Computer Vision

Professor John Daugman

University of Cambridge

Computer Science Tripos, Part II
Lent Term 2015/16

Outline of Lectures

1. **Overview. Goals of computer vision; why they are so difficult.**
2. **Image sensing, pixel arrays, CCD cameras. Image coding.**
3. **Biological visual mechanisms, from retina to visual cortex.**
4. **Mathematical operations for extracting structure from images.**
5. **Edge detection operators; gradients; zero-crossings of Laplacian.**
6. **Multi-resolution. Active Contours. Wavelets as primitives; SIFT.**
7. **Higher brain visual mechanisms; streaming; reciprocal feedback.**
8. **Texture, colour, stereo, and motion descriptors. Disambiguation.**
9. **Lambertian and specular surface properties. Reflectance maps.**
10. **Shape description. Codons; superquadrics and surface geometry.**
11. **Perceptual organisation and cognition. Vision as model-building.**
12. **Lessons from neurological trauma and deficits. Visual illusions.**
13. **Bayesian inference. Classifiers; probabilistic decision-making.**
14. **Model estimation. Machine learning and statistical methods.**
15. **Optical character recognition. Content-based image retrieval.**
16. **Face detection, face recognition, and facial interpretation.**

Aims of this course:

– to introduce the principles, models and applications of computer vision, as well as some mechanisms used in biological visual systems that might inspire design of artificial ones. At the end of the course you should:

- understand visual processing from both “bottom-up” (data oriented) and “top-down” (goals oriented) perspectives;
- be able to decompose visual tasks into sequences of image analysis operations, representations, algorithms, and inference principles;
- understand the roles of image transformations and their invariances;
- describe detection of features, edges, shapes, motion, and textures;
- describe some key aspects of how biological visual systems work;
- consider ways to try to implement biological visual strategies in computer vision, despite the enormous differences in hardware;
- be able to analyse the robustness, brittleness, generalisability, and performance of different approaches in computer vision;
- understand roles of machine learning in computer vision, including probabilistic inference, discriminative and generative methods;
- understand in depth at least one major vision application domain, such as face detection, recognition, or interpretation.

Recommended books and online resources

- Forsyth, D.A. & Ponce, J. (2003). *Computer Vision: A Modern Approach*.
- Shapiro, L. & Stockman, G. (2001). *Computer Vision*. Prentice Hall.
- Duda, R.O., Hart, P.E., & Stork, D.G. (2001) *Pattern Classification* (2nd Ed).

- CVonline: "Evolving, Distributed, Non-Proprietary, On-Line Compendium of Computer Vision" (Univ. of Edinburgh; updated Aug. 2015; includes many Wikipedia links): http://homepages.inf.ed.ac.uk/rbf/CVonline/
- Matlab Functions for Computer Vision and Image Processing (updated July 2015): http://www.peterkovesi.com/matlabfns/index.html
- Annotated Computer Vision Bibliography (updated 1 Jan. 2016): http://iris.usc.edu/Vision-Notes/bibliography/contents.html
- A collection of **Written Exercises** for this course (past Tripos Questions) is provided on the course website, with weekly assignments. These will be reviewed in a series of **Examples Classes** (within the lecture slots).
- A collection of **Practical Exercises** for this course developed by C Richardt, T Baltrusaitis, and L Swirski is provided here: http://www.cl.cam.ac.uk/~ls426/computervision/

1. Examples of computer vision applications and goals:

- automatic face recognition, and interpretation of facial expression
- tracking of persons and objects; pose estimation; gesture recognition

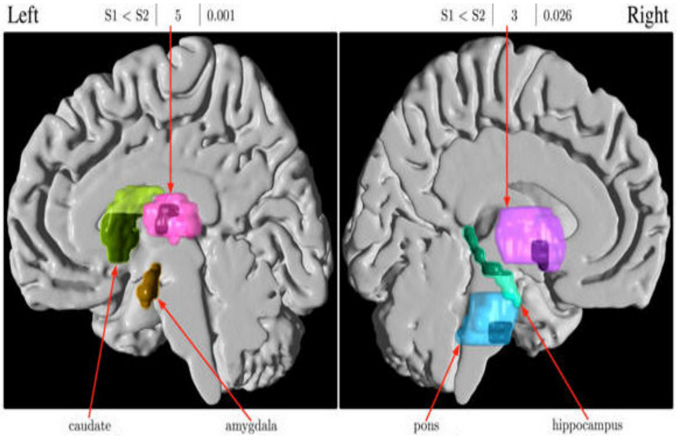
- object and pattern recognition; 3D scene reconstruction from images
- biometric-based visual determination of personal identity
- image search and content-based image retrieval; scene understanding

(some computer vision applications and goals, con't)

- vision-based autonomous robots; driverless cars
- motion estimation; collision avoidance; depth and surface inference

(some computer vision applications and goals, con't)

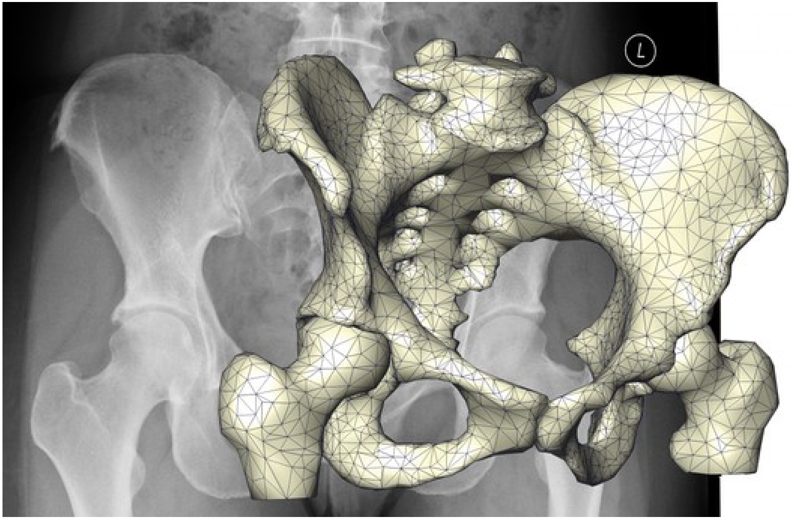
- 3D assessment of tissue and organs from non-invasive scanning
- automated medical image analysis, interpretation, and diagnosis



- neural/computer interface; interpretive prostheses for the blind
- optical character recognition (OCR): recognition of handwritten or printed characters, words, or numbers; e.g. car registration plates

(some computer vision applications and goals, con't)

- 3D reconstruction from radiological scans, and design of prostheses

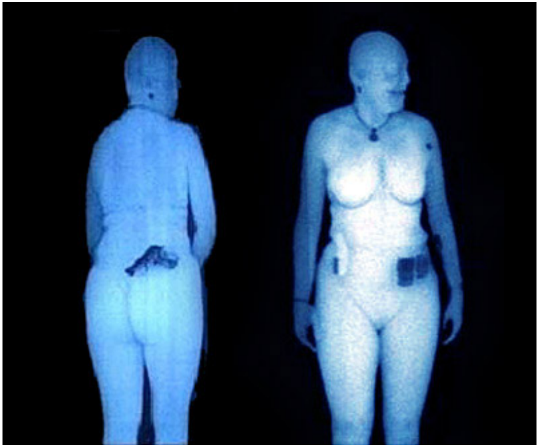


(some computer vision applications and goals, con't)

- robotic manufacturing: manipulation and assembly of parts
- agricultural robots: weeding, harvesting, and grading of produce

(some computer vision applications and goals, con't)

- anomaly detection; event detection; automated surveillance and security screening of passengers at airports



1(b). Why the goals of computer vision are so difficult

In many respects, computer vision is an "AI-complete" problem. Building general-purpose vision machines would entail, or require, solutions to most of the general goals of artificial intelligence:

- it would require finding ways of building flexible and robust visual representations of the world;
- maintaining and updating them, with machine learning;
- and interfacing the representations with attention, goals and plans.

Like other problems in AI, the challenge of vision can be described in terms of building a *signal-to-symbol converter.* The external world presents itself only as physical signals on sensory surfaces (such as a camera, retina, microphone...), which *explicitly* express very little of the information required for intelligent understanding of the environment.

These signals must be converted ultimately into symbolic representations whose manipulation allows the machine or organism to understand and to interact intelligently with the world.

(Why the goals of computer vision are so difficult, con't)

Although vision seems like such an effortless, immediate faculty for humans and other animals, it has proven to be exceedingly difficult to automate. Some of the reasons for this include the following:

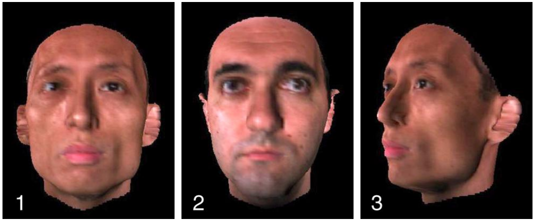
1. An image is a two-dimensional optical projection, but the world we wish to make sense of visually is three-dimensional. In this respect, vision is *"inverse optics:"* we must invert the 3D $\rightarrow$ 2D projection in order to recover world properties (object properties in space); but the 3D $\leftarrow$ 2D inversion of such a projection is, strictly speaking, mathematically impossible: there is no unique solution.

   In another respect, vision is *"inverse graphics:"* graphics begins with a 3D world description (in terms of object and illuminant properties, viewpoint, etc.), and "merely" computes the resulting 2D image, with its occluded surfaces, shading, gradients, perspective, etc. Vision has to perform exactly the inverse of this process!

   A classic example in computer vision is face recognition. Humans perform this task effortlessly, rapidly, reliably, and unconsciously.

(Why the goals of computer vision are so difficult, con't)

(We don't even know quite how we do it; like so many tasks for which our neural resources are so formidable, we have little "cognitive penetrance" or understanding of how we actually perform face recognition.) Consider these three facial images *(from Pawan Sinha, MIT, 2002)*:



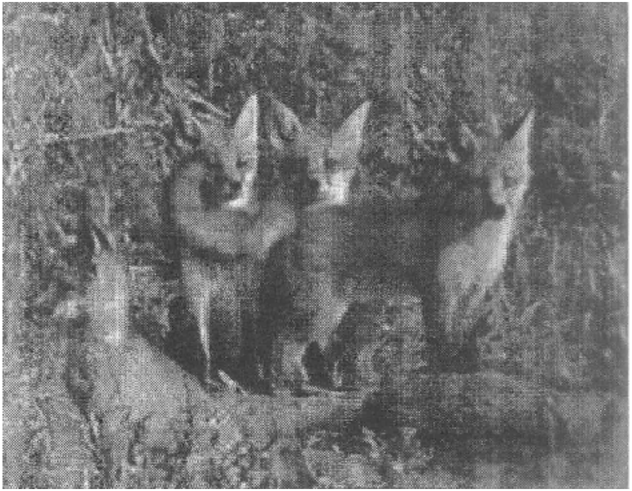
Which two pictures show the same person?

Unlike humans, classical computer vision algorithms would select 1 and 2 as the same person, since those images are more similar than 1 and 3.

However, recently remarkable progress has been made towards achieving good pose-invariant face recognition with Google's "FaceNet", based on a convolutional neural network and "deep learning" from a huge database of hundreds of millions of labelled example face images, in different poses.

(Why the goals of computer vision are so difficult, con't)

2. Few visual tasks can be performed in a purely data-driven way ("bottom-up" image analysis). Consider this image: the foxes are well camouflaged by their textured backgrounds; the foxes occlude each other; they appear in different poses, perspective angles; etc.



(Why the goals of computer vision are so difficult, con't)

The image of foxes was intentionally noisy, grainy, and monochromatic, in order to highlight how remarkable is the fact that we (humans) can easily process and understand the image despite such impoverished data.

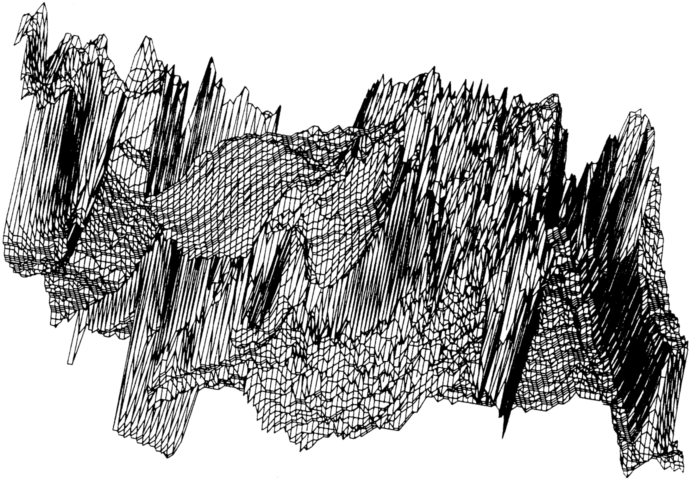
How can there possibly exist mathematical operators for such an image that can, despite its poor quality:

- perform the figure-ground segmentation of the scene (into its objects, versus background clutter)
- infer the 3D arrangements of objects from their mutual occlusions
- infer surface properties (texture, colour) from the 2D image statistics
- infer volumetric object properties from their 2D image projections
- and do all of this in "real time?" (This matters quite a lot in the natural world, "red in tooth and claw", since survival depends on it.)

Here is a video demo showing that computer vision algorithms can infer 3D world models from 2D (single) images, and navigate within them:
`http://www.youtube.com/watch?v=VuoljANz4EA` .

(Why the goals of computer vision are so difficult, con't)

Consider now the actual image data of a face, shown as a pixel array with greyscale value plotted as a function of (x,y) pixel coordinates. Can you see the face in this image, or even segment the face from its background, let alone recognise the face? In this format, the image reveals both the complexity of the problem and the poverty of the signal data.



(Why the goals of computer vision are so difficult, con't)

This "counsel of despair" can be given a more formal statement:

3. Most of the problems in vision are *ill-posed,* in Hadamard's sense that a *well-posed* problem must have the following set of properties:

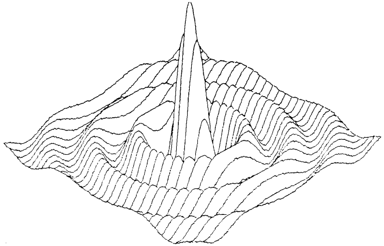
- its solution exists;
- its solution is unique;
- its solution depends continuously on the data.

Clearly, few of the tasks we need to solve in vision are well-posed problems in Hadamard's sense. Consider for example these tasks:

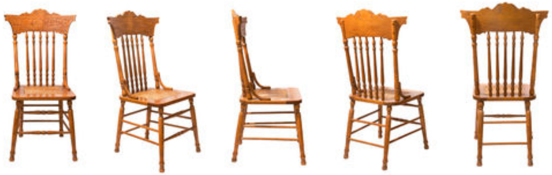
- infering depth properties from an image
- infering surface properties from image properties
- infering colours in an illuminant-invariant manner
- infering structure from motion, shading, texture, shadows, ...

(Why the goals of computer vision are so difficult, con't)

- inferring a 3D shape unambiguously from a 2D line drawing:

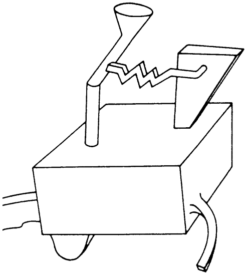


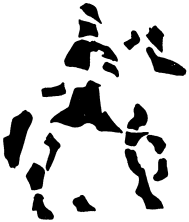
- interpreting the mutual occlusions of objects, and stereo disparity
- recognising a 3D object regardless of its rotations about its three axes in space (e.g. a chair seen from many different angles): pose-invariant recognition



(Why the goals of computer vision are so difficult, con't)

- understanding an object that has never been seen before:





For a chess-playing robot, the task of visually identifying an actual chess piece in 3D (*e.g.* a knight, with pose-invariance and "design-invariance") is a much harder problem than *playing* chess! (The latter problem was solved years ago, and chess-playing algorithms today perform at almost superhuman skill levels; but the former problem remains barely solved.)

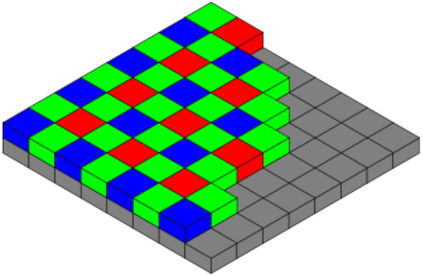
...but enough counsel of despair. Let us begin with understanding what an image array is.

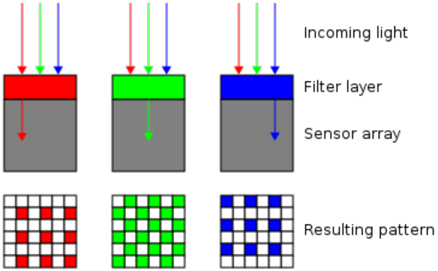
2. Image sensing, pixel arrays, CCD cameras, image coding

- A CCD video camera contains a dense array of independent sensors, which convert incident photons focused by the lens onto each point into a charge proportional to the light energy there.
- The local charge is "coupled" (hence CCD) capacitively to allow a voltage (V=Q/C) to be read out in a sequence scanning the array.
- The number of pixels (picture elements) ranges from a few 100,000 to many millions (*e.g.* 6 MegaPixel) in an imaging array that is about 1 cm^2 in size, so each pixel sensing element is only about 3 microns in width.
- The photon flux into such small catchment areas is a factor limiting further increases in resolution by simply building denser imaging arrays. Note also that 3 microns is only six times larger than the wavelength of a photon of light in middle of the visible spectrum (yellow $\sim$ 500 nanometers or nm), so quantum mechanics already limits the further resolution possible in sensors sized about 1 cm^2.

(Image sensing, pixel arrays, CCD cameras, con't)

- Spatial resolution of the image is thus determined both by the density of elements in the CCD array, and by the properties of the lens which is forming the image: optical figure-of-merit.
- Luminance resolution (the number of distinguishable grey levels) is determined by the number of bits per pixel resolved by the digitizer, and by the inherent signal-to-noise ratio of the CCD array.
- Colour information arises (conceptually if not literally) from three separate CCD arrays preceded by different colour filters, or mutually embedded as Bayer subpopulations within a single CCD array:





Data in video streams

Composite video uses a high-frequency "chrominance burst" to encode colour; or in S-video there are separate "luma" and "chroma" signals; or there may be separate RGB colour channels. Colour information requires much less resolution than luminance; some coding schemes exploit this.

A framegrabber or a strobed sampling block in a digital camera contains a high-speed analogue-to-digital converter which discretises this video signal into a byte stream, making a succession of frames.

Conventional video formats include NTSC (North American standard): 640×480 pixels, at 30 frames/second (actually there is an interlace of alternate lines scanned out at 60 "fields" per second); and PAL (European, UK standard): 768×576 pixels, at 25 frames/second.

Note what a vast flood of data is a video stream, even without HDTV:

768×576 pixels/frame $\times$ 25 frames/sec $=$ 11 million pixels/sec. Each pixel may be resolved to 8 bits in each of the three colour planes, hence 24×11 million $=$ 264 million bits/sec. How can we possibly cope with this data flux, let alone understand the objects and events it encodes?

Image formats and sampling theory

Images are represented as rectangular arrays of numbers (1 byte each), sampling the image intensity at each pixel position. A colour image may be represented in three separate such byte arrays called "colour planes", containing red, green, and blue components as monochromatic images. An image with an oblique edge within it might include this array:

0	0	0	1	1	0
0	0	1	2	10	0
0	1	2	17	23	5
0	3	36	70	50	10
1	10	50	90	47	12
17	23	80	98	85	30

There are many different image formats used for storing and transmitting images in compressed form, since raw images are large data structures that contain much redundancy (e.g. correlations between nearby pixels) and thus are highly compressible. Different formats are specialised for compressibility, manipulability, or for properties of browsers or devices.

Examples of image formats and encodings

- `.jpeg` - for compression of continuous-tone and colour images, with a controllable "quality factor". Tiles of Discrete Cosine Transform (DCT) coefficients are quantised, with frequency-dependent depth.
- `.jpeg2000` - a superior version of `.jpeg` implemented with smooth Daubechies wavelets to avoid block quantisation artifacts.
- `.mpeg` - a stream-oriented, compressive encoding scheme used for video and multimedia. Individual image frames are `.jpeg` compressed, but an equal amount of temporal redundancy is removed by inter-frame predictive coding and interpolation.
- `.gif` - for sparse binarised images; 8-bit colour. Very compressive; favoured for websites and other bandwidth-limited media.
- `.png` - using lossless compression, the portable network graphic format supports 24-bit RGB.
- `.tiff` - A complex umbrella class of tagged image file formats. Non-compressive; up to 24-bit colour; randomly embedded tags.
- `.bmp` - a non-compressive bit-mapped format in which individual pixel values can easily be extracted. Non-compressive.

(Image formats and sampling theory, con't)

- Various colour coordinates are used for "colour separation", such as HSI (Hue, Saturation, Intensity), or RGB, or CMY vector spaces.
- Regardless of the sensor properties and coding format, ultimately the image data must be represented pixel by pixel. For compressed formats, the image payload is actually in a (Fourier-like) transform domain, and so to retrieve an array of numbers representing image pixel values, essentially an inverse transform must be performed on the compressive transform coefficients.
- Typically a monochromatic image is resolved to 8 bits/pixel. This allows 256 different intensity values for each pixel, from black (0) to white (255), with shades of grey in between.
- A full-colour image may be quantised to this depth in each of the three colour planes, requiring a total of 24 bits per pixel. However, it is common to represent colour more coarsely, or even to combine luminance and chrominance information in such a way that their *total* information is only 8 or 12 bits/pixel.

(Image formats and sampling theory, con't)

- How much information does an image contain? Bit count does not relate to optical properties, nor to frequency analysis.
- Nyquist's Sampling Theorem says that the highest *spatial frequency* component of information contained in an image equals one-half the sampling density of the pixel array.
- Thus a pixel array with 640 columns can represent spatial frequency components of image structure no higher than 320 cycles/image.
- Likewise, if image frames are sampled in time at 30 per second, then the highest *temporal frequency* component of information contained within a moving sequence is 15 Hertz.
- Because quantised image information is thus fundamentally discrete, the operations from calculus which we might want to perform on an image, like differentiation (to find edges) or integration (to perform convolutions or transforms), must be done in their discrete forms.
- The discrete form of a derivative is a finite difference. The discrete form of an integral is a (suitably normalised) summation. But it is commonplace to represent such operations using their (usually 2D) notations from continuous mathematics: $\frac{d}{dx}$, ∇^2, and $\iint dx\,dy$.

3. Biological visual mechanisms: retina to visual cortex