NUI Galway
OÉ Gaillimh

Introduction – Representation of images and image types

EE551
Week 1 – Introduction to Image Processing

Ref. chapters 1 & 2, Gonzalez & Woods

2023

NUI Galway
OÉ Gaillimh

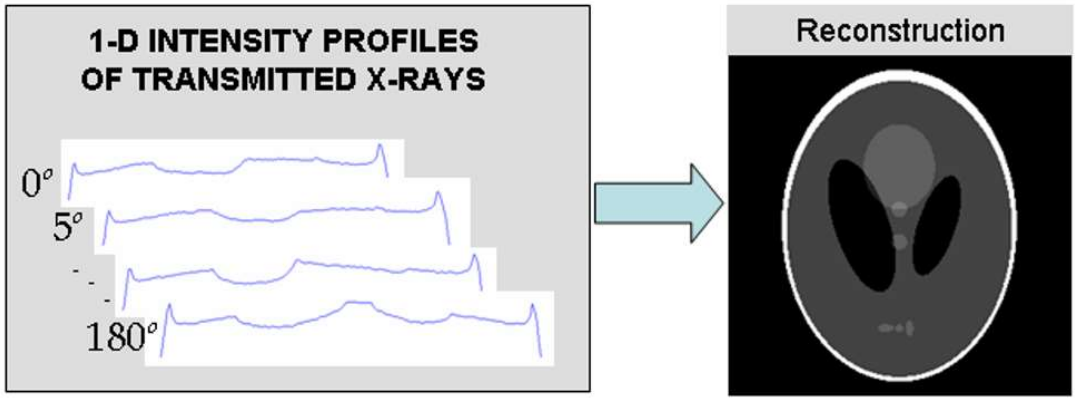
Image Processing

- A “catch-all” phrase that encompasses

Topic	Example
Reconstruction	(e.g. CAT scan)
Restoration	(e.g. Deblurring in astronomy)
Understanding/ interpretation	(e.g. Automated recognition)
Enhancement	(e.g. soft-focus effects, smoothing, sharpening)

Image Reconstruction

- Medical CAT (computer assisted tomography) scanners produce images of 2-D sections of the human body from a set of 1-D X-ray transmission measurements through the patient
- The basic 1-D projections thus obtained are not images in themselves, and it requires a relatively complex mathematical procedure to produce the 2-D anatomical sections.
- This is fundamentally an example of image *reconstruction*



NUI Galway
OÉ Gaillimh

Image Processing

- A digital colour image is captured of an old and slightly damaged but valuable painting
- Before manual restoration is attempted, we wish to consider how to achieve the best restoration of a computerised digital version
- As the input is an image, and the output is an image, this is basically an example of *processing*

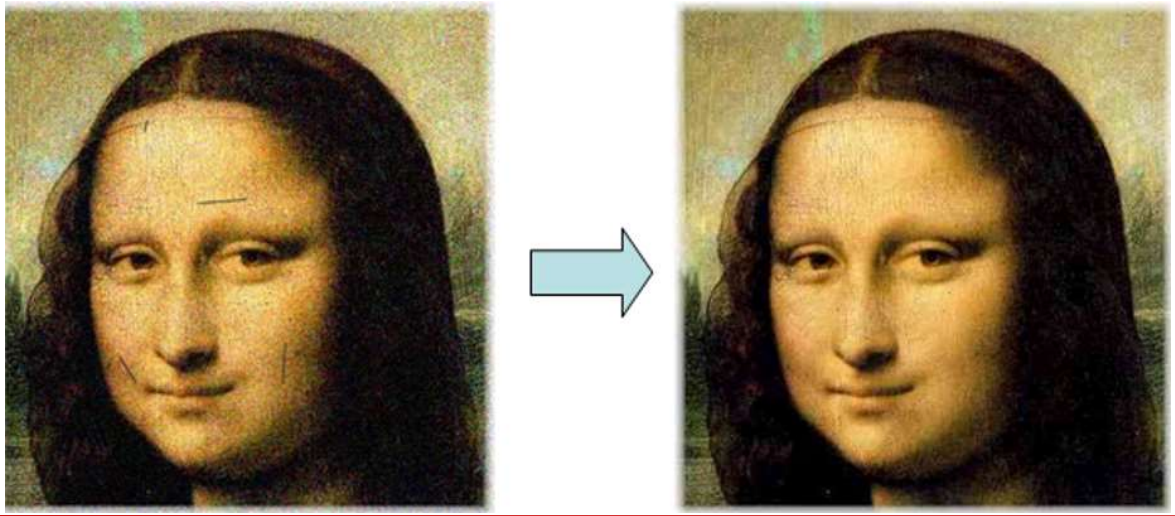


Image Analysis

- Automated face recognition systems attempt to analyse a human face and extract from it a set of parameters/measurements (a signature), which are compared against a database of similar signatures to effect a match
- This overall task is fundamentally one of *analysis* (though in practice, some preliminary processing will almost certainly be involved)

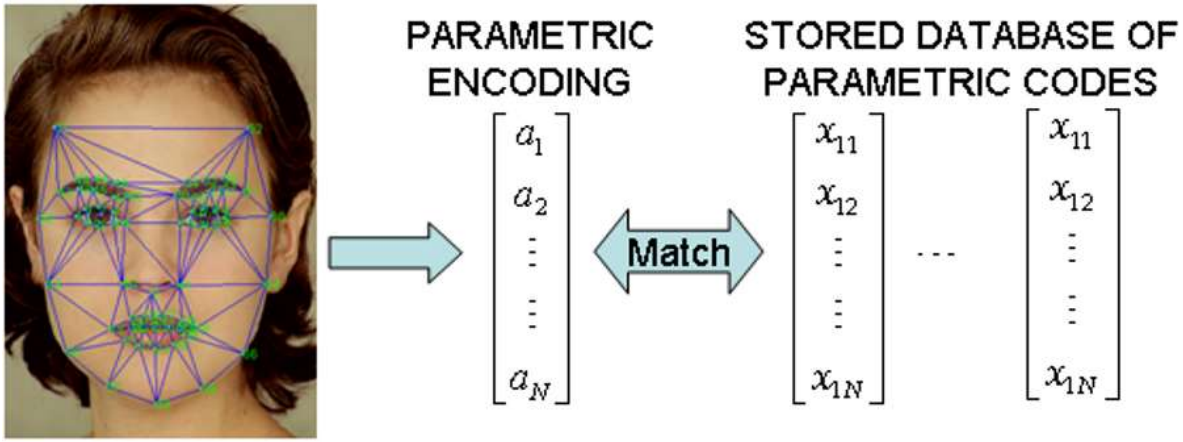
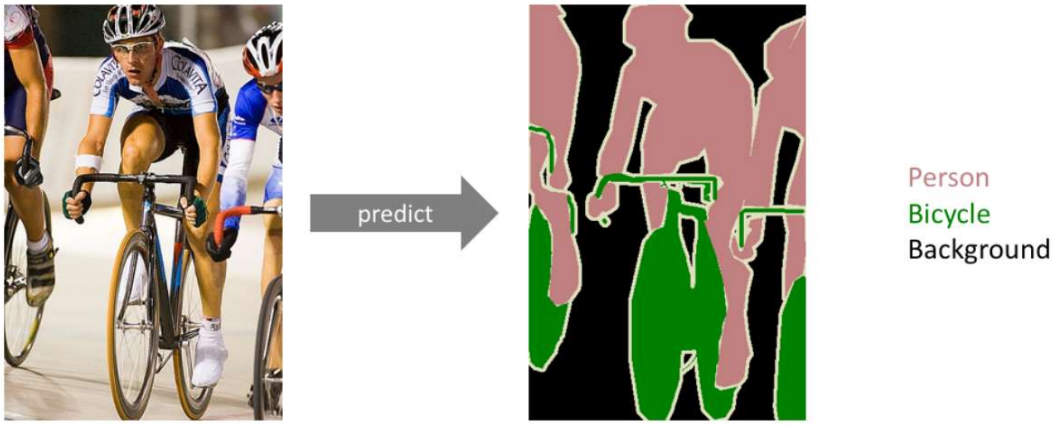


Image Understanding

- In certain instances, we may wish to understand the “basic nature” of an image. This can be useful for archiving, and also a preliminary step to decide whether further tasks of processing etc are required
- For example, does an image predominantly contain man-made structures or natural vegetation?
- In this conceptual example, a series of measurements (unspecified here) enable us to label the image as one containing *a person, a bicycle and the background*



Source: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/#devkit

Image Processing

- A “catch-all” phrase that encompasses

	Reconstruction	Analysis	Understanding	Processing
Input	Measurements / data	Image	Image	Image
Output	Image	Measurements / decisions	High Level Description	Image

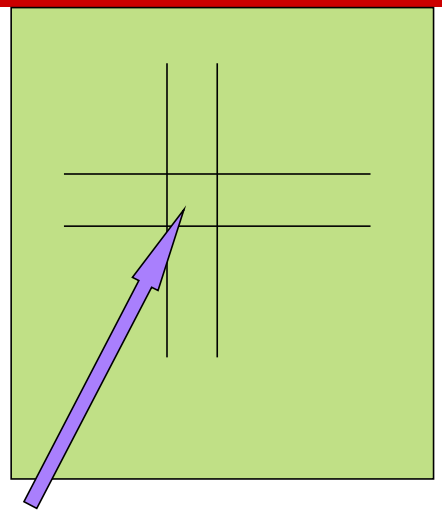
NUI Galway
OÉ Gaillimh

What is an image?

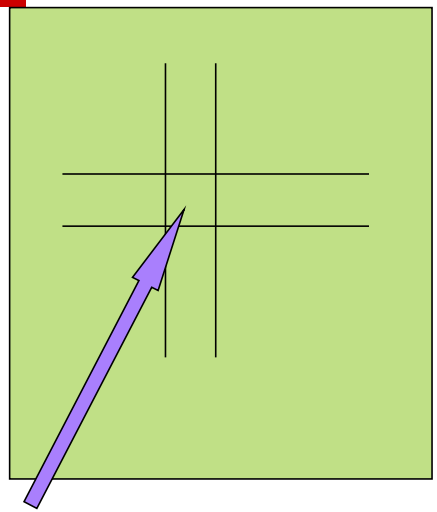
- We all have a fairly clear, intuitive idea of what an image is. Many formal definitions are possible. One definition is "the pictorial representation of information which is amenable to interpretation and analysis"
- In digital image processing, we are usually concerned with a 2-D array of pixels each of which has a numerical value (or values) associated with it.
- 2 most common examples
  - grey-scale (or intensity) images
  - True colour images.

NUI Galway
OÉ Gaillimh

Digital images – pixel description



Pixel (i, j)
i – denotes row or y index
j – denotes column or x index
1 value (e.g. 224) associated with each pixel

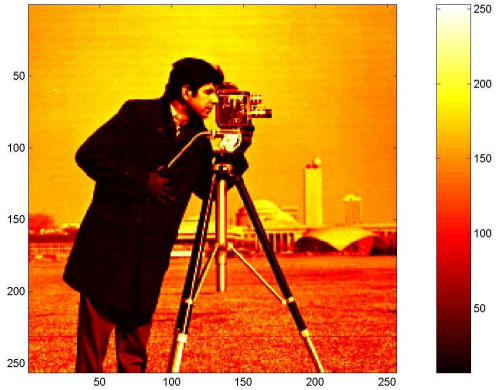


Pixel (i, j)
i – denotes row or y index
j – denotes column or x index
3 values associated with each pixel
(e.g. R=28, G=124, B=107)

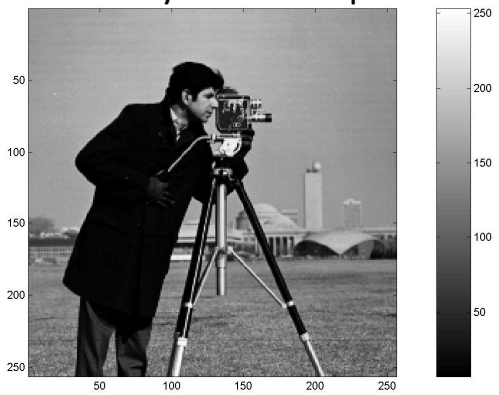
Colour maps

- An array of numbers is not in itself an “image”
- We must assign a shade of grey or a specific colour to each range of values
  - i.e. define a colourmap

Hot colour map



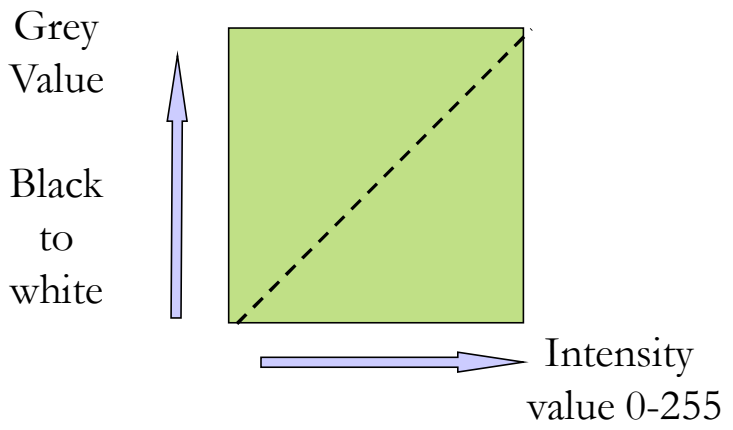
Grey colour map



Increasing Intensity

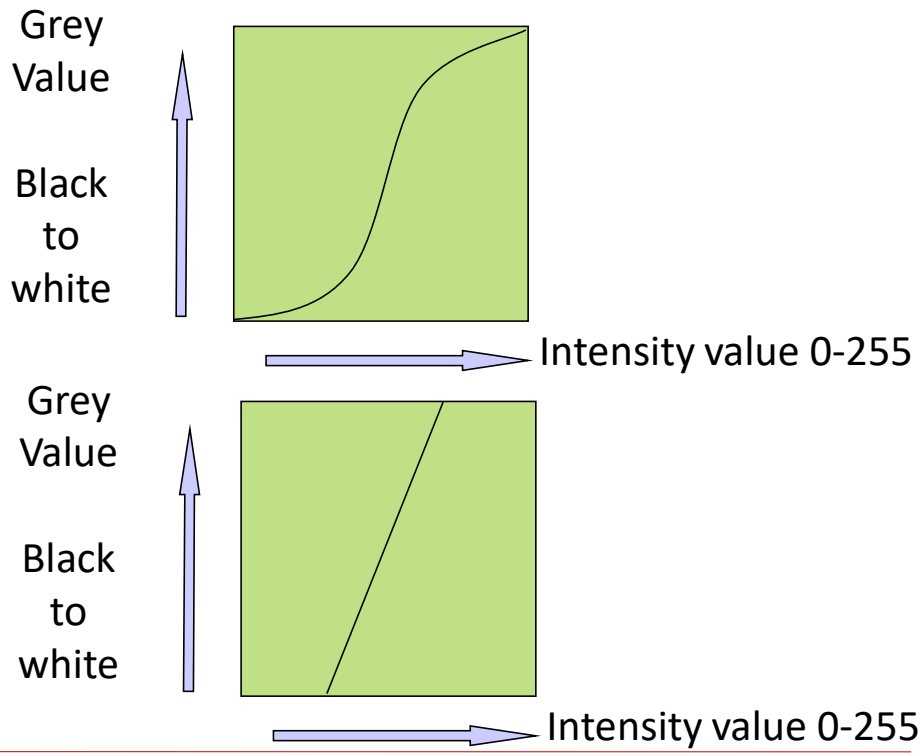
Greyscale – linear tone mapping

- The human visual system is only capable of distinguishing between about 30 shades of grey
- How we assign numerical values to shades of grey is an important issue. In general, it depends on what range of values in the data we are most interested in
- The simplest approach is to have a *linear mapping* between values and grey-scale
- Note: this mapping operation is typically called *tone mapping*



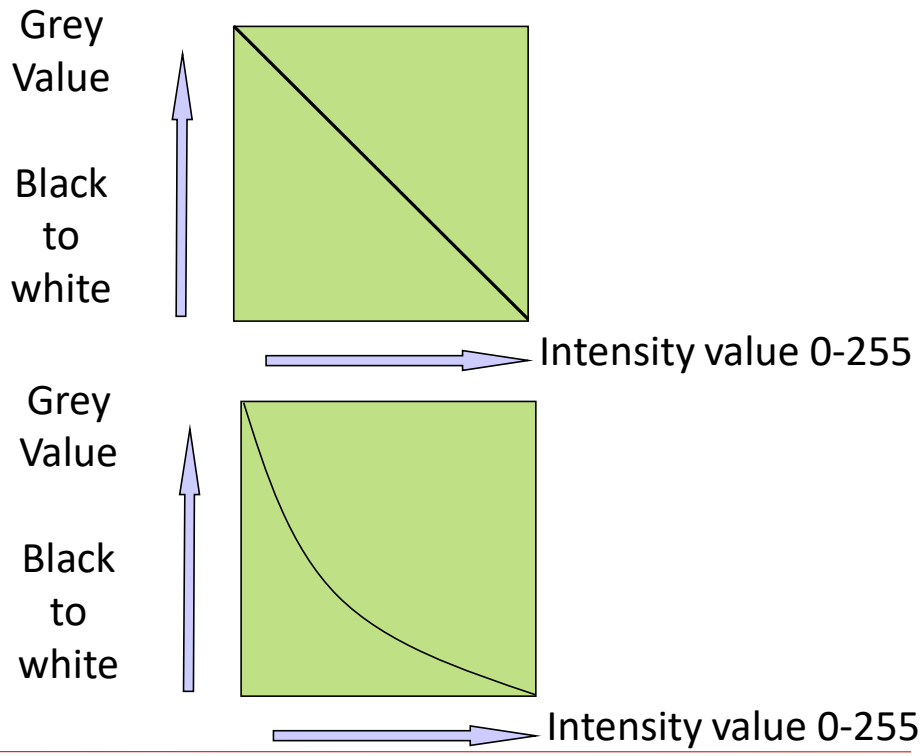
Greyscale – tone mapping

- Image contrast can be enhanced by applying an "*S-curve*", shown opposite
  - This type of contrast enhancement would typically be used in photography
- Alternatively, a *histogram stretch* operation can also be used to enhance contrast, by remapping a smaller range of input tones to a wider range of output tones



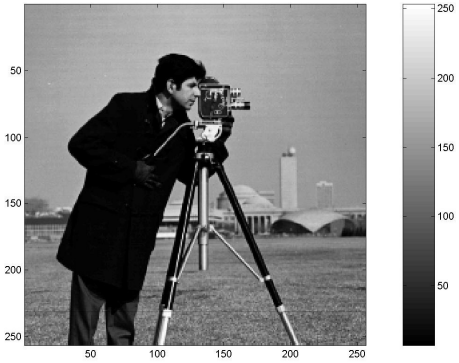
Greyscale – alternative tone mapping schemes

- It is also possible to invert tones, by using inverting tone mapping curves, as shown opposite
- Tone inversion is sometimes useful as a pre-processing step for detection algorithms



Tone mapping – example images

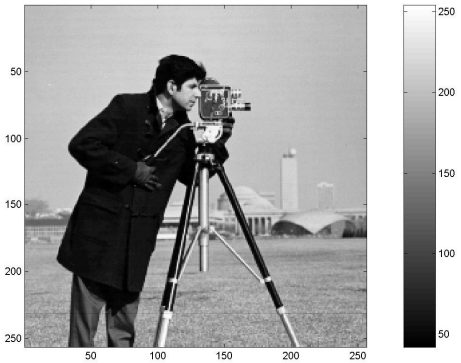
Linear



Tone inversion

Histogram stretch

Non-linear “gamma” = 0.5



Using colour maps

- We can display intensity images using colour maps if we wish (rather than the usual grayscale)
- Why should we wish to do this?
  - Because the human visual system can distinguish more distinct colours (approximately 250) than grey-scales.
  - Thus more detail is discernible and we can highlight differences existing in the data but not visible in a grey-scale representation.
- Python (e.g. Matplotlib) has many in-built colour maps.
  - E.g. hot, jet, cool, prism, H S V,....
  - Note: ensure you are using a perceptually uniform colour map!!!

Image Formats

- Manufacturers of imaging systems (both cameras and software) have defined many "standard" image types.
- Image quality and data storage requirements are, to some degree, conflicting requirements. Files have “headers” to identify them as images.
- Common image formats include the following-
  - TIF, GIF, BMP, JPEG, PIC,....
- Python supports most commonly used file formats
  - (i.e. can read and write all these image types)

Image Types

- Image type is something distinct from image format.
- Python supports/defines 4 image types –
  - Floating point arrays (not strictly images - no limits on minimum or maximum values).
  - Intensity images – values scaled to be between 0 and 1.
  - Binary images – pixel values are 0 OR 1 only.
  - Truecolor (RGB) images.
- Typically, intensity and RGB images may be of data type uint8. This assigns 8 bits to the values which range 0-255.
- Refer to example material…

Image formation and Basics of “Filtering”

- Consider some of the principles of image formation
- Not so concerned with the “physical” details of the sensors
- More concerned with the “signal processing” aspects
- Borrowing some concepts from linear filtering

Basic Model of Imaging System

- Imaging system (i.e. the camera) can be characterised by its Point Spread Function (PSF)
- Image = PSF * scene + Noise
- Noise is always present and we'll look at how to reduce it later in the module
- "*" is the *convolution* operator – describes what happens in the "spatial" domain i.e. operations on pixels
- Can also describe what happens in the *frequency* domain (later)
- The same basic principles apply not only in image formation (in a camera) but also in the "deliberate" image processing we will do on images

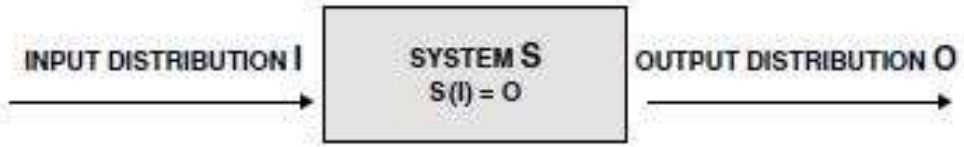


Figure 2.2 Systems approach to imaging. The imaging process is viewed as an operator *S* which acts on the input distribution *I* to produce the output *O*

Linearity

- Response of a linear system to a sum of inputs equals the sum of the individual responses, i.e.

$$S\{aX + bY\} = aS\{X\} + bS\{Y\}$$

- Linearity in an imaging system means that (in principle) all points in the input scene "contribute" to each point in the image
- In practice, this effect is limited to an "area" around each image point (some "neighbourhood")
- The overall effect at the output point is obtained by summing the individual contributions (linearity)

Filtering

- For image formation, we want the PSF of the imaging system to be as near to perfect as we can get
- Ideally, in image formation, each point in a scene would be “perfectly” transformed to the image
- This would require the PSF to be a “trivial” function (==1, or a “delta” function)
- In practice, this is not the case and an imperfect PSF causes some “smearing” of the output point because of the influence of adjacent points
- In other image processing applications, we deliberately design “filters” to have particular characteristics

Mechanics of Filtering (very brief introduction)

- The “filter” is described by its “kernel” (analogous to the impulse response)
- The kernel “slides” across the image, a pixel at a time
- At each location, the output is a weighted sum of the individual input pixels (linearity)

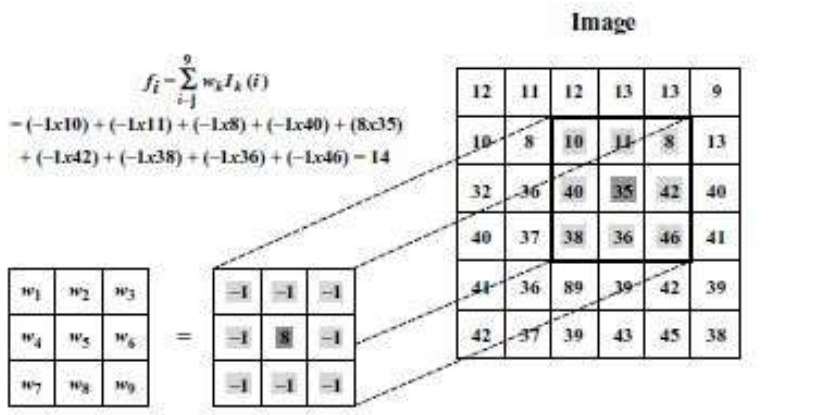


Figure 2.12 Discrete convolution. The centre pixel of the kernel and the target pixel in the image are indicated by the dark grey shading. The kernel is 'placed' on the image so that the centre and target pixels match. The filtered value of the target pixel is then given by a linear combination of the neighbourhood pixels, the specific weights being determined by the kernel values. In this specific case the target pixel of original value 35 has a filtered value of 14

Questions?