than colour
- This can be used in compression – use more bits for Y than U or V
- YCrCb is used in JPEG, and was used in analogue broadcast TV

Hue-Saturation-Value

- The way we *describe* colour is different again
- We talk about general hues – red, green, etc.
- We talk about 'bright' or 'strong' colours



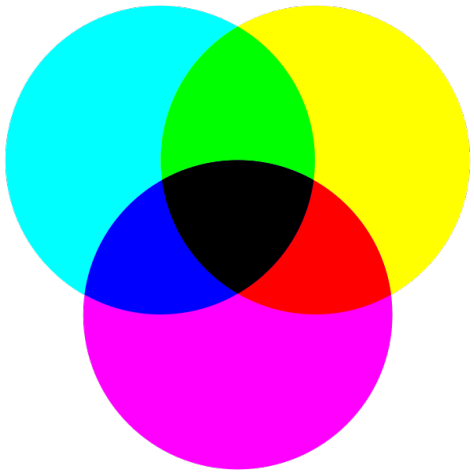
- Hue is the angle around the colour wheel
- Saturation is how *strong* the colour is
- Value is how light or dark the colour is

- Often used in colour-pickers, since it is quite intuitive
- Related (but different) is HSL (Hue-Saturation-Lightness)

HSV_cone.png CC-BY-SA-3.0, Moongateclimber `http://commons.wikimedia.org/wiki/File:HSV_cone.png`

Cyan-Magenta-Yellow(-blacK)

- RGB is an additive colour model – we add light
- In printing, subtractive models are needed

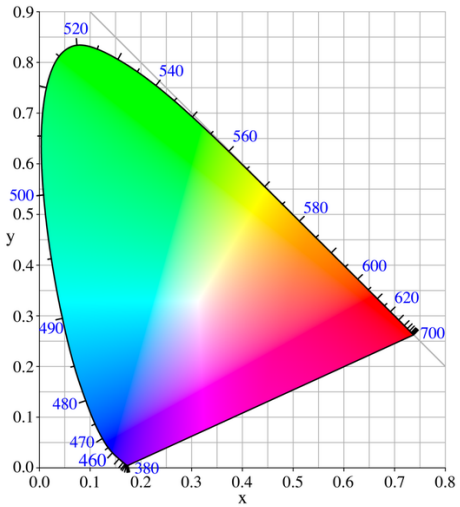


  - Cyan ink subtracts red light
  - Magenta ink subtracts green light
  - Yellow ink subtracts blue light

- In theory, C+M+Y = Black, but this is hard to achieve
- In practice, a true black ink is used – CMYK

CIE XYZ

- Based on scientific approach to perception of colour
- Several different versions, the one below is CIE 1931



- This area covers all the colours we can see
- Around the edge are pure wavelengths
- Third axis (not shown) is brightness
- The line between any two points is a mixture of the end point colours

- Any RGB or CMY process gives a triangle within the spectrum
- This is called the *gamut* of the display

Tutorials and Labs

- Next week's tutorial
  - Matrix and vector mathematics
  - Have a go *before the tutorial*
  - Come along if you have any questions or issues
- Monday's Lab:
  - Blender practice
  - Building a building