1. Colourspaces

Colour images are encoded as triplets of values.
RGB is an **additive** colour model that is used for **light-emitting** devices, e.g., CRT displays. **CMY** is a **subtractive** model that is used for **printers**.
RGB and CMY are used for imaging, YUV (PAL) and YIQ (NTSC) are used for video. YUV & YIQ use properties of the human eye to prioritize information. Y is the black and white (**luminance**) image, U and V are the colour difference (**chrominance**) images. **YCbCr** is used in JPEG and MPEG (and is sensitive to **skin tones**).
HSV (Hue, Saturation, and Value/brightness) and HLS (Hue, Lightness, and Saturation) are also commonly used.
Generally speaking, the translation from RGB to CMY is a simple one:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

The RGB and CMY colour spaces can each be represented by a cube.

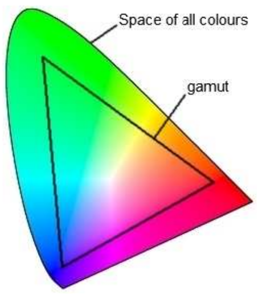
The **CIE-XYZ** colour space can be represented on a two-dimensional space by manipulating the RGB vector, such that $X + Y + Z = 1$. Given this, only two value are required to describe **chromacity** (Y is **luminance**)

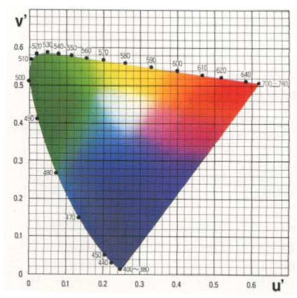
$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$x = \frac{X}{X+Y+Z} \qquad y = \frac{Y}{X+Y+Z} \qquad z = 1 - (x + y)$$

Different standards, including YUV (PAL) and YIQ (NTSC) place different values in the 3x3 matrix.

A colour **gamut** is a certain complete subset of colours, that portion of the visible colour space that can be represented, detected, or reproduced, e.g. by a monitor or a printer.
The curved edge of the colour space shows all the colours of a **pure** wavelength.





CIE-u'v' shifts the positions of the colours on the x-y plane for the **human** eye, such that equal linear distances are equally detectable.

This reflects the fact that the human eye can detect changes in blue more readily than changes in green.

Grassman's Laws:
(i) mixing two lights produces the **same** results as mixing each of the R, G, B components, i.e.

given

$$T_1 = w_1R + w_2G + w_3B$$
$$T_2 = w_4R + w_5G + w_6B$$

then

$$T_1 + T_2 = (w_1 + w_4)R + (w_2 + w_5)G + (w_3 + w_6)B$$

(ii) if the weights of two lights match, then the lights **match**, i.e.

given

$T_1 = w_1R + w_2G + w_3B$

$T_2 = w_4R + w_5G + w_6B$

then

if $w_1 = w_4$, $w_2 = w_5$ *and* $w_3 = w_6$

then $T_1 = T_2$

(iii) applying a linear product to a colour produces the **same** result as applying the product to each R, G, B component, i.e.

if

$T_1 = w_1R + w_2G + w_3B$

then

$k \cdot T_1 = k \cdot w_1R + k \cdot w_2G + k \cdot w_3B$

Colour **quantization** is concerned with **efficiencies**. Reducing the number of colours in an image can *compress* the size of the image without noticeable degradation.

There are different formulae (standards) for converting from **colour to greyscale**, for example,

ITU standard (*D65*): Grey scale = Y = .222*Red + .707*Green + .071*Blue

This reflects the fact that the eye derives more **brightness** from green than it does from blue.

Greyscale to colour is a little trickier. Since there is no colour information in greyscale, only luminance,

Note: The **mixing** of light can appear to give new colours, e.g. blue + yellow = green, but this is **different** from pure green light. It is the way the colour receptors and the brain interpret the blue and yellow, effectively *averaging* the wavelength, that creates the illusion of green light. There is no wavelength for ‘white light’ – it too is an illusion!

2. Segmentation

Segmentation is a *process* not a *task*.

Images can be segmented according to their **features**, which includes **textures**. Features can be grouped based on proximity, similarity, enclosure and continuity. Used in medical applications (CT, MRI, microscopy).

Objects consist of many **regions**. Identifying *objects* is a higher-level function.

There are four popular segmenting approaches:

threshold techniques – uses intensity levels, ignores spatial information, blurred boundaries are difficult to identify;

boundary (or edge)-based – use contour detection, weak at connecting broken contours;

area (or region)-based – group neighbouring pixels of similar intensity, over-stringency can cause fragmentation, leniency overmerges and blurs;

connectivity-preserving – aka *active contour*, flexible but can get stuck in a local minima.

Areas can be identified by **histograms**, ***K*-means**, **watershed**, **seed region growing**, intensity, colour and other methods. Most have similar problems in establishing the **number** and **locations** of areas.

For video, both intra-frame and inter-frame segmentation can take place. Need to break video into scenes (i.e. similarity) and check for differences between frames in a scene.

3. Coding & Compression

Coding is **not** the same as *compression*! (think about a large bitmap image)
'**Physical**' coding works on the data regardless of the information it contains; '**logical**' coding uses metadata and **knowledge** about the data to sensibly code/decode.
Compression can be **lossless**, e.g. Huffman encoding ~ optimal, variable-length, using prior probability knowledge – a cookbook; run-length encoding; Lampel-Ziv-Welch ~ dictionary-based encoding, needing no prior knowledge.
Compression can be **lossy**: JPEG is an ISO standard for colour & grey-scale, has 4 modes of operation: sequential, progressive, hierarchical & lossless.

Discrete Cosine Transform (**DCT**), as used in JPEG, helps separate the image into parts of differing importance, similar to discrete Fourier Transform: they transforms a signal or image from the spatial domain to the frequency domain.
Quantization is a process that determines what information can be discarded without a significant loss in quality. Quantisation Tables are derived from empirical experimentation, not theory. By taking the top-left values from the quantization tables, most of the image can be reproduced, discarding the rest.
The eye is **not** very perceptive to small **colour** changes, but **is** sensitive to small changes in **intensity**. YUV, YcbCr, etc, can retain the brightness element but reduce colour information.
Moving Pictures Experts Group, **MPEG-1** part 3 (audio) is used for **mp3**, MPEG-2 is used for **HDTV**. MPEG-4 combines communications and information algorithms, separates object planes (background/foreground).

Image **entropy** is a quantity which is used to describe the 'business' of an image. A flat image with little contrast will have a low entropy. An image of high contrast such as the heavily cratered surface of the moon would be a **high entropy** image,

$$\text{Entropy} = \sum_i p_i \cdot log_2 \cdot p_i$$

Main motivation for compression are reduced **storage** & **bandwidth**. Considerations include quality, **complexity** & delay (important for **real-time** decompression). There is no single "correct" compression algorithm. It depends on the image.

4. Linear filtering

Filters can **reduce noise**, **accentuate edges**, etc. They are **simple** to apply and can be analysed **theoretically**. Includes smoothing (blurring) and edge detection.
Linear filtering *does things* to images.
Images are stored as arrays of numbers. Linear filtering selects groups of pixels and produces a new number. Result is a slightly smaller image.

e.g. a 'block averaging' mask to average four pixels would be:

$$\begin{pmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{pmatrix}$$

$\boxed{\text{KEY}}$ applying a mask is **not** matrix multiplication, but is an **overlay**, i.e. sum the product of each pixel and its corresponding mask pixel.

The size of the resultant image can be fractionally smaller, unless we add artificial values around the edge to preserve the size.

The values of the mask dictate the effect of the filter,
e.g. **Sobel** filters

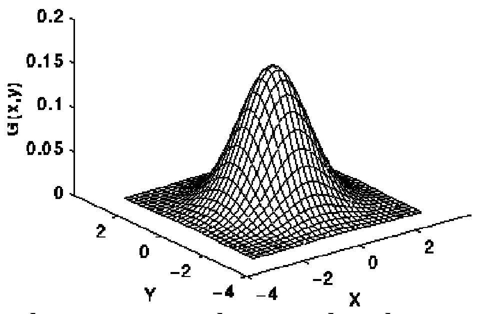
$\begin{pmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{pmatrix} \leftarrow$ accentuates horizontal lines

accentuates vertical lines $\rightarrow \begin{pmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{pmatrix}$

Combining the results of the two filters highlights both horizontal and vertical changes.
Prewitt filters perform a similar function.

Gaussian filters, e.g. $\frac{1}{16}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$ approximate to a smoothing effect

2D Gaussian distribution with mean (0,0) and $\sigma = 1$.



The idea of Gaussian smoothing is to use this 2-D distribution as a `point-spread' function, and this is achieved by convolution. Since the image is stored as a collection of discrete pixels we need to produce a discrete approximation to the Gaussian function before we can perform the convolution. In theory, the Gaussian distribution is non-zero everywhere, which would require an infinitely large convolution kernel, but in practice it is effectively zero more than about three standard deviations from the mean, and so we can truncate the kernel at this point.
Other filters include **mean** filters, **median** filters and **frequency** filters. Compass filters involve rotating the values in the filter, producing 8 different variants for a 3x3 filter.

The **Fourier Transform** is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or **frequency domain**, while the input image is the **spatial domain** equivalent. In the Fourier domain image, each point represents a particular frequency contained in the spatial domain image. The Fourier Transform is used in a wide range of applications, such as image **analysis**, image **filtering**, image **reconstruction** and image **compression**.

5. Non-Linear filtering

The **threshold** operator is non-linear because individually, corresponding pixels in the two images A and B may be below the threshold, whereas the pixel obtained by adding A and B may be above threshold.

Similarly, an **absolute** value operation is non-linear:

$$|-1+1| \neq |-1| + |1|$$

as is the exponential operator:

$$exp(1+1) \neq exp(1) + exp(1)$$

6. Histogram Equalisation

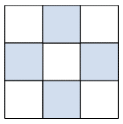
Histogram equalization employs a non-linear mapping which re-assigns the intensity values of pixels such that the output uniform distribution of intensities (i.e. a flat histogram). This technique is used in image comparison processes (because it is effective in detail enhancement).

For an 8-bit greyscale, for example, there are 256 bins. For a 1000x1000 pixel image (i.e. 10^6 pixels) we want to place approximately 4,000 (10^6/256) pixels in each bin, which will deliver an image of **uniform distribution of intensities**. This involves changing the intensity of pixels in order to 'move' them to another bin. The resultant image is not what was seen, but is what can be extracted from the original image.

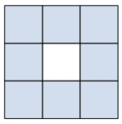
7. Pixel relationships

A pixel *p* at co-ordinates (*x*, *y*) has four *horizontal* and *vertical* neighbours. These are known as the **4-neighbours** of *p*, and are denoted by $N_4(p)$. The four diagonal neighbours of *p* are denoted by $N_D(p)$. The combination of the two are known as the **8-neighbours** of *p*, and are denoted by $N_8(p)$.

4-neighbours:



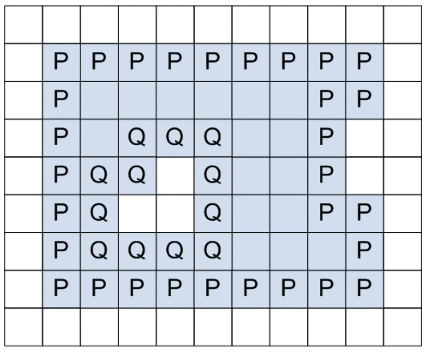
8-neighbours:



A digital **4-path** is a list of pixels such that each pixel in the list is a 4-neighbour of the next pixel in the list. The 4-path is **closed** if the first and last pixels in the list are the same.

A set *S* of pixels is a **4-region** if any two pixels in *S* are joined by a 4-path entirely contained in *S*.

This example shows two 4-paths *P* and *Q*, bounding a 4-region (shaded blue).



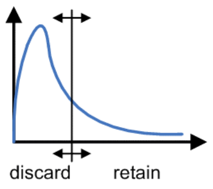
	P	P	P	P	P	P	P	P	P	
	P							P	P	
	P		Q	Q	Q			P		
	P	Q	Q		Q			P		
	P	Q			Q			P	P	
	P	Q	Q	Q	Q				P	
	P	P	P	P	P	P	P	P	P	

8. Structure detection (line, corner, edge)

Method: Apply one or several **linear filters** which have **large responses**, to the structure/image in questions and choose a threshold level, then detect those parts of the image where the filter value exceeds the threshold.

This example filter, $\begin{array}{|c|c|c|}\hline -1 & -1 & -1 \\ \hline -1 & 8 & -1 \\ \hline -1 & -1 & -1 \\ \hline \end{array}$ has the useful property that its entities add up to *zero*, meaning that for uniform areas of the image, the response is 0.

A histogram for response levels after applying a filter should ideally show a decaying curve, since we wish to discard *most* of the image and focus on a small part of it.



Edges – look for large differences in grey levels, with a common direction. Think of an image as a contour map and grey levels as heights. Look for large gradients. Sobel filters offer a computationally cheap and practical way to identify edges.

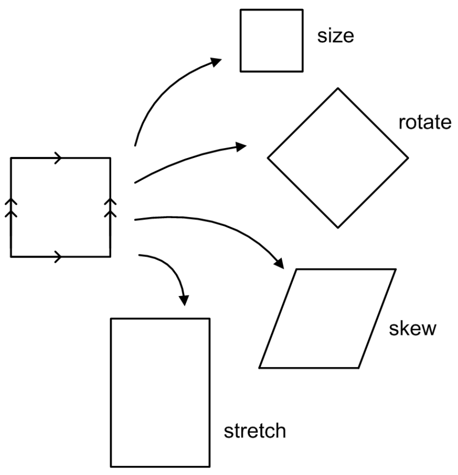
Hough Transform method for line detection, was so good it was patented! Record all the structures on which each point lies, then look for structures that get many votes, e.g. for grouping points that lie on lines, take each point of interest in turn and vote for all lines that could go through it; do this for each point. The line(s) of interest will be those with the most votes.

The formula for a line is $x{\cdot}cos\theta + x{\cdot}cos\theta + r = 0$.

Problems with the Hough Transform include difficulties with **noise**, and choosing the appropriate **granularity**.

9. Affine transformations

One in which parallel lines **remain parallel**



1D:

$y = ax + b$

2D:

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} ab \\ cd \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} c_1 \\ c_2 \end{pmatrix}$$

3D:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix}$$

$\begin{pmatrix} \cos\theta - \sin\theta \\ \sin\theta - \cos\theta \end{pmatrix}$ is a rotation; $\begin{pmatrix} a & 0 \\ 0 & n \cdot a \end{pmatrix}$ is a skew; $\begin{pmatrix} x \\ y \end{pmatrix}$ is a translation.

10. Stereo Image matching

It is desirable to be able to match points in one image to similar points in another. We can do this by calculating the **correlation** between similar points.
To calculate the correlation between two areas of an image,
(i) calculate the **mean average** value of pixel for each area;
(ii) **subtract** the area average from each pixel in that area;
(iii) treating each area as a vector, calculate the **correlation between both vectors**.

e.g. to compute the correlation between

15	44
18	31

and

15	56
40	21

(i) mean average m_1 = (15+44+18+31)/4 = 27
m_2 = (15+56+40+21)/4 = 33
(ii) subtract mean v_1 = (15-27, 44-27, 18-27, 31-27) = (-12, 17, -9, 4)
v_2 = (15-33, 56-33, 40-33, 21-33) = (-18, 23, 7, -12)
(iii) correlation $\rho = v_1 \cdot v_2 / |v_1||v_2| = 0.667$

It is a useful fact that if we **scale** the grey levels, or make the images brighter or duller, the **correlation** is still **the same**.

In general, for a given area in one image, e.g. 5x5, limit the search to a larger area in the second image, e.g. 40x40, but in a similar position within the image, and search for the area with the largest correlation, i.e. correlation closest to 1. The size of the search area is critical – too small and the match may be missed, too large and false matches may occur.

11. Pinhole Camera and Stereo

The pinhole camera flatten a **3D image** onto a **2D plane**. A small pinhole provides a sharp but dim image; a larger pinhole gives a brighter but blurred image.
The point on the image radiating light becomes a *patch* on the 2D plane. This is why **pinhole images are** by necessity **blurred**.

The origin of the World Co-ordinate System is at the optical centre, i.e. the pinhole.
The origin of the 2D image is at (0,0, –1) in world co-ordinates.
Image points are at $\begin{pmatrix} -x/z \\ -y/z \end{pmatrix}$
For stereo pinholes, we can think of the optical centres as being *behind* the 2D image plane. This allows us to think of the images as being the 'right way up'.
The **epipolar** line is the projection of the plane into either image.
We know that a common point on an object lies on the same plane common to the epipolar line ∴ we only have to search a line in the other image to find the point of correlation we are looking for (rather than an area).