

# Lecture 9


## An Introduction to Computer Vision: Part I

Rob Gaizauskas

# Lecture Outline

- What is Computer Vision?
- Image Formation
- Early Image Processing Operations
  
- Reading: (Readings that begin with \* are **mandatory**)
  - \*Russell and Norvig (2010), Chapter 24 “Perception”

# Lecture Outline

- What is Computer Vision?
- Image Formation
- Early Image Processing Operations
- Object Recognition by Appearance  Tomorrow
- Reconstructing the 3D World
- Object Recognition from Structural Information
- Applications
- Reading: (Readings that begin with \* are **mandatory**)
  - \*Russell and Norvig (2010), Chapter 24 “Perception”

# What is Computer Vision?

- Szeliski (2011):

*(the study of) mathematical techniques for recovering the three-dimensional shape and appearance of objects in imagery*

- Wikipedia:

*Computer vision is a field that includes methods for acquiring, processing, analyzing, and understanding images and, in general, high-dimensional data from the real world in order to produce numerical or symbolic information, e.g., in the forms of decisions. A theme in the development of this field has been to duplicate the abilities of human vision by electronically perceiving and understanding an image. This image understanding can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory. Computer vision has also been described as the enterprise of automating and integrating a wide range of processes and representations for vision perception.*

[http://en.wikipedia.org/wiki/Computer\\_vision](http://en.wikipedia.org/wiki/Computer_vision)

# What is Computer Vision?

- Computer vision vs computer graphics
  - Computer vision aims to produce 3D models from image data
  - Computer graphics aims to produce image data from 3D models

One may be viewed as the *inverse* of the other

- This duality akin to that between natural language understanding and generation
  - NL understanding aims to produce meaning representations from NL data
  - NL generation aims to produce NL output from meaning representations

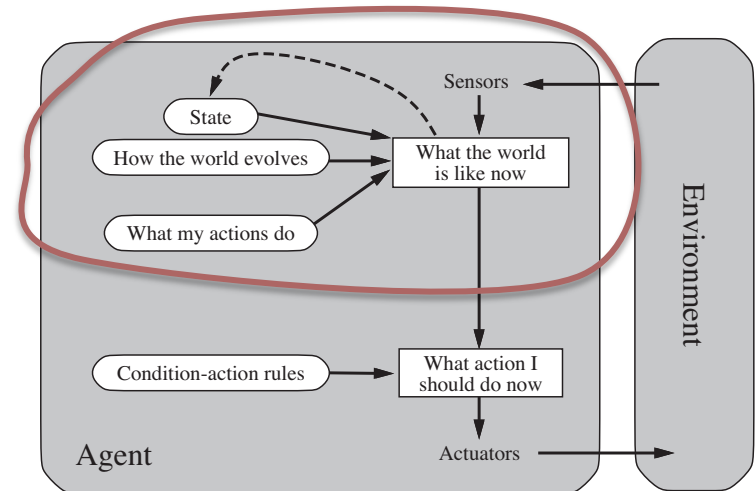
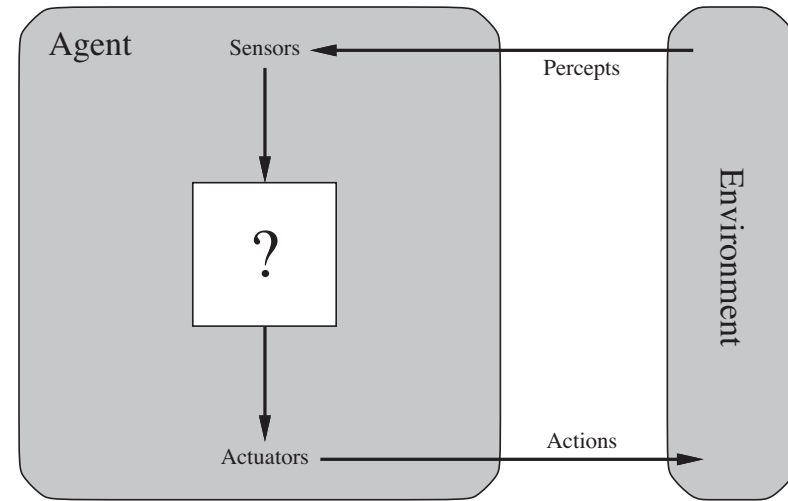
# Computer Vision and AI

- Intelligent agents are equipped with **sensors**
  - Sensors measure some aspect of the environment in which an agent is situated
  - The agent **interprets** sensor outputs to gain information about the world – this process is called **perception**
- Sensory **modalities** include:
  - Vision, hearing, touch, smell, taste
  - Radio, infrared, GPS, ...
  - Radar, ultrasound ... (active sensing)
- Visual perception appears to be a core capability for intelligent agents situated/operating in the physical world
  - The most intelligent natural agents all have advanced visual perceptual systems

# Perception and Agent Architecture

R&N Fig 2.1

- Recall the definition of agent from lecture 7:
  - “an **agent** is anything that can be viewed as perceiving its **environment** through **sensors** and acting on that environment through **actuators**” (R&N, p. 34)
- Vision is one mode of perception
  - For non-reflex agents, includes not just **sensing** light in the visible spectrum, but **interpreting** what is seen **as**, e.g., objects in spatial relations, participating in actions/ processes, etc.



# Computer Vision and AI

- Vision, like all perception, serves to further an agent's goals – it is not an end in itself
- Visual observations extremely rich in terms of
  - amount of detail they provide
  - Sheer volume of data
    - a robot video camera might produce  $1 \times 10^6$  24-bit pixels at 60Hz  $\approx$  10GB per minute
- Therefore key problem for agent:
  - Which aspects of the visual stimulus should be examined to help make good decisions and which aspects should be ignored?



# Computer Vision and AI

- Russell and Norvig distinguish 3 broad approaches:
  - **Feature extraction:**
    - Simple computations on sensor observations directly leading to behavioural response
    - E.g. in fruit fly (*Drosophila*) visual system directly linked to wing muscles allowing reflex escape response to visual inputs
  - **Recognition:**
    - Agent uses visual + other information to draw distinctions between objects encountered
    - E.g. labelling image yes/no depending on whether it contains edible food, a friend's face, etc.
  - **Reconstruction:**
    - Agent builds a geometric model of the world from an image or set of images

# Central Challenge for Object Recognition

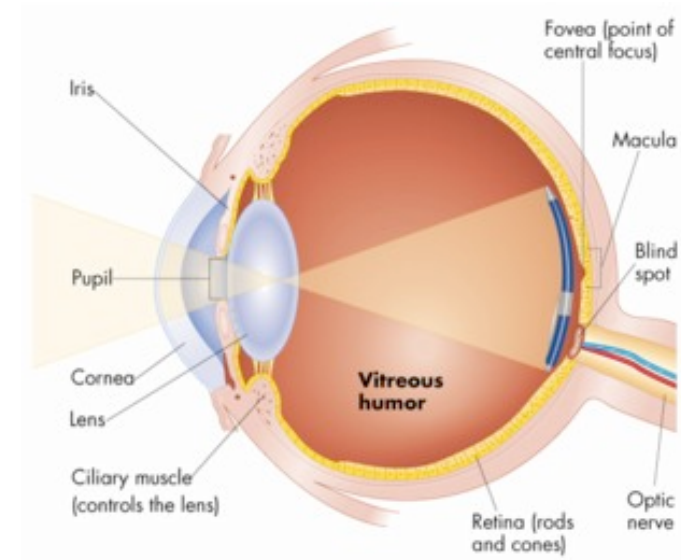
- Same object may appear very different in two images due to differences in
  - Illumination
  - Perspective
  - Distance
  - Partial occlusion
- Different objects may look very similar in two images, for the same reasons
- I.e. image ↔ object relation is highly ambiguous

# Lecture Outline

- What is Computer Vision?
- Image Formation
- Early Image Processing Operations
- Reading:
  - Russell and Norvig (2010), Chapter 24 “Perception”

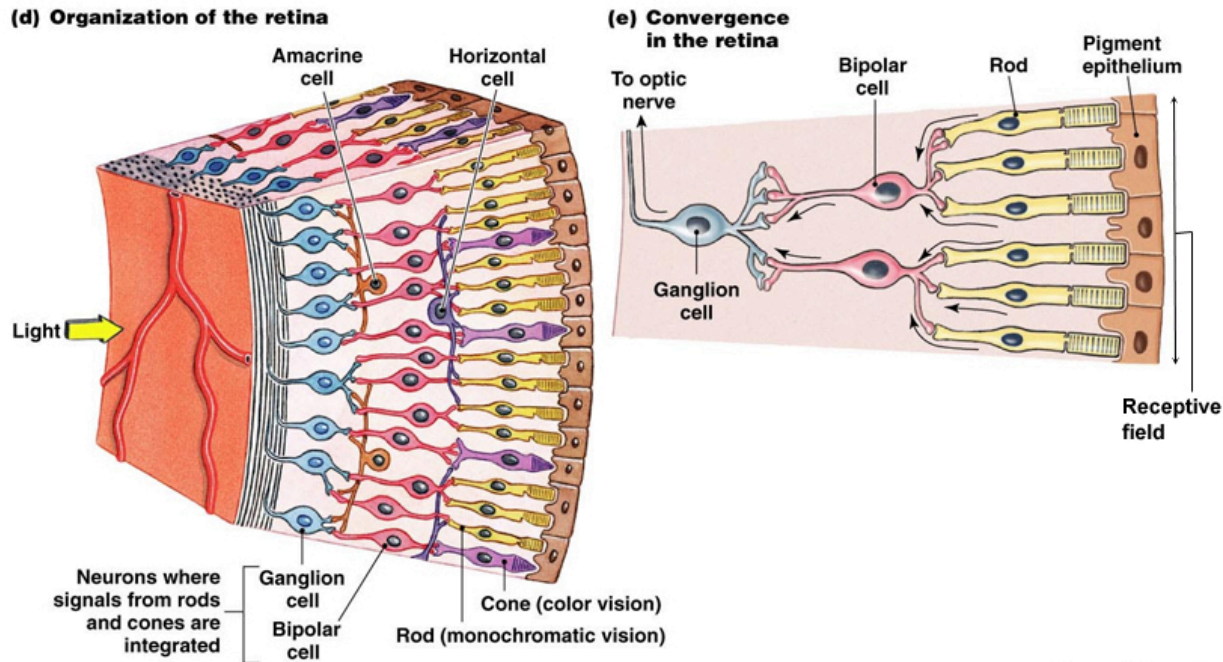
# Image Formation in the Eye

- Image sensors gather light from scattered objects in a **scene** + create a 2D **image**
- In the human eye images are formed on photoreceptors lining the retina
  - Cones: ( $\sim 5 \times 10^6$ ) responsible for colour vision – 3 types each sensitive to a different wavelength (“red, green, blue”)
  - Rods: ( $\sim 1 \times 10^8$ ) responsible for low/dim light vision
  - Cones mostly near centre of retina; rods evenly dispersed



# Human Eye: Rod and Cones

- Rods and cones pictorially



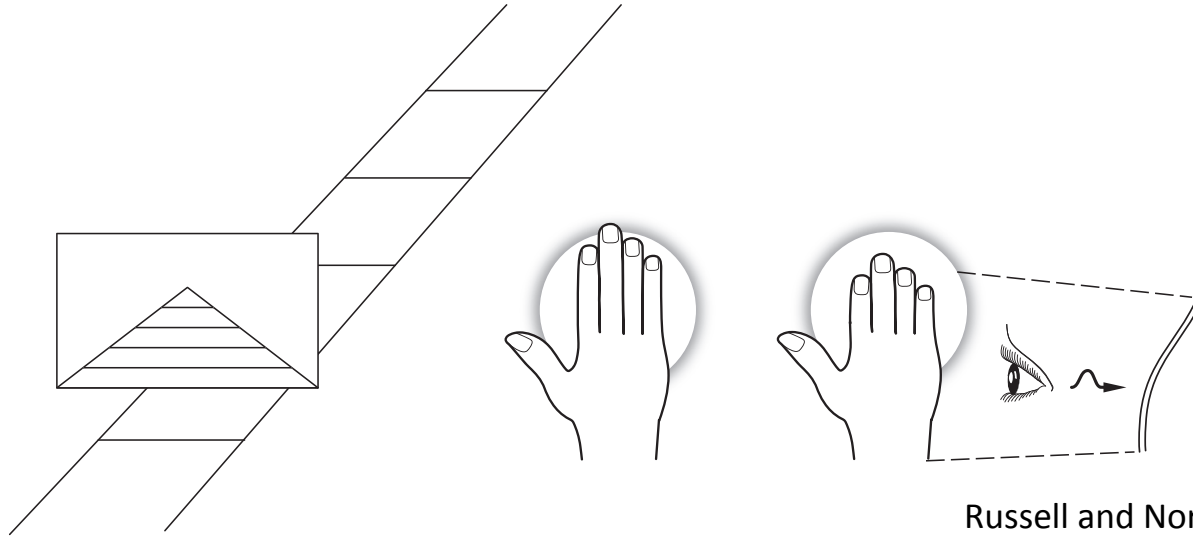
Copyright © 2007 Pearson Education, Inc., publishing as Benjamin Cummings.

Fig. 10-35

# Image Formation in Cameras

- In cameras the image formed on an image plane
  - On film coated with silver halides for traditional cameras
  - On a grid of millions of photosensitive **pixels** for digital cameras, where each pixel is either
    - A **complementary metal-oxide semiconductor** (CMOS), or
    - A **charge-coupled device** (CCD)
  - Photons arriving at sensor produce an effect whose strength is dependent on the photon's wavelength
  - Output of sensor is sum of all effects of photons observed in time window

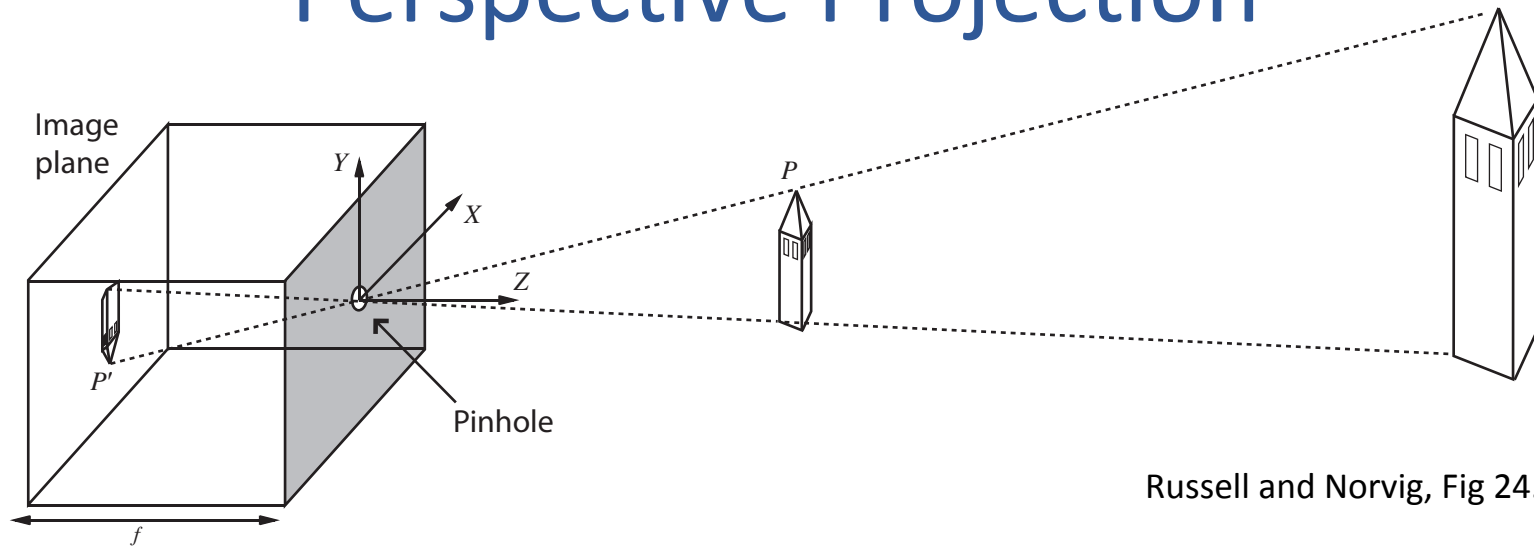
# Geometric Distortion in Imaging



Russell and Norvig, Fig 24.1

- Images on a 2D surface distort 3D scenes
  - Parallel lines appear to meet in the distance
  - Small hand blocks out moon
  - Tilted hand appears foreshortened

# Perspective Projection



Russell and Norvig, Fig 24.2

- In a pinhole camera
  - light passes through a small pinhole aperture on the front
  - Image is formed (inverted) on the rear plane
- Using 3D co-ordinate system with origin at pinhole, point  $P = (X, Y, Z)$  in scene is projected to  $P' = (x, y)$  in image plane with  $f$  the distance from pinhole to image plane

$$\frac{-x}{f} = \frac{X}{Z}, y = \frac{Y}{Z} \quad \longrightarrow \quad x = \frac{-fX}{Z}, y = \frac{-fY}{Z}$$



# Perspective Projection (cont)

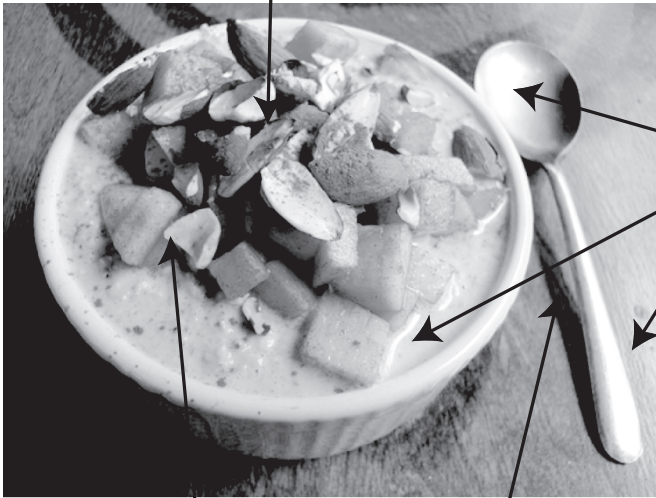
- Perspective Projection explains
  - why a small tower close up appears the same size in an image as a tall tower far away
  - Why two parallel lines appear to converge in the image at a single point (**vanishing point**)
- Need additional information to determine actual size of objects in scene – what sort of information?

# Lens Systems

- Pinhole cameras have problems
  - Need small aperture to keep image in focus – but few photons get through, yielding dark image
  - Can keep aperture open longer, but then get motion blur if objects in scene move
  - Or, can expand aperture, but then photons from multiple points in object fall across multiple points in image causing image blur
- Solution – **lens systems** – found in eyes and cameras
  - Focus light from nearby object locations to nearby locations in image plane
  - Can only focus light within a limited range of depths centred around a **focal plane**
  - Focal plane moved by adjusting lens, either by using muscles to change its shape (human eye) or by moving the lens back and forth (camera)

# Light and Shading

Diffuse reflection, bright



Specularities

Diffuse reflection, dark

Cast shadow

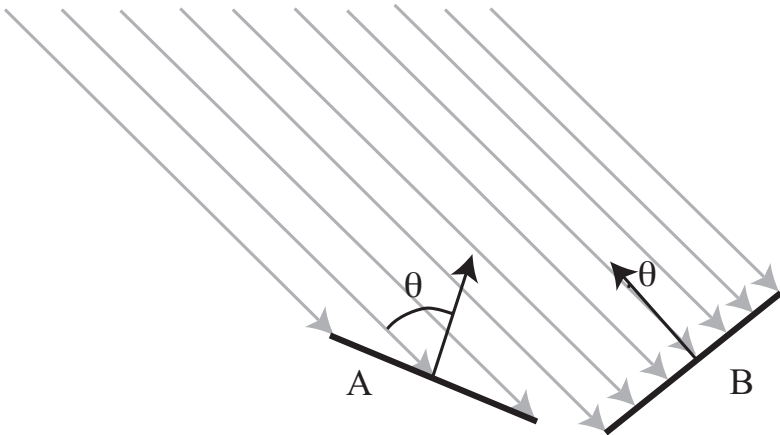
- Brightness of a pixel in an image is a function of the surface patch in the scene that projects to the pixel
- Image brightness is a good, though ambiguous, clue to object shape
- People can distinguish 3 main underlying causes of varying brightness and are able to reverse-engineer the object's properties

1. **Overall intensity:** white object in shadow may be less bright than black object in bright light, but eye can distinguish relative brightness and perceive white object as white
  2. **Reflection:** different points in scene reflect different amounts of light – perceived by people as lighter/darker and interpreted as texture or markings on object
  3. **Shading:** surface patches facing light brighter than those facing away from light – interpreted by people as coming from geometry of object
- Marking/shading can be confused – e.g. makeup on cheek may be mistaken as shading, making the face look thinner

# Light and Shading (cont)

- **Diffuse reflection**
  - Most surfaces reflect light evenly in all directions
    - E.g. most cloth, rough wooden/brick surfaces, vegetables, rough stone
  - Means surface brightness independent of viewing direction
- **Specular reflection**
  - Some surfaces, e.g. mirrors, do not reflect light evenly in all directions
  - Means what you see depends on viewing direction
  - Perfect mirrors produce what is called specular reflection
  - Some surface have small patches where specular reflection occurs – called **specularities**
    - E.g. brushed metal, wet floors, car windscreens

# Light and Shading (cont)



- Main source of illumination outside is the sun whose rays travel parallel to each other
- Modeled as **distant point light source**
  - Most important model of lighting
  - Works well for indoor scenes too
- A diffuse surface patch illuminated by a distant point light source reflects some proportion of the light it collects
  - This proportion is called the **diffuse albedo**
  - White paper/snow have high albedo (0.90); flat black velvet/charcoal have low albedo (~0.05)
- Brightness of a diffuse patch  $I$  given by **Lambert's cosine law**

$$I = \rho I_0 \cos \theta$$

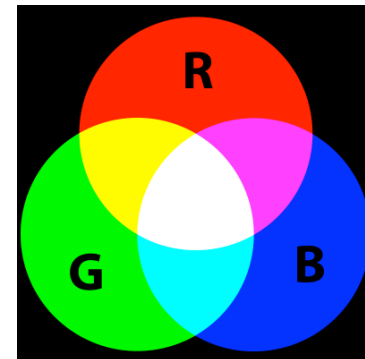
where  $\rho$  is the diffuse albedo,  $I_0$  is the intensity of the light source and  $\theta$  is the angle between the direction of the light source and the surface normal

# Light and Shading (cont)

- Lambert's law predicts
  - bright image pixels come from surface patches that face light
  - dark image pixels come from surface patches tangential to light source direction
- Thus shading provides some shape information
- A surface not reached by light is in shadow
- Shadows rarely uniformly black – surface receives reflected light from other sources (e.g. sky, other surfaces)
  - called interreflections – can have significant effect on brightness of other surfaces
  - Sometimes modeled by adding a constant ambient illumination term to predicted intensity

# Colour

- Light arriving at eye has different amounts of energy at different wavelengths
- Human eye responds to light in the 380-750nm wavelength region, using 3 types of receptor cells (cones) with peak receptiveness at
  - 420nm (blue)
  - 540nm (green)
  - 570nm (red)
- Principle of trichromancy:  
for any spectral density function we can create another by mixing any three sufficiently wavelength-separated colours (e.g. red, green, blue) such that a human cannot distinguish the difference



## Colour (cont)

- Principle of Trichromancy means TV/computer screens just need R/G/B elements
- In computer vision
  - Surfaces can be modelled with 3 albedos for R/G/B
  - Light sources can be modelled with 3 R/G/B intensities
  - Lambert's law can then be applied to get 3 R/G/B pixel values
- This model correctly predicts that the same surface will produce different coloured image patches under different-coloured lights
- Human eye good at ignoring effects of different-coloured lights and can estimate colour of surface under white light – called **colour constancy**
  - Accurate colour constancy algorithms now exist (“auto white balance” function of digital cameras)



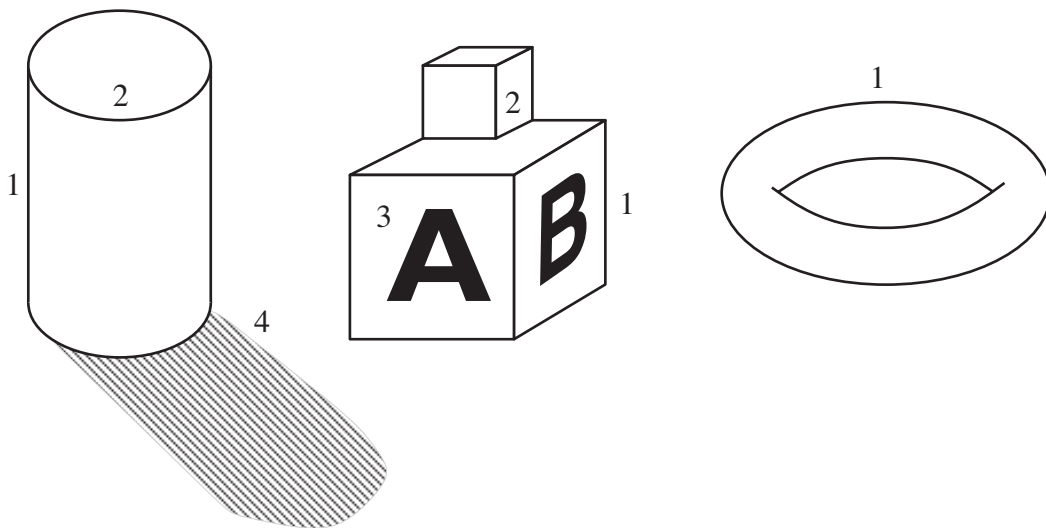
# Lecture Outline

- What is Computer Vision?
- Image Formation
- Early Image Processing Operations
- Reading:
  - Russell and Norvig (2010), Chapter 24 “Perception”

# Early Image Processing Operations

- Studying process of image formation lets us see how light reflects off objects in scene to form an image of, e.g.,  $5 \times 10^6$  3-byte pixels
- How do we start analyzing this data?
- Three early/low-level image processing operations
  1. Edge detection
  2. Texture analysis
  3. Optical flow
- Called early/low-level because first in pipeline of operations
  - Local – carried out on one part of image taking into account only pixels very nearby
  - Relatively knowledge-free – don't need to take into account objects that might be present in scene
- Good candidates for implementation in parallel hardware
  - graphics processing unit (GPU) or eye
- One mid-level operation: **segmentation of images** into regions

# Edge Detection



## Types of Edges

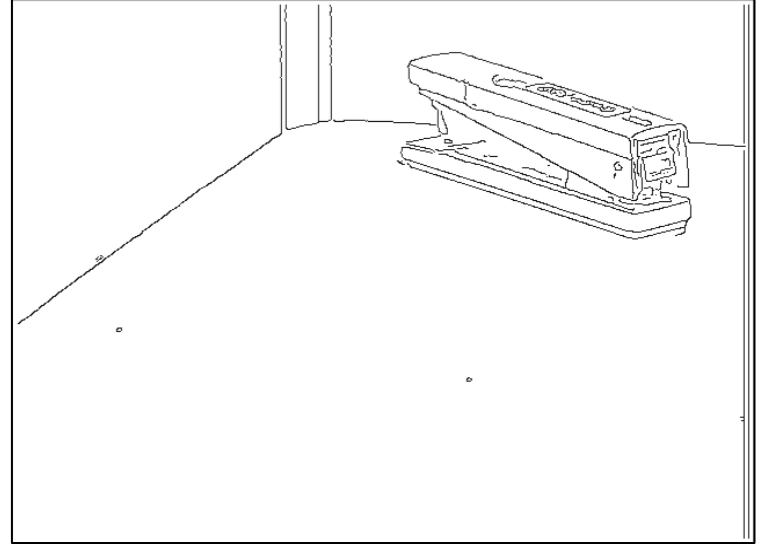
1. Depth discontinuities
2. Surface orientation discontinuities
3. Reflectance discontinuities
4. Illumination discontinuities (shadows)

- **Edges:** straight lines/curves in image across which there is a significant change in image brightness
- Goal: abstract away from messy multi-Mb image towards a more compact/abstract representation
- Motivation: edge contours correspond to important scene contours

# Edge Detection



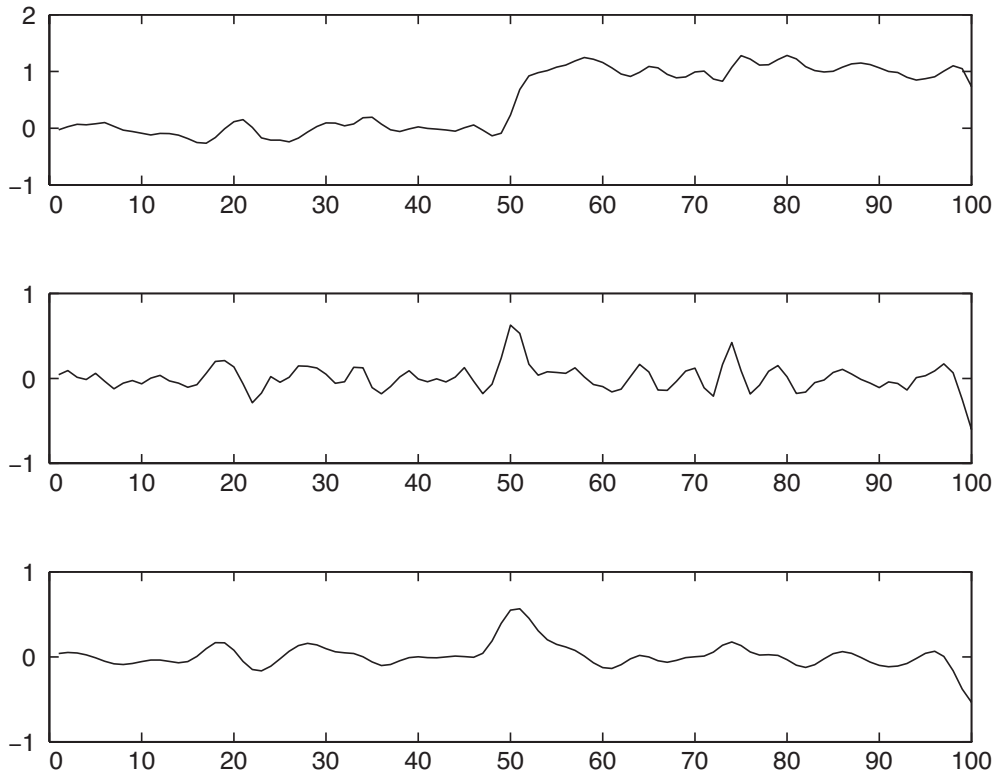
Image of stapler on desk



Edges detected in image (left)

- As figures show, edge detection does not result in ideal line drawing
  - Gaps in edges; noise edges that do not correspond to anything significant in the image
  - These can be corrected for by further processing

# Edge Detection



- How are edges detected?
- Consider profile of image brightness along 1D cross-section at 90° to an edge, e.g. edge between left end of desk and wall in stapler photo
- Looks like top graph at left
- Notice edges correspond to locations in image where brightness changes sharply

- Suggests differentiating image and looking for places where derivative  $I'(x)$  is large (middle graph above)
- Almost works – peak at 50 – but spurious peaks elsewhere (75) due to noise
- Solution is to smooth image first (bottom graph above)

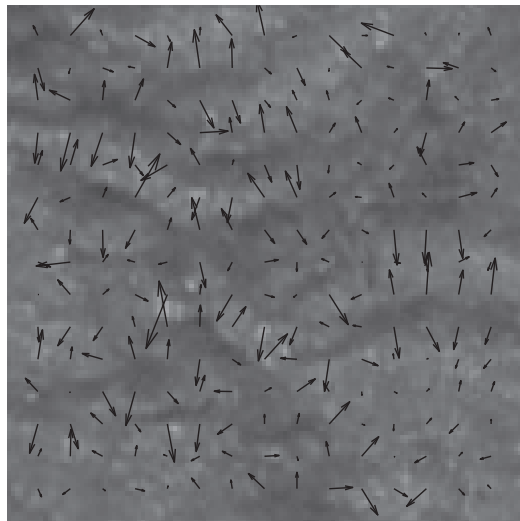
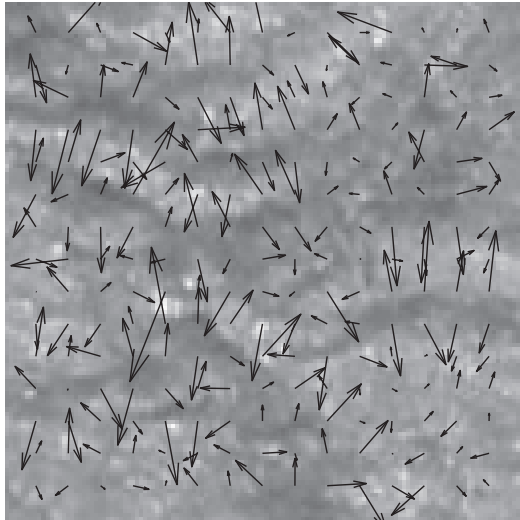
# Edge Detection

- Removing noise:
  - Noise arises from statistical fluctuations in the behaviour of the sensor (pixels in CCDs absorb photons and emit electrons – this is a physical process and measurement of it is bound to involve noise/irregularities)
  - Can model noise as Gaussian probability distribution, with each pixel independent of the others
    - Can smooth image by assigning to each pixel average of its neighbours
    - How many neighbours? – 1 pixel away, 2?, 3?
    - Good approach is to use weighted average that weights nearest pixels more and decrease weights for more distant pixels
    - **Gaussian filter** does this (“Gaussian Blur” in Photoshop)
  - Gaussian filter replaces intensity  $I(x_0, y_0)$  with sum over all  $(x, y)$  pixels of  $I(x, y)N_\sigma(d)$ , where  $d$  is distance from  $(x_0, y_0)$  to  $(x, y)$ 
    - This operation is called **convolution** and is written  $I * N_\sigma$

# Edge Detection

- Algorithm can be generalized from one to two dimensions
- Compute gradient of intensity function  $I(x,y)$
- Magnitude of gradient signals presence of edge
- Direction of gradient gives edge **orientation** at any point
- Again, smooth image by convolving with Gaussian before computing gradient
- Given gradient, obtain edges by
  - Finding edge points
    - Look at nearby points backwards and forwards along direction of gradient – pick point with largest gradient magnitude above suitable threshold (otherwise not an edge point)
  - Linking edge points together
    - Assume any two neighbouring edge points with consistent orientation belong to same edge curve

# Texture



- In everyday use **texture** is the visual feel of a surface
- In computational vision **texture** refers to a “spatially repeated pattern that can be sensed visually”
  - E.g. stitches on sweater, spots on leopard, blades of grass on a lawn, pebbles on beach
- Texture only makes sense as a property of a multi-pixel patch
  - Can detect by calculating orientation (direction of intensity gradient) of each point in a patch and then computing histogram of orientations for the patch



# Texture

- Edge orientations/texture largely invariant to changes illumination
  - Useful clue for object recognition, since other clues, e.g. edges, yield different results in different lighting conditions
- Edge detection does not work as well for textured objects as it does for smooth objects
  - Edges get lost amongst texture elements
- Solution: look for changes in texture, as looked for changes in brightness
  - Changes in orientation histograms suggest boundaries between patches
  - E.g. patch on a tiger and patch on grassy background will have very different orientation histograms

# Optical Flow



- **Optical flow** is the apparent motion in a video image sequence that arises when videoing a moving object or when the camera is moving relative to an object
  - Optical flow measures direction and speed of motion of features **in the image** not in the scene
- Optical flow gives useful information about scene structure
  - E.g. in video from moving train distant object have slower apparent motion than close objects, so rate of apparent motion can tell us about distance

# Optical Flow

- Optical flow vector field represented at each point  $(x,y)$  by components  $v_x(x,y)$  and  $v_y(x,y)$  in  $x$  and  $y$  directions respectively
- To measure flow need to find corresponding points between frames
- Simple idea: image patches around corresponding points have similar intensity patterns
- So compare block of pixels centered at pixel  $p$ ,  $(x_0, y_0)$  at time  $t_0$  with pixel blocks centered at various candidate pixels  $(x_0 + D_x, y_0 + D_y)$  at time  $t_0 + D_t$
- One similarity measure is sum of squared differences:

$$SSD(D_x, D_y) = \sum_{(x,y)} (I(x, y, t) - I(x + D_x, y + D_y, t + D_t))^2$$

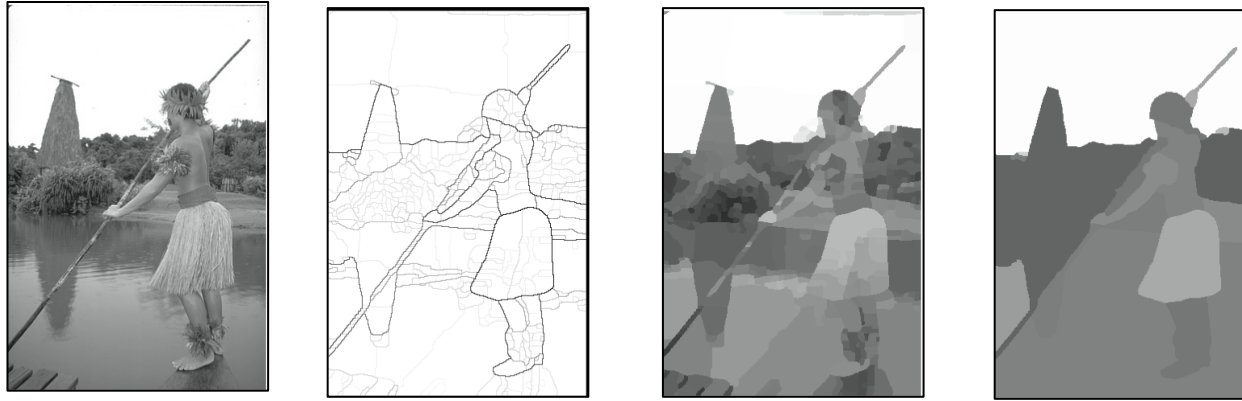
where  $(x,y)$  ranges over pixels in the block centered at  $(x_0, y_0)$

- Search for  $(D_x, D_y)$  that minimize SSD
- Optical flow at  $(x_0, y_0)$  is then  $(v_x, v_y) = (D_x/D_t, D_y/D_t)$

# Segmentation of Images

- **Segmentation** is the process of dividing an image into regions of similar pixels
- Within an object/region properties of pixels – e.g. brightness, colour, texture tend to be similar
  - Across inter-object boundary there is typically a large change in one or more of these properties
- Two main approaches to segmentation:
  1. Detect boundaries of regions
  2. Detect regions themselves

# Segmentation of Images



## 1. Detect boundaries of regions

- E.g. use a trained classifier to determine whether a boundary splits a disk-shaped patch along a particular diameter by comparing features of two sides of disk (2<sup>nd</sup> figure from left above)

## 2. Detect regions themselves

- E.g. use clustering algorithm to group pixels with similar properties (rightmost 2 figures above)

# References

- Snowden, Robert, Thompson, Peter and Troschianko, Tom (2012) Basic Vision: An Introduction to Visual Perception (revised edition). Oxford.
- Szeliski, Richard (2011). Computer Vision: Algorithms and Applications. Springer.
- Russell, Stuart and Norvig, Peter (2010) Artificial Intelligence: A Modern Introduction (3<sup>rd</sup> ed). Pearson. Chapter 24.
- Wikipedia: Computer Vision. [http://en.wikipedia.org/wiki/Computer\\_vision](http://en.wikipedia.org/wiki/Computer_vision) (visited 01/11/15).