Digital Image Processing (CS/ECE 545) Lecture 1: Introduction to Image Processing and ImageJ

Prof Emmanuel Agu

Computer Science Dept.

Worcester Polytechnic Institute (WPI)

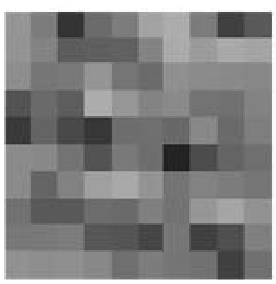
What is an Image?

- 2-dimensional matrix of Intensity (gray or color) values

Set of Intensity values

Image coordinates are integers

$$I(u,v) \in \mathbb{P} \quad \text{and} \quad u,v \in \mathbb{N}.$$



$F(x,y)$

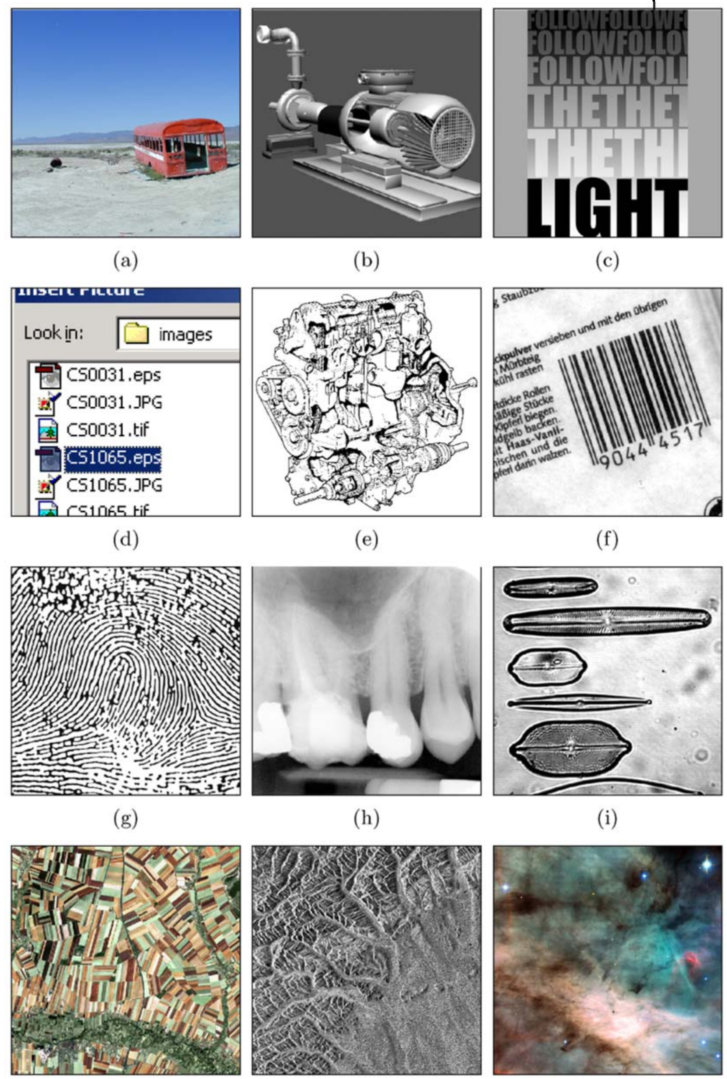
$\rightarrow$

148	123	52	107	123	162	172	123	64	89	$\cdots$
147	130	92	95	98	130	171	155	169	163	$\cdots$
141	118	121	148	117	107	144	137	136	134	$\cdots$
82	106	93	172	149	131	138	114	113	129	$\cdots$
57	101	72	54	109	111	104	135	106	125	$\cdots$
138	135	114	82	121	110	34	76	101	111	$\cdots$
138	102	128	159	168	147	116	129	124	117	$\cdots$
113	89	89	109	106	126	114	150	164	145	$\cdots$
120	121	123	87	85	70	119	64	79	127	$\cdots$
145	141	143	134	111	124	117	113	64	112	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	

$I(u,v)$

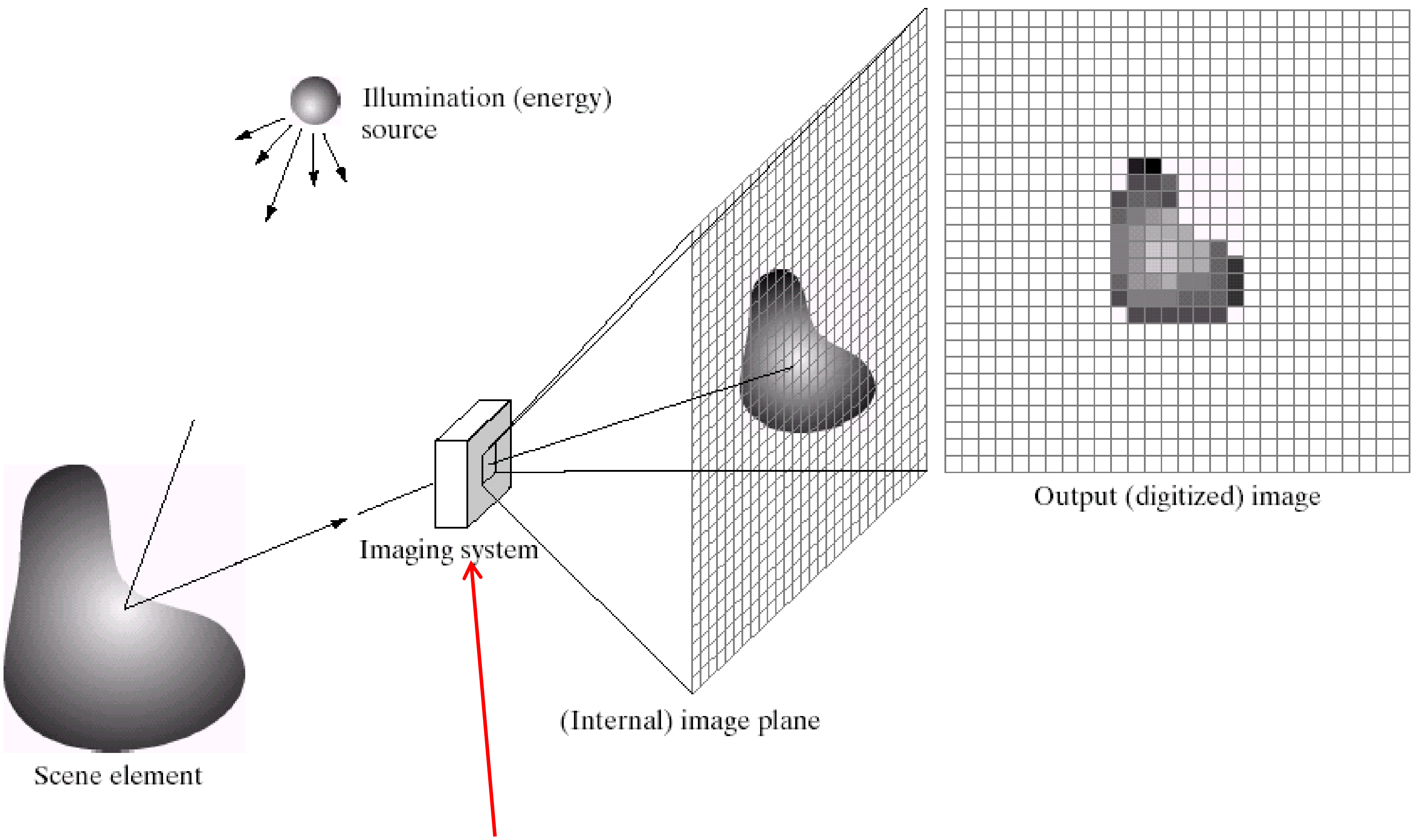
Example of Digital Images

a) Natural landscape
b) Synthetically generated scene
c) Poster graphic
d) Computer screenshot
e) Black and white illustration
f) Barcode
g) Fingerprint
h) X-ray
i) Microscope slide
j) Satellite Image
k) Radar image
l) Astronomical object



(a) (b) (c)
(d) (e) (f)
(g) (h) (i)
(j) (k) (l)

Imaging System

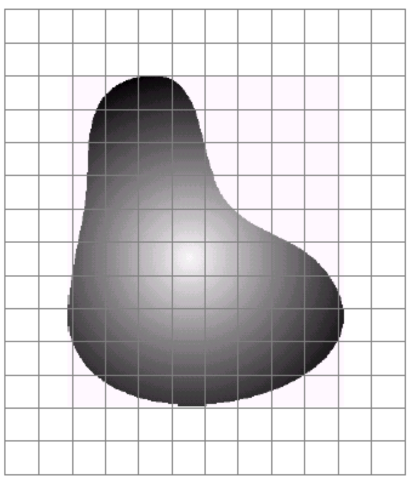


Example: a camera
Converts light to image

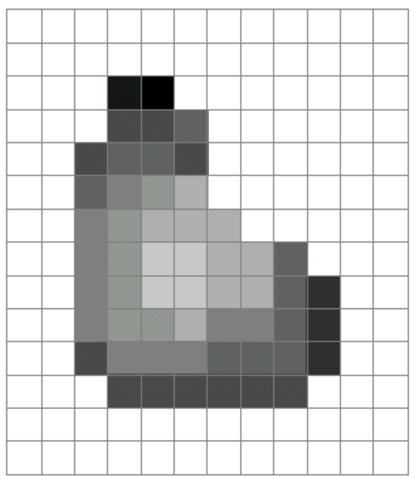
Credits: Gonzales and Woods

Digital Image?

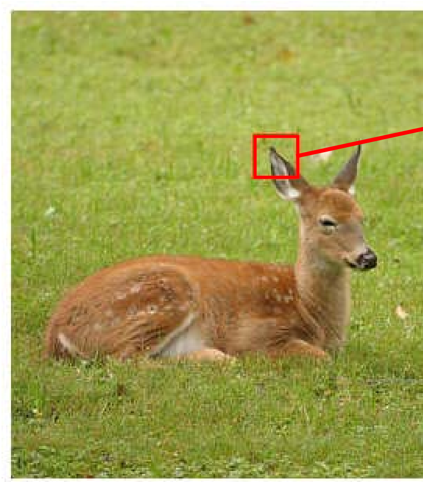
- **Remember:** *digitization* causes a digital image to become an *approximation* of a real scene



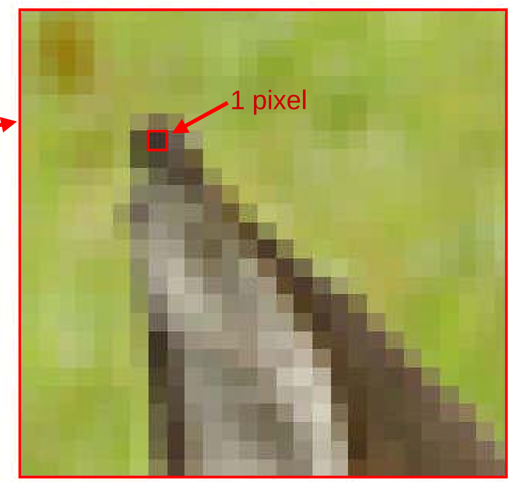
Real image



Digital Image
(an approximation)



Real image



Digital Image
(an approximation)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Digital Image

- Common image formats include:
  - 1 values per point/pixel (B&W or Grayscale)
  - 3 values per point/pixel (Red, Green, and Blue)
  - 4 values per point/pixel (Red, Green, Blue, + “Alpha” or Opacity)

Grayscale

RGB

RGBA

- We will start with gray-scale images, extend to color later

What is image Processing?

- **Algorithms that alter an input image to create new image**
- **Input is image, output is image**

Original Image

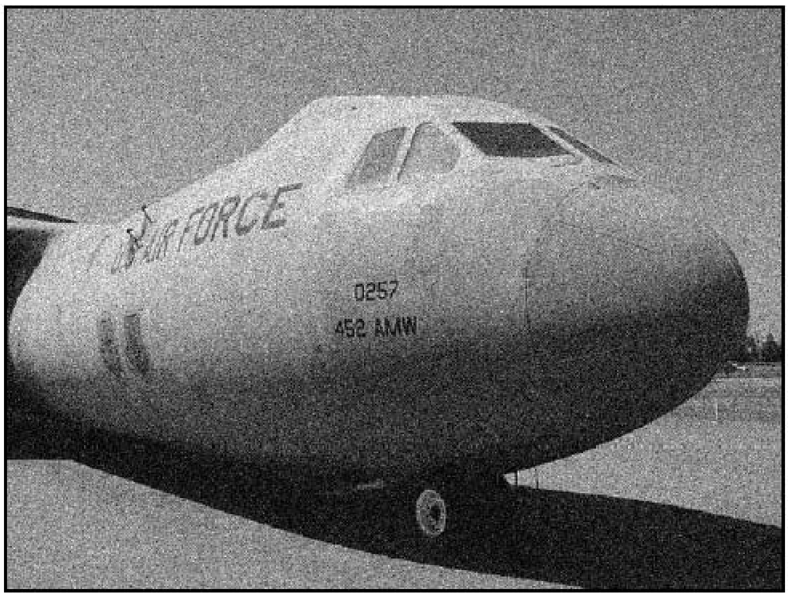
Image Processing Algorithm (e.g. Sobel Filter)

Processed Image

- **Improves an image for human interpretation in ways including:**
  - Image display and printing
  - Image editting
  - Image enhancement
  - Image compression

Example Operation: Noise Removal

Noisy Image

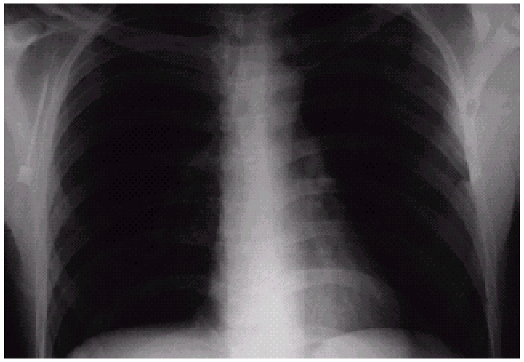


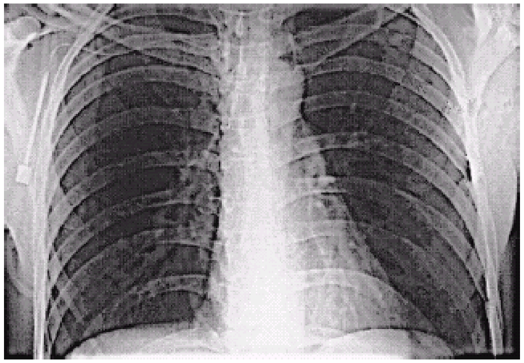
Denoised Image

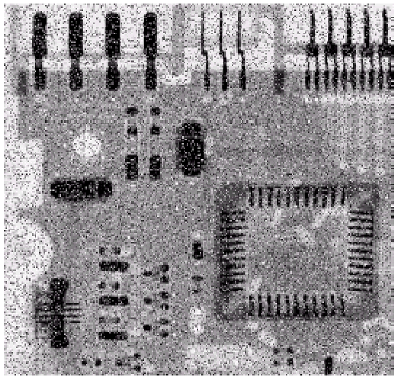
Think of noise as white specks on a picture (random or non-random)

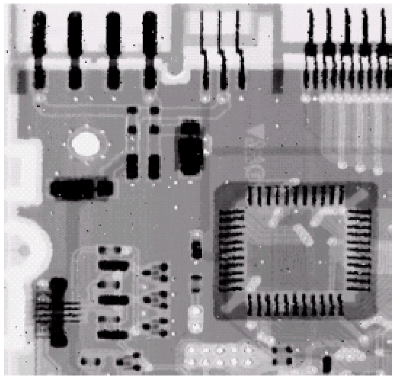
Examples: Noise Removal

Images taken from Gonzalez & Woods, Digital Image Processing (2002)









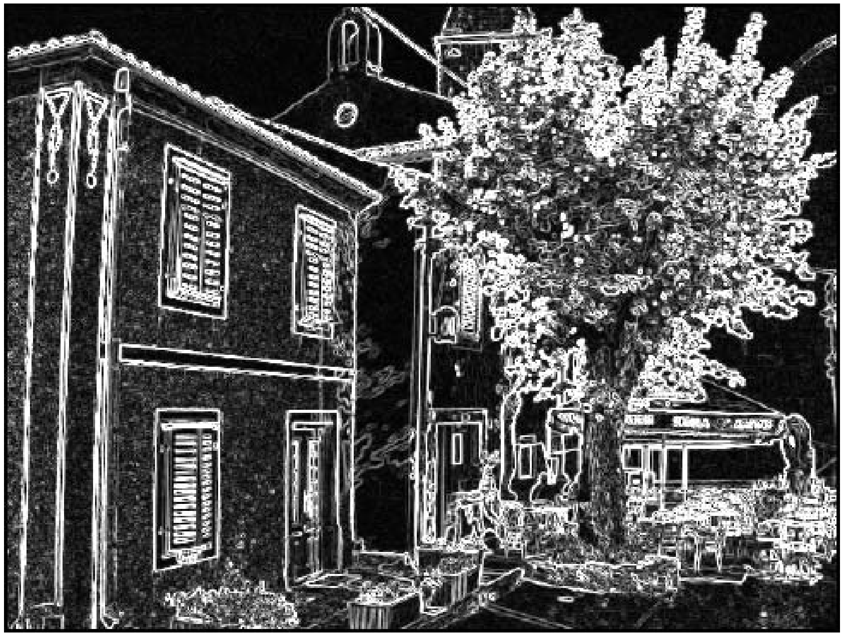
Example: Contrast Adjustment

Low Contrast

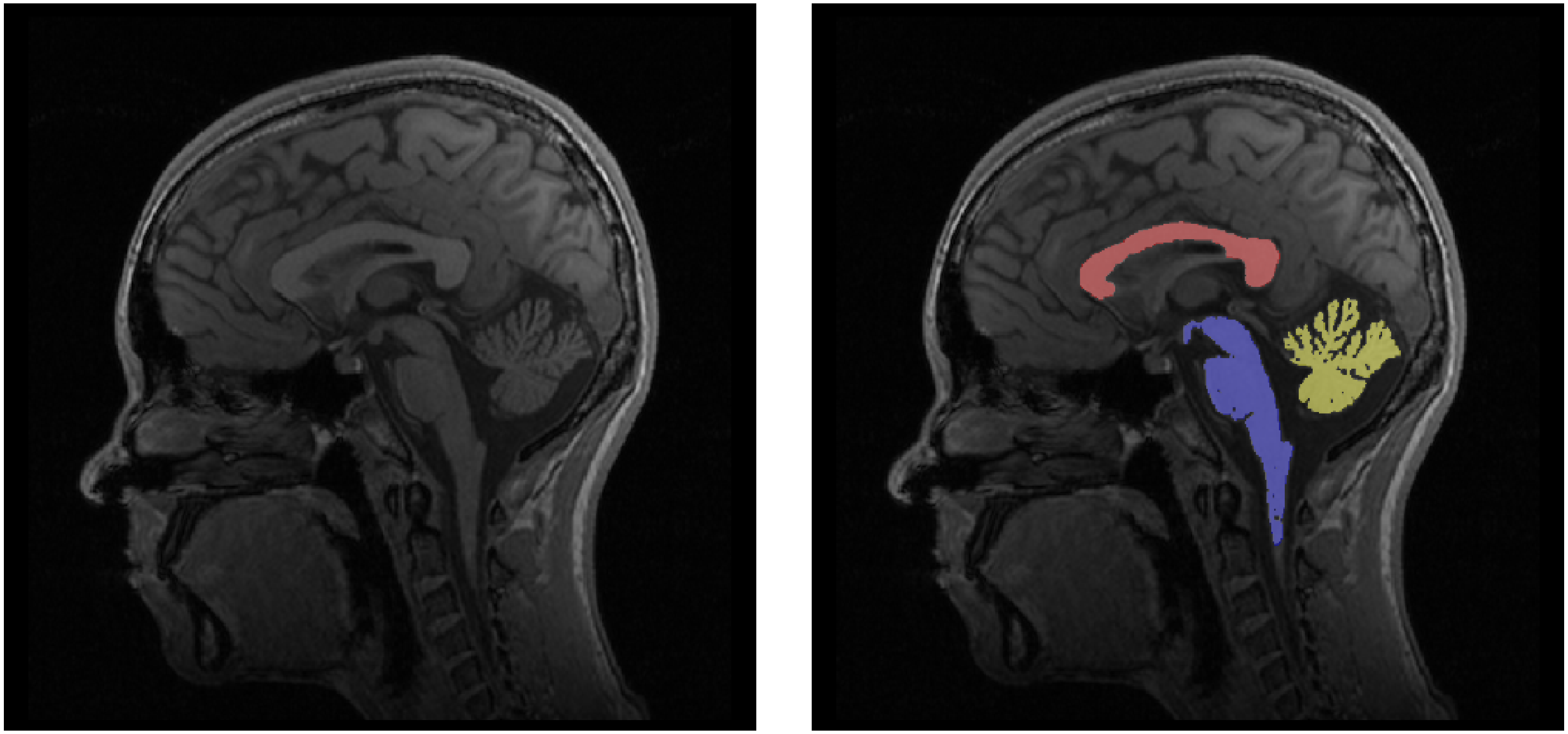
Original Contrast

High Contrast

Example: Edge Detection



Example: Region Detection, Segmentation



Example: Image Compression

Original, 2.1MB

JPEG Compression, 308KB (15%)

Example: Image Inpainting

Damaged Image

Restored Image

Credit: M. Bertalmio, G. Sapiro, V. Caselles, C. Ballester: Image Inpainting, SIGGRAPH 2000

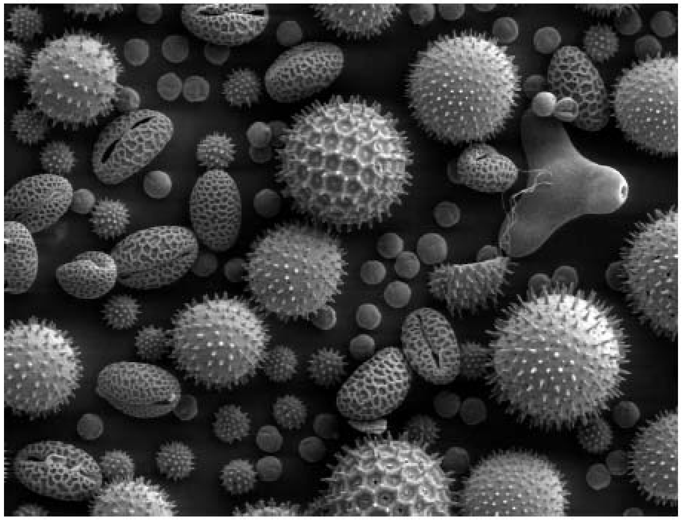
Inpainting? Reconstruct corrupted/destroyed parts of an image

Examples: Artistic (Movie Special)Effects

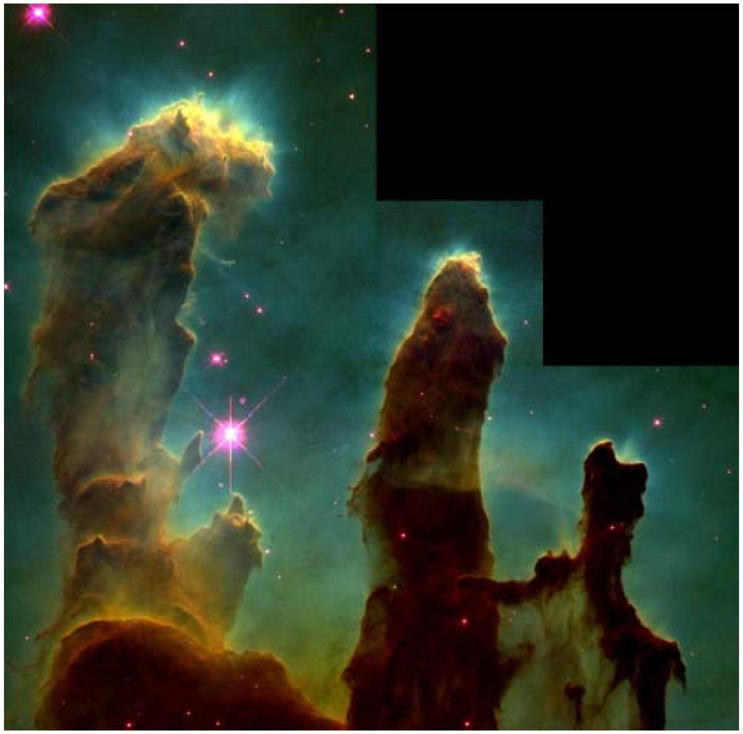
Applications of Image Processing

Astronomy

Biology



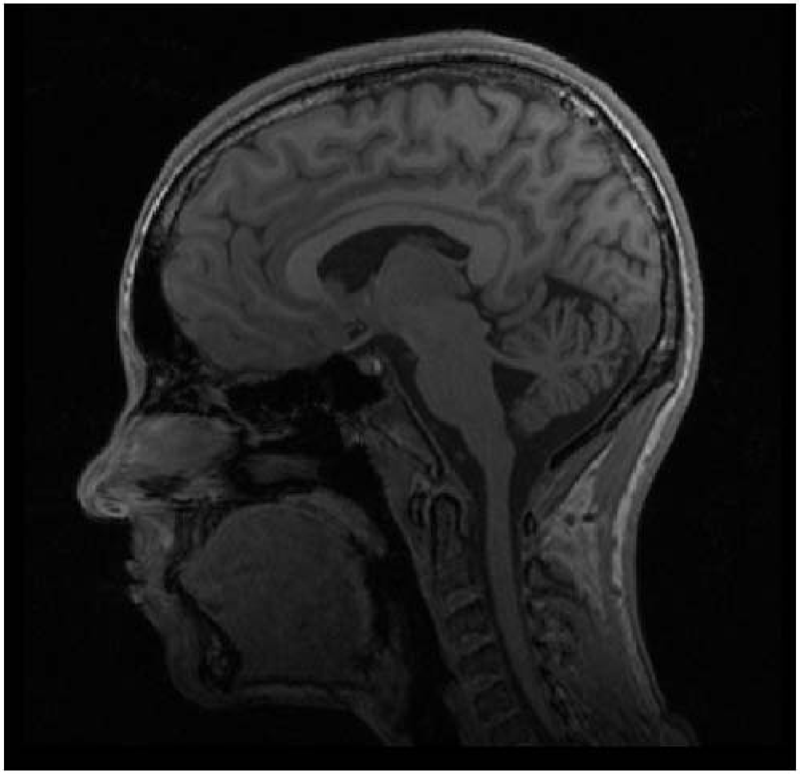
Credit: Dartmouth Electron Microscopy Facility



Credit: NASA, Jeff Hester, and Paul Scowen (Arizona State)
More info here

Applications of Image Processing

Medicine

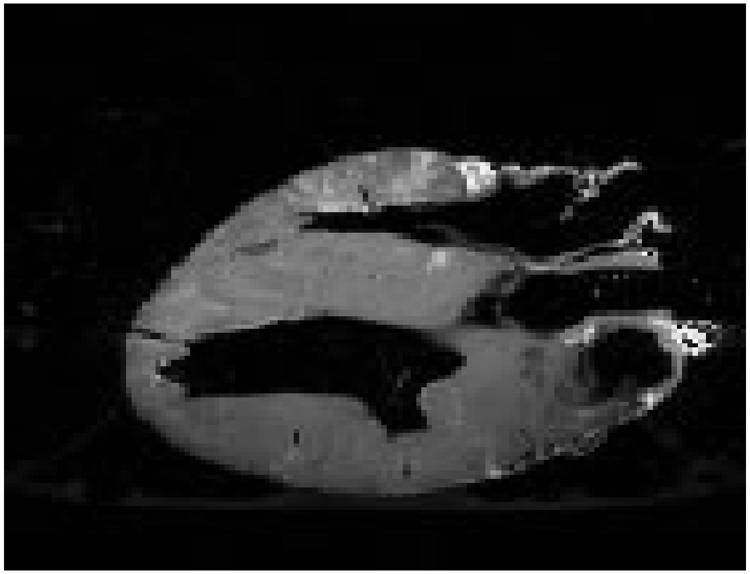


Credit: Dr. Janet Lainhart, UofU Psychiatry

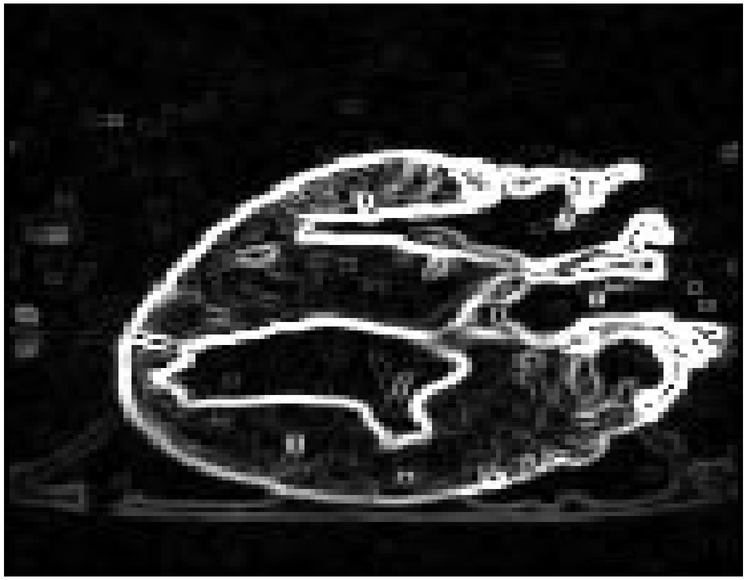
Security, Biometrics

Applications of Image Processing: Medicine

Images taken from Gonzalez & Woods, Digital Image Processing (2002)



Original MRI Image of a Dog Heart



Edge Detection Image

Applications of Image Processing

Satellite Imagery

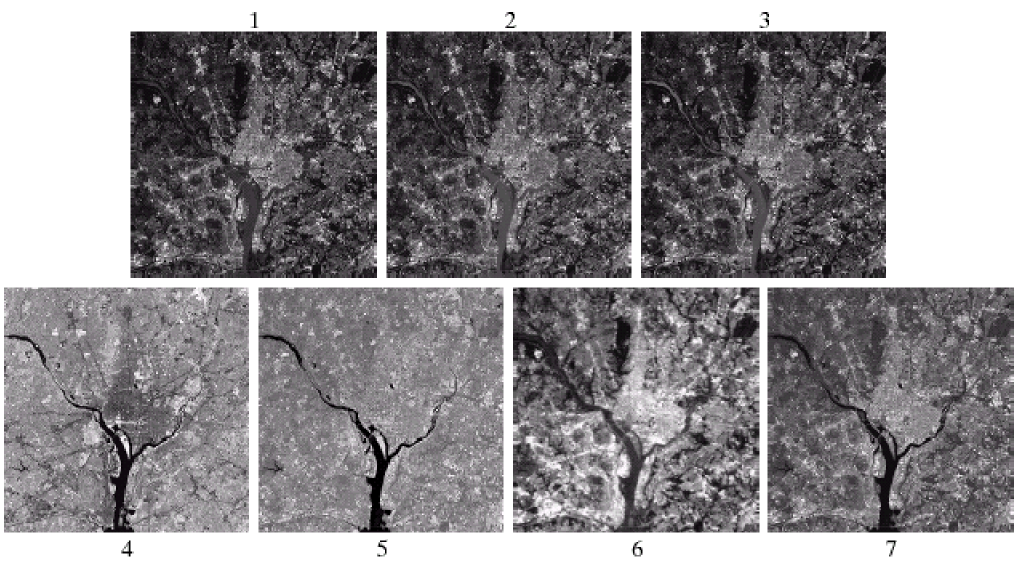
Credit: NASA

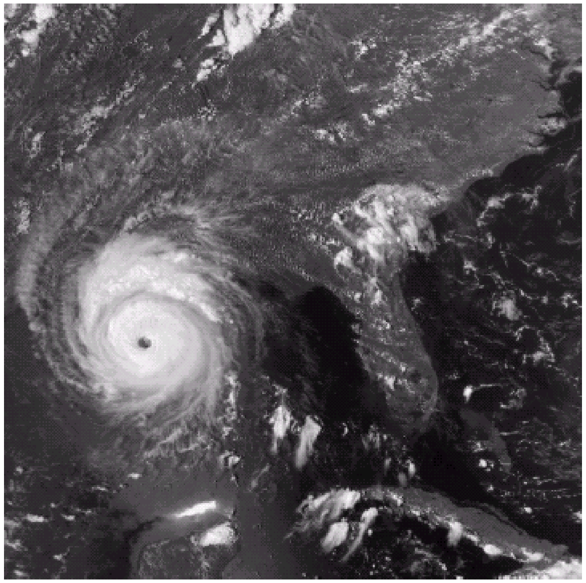
Personal Photos

Credit: Tom Fletcher

Applications of Image Processing: Geographic Information Systems (GIS)

- Terrain classification
- Meteorology (weather)

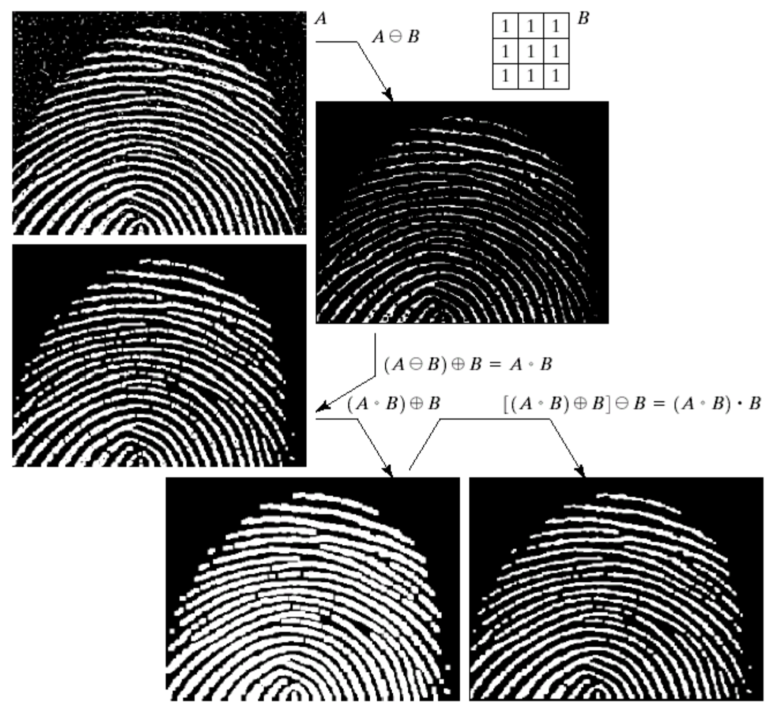




Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Applications of Image Processing: Law Enforcement

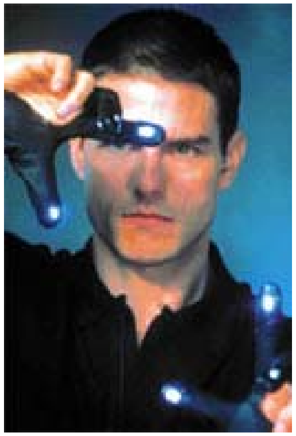
- Number plate recognition for speed cameras or automated toll systems
- Fingerprint recognition

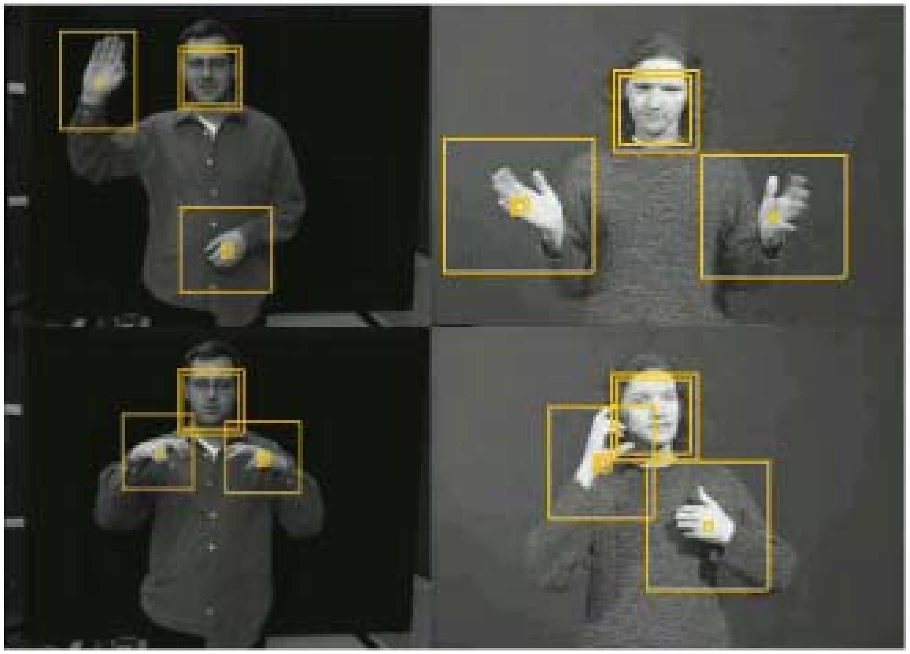


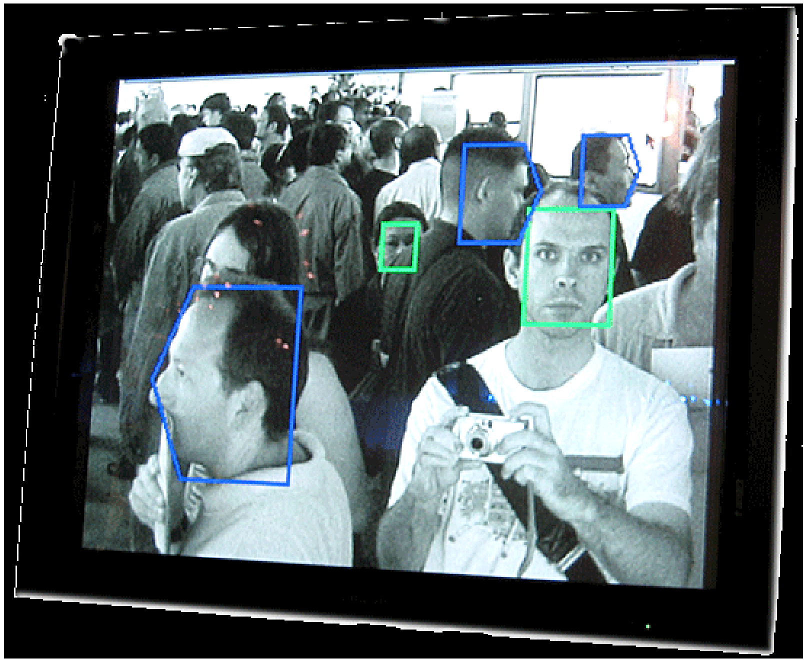
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Applications of Image Processing: HCI

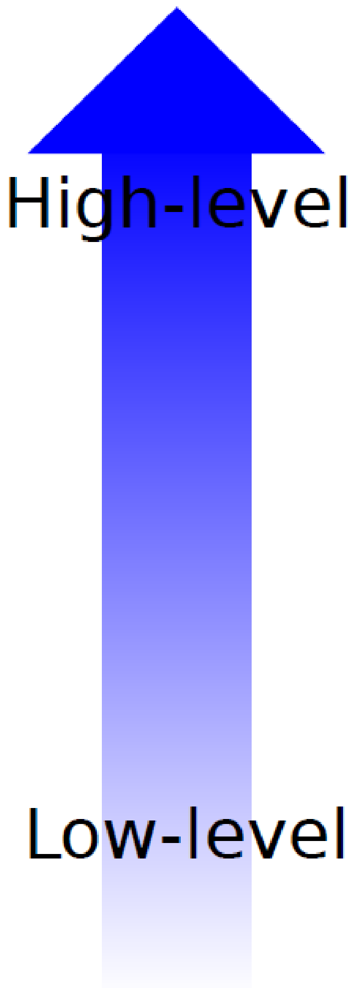
- Face recognition
- Gesture recognition







Relationship with other Fields



Computer Vision

Object detection, recognition, shape analysis, tracking
Use of Artificial Intelligence and Machine Learning

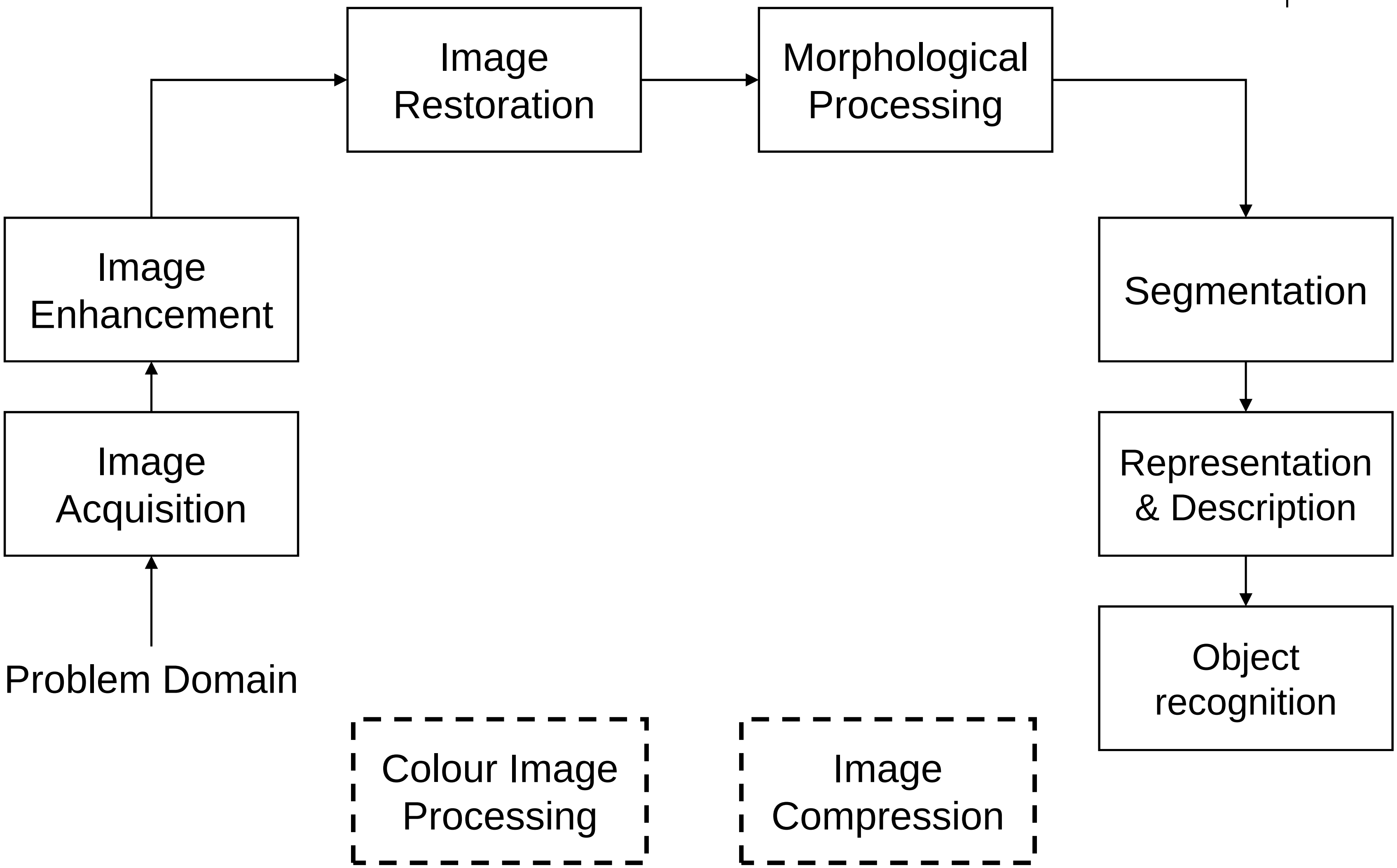
Image Analysis

Segmentation, image registration, matching

Image Processing

Image enhancement, noise removal, restoration,
feature detection, compression

Key Stages in Digital Image Processing



Key Stages in Digital Image Processing: Image Aquisition

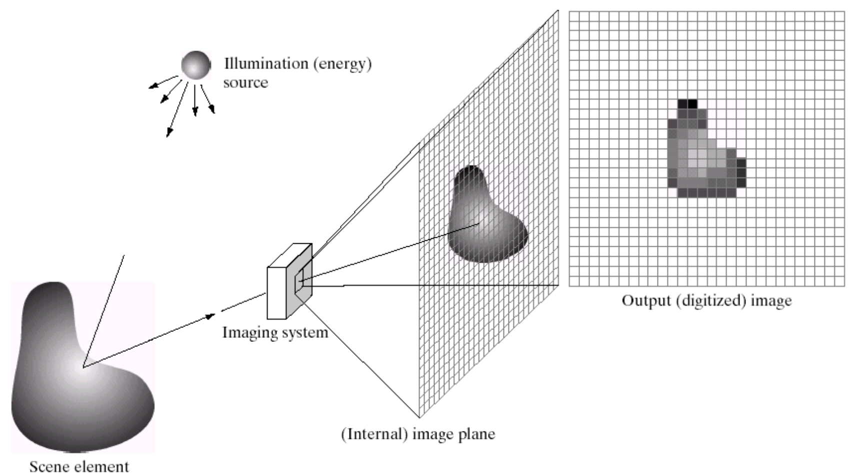
Image Restoration

Morphological Processing

Image Enhancement

Segmentation

Image Acquisition



Representation & Description

Example: Take a picture

Problem Domain

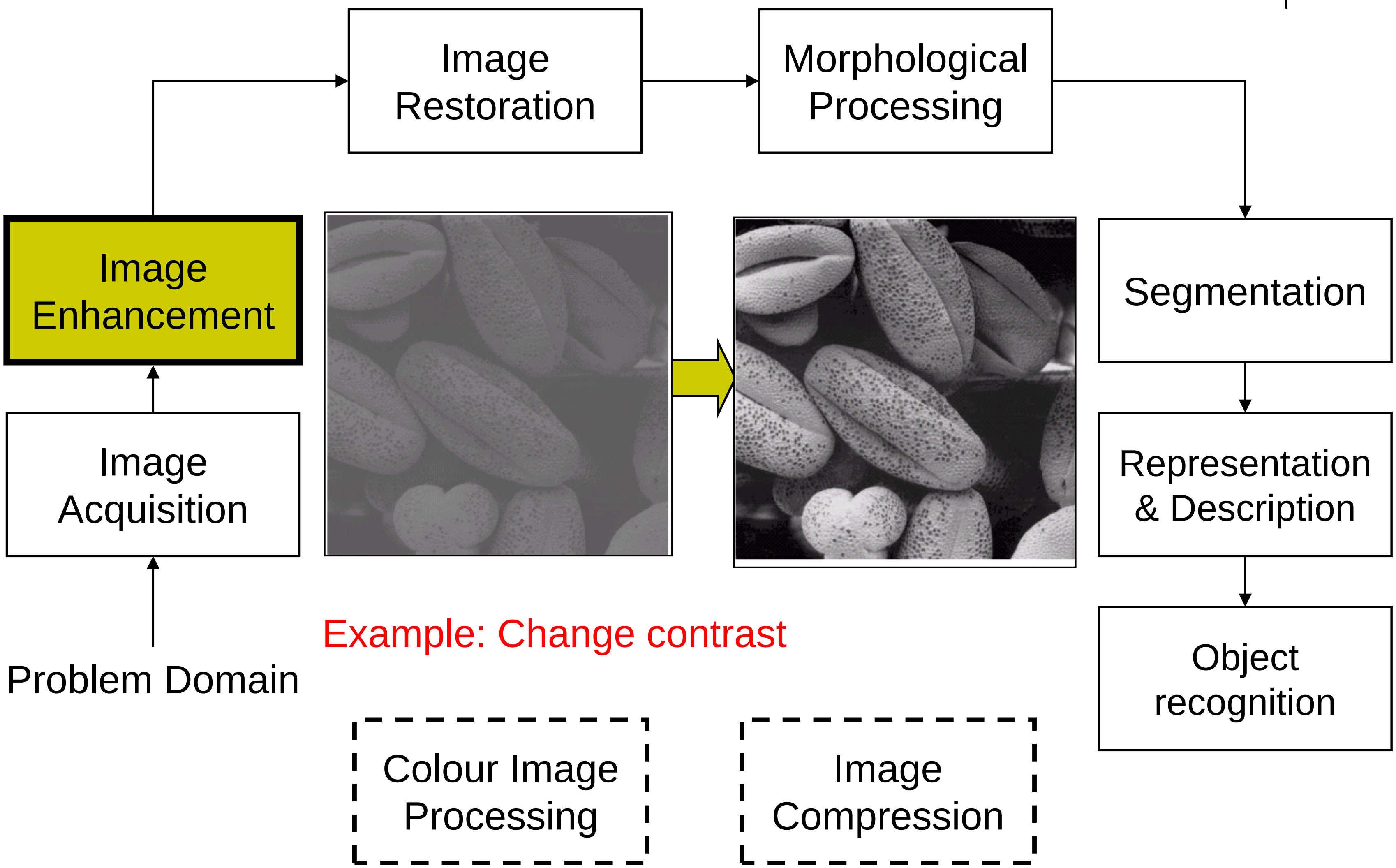
Object recognition

Colour Image Processing

Image Compression

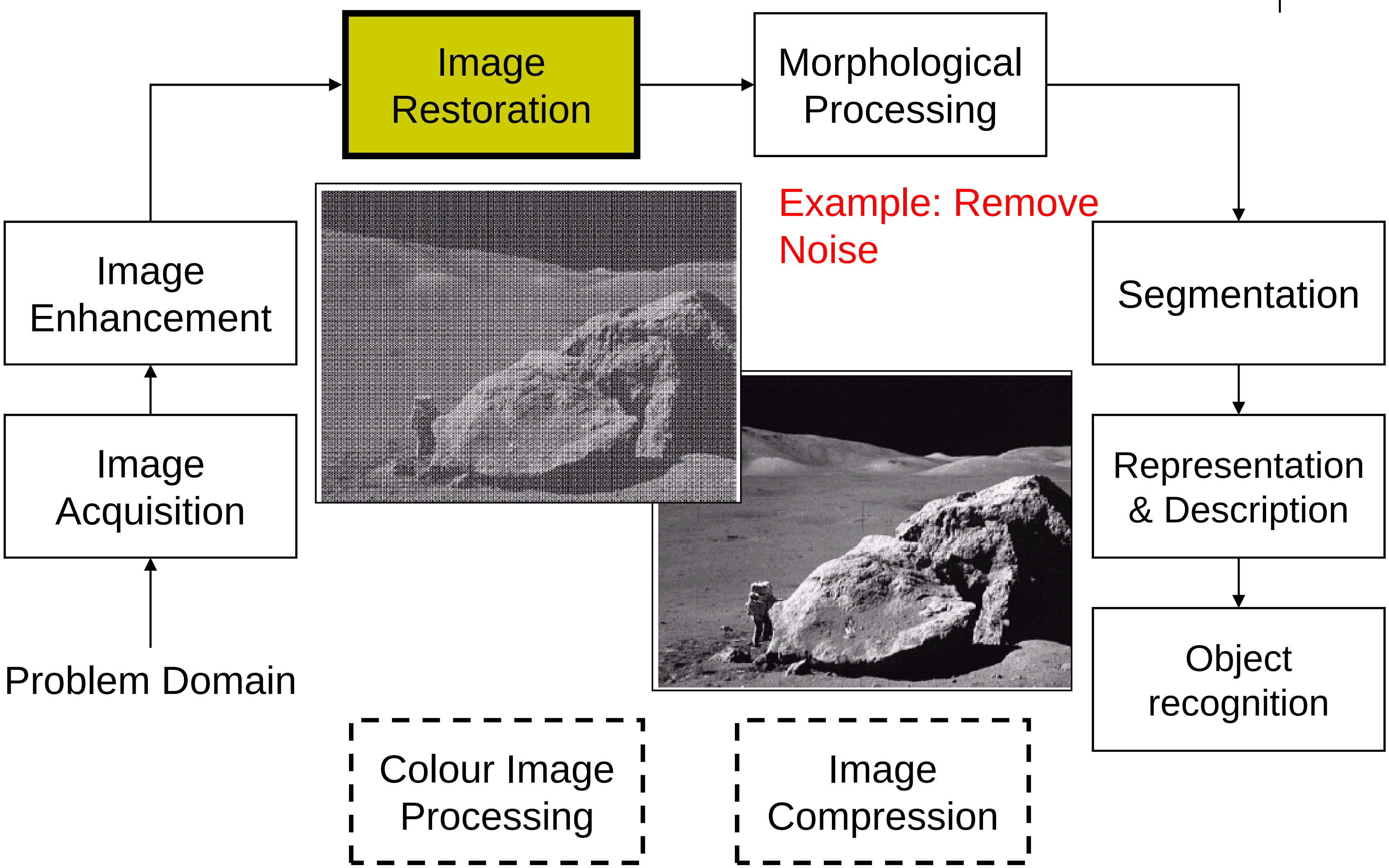
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Key Stages in Digital Image Processing: Image Enhancement



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Key Stages in Digital Image Processing: Image Restoration



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Key Stages in Digital Image Processing: Morphological Processing

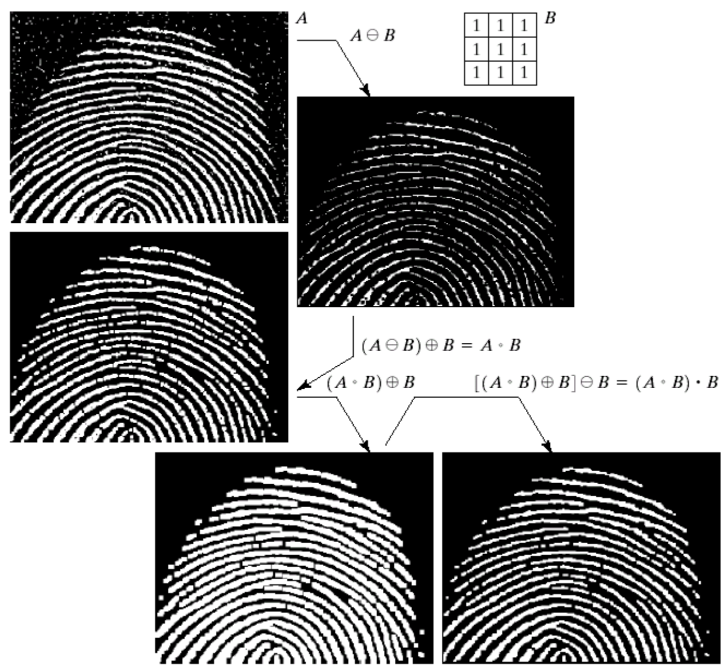
Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Restoration

Morphological Processing

Extract attributes useful for describing image

Image Enhancement



Segmentation

Image Acquisition

Representation & Description

Problem Domain

Object recognition

Colour Image Processing

Image Compression

Key Stages in Digital Image Processing: Segmentation

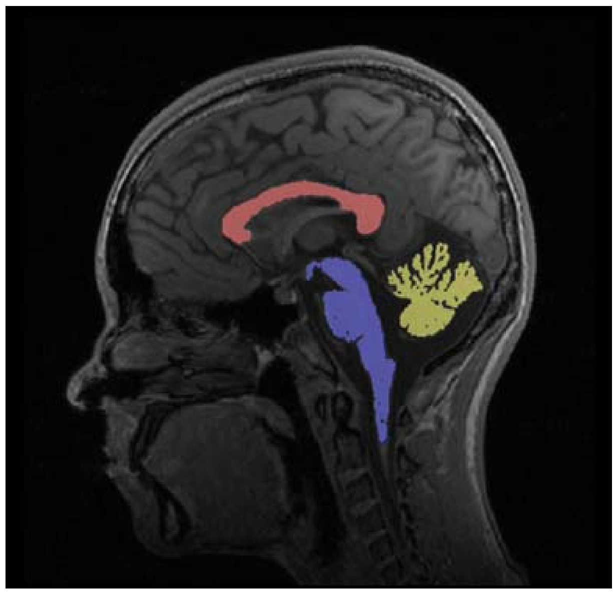
Image Acquisition → Image Enhancement → Image Restoration → Morphological Processing → Segmentation → Representation & Description → Object recognition

Problem Domain

Colour Image Processing

Image Compression

Divide image into constituent parts



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Key Stages in Digital Image Processing: Object Recognition

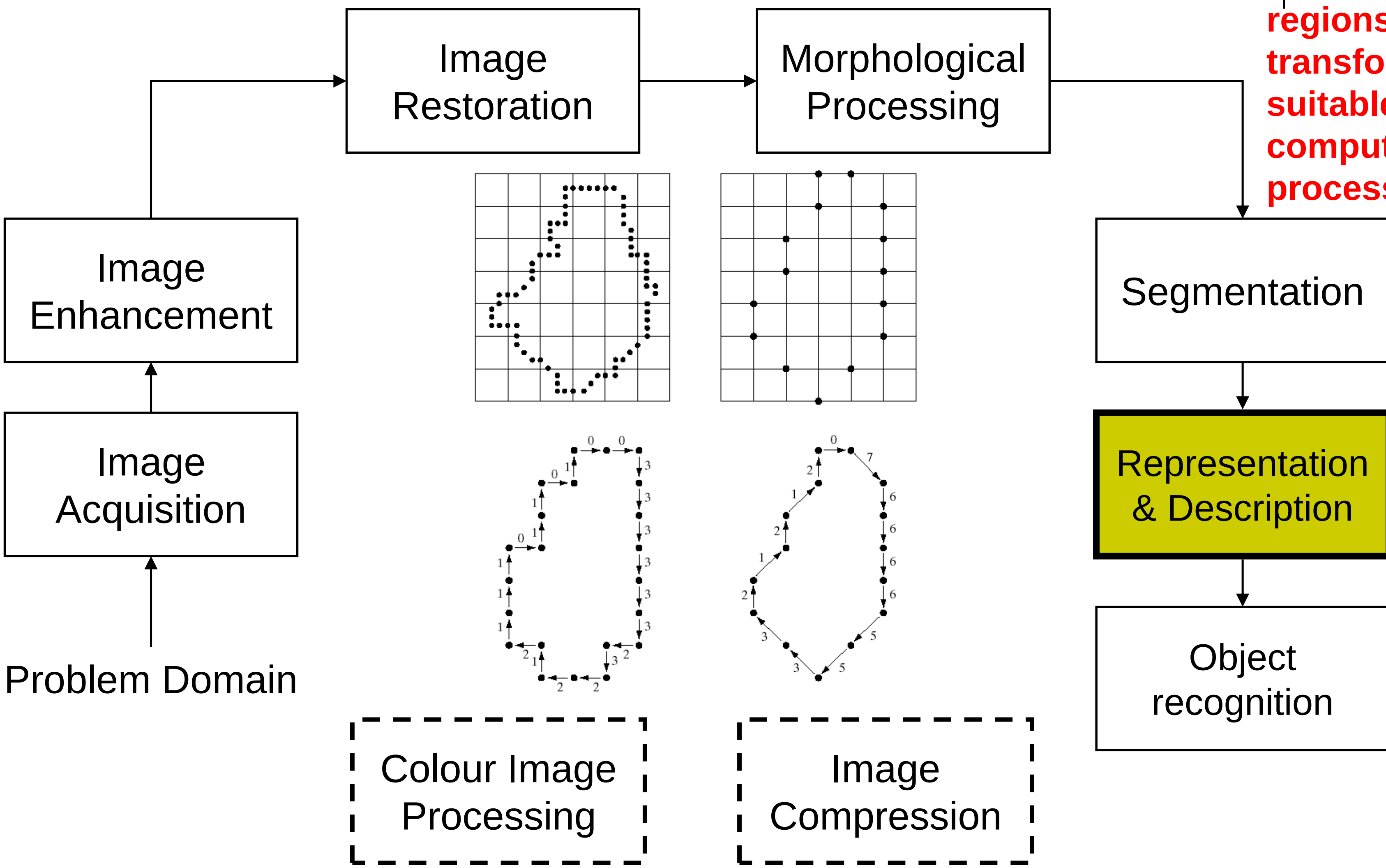
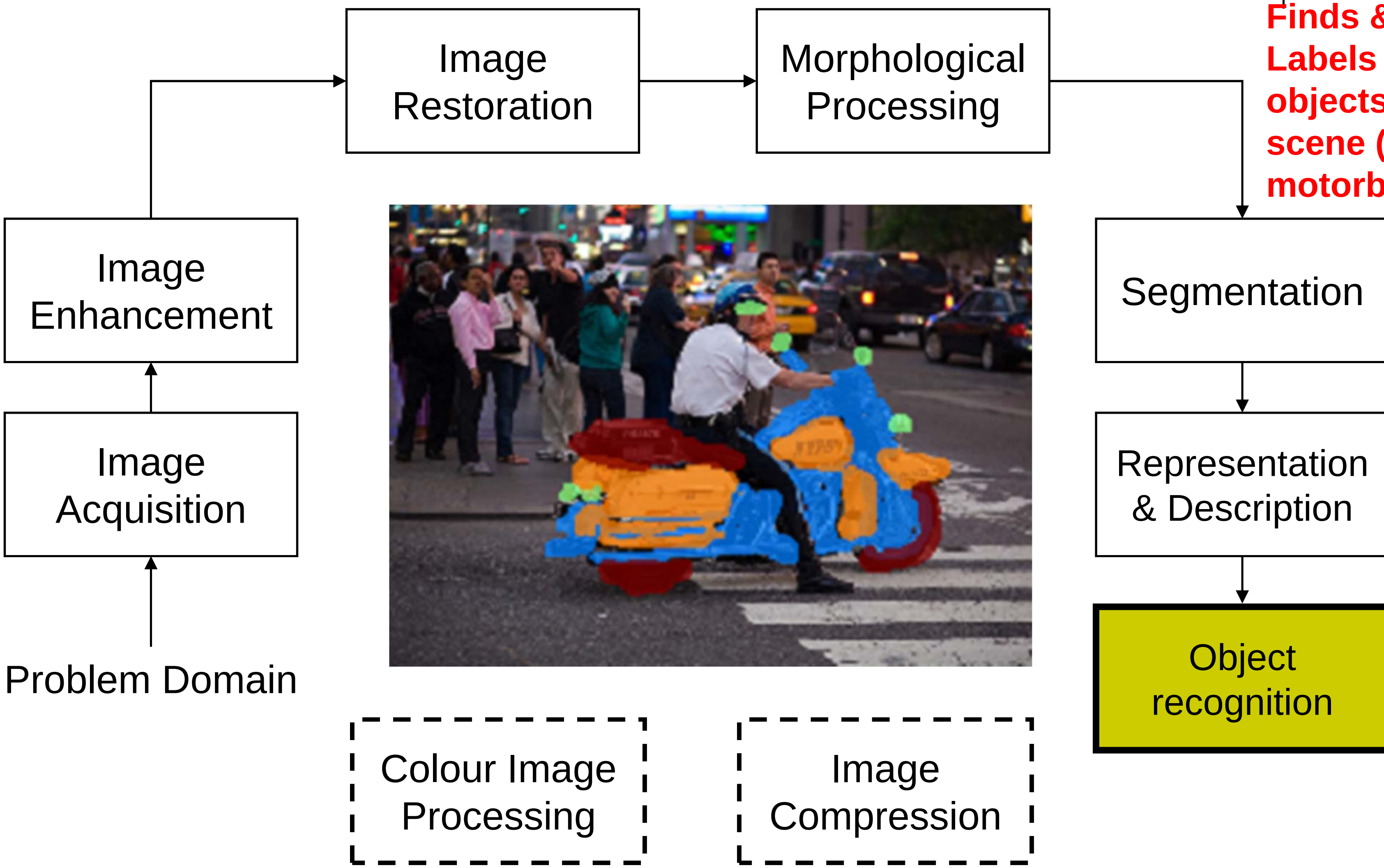


Image regions transformed suitable for computer processing

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Key Stages in Digital Image Processing: Representation & Description

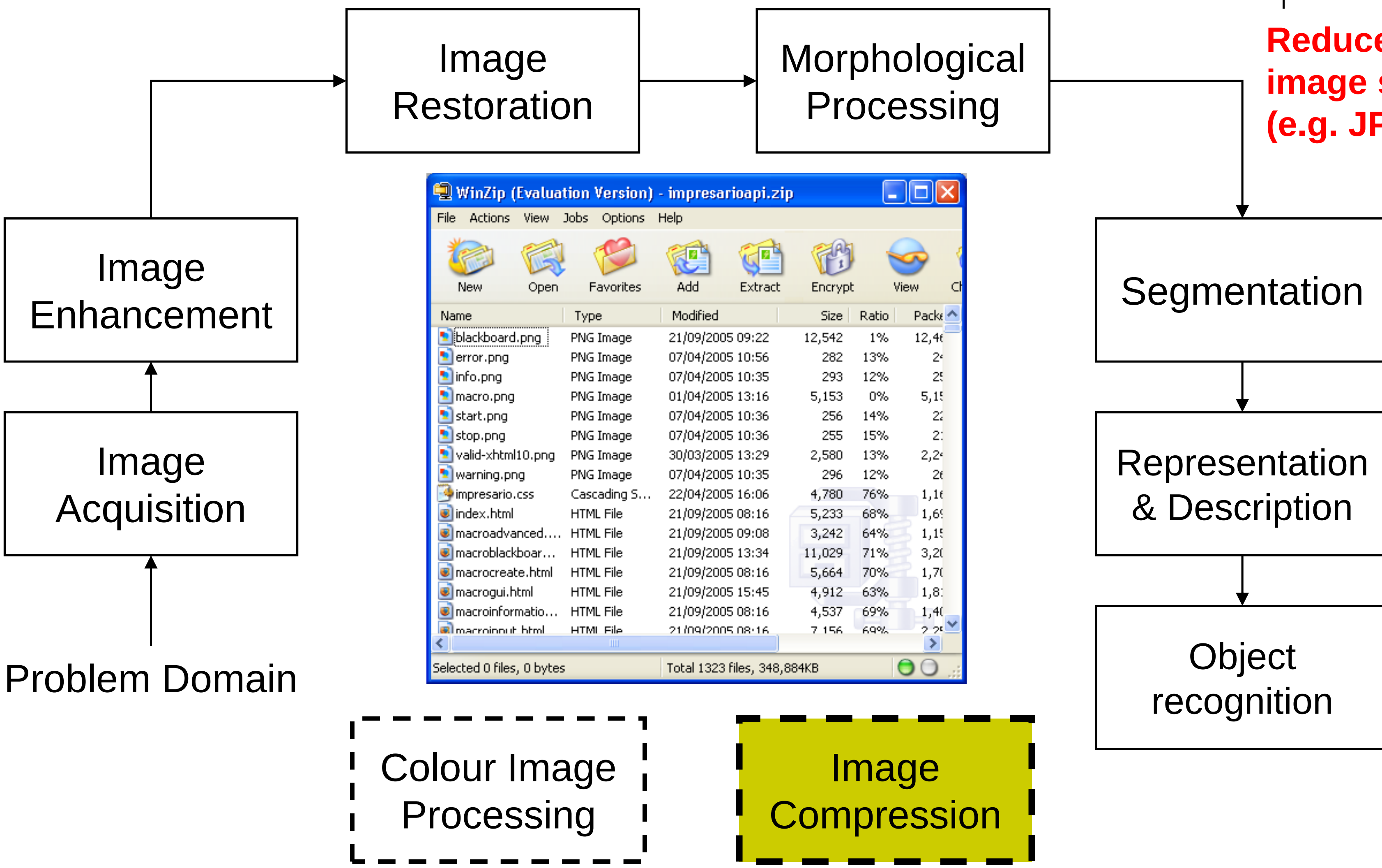


Finds & Labels objects in scene (e.g. motorbike)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Key Stages in Digital Image Processing: Image Compression

Reduce image size (e.g. JPEG)

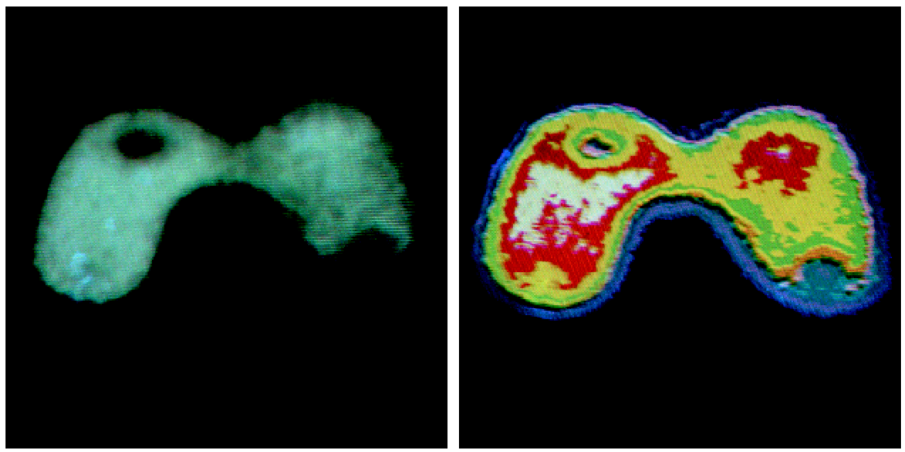


Key Stages in Digital Image Processing: Colour Image Processing

Image Restoration

Morphological Processing

Image Enhancement



Segmentation

Image Acquisition

Representation & Description

Object recognition

Problem Domain

Consider color images (color models, etc)

Colour Image Processing

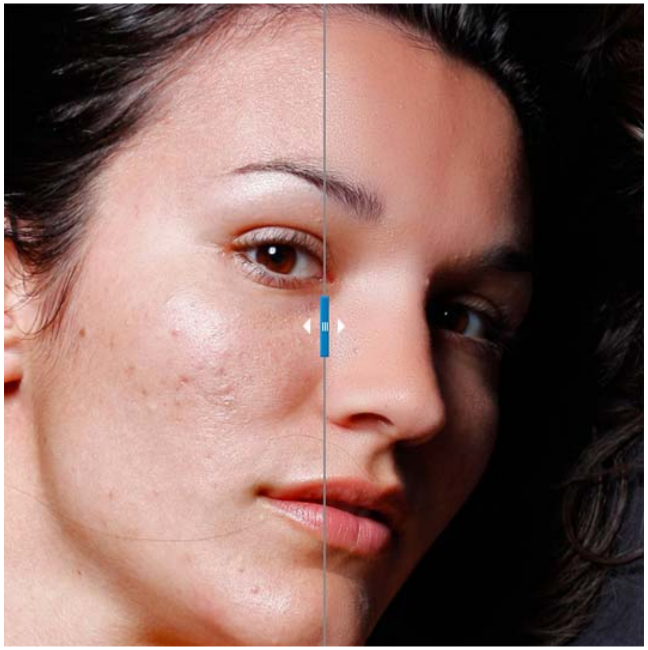
Image Compression

Mathematics for Image Processing

- Calculus
- Linear algebra
- Probability and statistics
- Differential Equations (PDEs and ODEs)
- Differential Geometry
- Harmonic Analysis (Fourier, wavelet, etc)

About This Course

- Image Processing has many aspects
  - **Computer Scientists/Engineers** develop tools (e.g. photoshop)
    - **Requires** knowledge of maths, algorithms, programming
  - **Artists** use image processing tools to modify pictures
    - **DOES NOT** require knowledge of maths, algorithms, programming



Example: Portraiture photoshop plugin

Example: Knoll Light Factory photoshop plugin

Example: ToonIt photoshop plugin

About This Course

- Most hobbyists follow artist path. Not much math!
- This Course: Image Processing for computer scientists and Engineers!!!
- Teaches concepts, uses ImageJ as concrete example
- ImageJ: Image processing library
  - Includes lots of already working algorithms,
  - Can be extended by programming new image processing techniques
- Course is **NOT**
  - just about programming ImageJ
  - a comprehensive course in ImageJ. (Only parts of ImageJ covered)
  - about using packages like Photoshop, GIMP

About This Course

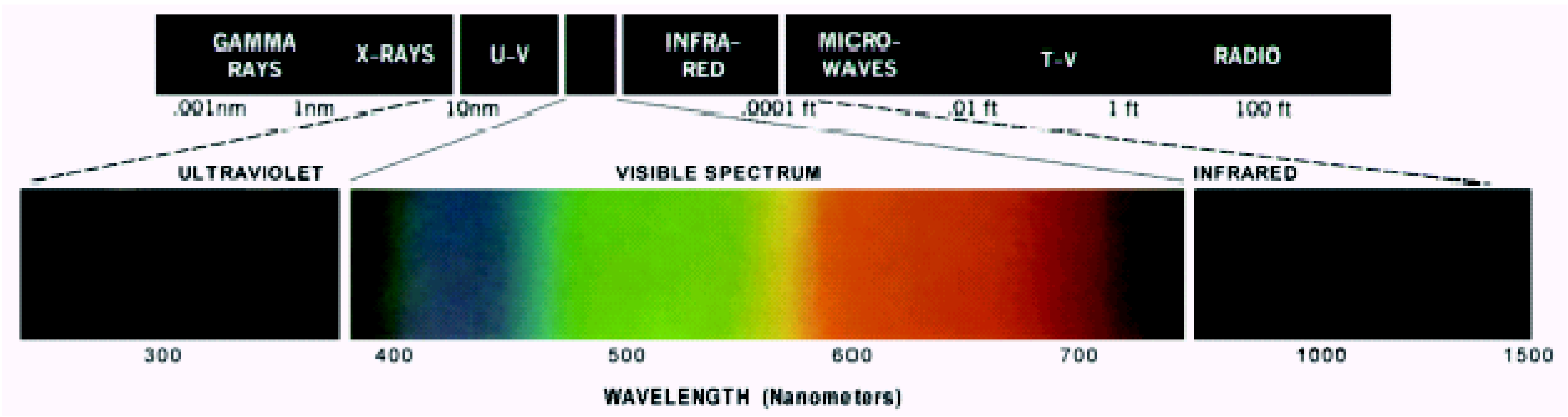
- Class is concerned with:
  - How to implement image processing algorithms
  - Underlying mathematics
  - Underlying algorithms
- This course is a lot of work. Requires:
  - Lots of programming in Java (maybe some MATLAB)
  - Lots of math, linear systems, fourier analysis

Administrivia: Syllabus Summary

- 2 Exams (50%), 5 Projects (50%)
- Projects:
  - Develop ImageJ Java code on any platform but must work in Zoolab machine
  - May discuss projects but turn in individual projects
- Class website: http://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/
- Text:
  - *Digital Image Processing: An Algorithmic Introduction using Java* by Wilhelm Burger and Mark J. Burge, Springer Verlag, 2008
- Cheating: Immediate 'F' in the course
- My advice:
  - Come to class
  - Read the text
  - Understand concepts before coding

Light And The Electromagnetic Spectrum

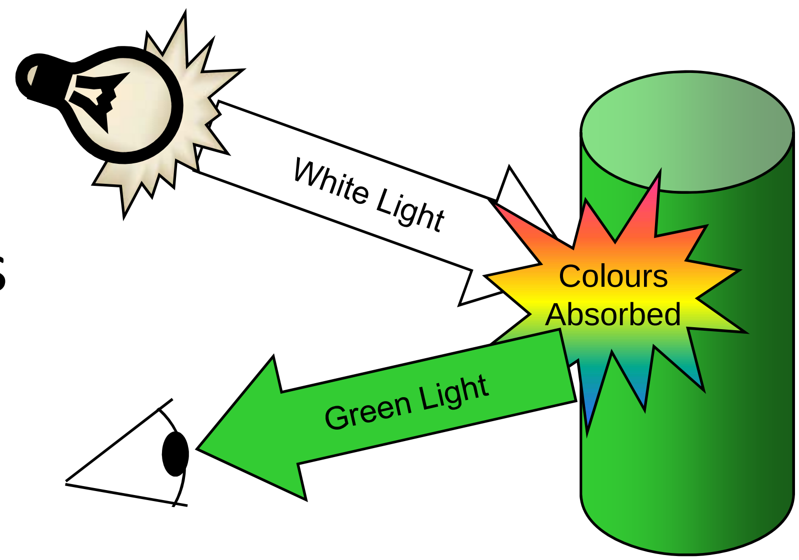
- Light: just a particular part of electromagnetic spectrum that can be sensed by the human eye
- The electromagnetic spectrum is split up according to the wavelengths of different forms of energy



Reflected Light

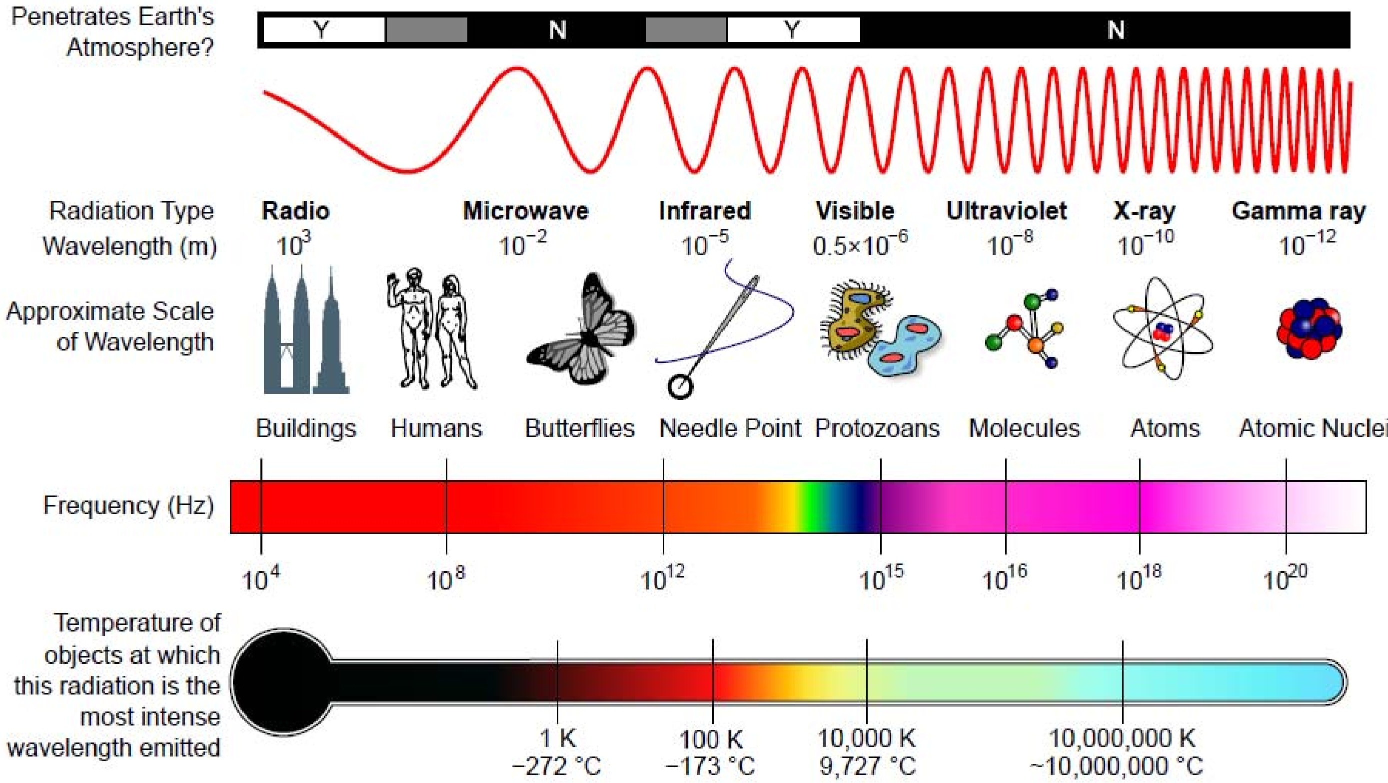
- The colours humans perceive are determined by nature of light reflected from an object

- For example, if white light (contains all wavelengths) is shone onto green object it absorbs most wavelengths absorbed except green wavelength (color)



Electromagnetic Spectrum and IP

- Images can be made from any form of EM radiation



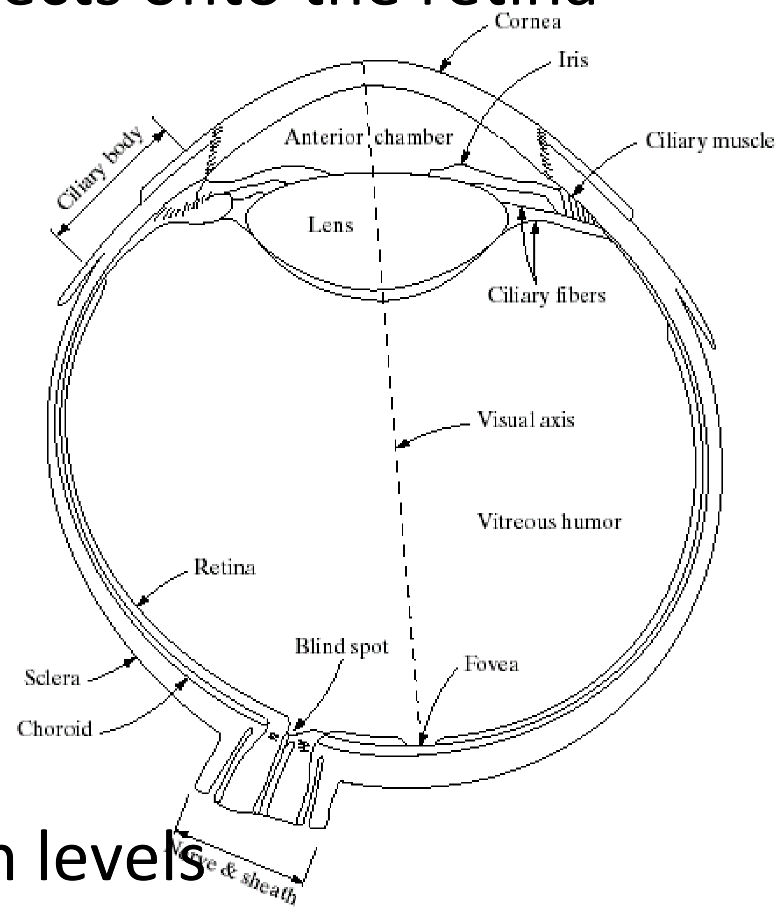
From Wikipedia

Images from Different EM Radiation

- Radar imaging (radio waves)
- Magnetic Resonance Imaging (MRI) (Radio waves)
- Microwave imaging
- Infrared imaging
- Photographs
- Ultraviolet imaging telescopes
- X-rays and Computed tomography
- Positron emission tomography (gamma rays)
- Ultrasound (not EM waves)

Human Visual System: Structure Of The Human Eye

- The lens focuses light from objects onto the retina
- Retina covered with light receptors called ***cones*** (6-7 million) and ***rods*** (75-150 million)
- Cones concentrated around fovea. Very sensitive to colour
- Rods more spread out and sensitive to low illumination levels



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image Formation In The Eye

- Muscles in eye can change the shape of the lens allowing us focus on near or far objects
- An image is focused onto retina exciting the rods and cones and send signals to the brain

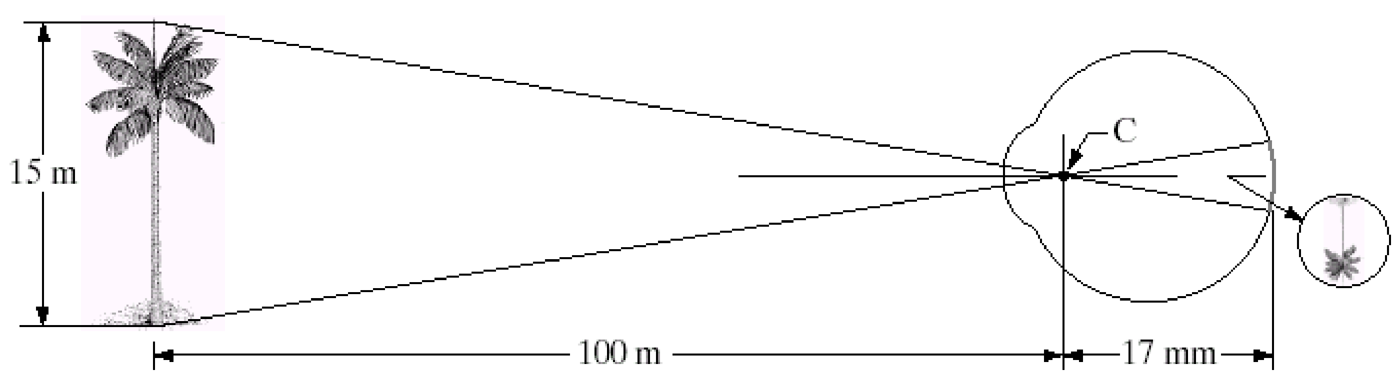
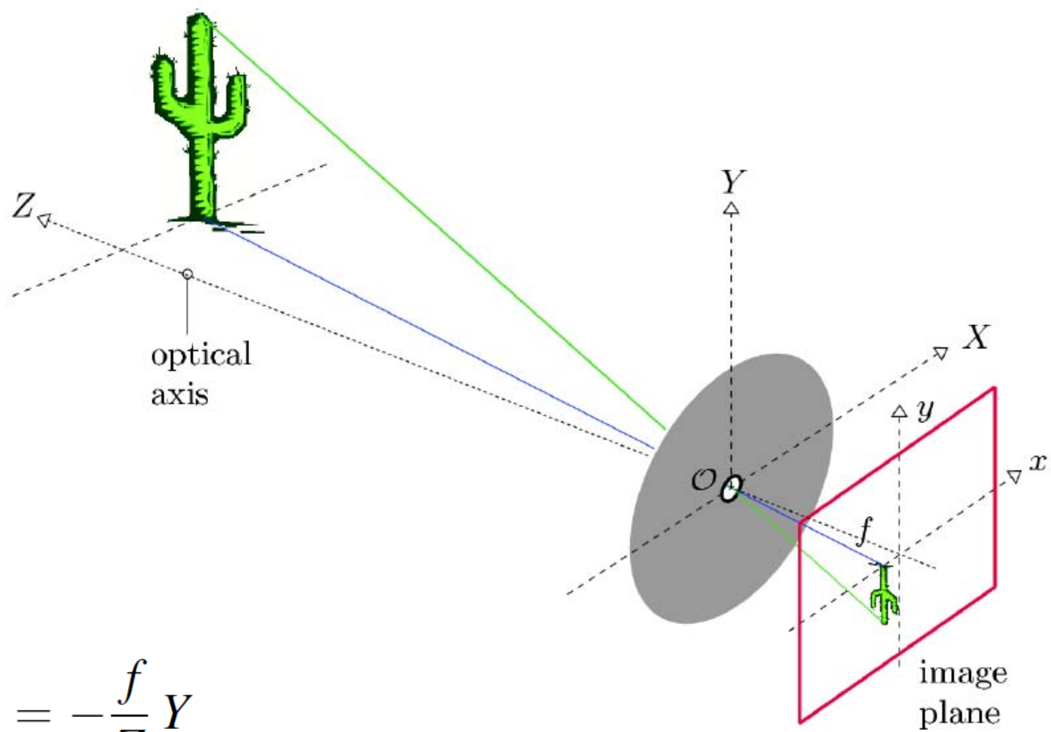


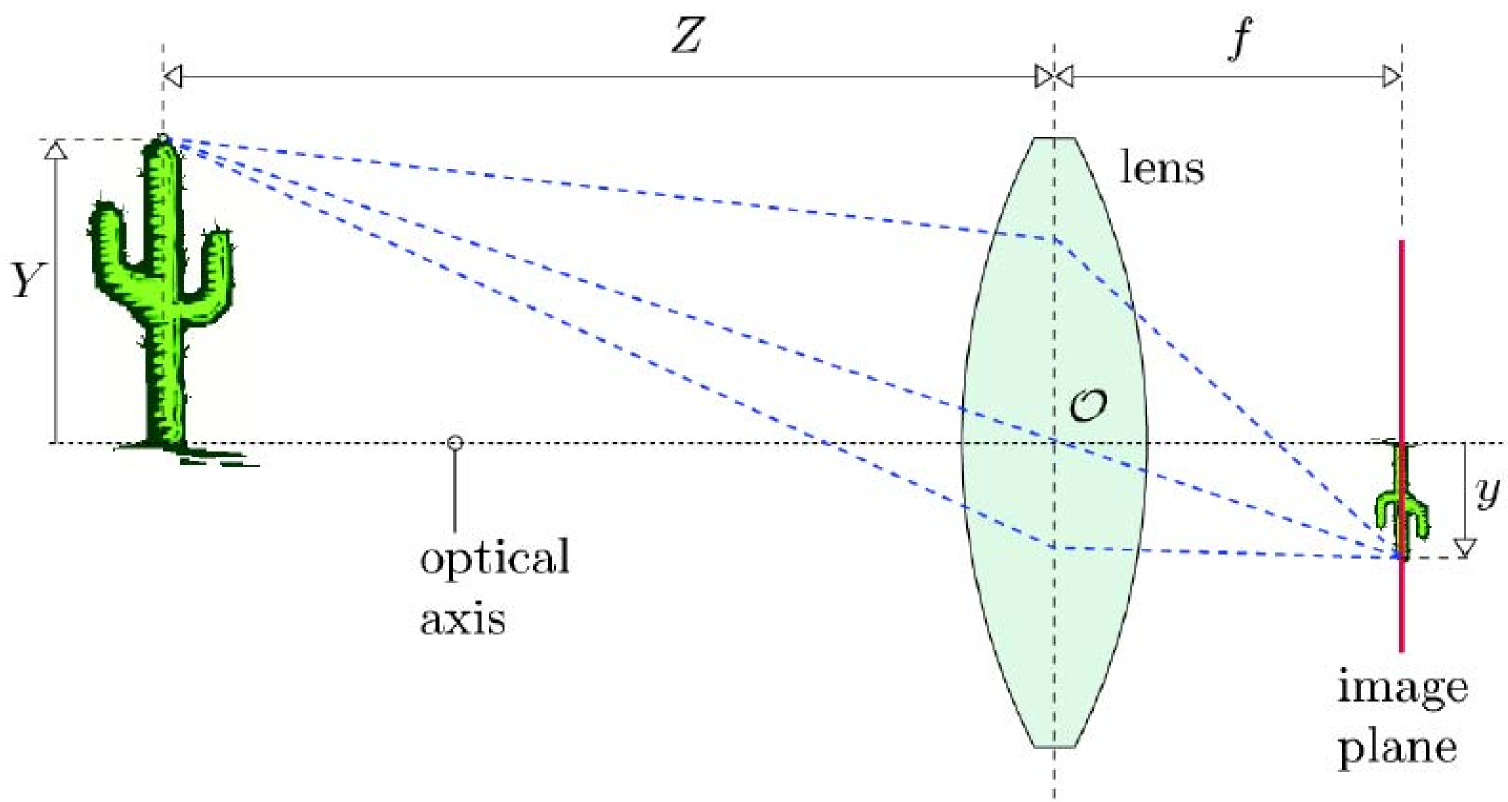
Image Formation

- The Pinhole Camera (abstraction)
  - First described by ancient Chinese and Greeks (300-400AD)



$$x = -\frac{f}{Z}X, \qquad y = -\frac{f}{Z}Y$$

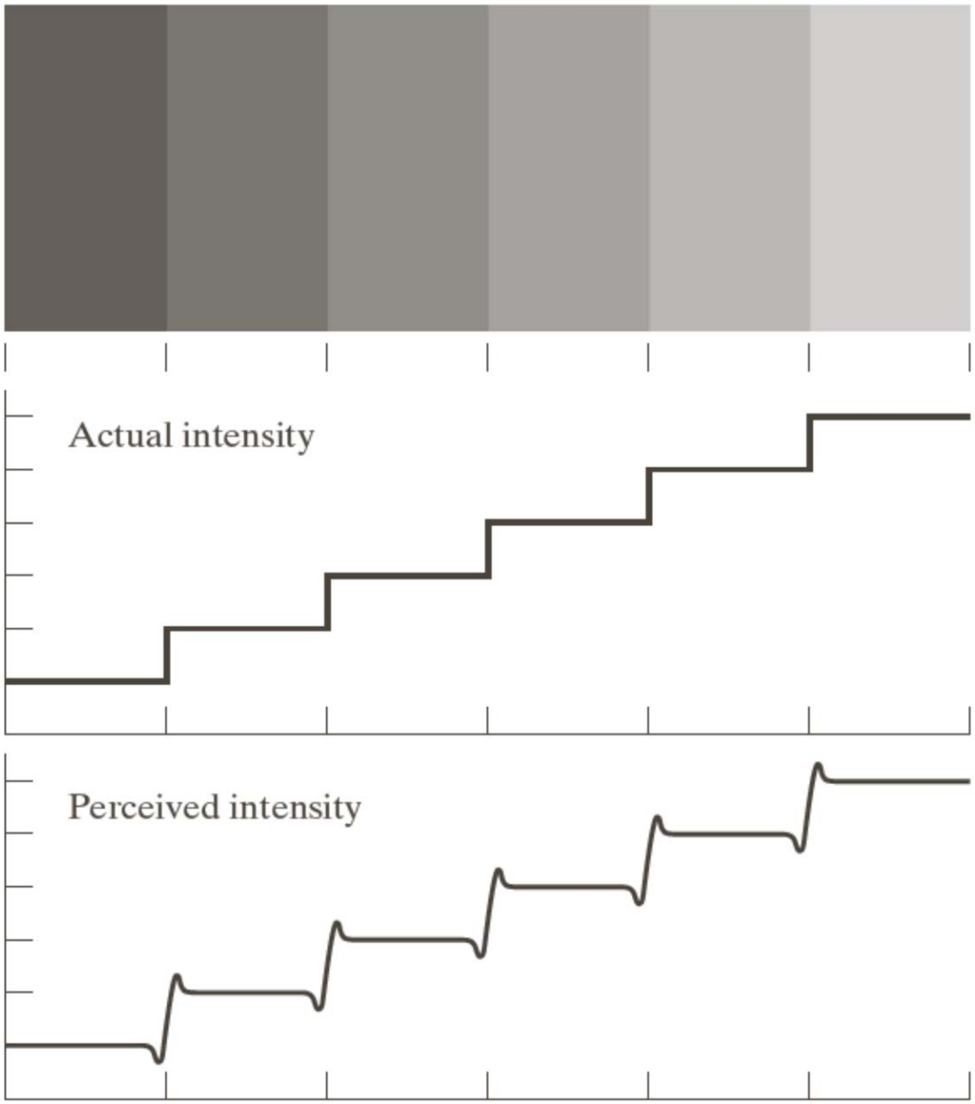
Thin Lens



Brightness Adaptation & Discrimination

- The human visual system can perceive approximately 10^{10} different light intensity levels
- However, at any one time we can only discriminate between a much smaller number – ***brightness adaptation***
- Similarly, ***perceived intensity*** of a region is related to the light intensities of the regions surrounding it

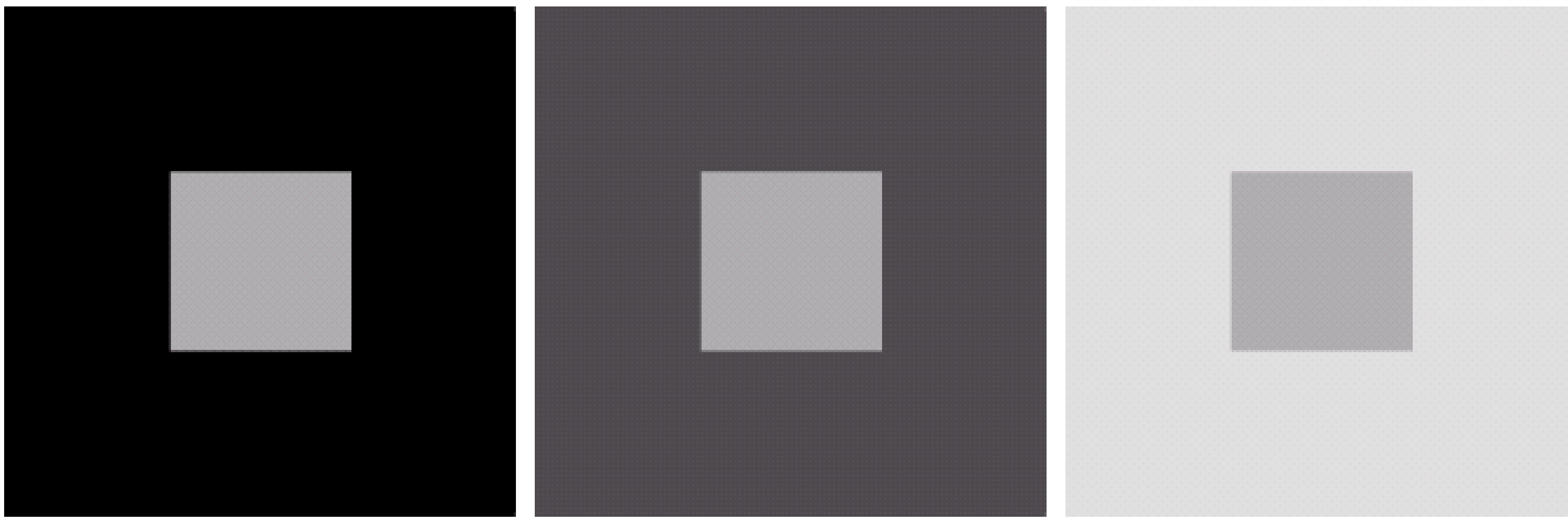
Brightness Adaptation & Discrimination: Mach Band Effect



Perceived intensity overshoots or undershoots at areas of intensity change

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Brightness Adaptation & Discrimination

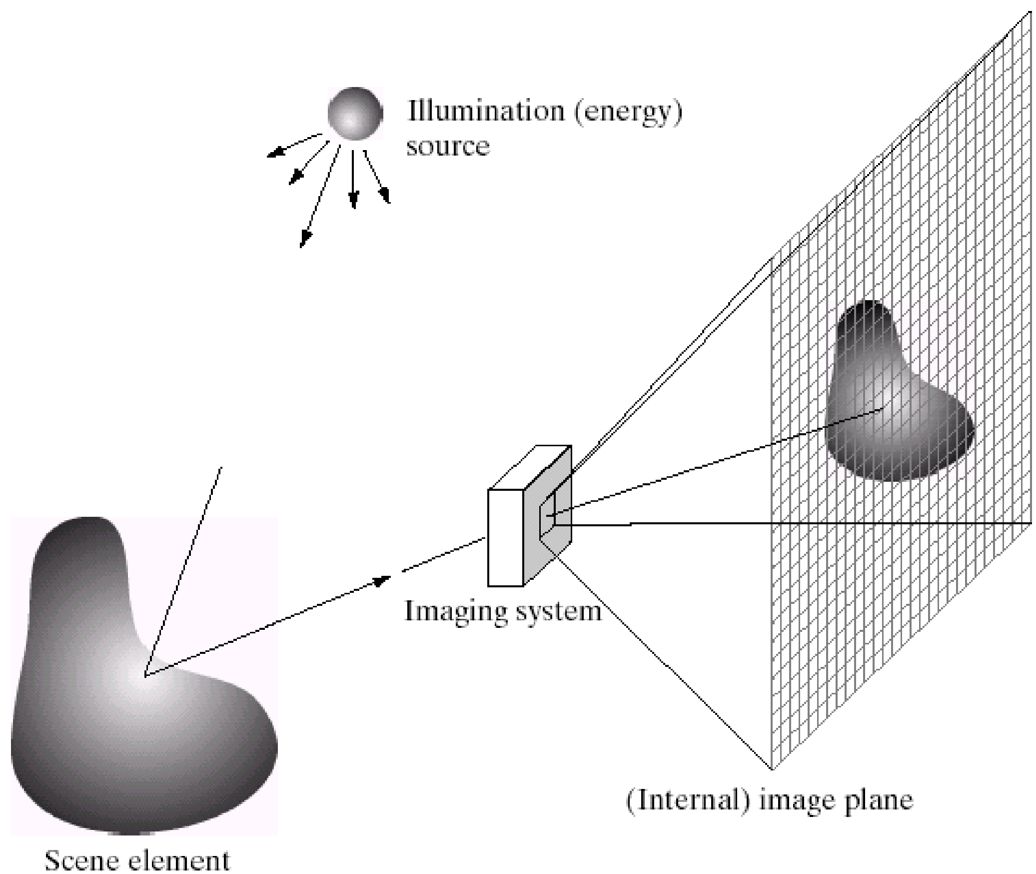


An example of *simultaneous contrast*

All inner squares have same intensity but appear darker as outer square (surrounding area) gets lighter

Image Acquisition

● Images typically generated by ***illuminating a scene*** and absorbing energy reflected by scene objects

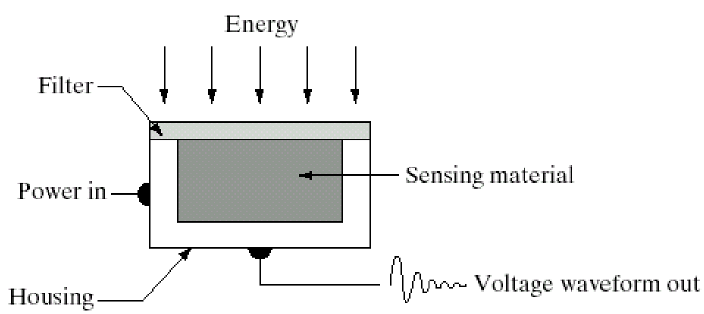


Images taken from Gonzalez & Woods, Digital Image Processing (2002)

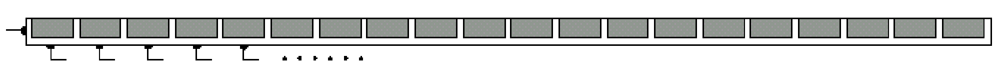
Image Sensing

- Incoming energy (e.g. light) lands on a sensor material responsive to that type of energy, generating a voltage

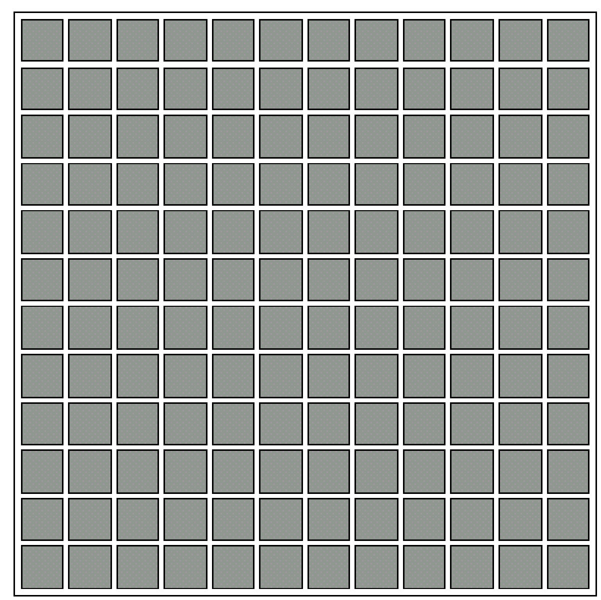
- Collections of sensors are arranged to capture images



Imaging Sensor



Line of Image Sensors

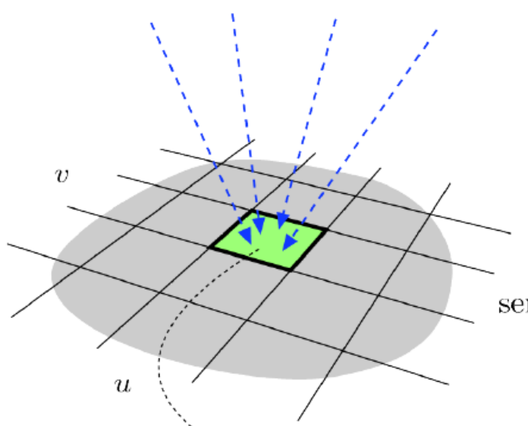


Array of Image Sensors

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Spatial Sampling

- Cannot record image values for all (x,y)
- Sample/record image values at discrete (x,y)
- Sensors arranged in grid to sample image



$F(x,y)$

148	123	52	107	123	162	172	123	64	89	⋯
147	130	92	95	98	130	171	155	169	163	⋯
141	118	121	148	117	107	144	137	136	134	⋯
82	106	93	172	149	131	138	114	113	129	⋯
57	101	72	54	109	111	104	135	106	125	⋯
138	135	114	82	121	110	34	76	101	111	⋯
138	102	128	159	168	147	116	129	124	117	⋯
113	89	89	109	106	126	114	150	164	145	⋯
120	121	123	87	85	70	119	64	79	127	⋯
145	141	143	134	111	124	117	113	64	112	⋯
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	

$I(u,v)$

Image (Spatial) Sampling

- A digital sensor can only measure a limited number of **samples** at a **discrete** set of energy levels

- **Sampling** can be thought of as:

  Continuous signal x comb function

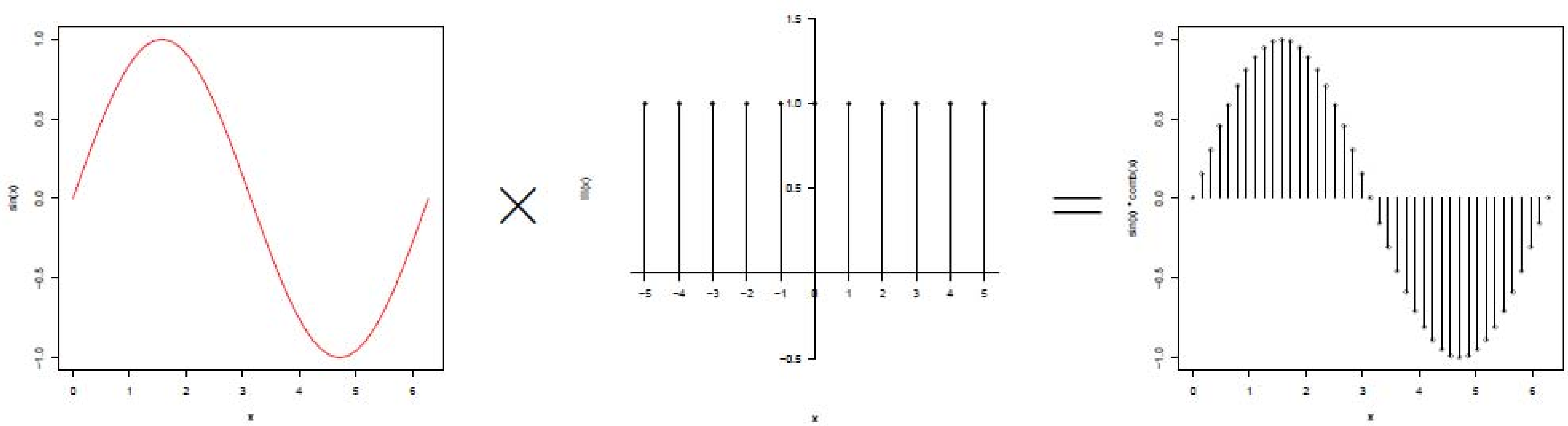


Image Quantization

- ***Quantization***: process of converting continuous **analog** signal into its digital representation
- Discretize image *I(u,v)* values
- Limit values image can take

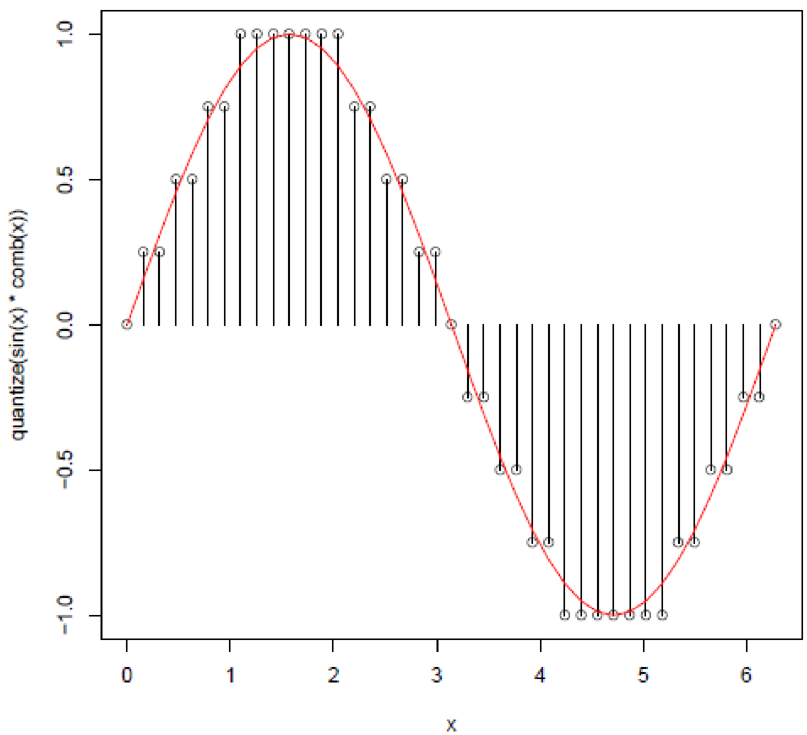
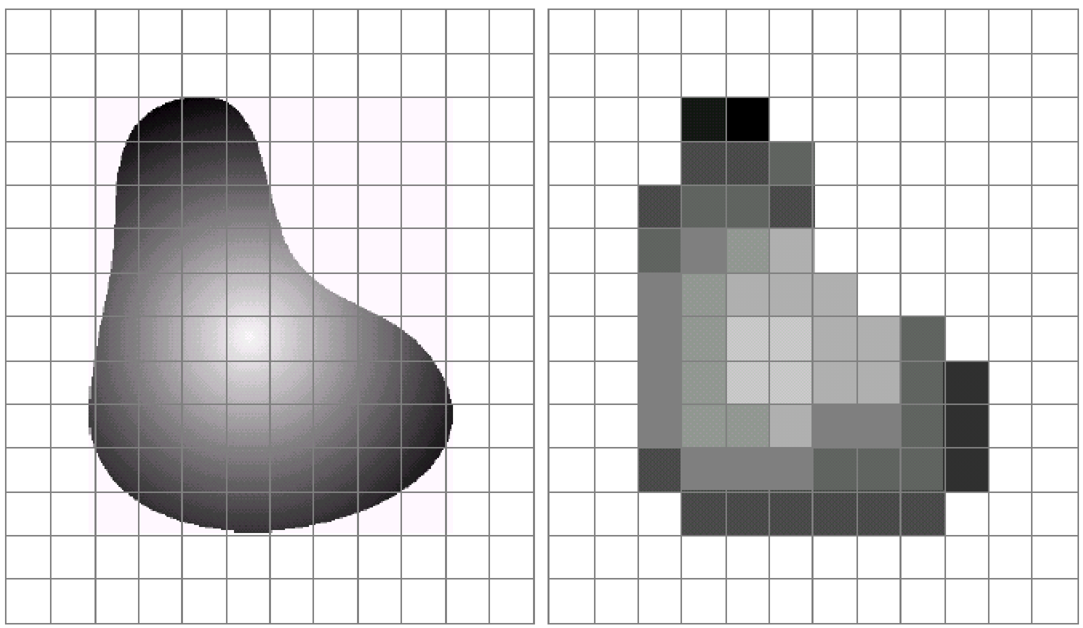


Image Sampling And Quantization

- Sampling and quantization generates **approximation** of a real world scene



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Image as Discrete Function

After spatial sampling and quantization, an image is a discrete function. The image domain Ω is now discrete:

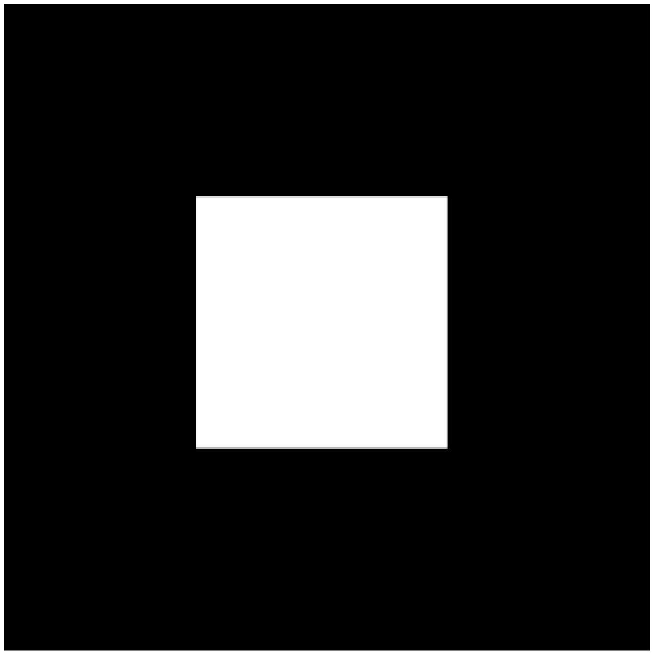
$$\Omega \subset \mathbb{N}^2,$$

and so is the image range:

$$I : \Omega \to \{1, \ldots, K\},$$

where $K \in \mathbb{N}$.

Image as a Function



A simple image

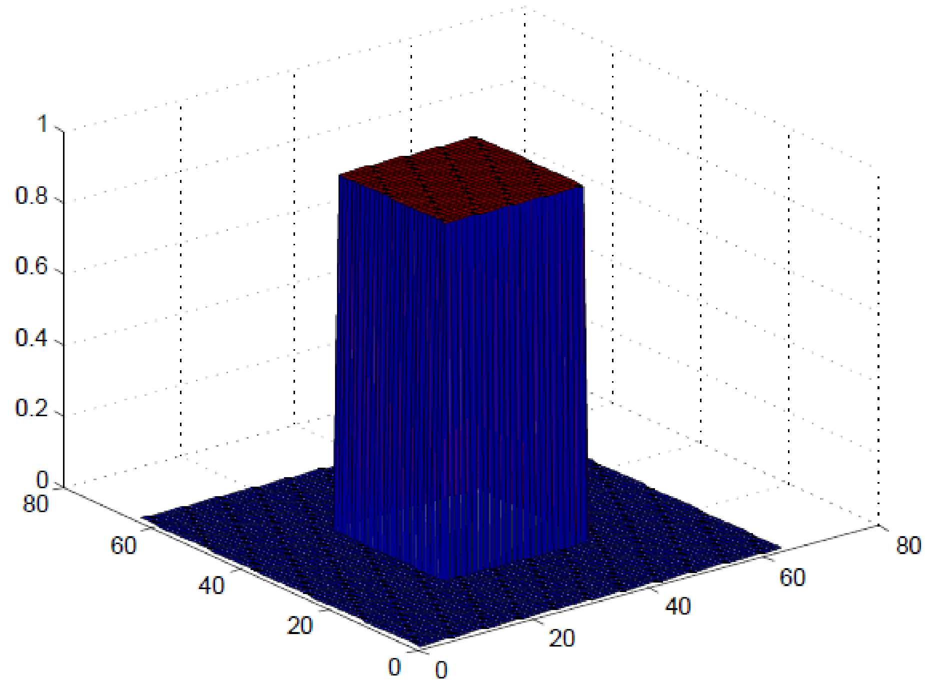
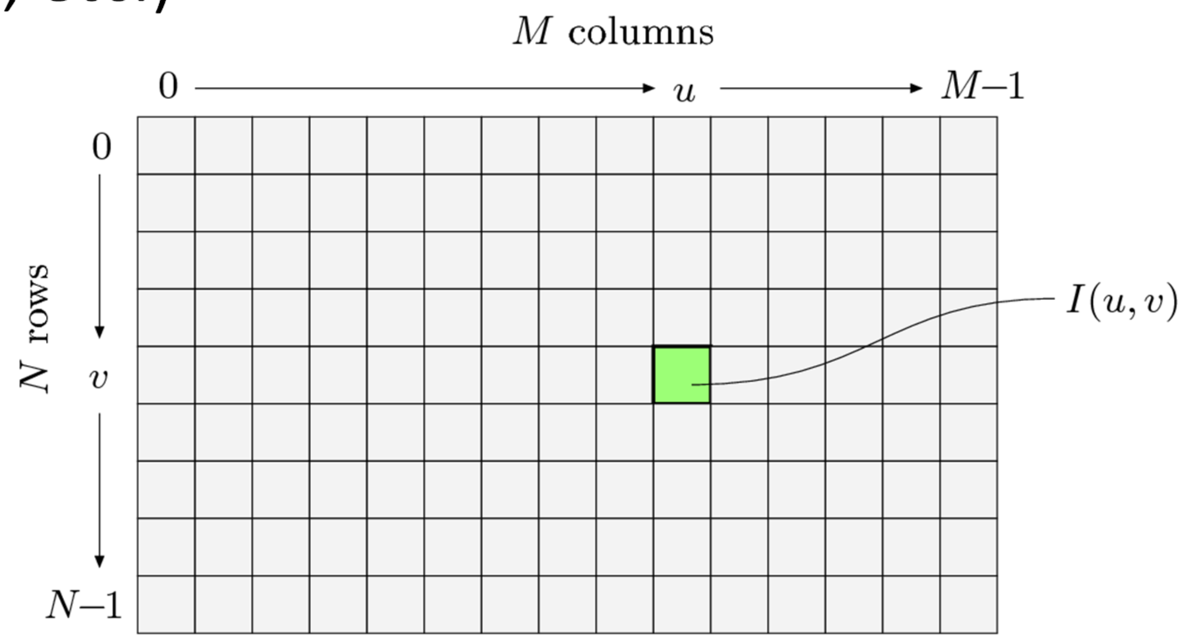


Image function as a height field

Representing Images

- Image data structure is 2D array of pixel values
- Pixel values are gray levels in range 0-255 or RGB colors
- Array values can be any data type (bit, byte, int, float, double, etc.)



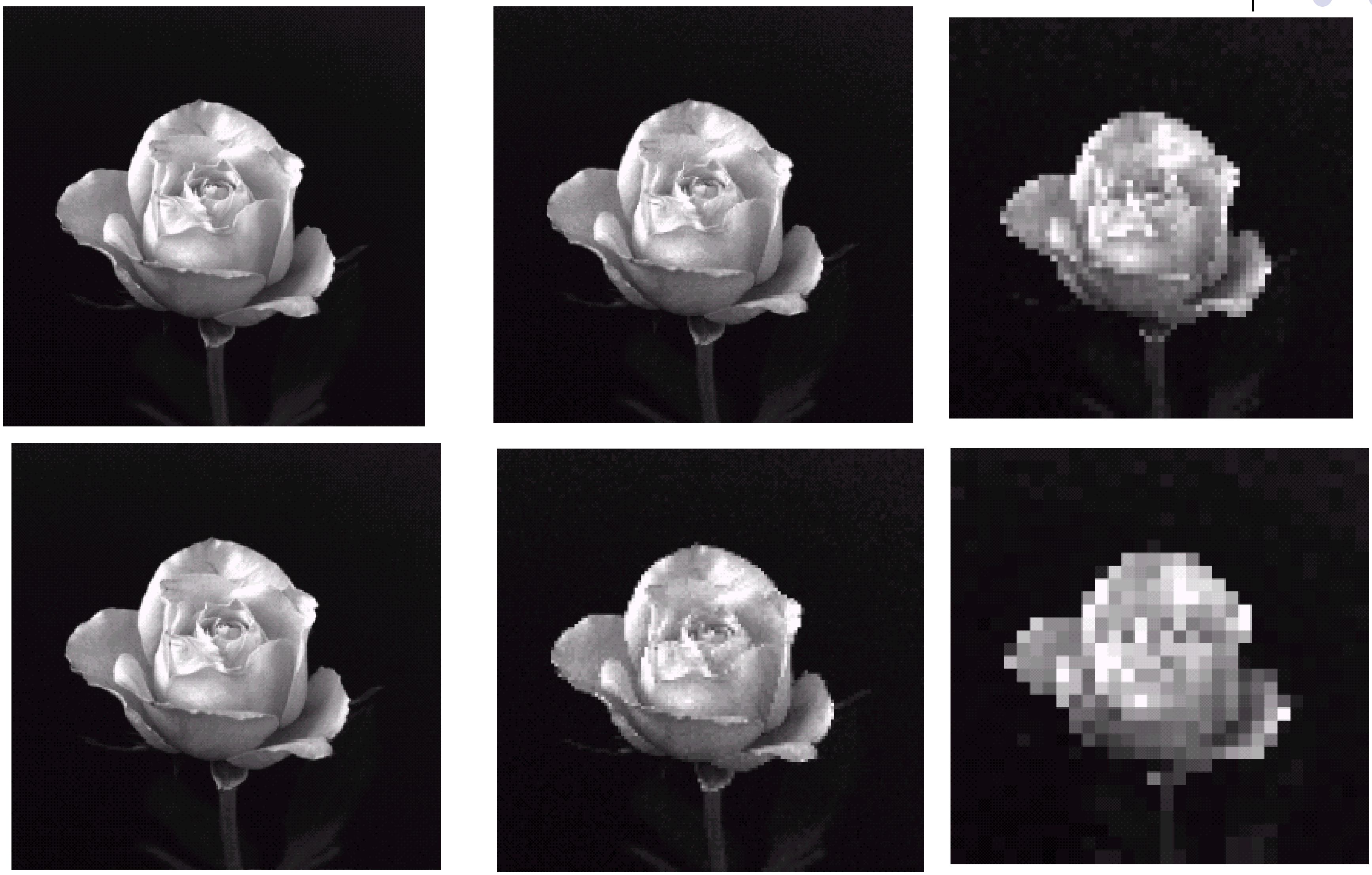
Spatial Resolution

- *The* ***spatial resolution*** of an image is determined by how fine/coarse sampling was carried out
- **Spatial resolution:** smallest discernable image detail
  - Vision specialists talk about image resolution
  - Graphic designers talk about *dots per inch* (DPI)

Spatial Resolution

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Spatial Resolution: Stretched Images



Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Intensity Level Resolution

- ***Intensity level resolution*:** number of intensity levels used to represent the image
  - The more intensity levels used, the finer the level of detail discernable in an image
  - Intensity level resolution usually given in terms of number of bits used to store each intensity level

Number of Bits	**Number of Intensity Levels**	**Examples**
1	2	0, 1
2	4	00, 01, 10, 11
4	16	0000, 0101, 1111
8	256	00110011, 01010101
16	65,536	1010101010101010

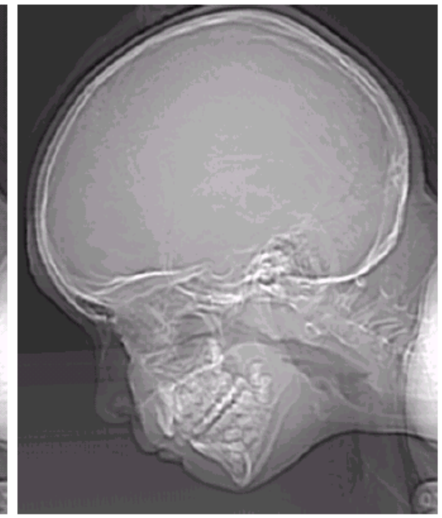
Intensity Level Resolution

256 grey levels (8 bits per pixel)

128 grey levels (7 bpp)

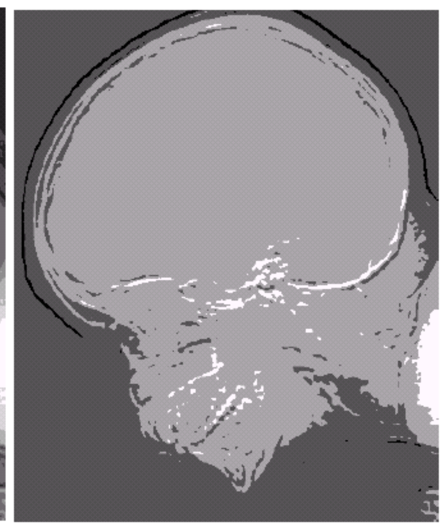
64 grey levels (6 bpp)

32 grey levels (5 bpp)



16 grey levels (4 bpp)

8 grey levels (3 bpp)

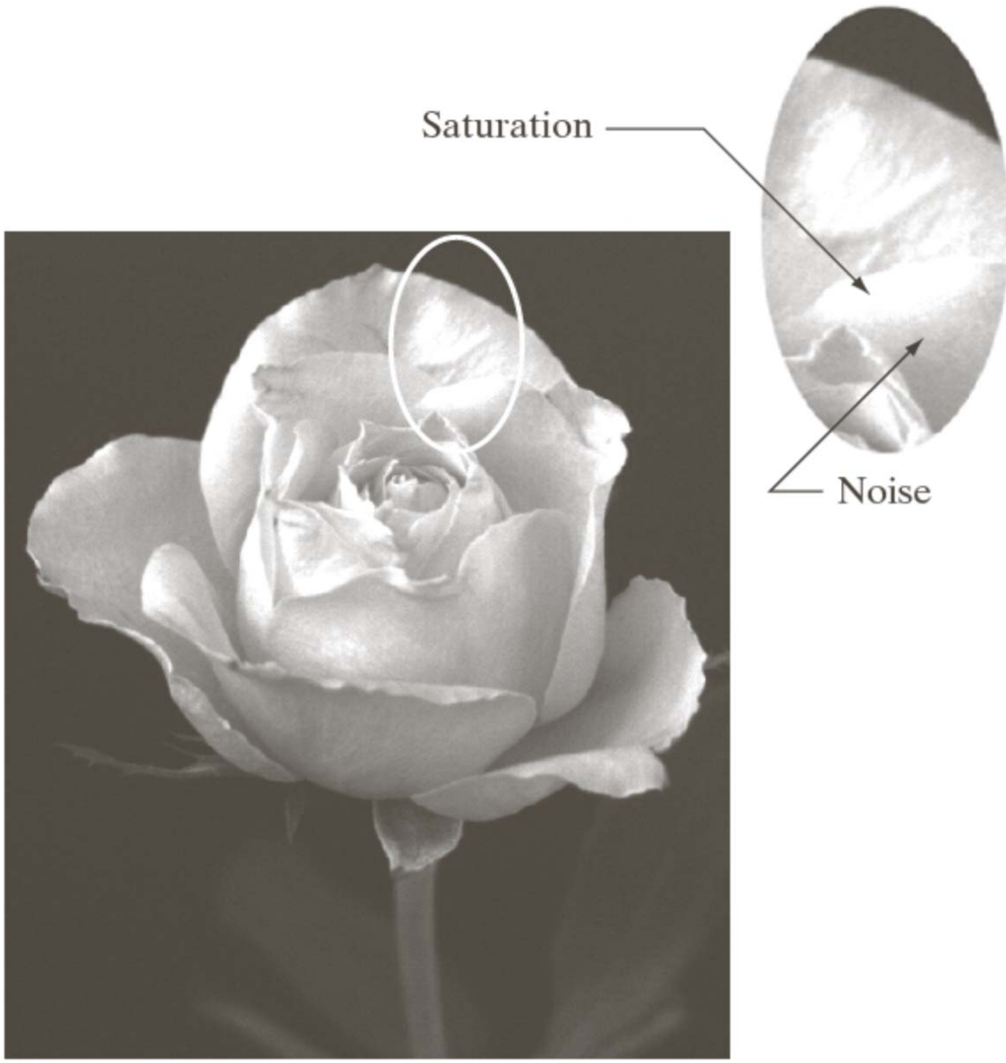


4 grey levels (2 bpp)

2 grey levels (1 bpp)

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Saturation & Noise



Saturation: highest intensity value above which color is washed out

Noise: grainy texture pattern

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Resolution: How Much Is Enough?

- The big question with resolution is always *how much is enough*?
  - Depends on what is in the image *(details)* and what you would like to do with it *(applications)*
  - Key questions:
    - Does image look aesthetically pleasing?
    - Can you see what you need to see in image?

Resolution: How Much Is Enough?

- **Example:** Picture on right okay for counting number of cars, but not for reading the number plate

Intensity Level Resolution

Images taken from Gonzalez & Woods, Digital Image Processing (2002)

Low Detail

Medium Detail

High Detail

Image File Formats

- Hundreds of image file formats. Examples
  - Tagged Image File Format (TIFF)
  - Graphics Interchange Format (GIF)
  - Portable Network Graphics (PNG)
  - JPEG, BMP, Portable Bitmap Format (PBM), etc
- Image pixel values can be
  - **Grayscale:** 0 – 255 range
  - **Binary:** 0 or 1
  - **Color:** RGB colors in 0-255 range (or other color model)
  - **Application specific** (e.g. floating point values in astronomy)

How many Bits Per Image Element?

Grayscale (Intensity Images):

Chan.	*Bits/Pix.*	*Range*	*Use*
1	1	$0 \ldots 1$	Binary image: document, illustration, fax
1	8	$0 \ldots 255$	Universal: photo, scan, print
1	12	$0 \ldots 4095$	High quality: photo, scan, print
1	14	$0 \ldots 16383$	Professional: photo, scan, print
1	16	$0 \ldots 65535$	Highest quality: medicine, astronomy

Color Images:

Chan.	*Bits/Pix.*	*Range*	*Use*
3	24	$[0 \ldots 255]^3$	RGB, universal: photo, scan, print
3	36	$[0 \ldots 4095]^3$	RGB, high quality: photo, scan, print
3	42	$[0 \ldots 16383]^3$	RGB, professional: photo, scan, print
4	32	$[0 \ldots 255]^4$	CMYK, digital prepress

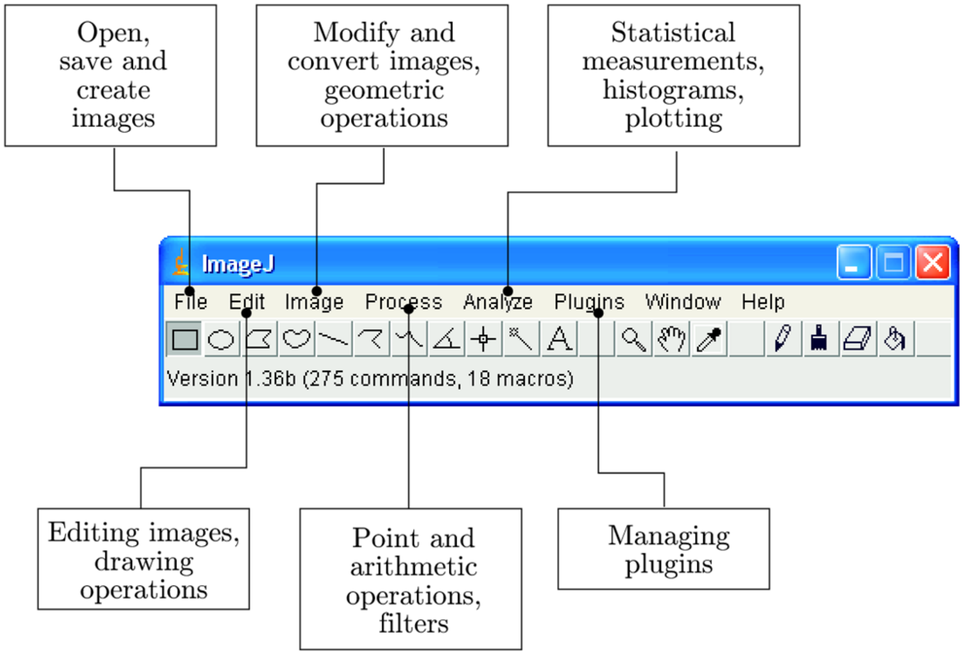
Special Images:

Chan.	*Bits/Pix.*	*Range*	*Use*
1	16	$-32768 \ldots 32767$	Whole numbers pos./neg., increased range
1	32	$\pm 3.4 \cdot 10^{38}$	Floating point: medicine, astronomy
1	64	$\pm 1.8 \cdot 10^{308}$	Floating point: internal processing

Introduction to ImageJ

- **ImageJ:** Open source Java Image processing software
- Developed by Wayne Rasband at Nat. Inst for Health (NIH)
  - Many image processing algorithms **already implemented**
  - New image processing algorithms can also be implemented easily
  - Nice click-and-drag interface

Wayne Rasband (right)

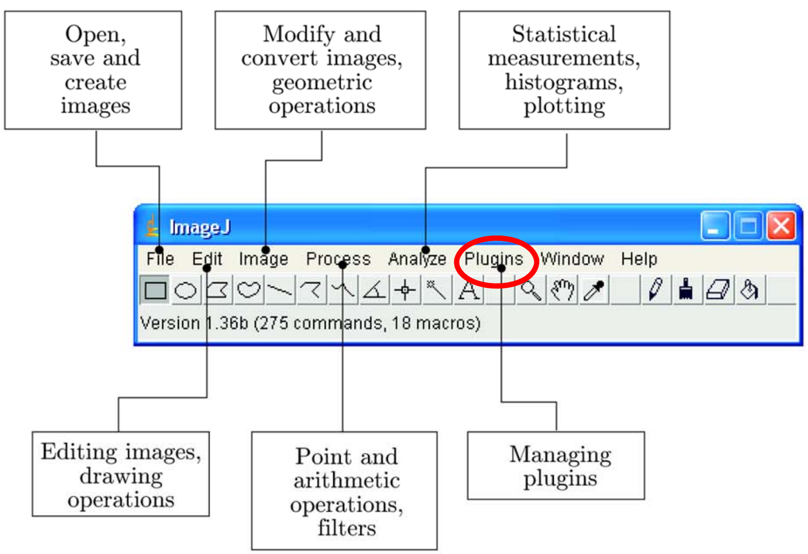


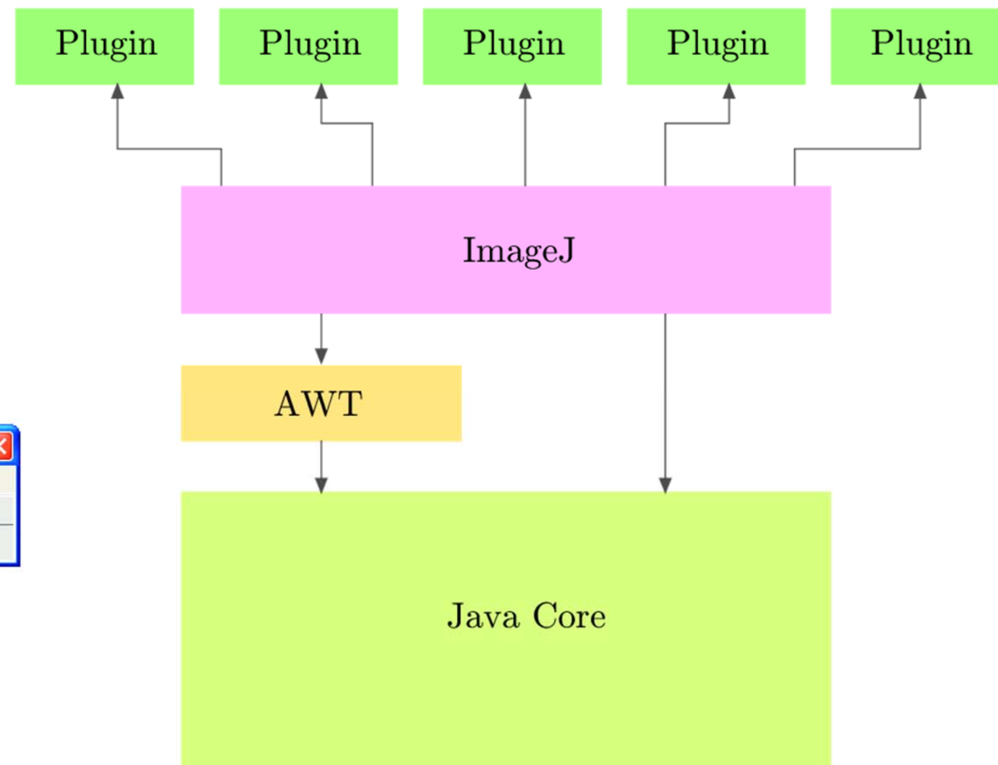
ImageJ: Key Features

- **Interactive tools** for image processing of images
  - Supports many image file formats (JPEG, PNG, GIF, TIFF, BMP, DICOM, FITS)
- **Plug-in mechanism** for implementing new functionality, extending ImageJ
- **Macro language + interpreter:** Easy to implement large blocks from small pieces without knowing Java

ImageJ Software Architecture

- ImageJ uses Java's windowing system (AWT) for display
- Programmer writes plugins to extend ImageJ
  - Already implemented plugins available through ImageJ's plugins menu





ImageJ Plugins

- **Plugins:** Java classes that implement an interface defined by ImageJ
- Two types of plugins
  - **Plugin:** Requires no image to be open first
  - **PlugInFilter:** Passed currently open image, operates on it
- We will mostly focus on **PlugInFilters**
- Two methods defined
  - **int setup(String *arg,* ImagePlus *im*):**
    - Does initialization, verifies plugin capabilities matches input image
  - **int run(ImageProcessor *ip*):**
    - Does actual work. Passed image (ip), modifies it, creates new images

First ImageJ Example: Invert Image

- **Task:** Invert 8-bit grayscale *(M x N)* image
- Basically, replace each image pixel with its complement

$$I(u,v) \leftarrow 255 - I(u,v)$$

- We shall call plugIn **My_Inverter**
  - Name of Java Class: **My_Inverter**
  - Name of source file: **My_Inverter.java**
  - "_" underscore makes ImageJ recognize source file as plugin
  - After compilation, automatically inserted into ImageJ menu

First ImageJ Example: Invert Image

```
import ij.ImagePlus;
import ij.plugin.filter.PlugInFilter;
import ij.process.ImageProcessor;

public class My_Inverter implements PlugInFilter {

  public int setup (String arg, ImagePlus im) {
    return DOES_8G; // this plugin accepts 8-bit grayscale images
  }

  public void run (ImageProcessor ip) {
    int w = ip.getWidth();
    int h = ip.getHeight();

    // iterate over all image coordinates
    for (int u = 0; u < w; u++) {
      for (int v = 0; v < h; v++) {
        int p = ip.getPixel(u, v);
        ip.putPixel(u, v, 255-p); // invert
      }
    }
  }

} // end of class My_Inverter
```

Indicates plugIn handles 8-bit grayscale images

Retrieves width and height of input image

Loops over all image pixels

Sets each pixel to its compliment (255 – original pixel value)

Compiling ImageJ Plugins

1. Place plugIn source code (My_Inverter.java) in sub-directory of ImageJ install location <ij>/plugins/
2. Open grayscale image from samples (since plugin requires image to be open)
3. Compile in run plugin by going to menu

   `Plugins->Compile and Run…`

- **Note:** On startup, ImageJ loads all plugins in the <ij>plugins/ sub-directory
- ImageJ can also be used with eclipse IDE (large programs)

References

- Wilhelm Burger and Mark J. Burge, Digital Image Processing, Springer, 2008
- University of Utah, CS 4640: Image Processing Basics, Spring 2012
- Gonzales and Woods, Digital Image Processing (3rd edition), Prentice Hall
- Digital Image Processing slides by Brian Mac Namee