Computer Vision

Computer Science Tripos (Pt II, II-Gen, Dipl), **Lent Term**

16 Lectures by J G Daugman

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 primary cortex.**
4. **Mathematical operations for extracting structure from images.**
5. **Edge detection operators; the Laplacian and its zero-crossings.**
6. **Scale-space; multi-resolution. Wavelets as visual primitives.**
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. **Infering 3D shape from shading: surface geometry. Codons.**
11. **Perceptual psychology and cognition. Vision as model-building.**
12. **Lessons from neurological trauma and deficits. Visual illusions.**
13. **Bayesian inference in vision. Classifiers; probabilistic methods.**
14. **Object-centred coords. Appearance-based vs model-based vision.**
15. **Vision as a set of inverse problems. Regularisation.**
16. **Case study: face detection and recognition; facial interpretation.**

Syllabus for Computer Vision

Aims

The aims of this course are to introduce the principles, models and applications of computer vision, as well as some mechanisms used in biological visual systems that may inspire design of artificial ones. The course will cover: image formation, structure, and coding; edge and feature detection; neural operators for image analysis; texture, colour, stereo, motion; wavelet methods for visual coding and analysis; interpretation of surfaces, solids, and shapes; data fusion; probabilistic classifiers; visual inference and learning; and face recognition.

Lectures

- Goals of computer vision; why they are so difficult. How images are formed, and the ill-posed problem of making 3D inferences from them about objects and their properties.
- Image sensing, pixel arrays, CCD cameras, framegrabbers. Elementary operations on image arrays; coding and information measures.
- Biological visual mechanisms from retina to cortex. Photoreceptor sampling; receptive field profiles; spike trains; channels and pathways. Neural image encoding operators.
- Mathematical operators for extracting image structure. Finite differences and directional derivatives. Filters; convolution; correlation. 2D Fourier domain theorems.
- Edge detection operators; the information revealed by edges. The Laplacian operator and its zero-crossings. Logan's Theorem.
- Scale-space, multi-resolution representations, causality. Wavelets as visual primitives.
- Higher level visual operations in brain cortical areas. Multiple parallel mappings; streaming and divisions of labour; reciprocal feedback through the visual system.
- Texture, colour, stereo, and motion descriptors. Disambiguation and the achievement of invariances.
- Lambertian and specular surfaces. Reflectance maps. Discounting the illuminant when infering 3D structure and surface properties.
- Inferring 3D shape from shading: surface geometry. Boundary descriptors; Fundamental Theorem of Curves; codons.
- Perceptual psychology and visual cognition. Vision as model-building and graphics in the brain. Learning to see.
- Lessons from neurological trauma and visual deficits. Visual illusions and what they may imply about how vision works.
- Bayesian inference in vision; knowledge-driven interpretations. Classifiers. Probabilistic methods in vision.
- Object-centred coordinates. Solid parameterisation and superquadrics. Appearance-based *versus* volumetric model-based vision.
- Vision as a set of inverse problems; mathematical methods for solving them: energy minimisation, relaxation, regularisation.
- Approaches to face detection, face recognition, and facial interpretation.

Objectives

At the end of the course students 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, specific algorithms, and inference principles
- understand the roles of image transformations and their invariances in pattern recognition and classification
- be able to analyse the robustness, brittleness, generalisability, and performance of different approaches in computer vision
- be able to describe key aspects of how biological visual systems encode, analyse, and represent visual information
- be able to think of ways in which biological visual strategies might be implemented in machine vision, despite the enormous differences in hardware
- understand in depth at least one important application domain, such as face recognition, detection, or interpretation

Recommended book

* Shapiro, L. & Stockman, G. (2001). *Computer Vision.* Prentice Hall.

1 Overview. Goals of computer vision; why they are so difficult.

Computer vision seeks to generate intelligent and useful descriptions of visual scenes and sequences, and of the objects that populate them, by performing operations on the signals received from video cameras.

Some examples of computer vision applications and goals:

- automatic face recognition, and interpretation of expression
- visual guidance of autonomous vehicles
- automated medical image analysis, interpretation, and diagnosis
- robotic manufacturing: manipulation, grading, and assembly of parts
- OCR: recognition of printed or handwritten characters and words
- agricultural robots: visual grading and harvesting of produce
- smart offices: tracking of persons and objects; understanding gestures
- biometric-based visual identification of persons
- visually endowed robotic helpers
- security monitoring and alerting; detection of anomaly
- intelligent interpretive prostheses for the blind
- tracking of moving objects; collision avoidance; stereoscopic depth
- object-based (model-based) compression of video streams
- general scene understanding

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, and interfacing them 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 videocamera, 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 interact intelligently with the world.

Although vision seems like such an effortless and immediate faculty for humans and other animals, it has proven 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 need to invert the 3D $\longrightarrow$ 2D projection in order to recover world properties (object properties in space); but the 2D $\longrightarrow$ 3D inversion of such a projection is, strictly, mathematically impossible.

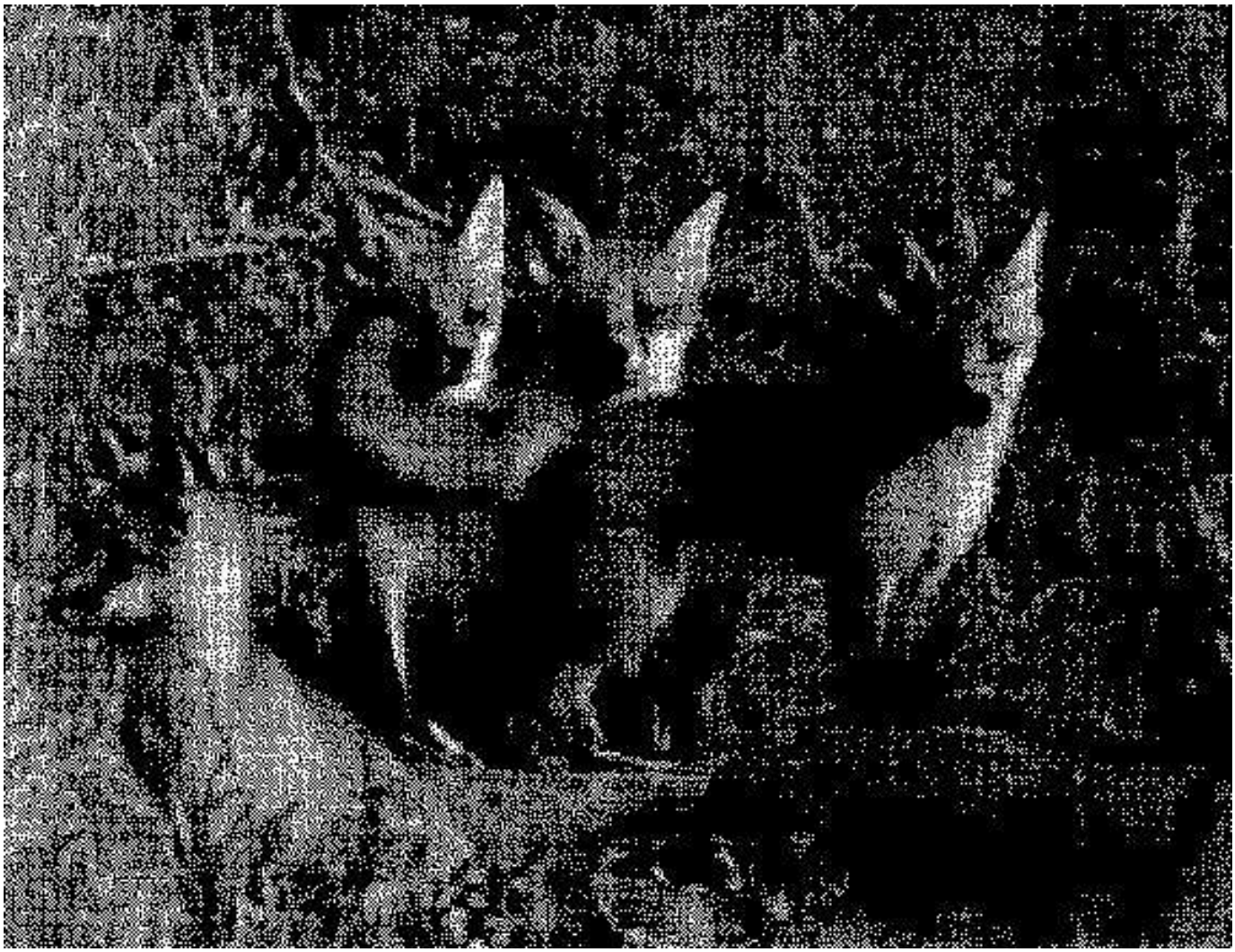
   In another respect, vision is *"inverse graphics:"* graphics begins with a 3D world description (in terms of object and illuminant properties, viewpoint, etc.), and "merely" computes the resulting 2D image, with its occluded surfaces, shading and shadows, gradients, perspective, etc. Vision has to perform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

   A classical and central problem in computer vision is face recognition. Humans perform this task effortlessly, rapidly, reliably, and unconsciously. (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?

   Most current computer algorithms select 1 and 2 as the same person, since those images are more similar than 1 and 3.

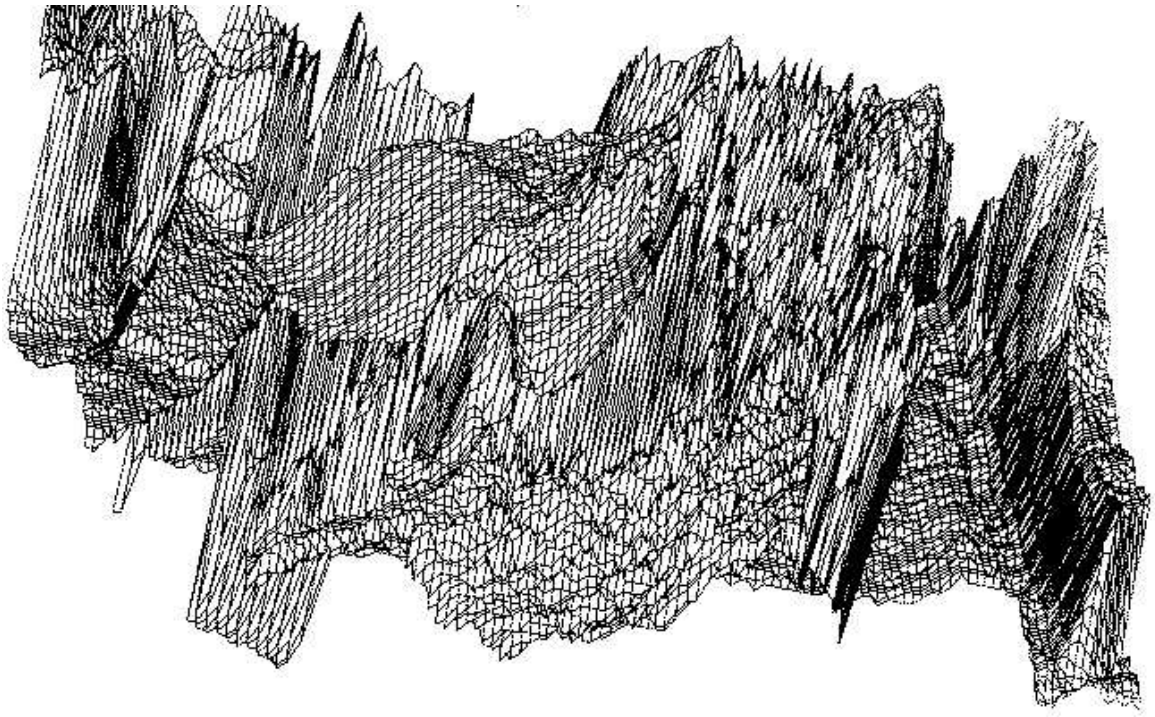
2. Very few visual tasks can be successfully performed in a purely data-driven way ("bottom-up" image analysis). Consider the next image example: the foxes are well camouflaged by their textured backgrounds; the foxes occlude each other; they appear in several different poses and perspective angles; etc. How can there possibly exist mathematical operators for such



an image that can:

- perform the figure-ground segmentation of the scene (into its objects and background)
- 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.)

Consider now the actual image data of a face, shown as a pixel array with luminance 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 recognize the face? In this form, the image reveals both the complexity of the problem and the poverty of the data.



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

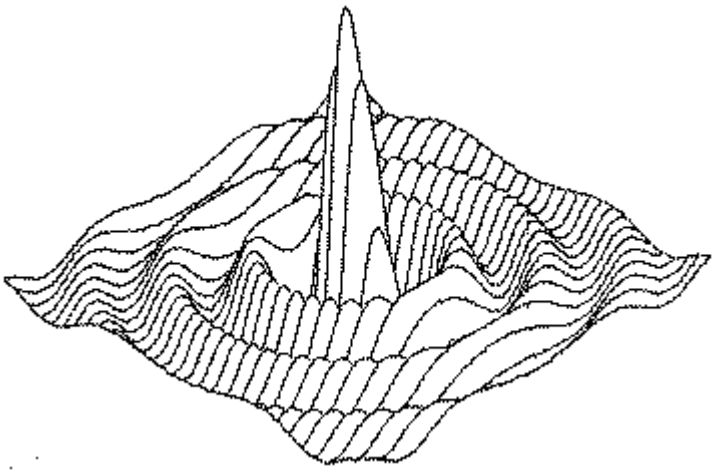
Most of the problems we need to solve in vision are *ill-posed,* in Hadamard's sense that a *well-posed* problem must have the following set of properties:

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

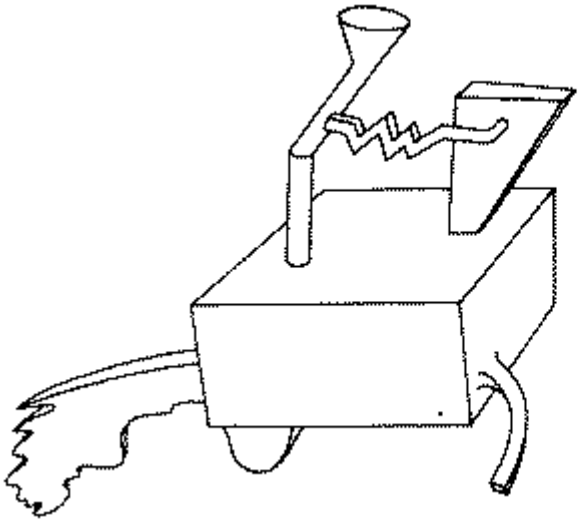
Clearly, few of the tasks we need to solve in vision are well-posed problems in Hadamard's sense. Consider for example the problems of:

- 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, ...

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



- interpreting the mutual occlusions of objects, and stereo disparity
- recognizing a 3D object regardless of its rotations about its three axes in space (e.g. a chair seen from many different angles)
- understanding an object that has never been seen before:



- etc. ...

...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) is typically about 442,000 within a grid that is 1/3" x 1/3" in size, so each pixel sensor is a mere 13 microns across (0.013 mm).

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. 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 subpopulations within a single CCD array. It is encoded into the camera output either as a high-frequency "chrominance burst" (to be separately demodulated and decoded); or else signaled on a separate channel ("luma" and "chroma" portions of an S signal); or else provided as three separate RGB colour channels (red, green, blue).

A framegrabber contains a high-speed analogue-to-digital converter (ADC) which discretizes this video signal into a byte stream. Standard pixel array formats are:

- NTSC (North American standard): 640 x 480 pixels, at 30 frames/second (actually there is an interlace of alternate lines scanned out at 60 "fields" per second)
- PAL (European, UK standard): 768 x 576 pixels, at 25 frames/second (actually alternate lines are scanned out at 50 "fields" per second)

Note what a vast quantity of data a video sequence contains! 768 x 576 pixels/frame x 25 frames/second = 11 million pixels per second. Each pixel may be resolved to 8 bits in each of the three colour planes, hence 24 x 11 million = 264 million bits per second! How can we possibly cope with this data flux, let alone understand the objects and events creating such an image stream? ...

2.1 Image formats and sampling theory

Images are represented as rectangular arrays of numbers representing image intensities at particular locations. Each element of such an array is called a pixel, for picture element. A colour image may be represented in three separate such arrays called "colour planes," containing red, green, and blue components as monochromatic images. An image with an oblique edge might look like:

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 specialized for compressibility, manipulability, or the properties of printers and browsers. Some examples:

- `.jpeg` - ideal for variable compression of continuous colour images, with a "quality factor" (typically 75) that can be specified. Useful range of DCT compression goes from 100:1 ("lossy") to about 10:1 (almost lossless).
- `.mpeg` - a stream-oriented, compressive encoding scheme used mainly for video (but also multimedia). Individual image frames are `.jpeg` compressed, but an equal amount of redundancy is removed temporally by inter-frame predictive coding and interpolation.
- `.gif` - ideal for sparse binarized images. Only 8-bit colour. Very compressive and favoured for web-browsers and other bandwidth-limited media.
- `.tiff` - a complex, full-colour format (up to 24 bits per pixel with 3 colour planes). Favoured for scanners, quality printing and publishing.
- `.bmp` - a non-compressive bit-mapped format in which individual pixel values can easily be extracted.

In addition there are varieties of colour coordinates used for "colour separation," such as HSI (Hue, Saturation, Intensity), or RGB (Red, Green, Blue), CMY, etc. But regardless of the sensor properties and coding format used, ultimately the image data must be represented numerically pixel by pixel. Typically this involves the conversion (e.g. by a tool such as `xv`) of the various compressed formats into `.bmp`, with header files for format and dimensions.

The total number of independent pixels in an image array determines the spatial resolution of the image. Independent of this is the grey-scale (or colour) resolution of the image, which is determined by the number of bits of information specified for each pixel. These separate dimensions are illustrated in the following family of images, showing the effects of differing quantization accuracies for spatial and luminance information.

It is typical for a monochromatic ("black & white") image to have resolution of 8 bits/pixel. This creates 256 different possible intensity values for each pixel, from black (0) to white (255), with all shades of grey in between. A full-colour image may be quantized 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.

Because quantized 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 normalized) *summation*. However, for the sake of conceptual familiarity, it is still commonplace in computer vision to represent such operations using their usual notations from continuous mathematics, with the understanding that the operation itself (as with everything else in Computer Science!) is of course discrete.

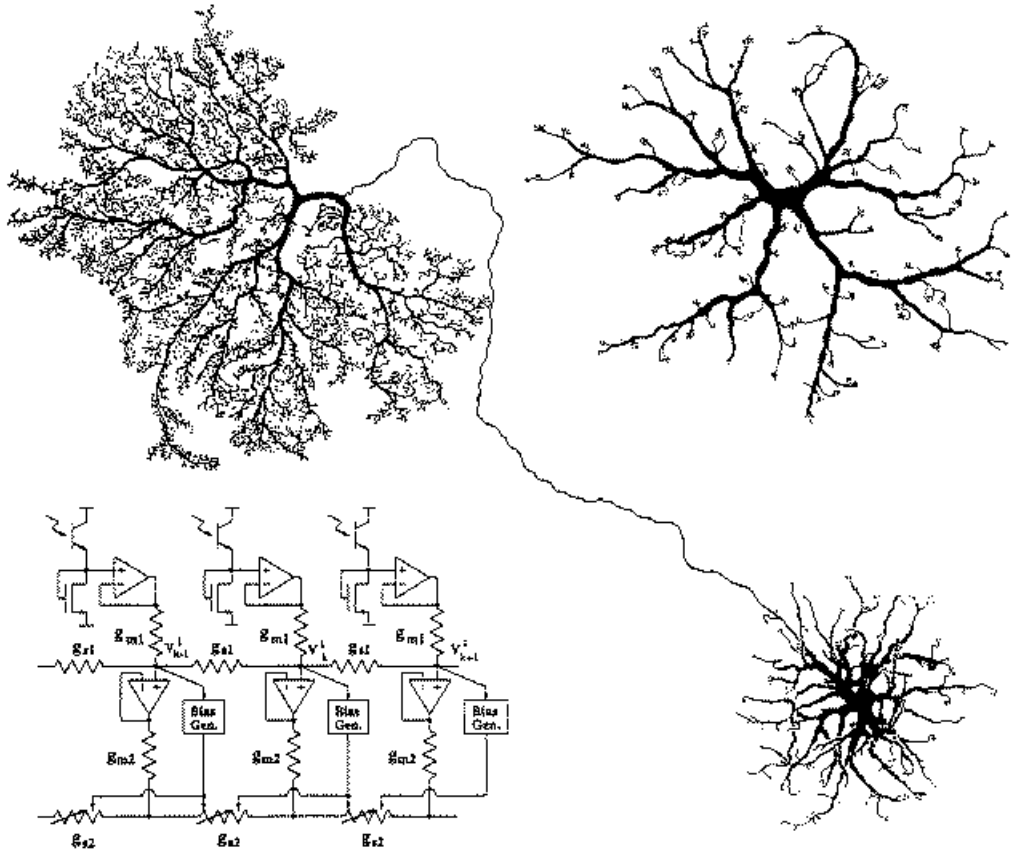
The discreteness of image arrays imposes an upper limit on the amount of information they can contain. One way to describe this is by the total bit count, but this does not relate to the optical properties of image information. A better way is through Nyquist's Theorem, which tells us that the highest *spatial frequency* component of information contained within the image is equal to one-half the sampling density of the pixel array. (Intuitively, this is because at least two samples are required to signify a single cycle of a sinewave: its peak and its trough.) Thus, a pixel array containing 640 columns can represent spatial frequency components of image structure no higher than 320 cycles/image. For the same reason, if image frames are sampled in time at the rate of 30 per second, then the highest *temporal frequency* component of information contained in the image sequence is 15 Hertz.

3 Biological visual mechanisms, from retina to primary cortex.

A strategy that has long inspired researchers in Computer Vision, whether they work on low-level problems (such as sensor design, image coding, and feature extraction), or high-level problems (such as pattern recognition, inference, and visual learning), is:

Neurobiological Visual Principles $\Longrightarrow$ Machine Vision

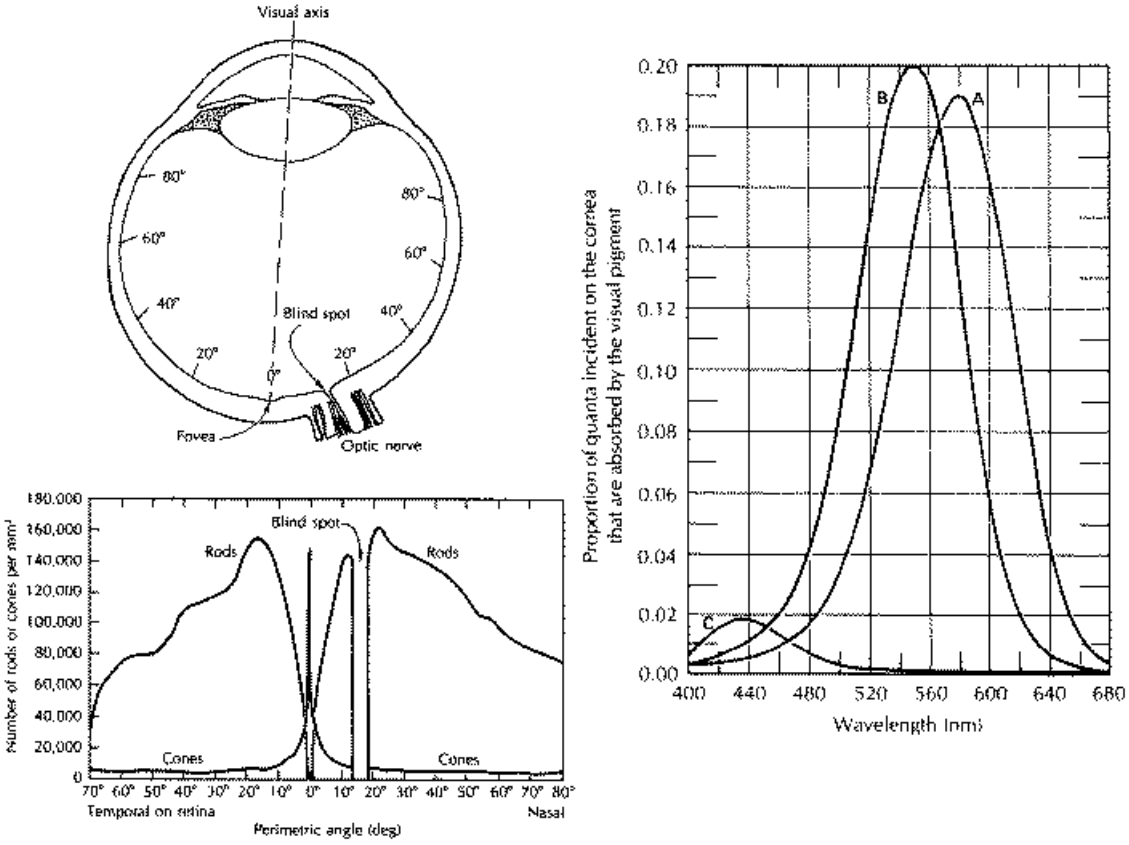
The structure of biological nervous tissue and the nature of events that occur in it are utterly different from those found in computing hardware. Yet since the only general-purpose visual systems that exist today are the biological ones, let us learn what we can from "wetware."



Neurons are sluggish but richly interconnected devices having both analogue and discrete aspects. Fundamentally they consist of an enclosing membrane that can separate electrical charge (hence there is generally a voltage difference between inside and out). The membrane is a lipid bilayer that has a capacitance of about 10,000 μFarads/cm^2, and it also has pores that are differentially selective to different ions (mainly Na^+, K^+, and Cl^-). These ion species enter or leave a neuron through protein pores studding its lipid membrane, acting as conductances (hence as resistors). The resistors for Na^+ and K^+ have the further crucial property that their resistance is not constant, but voltage-dependent. Hence as more positive ions (Na^+) flow into the neuron, the voltage becomes more positive on the inside, and this further reduces the membrane's resistance to Na^+, allowing still more to enter. This catastrophic breakdown in resistance to Na^+ constitutes a nerve impulse. Within about a msec a slower but opposite effect involving K^+ takes over, eventually restoring the original voltage. Following a short refractory period of about 2 msec during which ions are actively pumped back

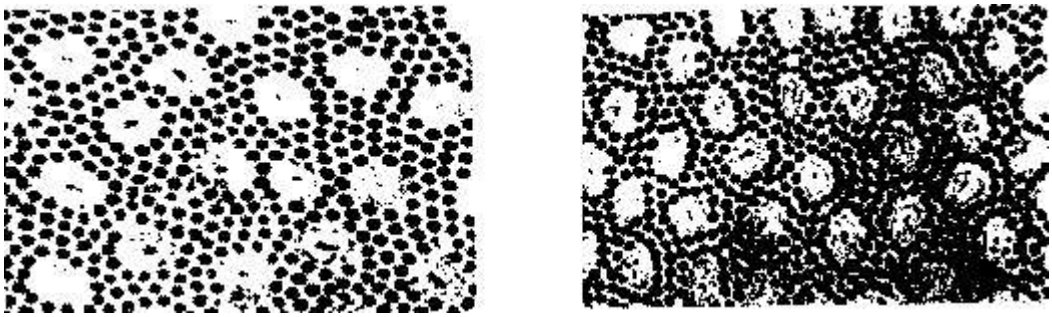
in opposite directions to reach their original electro-osmotic equilibrium concentrations, the neuron is ready for action again. Meanwhile, the impulse thus generated propagates down the axon, at a speed of about 100 m/sec. This signalling pulse can be described as discrete, but the antecedent summations of current flows into the neuron (from various influences by other neurons) which caused the catastrophic impulse are fundamentally analogue events.

Overall, the human brain contains about 100 billion neurons (10^{11}). On average each neuron may have connections with about 1,000 to 10,000 others, and so the total number of synapses (= junctions between neurons) in the brain is a staggering 10^{15}. Yet balanced against this massive connectivity, is the surprising sluggishness of neurons: as indicated above, the time course of nerve impulse generation prevents "clocking" of nerve pulses any faster than about 300 Hz. Neural activity is fundamentally asynchronous: there is no master clock on whose edges the events occur. A further contrast with computing systems is that it is rarely possible to distinguish between processing and communications, as we do in computing. In the brain, there are just impulses implementing both, by exchange of signals amongst neurons. It is not so much a hierarchical architecture as a parallel one, with reciprocal connections amongst different areas. About $2/3^{rds}$ of the brain receives visual input; we are quite fundamentally visual creatures. There are some 30 known different visual areas, of which the primary visual cortex in the occipital lobe at the back of the brain has been the most extensively studied.

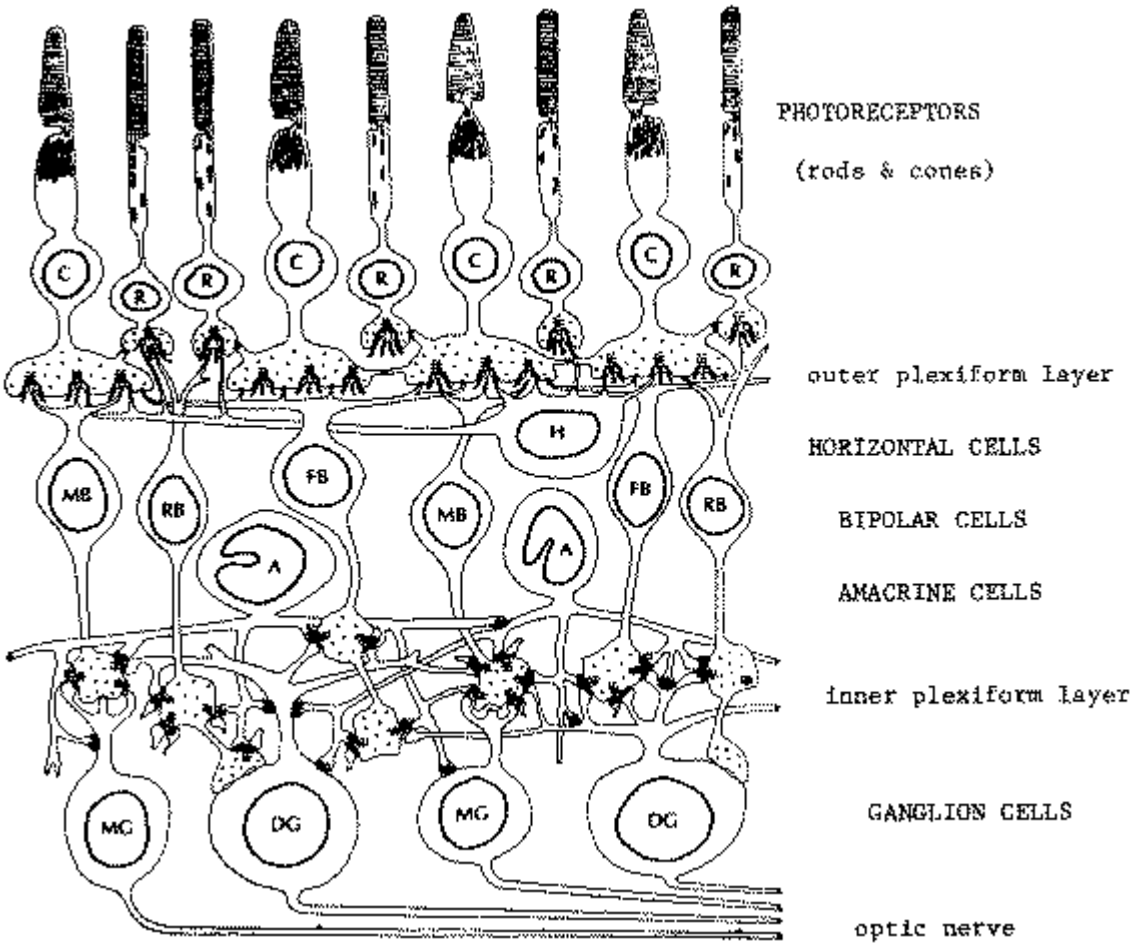


The mammalian eye is formed from a collapsed ventrical of the brain. The retina is about 1 mm thick and contains about 120 million light-sensitive photoreceptors, of which only 6 million are cones (in 3 wavelength-selective classes nominally red, blue, and green) and the vast remainder are rods which do not discriminate in wavelength. The visible spectrum of light consists of wavelengths in the range of 400nm - 700nm. Rods are specialised for much lower light intensities than cones; they subserve our "night vision" (hence the absence of perceived colour at night), and they pool together their responses (hence their much poorer spatial resolution). Cones exist primarily near the fovea, in about the central 20° (see diagram), where their responses remain individual and thus they detect with high spatial resolution. But cone

light sensitivity is much less than rods, functioning only at higher light levels, and so we really have a dual system with two barely overlapping dynamic ranges. The total dynamic range of human vision (range of light intensities that can be processed) is a staggering 10^{11} to 1. At the low end, we can reliably "see" individual photons (i.e. reliably have a visual sensation when at most a few photons reach the retina in a burst).

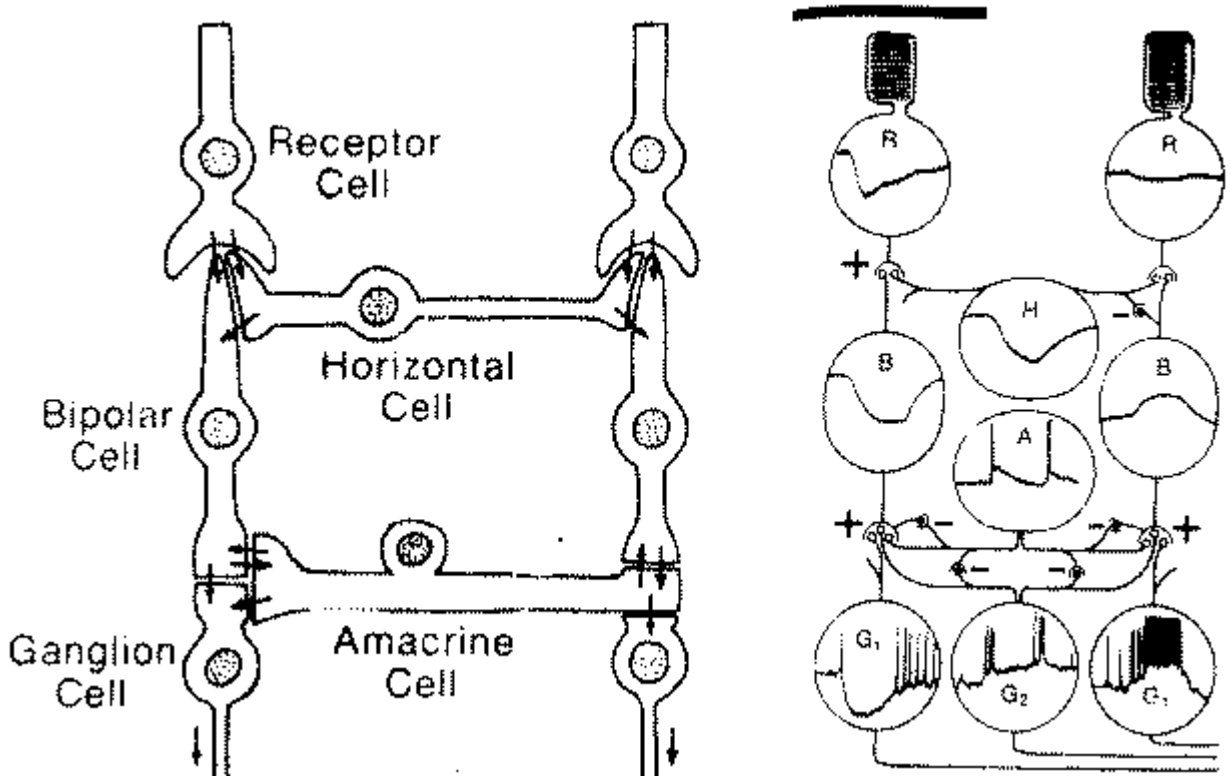


Rods and cones are distributed across the retina in jointly embedded hexagonal lattices but with varying relative densities, depending on eccentricity (distance from the fovea, measured in degrees). The hexagonal lattices are imperfect (incoherent rather than crystalline), which is believed to help prevent aliasing of high resolution information.



The retina is a multi-layered structure, containing 3 nuclear layers (of neurons) plus 2 plexiform layers (for interconnections amongst the neurons). Paradoxically, the photoreceptors are at the back, so light must first travel through all of the rest of the retina before being absorbed

by the pigments in the rods and cones. There are basically two directions of signal flows in the retina: longitudinal (photoreceptors → bipolar cells → ganglion cells); and lateral (via horizontal cells in the outer plexiform layer, and amacrine cells in the inner plexiform layer).



In only a very crude sense can one describe the retina as an "image capture" device like a camera, having analogue input phototransducers that convert photons into voltage changes, and discrete output devices that send pulses down the optic nerve. This simple view is quickly discarded by recognising that there are 120 million "input channels" (the photoreceptors, similar in a sense to pixels), but only 1 million "output channels" (the axons of the ganglion cells which constitute the optic nerve). Clearly the retina is already doing a lot of processing of the image, and it sends its coded results to the brain: not merely a raw converted image array. The retina *is a part* of the brain.

The nature of retinal signal processing might be summarised as:

- image sampling by photoreceptor transducers, with pooling of signals from rods
- spatial centre-surround comparisons implemented by bipolar cells (direct central input from photoreceptors, minus surround inhibition via horizontal cells)
- temporal differentiation by amacrine cells, subserving motion sensitivity
- separate coding of "sustained" versus "transient" image information by different classes of ganglion cells (large receptive fields ⇔ transient; small fields ⇔ sustained)
- initial colour separation by "opponent processing" channels (yellow vs blue; red vs green) coupled sometimes with spatial opponency (on-centre, off-surround)
- generation of nerve impulse spikes in a parallel temporal modulation code on the 1 million fibres of the optic nerve from each eye (= 2nd Cranial Nerve)

There is both convergence (fan-in) and divergence (fan-out) of signals through the retina:

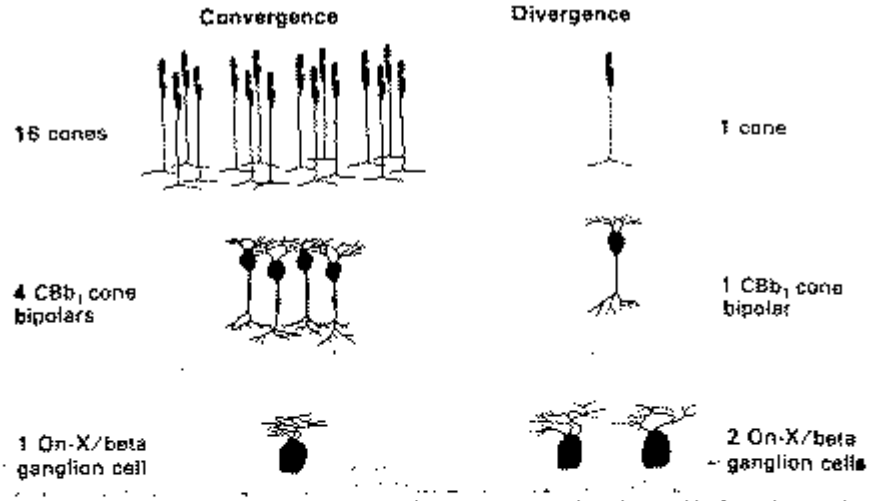
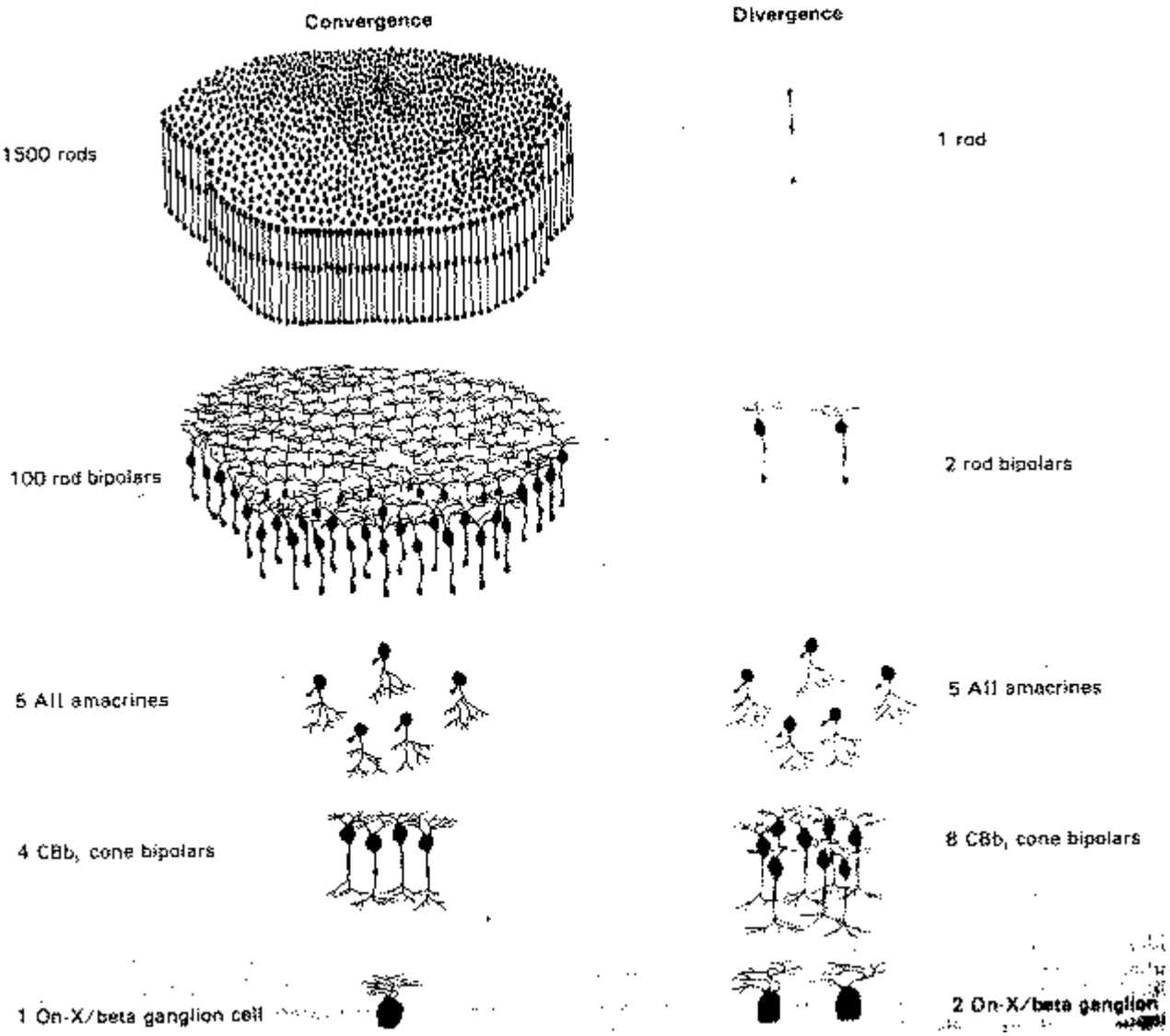


Fig. 5. Actual convergence and divergence through successive arrays along the cone bipolar pathway to the ON-beta ganglion cell in the area centralis. In this circuit connections between successive arrays are much narrower than their potential connections. Text describes structure–function relationships of this circuit.



3.1 Receptive field structure in the retina

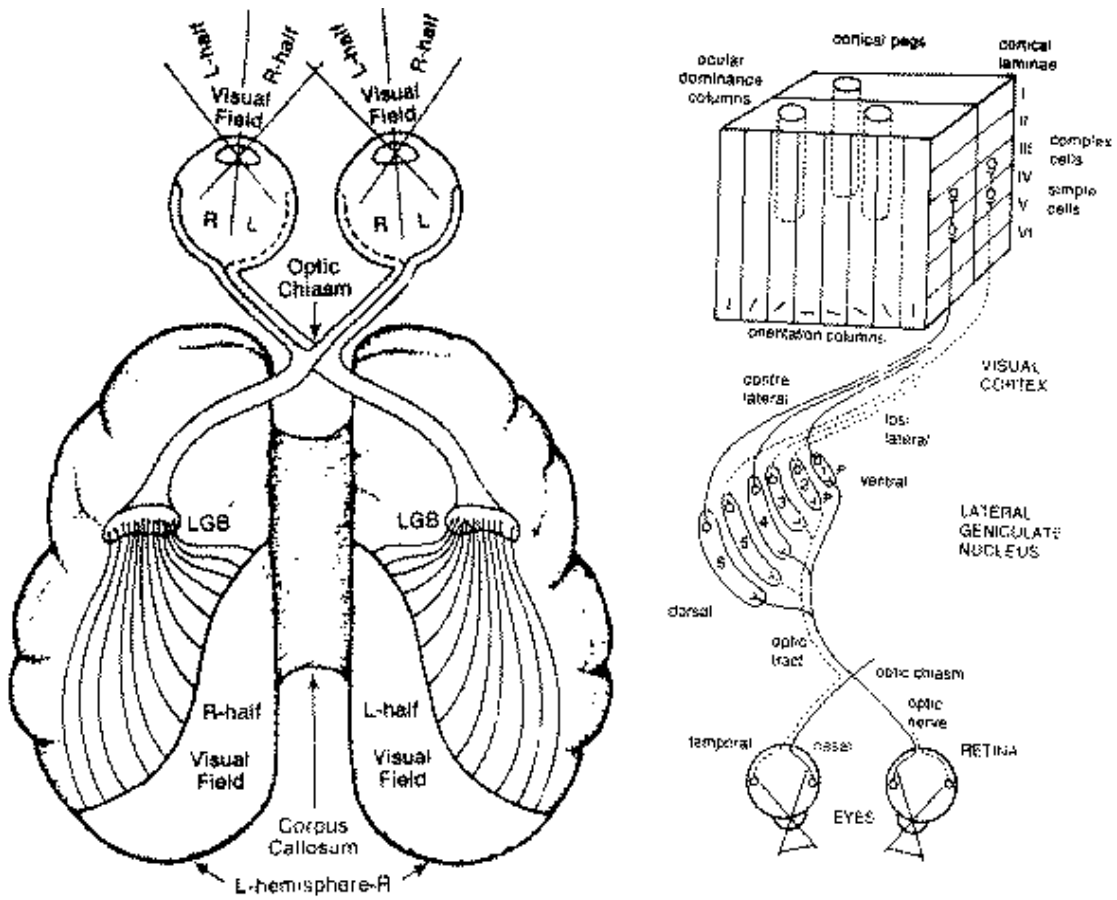
The spatial structuring of exhitatory and inhibitory influences amongst neurons in the retina gives them their properties as image operators. Similarly for the temporal structure of their interactions. In both space and time, retinal neurons can thus be described as filters; and to the extent that they act as linear devices (having the properties of proportionality and superposition of responses to components of stimuli), their behaviour can be fully understood (and even predicted for arbitrary images) through Fourier analysis and the other tools of linear systems analysis. An important aspect of retinal receptive fields – as distinct from those found in most neurons of the visual cortex – is that their spatial structure is isotropic, or circularly symmetric, rather than oriented.

- Photoreceptors respond to light by hyperpolarising (the voltage across the cell membrane becomes more negative inside, for vertebrates; the opposite is true for invertebrates). Their "receptive field" is just their own cross-section for absorbing light, a small disk about 3 μ in diameter on the human retina, about a minute of visual arc.
- Horizontal cells pool together the responses from large numbers of photoreceptors within a local area. With these "surround" signals, they inhibit bipolar cells (hence the name).
- Bipolar cells are the first to have a "centre-surround" receptive field structure: their response to light in a central disk is opposite from their response to light in the local surrounding area. Field boundaries are circular and roughly concentric (i.e. annular).
- Amacrine cells are "on-off" in temporal, as opposed to spatial, terms.
- Ganglion cells combine these spatial and temporal response properties and thus serve as integro-differential image operators with specific scales and time constants. Moreover they convert their responses to impulses in a spike frequency code, traveling down their axons which are the fibres of the optic nerve to the thalamus and thence on to the primary visual cortex in the brain.

3.2 Visual cortical architecture and receptive field structure

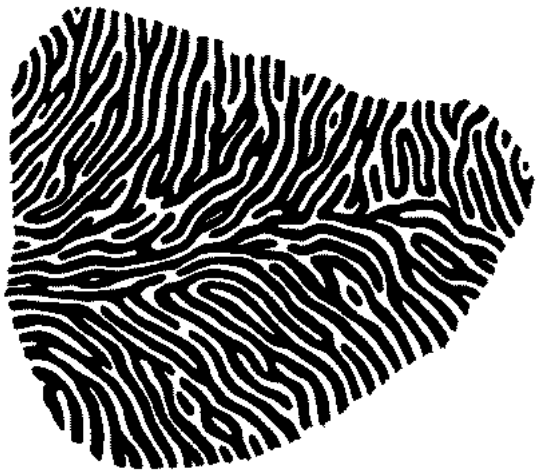
The optic nerve from each eye splits into two halves at the optic chiasm, each portion continuing on to only one of the two cerebral hemispheres of the brain. The optic nerve portion containing signals from the nasal half of each retina crosses over to project only to the contralateral (opposite) brain hemisphere; whereas the optic nerve portion bearing signals from the temporal half of each eye projects only to the ipsilateral (same side) brain hemisphere. Since the optical image on each retina is inverted, this means that the left-half of the visual world (relative to the point of gaze fixation) is directly "seen" only by the right brain; and the right-half of the visual world only by the left brain. It is almost interesting to ask why we don't see some kind of seam going down the middle... (Ultimately the two brain hemispheres share all of their information via a massive connecting bundle of 500 million commissural fibres called the corpus callosum.)

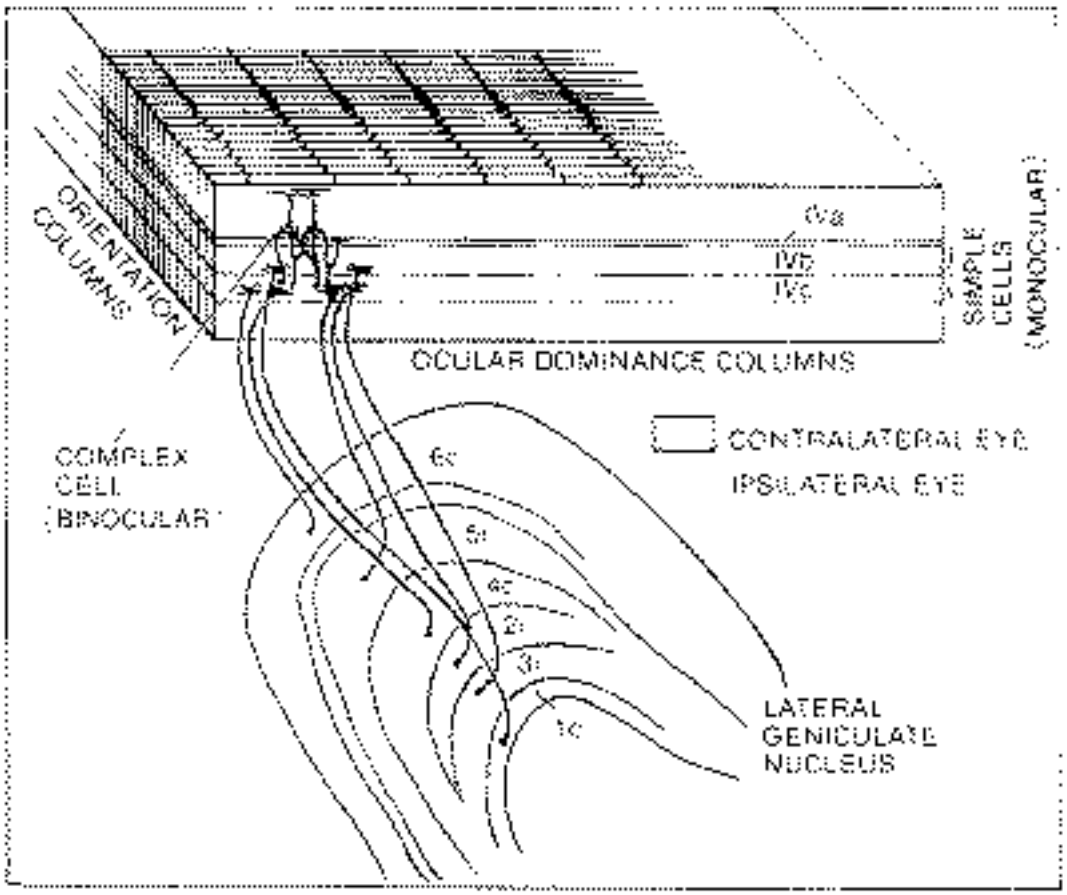
The optic nerve projections to each visual cortex pass first to a 6-layered structure called the lateral geniculate nucleus (LGN), in a polysensory organ of the midbrain called the thalamus. It is an intriguing fact that this so-called "relay station" actually receives 3 times more descending (efferent) fibres projecting back down from the cortex, as it does ascending (afferent) fibres from the eyes. Could it be that this confluence compares cortical feedback representing hypotheses about the visual scene, with the incoming retinal data in a kind of predictive coding or hypothesis testing operation? Several scientists have proposed that "vision is graphics" (i.e. what we see is really our own internally generated 3D graphics, modelled to fit the 2D retinal data, with the model testing and updating occuring here in the thalamus).



Schematic of the central visual pathway in the human. (from Popper and Eccles, 1977)

The right-eye and left-eye innervations from each LGN to the primary visual cortex in the occipital lobe of that hemisphere are inter-woven into "slabs," or columns, in which neurons receive input primarily from just one of the eyes. These ocular dominance columns have a cycle of about 1 mm and resemble fingerprints, as seen in the following figures. Clearly each hemisphere is trying to integrate together the signals from the two eyes in a way suitable for stereoscopic vision, by computing the relative retinal disparities of corresponding points in the two images. The disparities reflect the relative positions of the points in depth, as we will study later with stereoscopic visual algorithms.

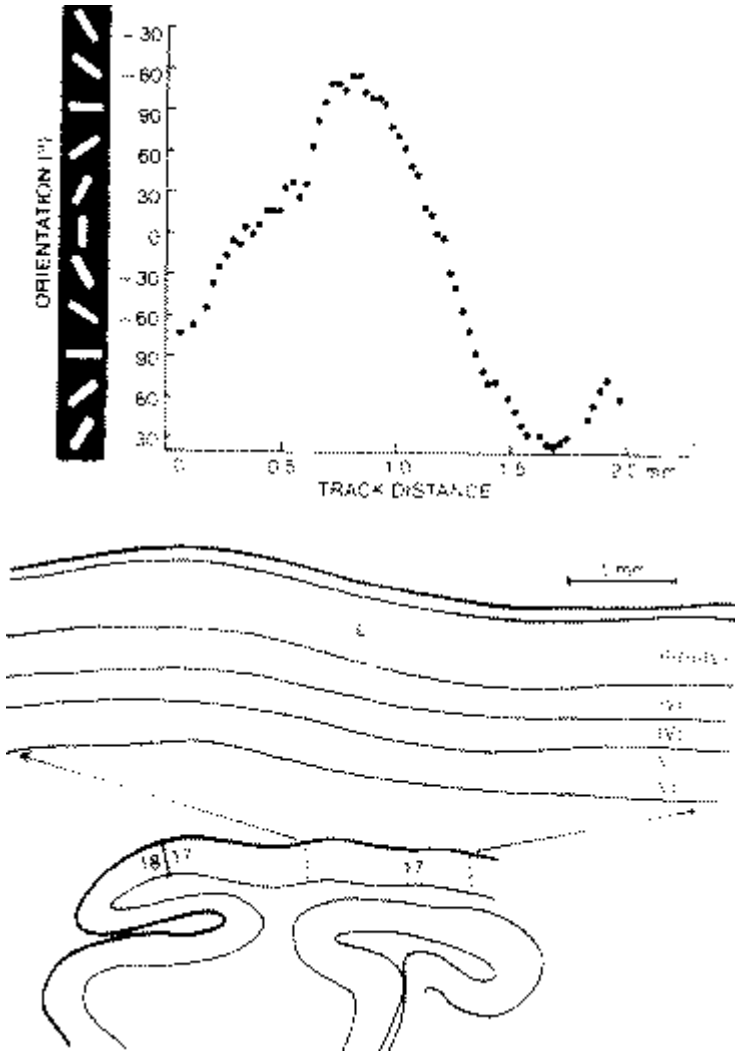




RELATION BETWEEN OCULAR DOMINANCE AND ORIENTATION COLUMNS. Scheme in which the ocular dominance and orientation columns run at right angles to each other. An example of a complex cell is shown in an upper layer, receiving its inputs from two simple cells that lie in two neighboring ocular dominance columns, but share the same orientation column. (From Hubel and Wiesel, 1972)

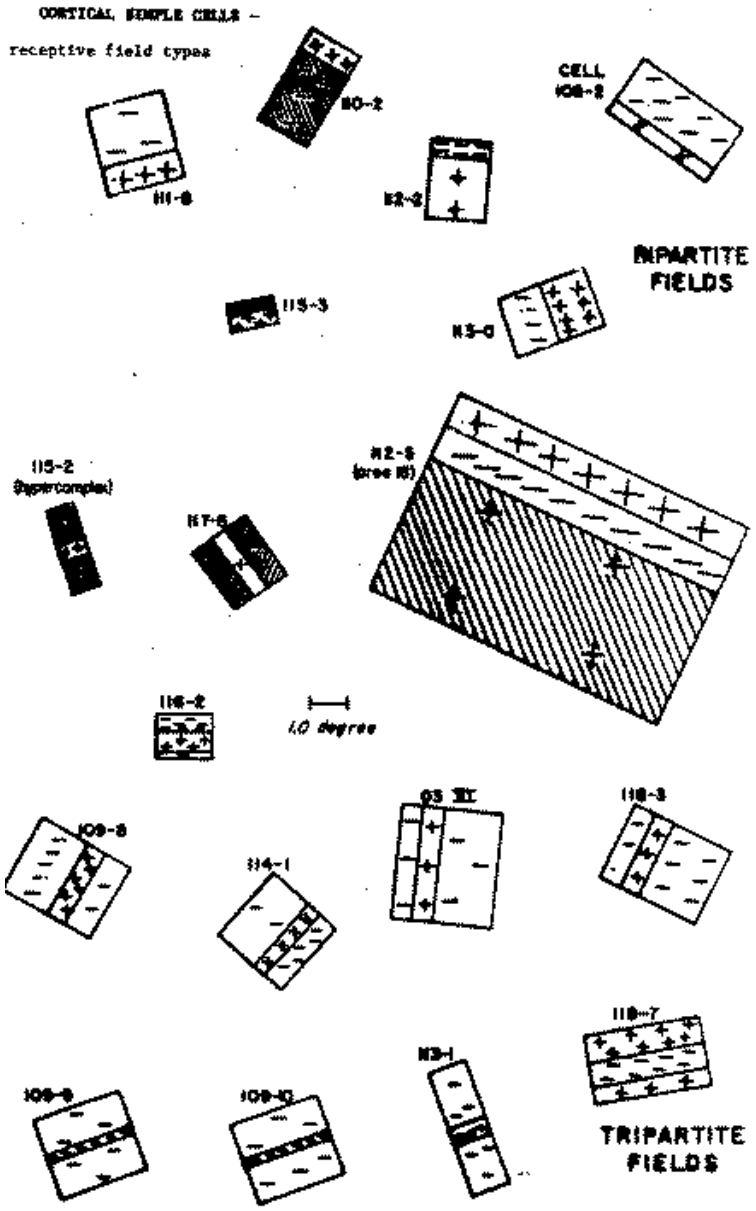
ANATOMICAL CONFIRMATION of ocular-dominance columns came from various staining methods and from axonal-transport autoradiographs such as those shown in color on page 85. This composite autoradiograph visualizing the pattern over an area some 10 millimeters wide was made by cutting out and pasting together the regions representing layer IV in a number of parallel sections: the one in bottom illustration on page 85 and others at different depths

Orthogonal to the ocular dominance columns in the cortical architecture, there runs a finer scale sequence of orientation columns. Neurons in each such column respond only to image structures that have a certain preferred orientation (such as bars or edges). The columns form a regular sequence of systematically changing preferred orientations. This is one of the most crystalline properties seen in visual cortical architecture:



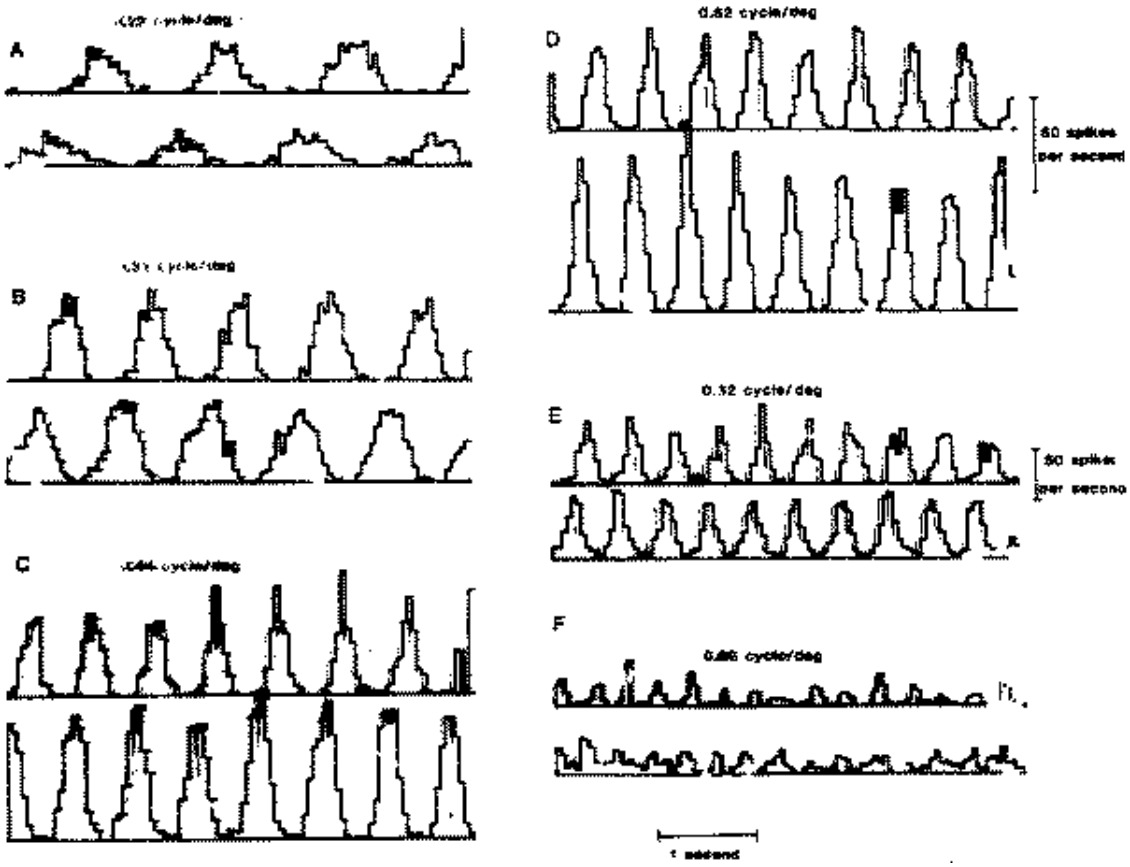
When individual neurons in the visual cortex are probed with microelectrodes during light stimulation of the retina, their functional properties are revealed by demarcating the region of visual space over which they respond (as indicated by a change in their firing rate). Areas where they are excited by light are indicated by + marks; areas where light inhibits them are indicated by − marks. Their plotted receptive fields then seem to reveal 5 main spatial "degrees of freedom:"

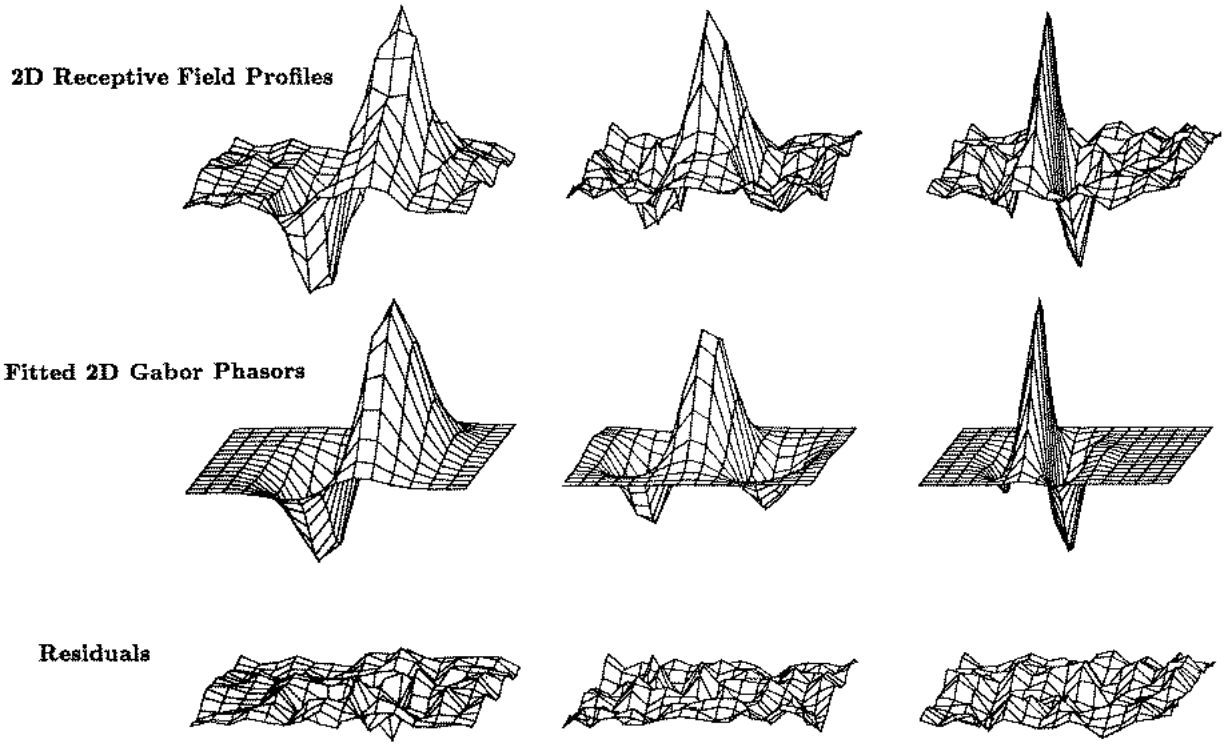
1. Position of their receptive field in visual space, both horizontally...
2. ...and vertically;
3. Size of their receptive field;
4. Orientation of the boundaries between excitatory and inhibitory regions;
5. Phase, or symmetry of the receptive field (bipartite or tripartite types).



The phase variable is particularly revealing. By recording from adjacent pairs of neurons simultaneously, with a kind of "double-barrelled" micro-electrode, it was found that neurons having the same receptive field location, the same field size and the same orientation preference, actually had a quadrature phase relationship. Adjacent neurons would form pairs whose modulated receptive field structure showed a 90° spatial phase offset. Several examples of such quadrature pairs of cortical visual neurons are shown in the following spike histograms recorded in response to a drifting sinusoidal luminance grating.

Finally, by plotting the actual amount by which a neuron is excited or inhibited by light, as a function of the coordinates of the stimulus within its receptive field, we obtain a 2D function called its receptive field profile. These turn out, for about 97% of the neurons, to be very closely described as 2D Gabor wavelets (or phasors). Some examples of empirically measured profiles are shown in the top row of the lower figure; the ideal theoretical form of such a wavelet (which we will define later) is shown in the middle row; and the difference between these two functions in the bottom row; the differences are nearly nil and statistically insignificant. So, it appears that the visual cortex of the brain evolved a knowledge of the valuable properties of such wavelets for purposes of image coding and analysis!


A
B
C
D
0.52 cycle/deg
E
0.32 cycle/deg
F
0.86 cycle/deg
50 spikes per second
1 second


2D Receptive Field Profiles
Fitted 2D Gabor Phasors
Residuals

4 Mathematical operations for extracting structure from images.

Nearly all image processing and feature encoding operations are based to some extent on the theory of Fourier Analysis.

Even if the operations never actually require computing a Fourier transform, their underlying principles and concepts (such as scale; edge or motion energy; filtering; directional derivative; textural signature; statistical structure; etc.) must be understood at least partially in "spectral" (i.e. Fourier) terms.

In addition to this explanatory role, Fourier analysis can be used directly to construct useful visual representations that are invariant under translation (change in position), rotation, and dilation (change in size). This is therefore the representation underlying many pattern classification and recognition applications, such as optical character recognition (OCR).

Finally, even many operations in pattern recognition that might not seem related in any way to Fourier analysis, such as computing correlations, convolutions, derivatives, differential equations, and diffusions, are much more easily implemented in the Fourier domain. (Powerful algorithms like the FFT make it easy to go back and forth rapidly between the image and Fourier domains).

For all of these reasons, we will review some principles and techniques of Fourier analysis with a view to understanding some of the basic operations in computer vision. Applications include edge detection operators, analysis of motion, texture descriptors, and wavelet-based feature detectors.

Consider an image as a black & white luminance distribution over the (x, y) plane: a real-valued (indeed, a positive-valued) two-dimensional function $f(x, y)$.

Any image can be represented by a linear combination of basis functions:

$$f(x, y) = \sum_k a_k \Psi_k(x, y) \tag{1}$$

where many possible choices are available for the expansion basis functions $\Psi_k(x, y)$. In the case of Fourier expansions in two dimensions, the basis functions are the bivariate complex exponentials:

$$\Psi_k(x, y) = \exp(i(\mu_k x + \nu_k y)) \tag{2}$$

where the complex constant $i = \sqrt{-1}$. A complex exponential contains both a real part and an imaginary part, both of which are simple (real-valued) harmonic functions:

$$\exp(i\theta) = \cos(\theta) + i \sin(\theta) \tag{3}$$

which you can easily confirm by the power-series that define the transcendental functions such as exp, cos, and sin:

$$\exp(\theta) = 1 + \frac{\theta}{1!} + \frac{\theta^2}{2!} + \frac{\theta^3}{3!} + \cdots + \frac{\theta^n}{n!} + \cdots, \tag{4}$$

$$\cos(\theta) = 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \frac{\theta^6}{6!} + \cdots, \tag{5}$$

$$\sin(\theta) = \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \frac{\theta^7}{7!} + \cdots, \tag{6}$$

(It has been said that the most remarkable and far-reaching relationship in all of mathematics is the simple yet counterintuitive "Euler Relation" implied by Eqt (3) above: $e^{i\pi} + 1 = 0$, which also contains the five most important mathematical constants, and symbolizes the subject of harmonic analysis.)

Fourier Analysis computes the coefficients a_k that yield an expansion of the image $f(x, y)$ in terms of complex exponentials:

$$f(x, y) = \sum_k a_k \exp(i(\mu_k x + \nu_k y)) \tag{7}$$

where the parameters μ_k and ν_k define the coordinates of the 2D Fourier domain. These (μ_k, ν_k) coordinates are called vector spatial frequencies, and the array of them must span the (μ, ν) Fourier plane in a uniform cartesian lattice.

It is often useful to think of the (μ, ν) Fourier plane as resolved into polar coordinates, where $\omega = \sqrt{\mu^2 + \nu^2}$ is (scalar) spatial frequency and $\phi = \tan^{-1}(\nu/\mu)$ is (scalar) orientation.

Each Fourier coefficient a_k is computed as the orthonormal projection of the entire image $f(x, y)$ onto the conjugate Fourier component $\exp(-i(\mu_k x + \nu_k y))$ associated with that coefficient:

$$a_k = \int_X \int_Y \exp(-i(\mu_k x + \nu_k y)) f(x, y) dx dy \tag{8}$$

Note that these computed Fourier coefficients a_k are complex-valued. To get a complete representation in the 2D Fourier domain for an image with n x n pixels, the number of (μ_k, ν_k) vector frequency components whose associated coefficients a_k must be computed is also n x n.

4.1 Useful Theorems of 2D Fourier Analysis

Many important steps in computer vision such as feature extraction and invariant pattern recognition depend at least partly on a small set of Fourier

theorems. We will review some main ones here, together with their direct consequences for practical computer vision applications. In every case, the input image is denoted $f(x, y)$, and its 2D Fourier transform (given by the set of computed coefficients a_k spanning the Fourier plane) is denoted by $F(\mu, \nu)$.

Shift Theorem: Shifting the original pattern in (x, y) by some 2D displacement (α, β) merely multiplies its 2DFT by $\exp(-i(\alpha\mu + \beta\nu))$. Thus the 2DFT of the shifted pattern $f(x-\alpha, y-\beta)$ is: $F(\mu, \nu)\exp(-i(\alpha\mu+\beta\nu))$.

Practical Application: The power spectrum of any isolated pattern is thus translation-invariant: it does not depend on where the pattern is located within the image, and so you don't have to find it first. The power spectrum is defined as the product of the pattern's 2DFT, $F(\mu, \nu)$, times its complex conjugate, $F^*(\mu, \nu)$, which just requires that the sign (–) of the imaginary part of $F(\mu, \nu)$ gets reversed. You can easily see that the power spectrum of the shifted pattern $f(x-\alpha, y-\beta)$, namely:

$$\exp(-i(\alpha\mu + \beta\nu))F(\mu, \nu)\exp(i(\alpha\mu + \beta\nu))F^*(\mu, \nu)$$

is equal to the power spectrum of the original unshifted pattern, namely: $F(\mu, \nu)F^*(\mu, \nu)$. Thus the power spectrum is translation-invariant.

Similarity Theorem: If the size of the original pattern $f(x, y)$ changes (shrinks/expands), say by a factor α in the x-direction, and by a factor β in the y-direction, becoming $f(\alpha x, \beta y)$, then the 2DFT of the pattern, $F(\mu, \nu)$, also changes (expands/shrinks) by the reciprocal of those factors and with similarly scaled amplitude. It becomes: $\frac{1}{|\alpha\beta|}F(\frac{\mu}{\alpha}, \frac{\nu}{\beta})$.

Rotation Theorem: If the original pattern $f(x, y)$ rotates through some angle θ, becoming $f(x\cos(\theta)+y\sin(\theta), -x\sin(\theta)+y\cos(\theta))$, then its 2DFT $F(\mu, \nu)$ also just rotates through the same angle. It becomes: $F(\mu\cos(\theta)+\nu\sin(\theta), -\mu\sin(\theta)+\nu\cos(\theta))$.

Practical Application: Size- and orientation-invariant pattern representations can be constructed by these relationships. Specifically, if the Fourier domain (μ, ν) is now mapped into log-polar coordinates (r, θ) where $r = \log(\sqrt{\mu^2+\nu^2})$ and $\theta = \tan^{-1}(\nu/\mu)$, then any dilation (size change) in the original pattern becomes simply a translation along the r-coordinate; and any rotation of the original pattern becomes simply a translation along the orthogonal θ-coordinate in this log-polar Fourier domain. But we saw earlier that translations are made immaterial by taking a power spectrum, and so these effects of dilation and rotation of the pattern are eliminated in such a representation.

Combined with the translation-invariant property of the power spectrum, we now see how it becomes possible to represent patterns in a manner that is independent of their position in the image, their orientation, and their size (i.e. the Poincaré group of transformations) These principles are routinely exploited in machine optical character recognition; in military recognition of aircraft profiles; and in "optical computing" generally.

Convolution Theorem: Let function $f(x, y)$ have 2DFT $F(\mu, \nu)$, and let function $g(x, y)$ have 2DFT $G(\mu, \nu)$. The convolution of $f(x, y)$ with $g(x, y)$, which is denoted $f * g$, combines these two functions to generate a third function $h(x, y)$, whose value at location (x, y) is equal to the integral of the product of functions f and g after one is flipped and undergoes a relative shift by amount (x, y):

$$h(x, y) = \int_{\alpha} \int_{\beta} f(\alpha, \beta) g(x - \alpha, y - \beta) d\alpha d\beta \tag{9}$$

Thus, convolution is a way of combining two functions, in a sense using each one to blur the other, making all possible relative shifts between the two functions when computing the integral of their product to obtain the output as a 2D function of these amounts of shift.

Convolution is extremely important in vision because it is the basis for filtering. It is also the essential neural operation in the brain's visual cortex, where each neurone's receptive field profile is convolved with the retinal image. In the above integral definition, if the minus (–) signs were simply replaced with (+) signs, the new expression would be the correlation integral.

The Convolution Theorem states that convolving two functions $f(x, y)$ and $g(x, y)$ together in the image domain, simply multiplies their two 2DFT's together in the 2D Fourier domain:

$$H(\mu, \nu) = F(\mu, \nu) G(\mu, \nu) \tag{10}$$

where $H(\mu, \nu)$ is the 2DFT of the desired result $h(x, y)$.

This is extremely useful as it is much easier just to multiply two functions $F(\mu, \nu)$ and $G(\mu, \nu)$ together, to obtain $H(\mu, \nu)$, than to have to convolve $f(x, y)$ and $g(x, y)$ together (if the kernel is larger than tiny, say larger than about 5 x 5) to obtain $h(x, y)$. Of course, exploiting the Convolution Theorem means going into the 2D Fourier Domain and computing the 2DFT's of $f(x, y)$ and $g(x, y)$, and then performing yet another

(inverse) FFT in order to recover $h(x, y)$ from the resulting $H(\mu, \nu)$. But with available powerful and fast 2D-FFT algorithms, this is very efficient.

Practical Application: Filtering. The starting-point of all feature extraction and image understanding operations is the filtering of an image $f(x, y)$ with some set of filters $g_k(x, y)$. Filtering is a linear operation implemented by the convolution of an image $f(x, y)$ with filter kernel(s) $g_k(x, y)$. The resulting output "image" $h_k(x, y)$ then normally undergoes non-linear operations of various kinds for image segmentation, motion detection, texture analysis, pattern recognition, and object classification.

The 2D discrete convolution of an image array with a 2D filter kernel can be represented algebraically in the following form, where the earlier continuous integrals have been replaced by discrete summations:

$$\text{result}(i, j) = \sum_m \sum_n \text{kernel}(m, n) \cdot \text{image}(i - m, j - n)$$

Simple C program for performing image convolutions

In the following simple example, the array **image** is being convolved with the (typically much smaller) array **kernel**, in order to generate a new image array **result** as the output of the convolution. (Problems with array boundaries have been ignored here for simplicity.) Discrete convolution such as illustrated here is the key operation for all image processing and front-end stages of computer vision.

```
int  i, j, m, n, sum, image[iend][jend],
     kernel[mend][nend], result [iend][jend];

for (i = mend; i < iend; i++) {
    for (j = nend; j < jend; j++) {
        sum = 0;
        for ( m = 0; m < mend; m++) {
            for ( n = 0; n < nend; n++ ) {
                sum += kernel[m][n] * image[i-m][j-n];
            }
        }
        result[i][j] = sum/(mend*nend);
    }
}
```

If we chose to perform the convolution in the Fourier domain because the kernel array is large, then of the four nested **for loops** in the C code

above, the inner two loops would be entirely eliminated. The innermost executable line inside the i and j **for loops** would simply be:

```
Result[i][j] = Kernel[i][j] * Image[i][j];
```

but the program would have to be preceded by FFTs of `kernel[i][j]` (trivial) and of `image[i][j]`, and followed by an FFT of `Result[i][j]`. Since the complexity of a 2D FFT is on the order of $n^2 \log_2(n)$ where n^2 is the number of pixels, plus n^2 multiplications in the innermost loop, the total complexity of the Fourier approach is $n^2(2\log_2(n)+1)$. In contrast, the number of multiplications in the explicit convolution above (not including all the array-addressing) is `iend*jend*mend*nend` (note that `iend*jend`$= n^2$). Hence you can calculate that the trade-off point occurs when the convolution kernel size `mend*nend` is about $\approx 2(\log_2(n)+1)$: a very small convolution kernel indeed, roughly 5 x 5 for a 512 x 512 image. For convolutions larger than this tiny one, the Fourier approach is faster.

Differentiation Theorem: Computing the derivatives of an image $f(x,y)$ is equivalent to multiplying its 2DFT, $F(\mu,\nu)$, by the corresponding frequency coordinate raised to a power equal to the order of differentiation:

$$\left(\frac{d}{dx}\right)^m \left(\frac{d}{dy}\right)^n f(x,y) \stackrel{2DFT}{\Longrightarrow} (i\mu)^m (i\nu)^n F(\mu,\nu) \tag{11}$$

A particularly useful implication of this theorem is that isotropic differentiation, which treats all directions equally (for which the lowest possible order of differentiation is 2nd-order, known as the Laplacian operator ∇^2) is equivalent simply to multiplying the 2DFT of the image by a paraboloid:

$$\nabla^2 f(x,y) \equiv \left(\frac{d^2}{dx^2} + \frac{d^2}{dy^2}\right) f(x,y) \stackrel{2DFT}{\Longrightarrow} -(\mu^2+\nu^2)F(\mu,\nu) \tag{12}$$

Practical Application: Multi-Resolution Edge Detection.

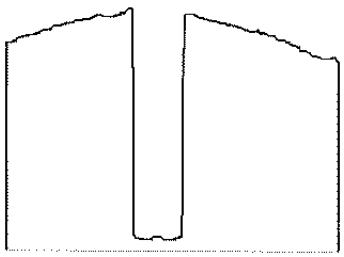
5 Edge detection operators; the information revealed by edges.

Computer vision applications invariably begin with *edge detection*, be the edges straight, curvilinear, or closed boundary contours. There are several reasons why edges are important, and why detecting the edges in a scene can be regarded as an elementary form of constructing a *signal-to-symbol converter:*

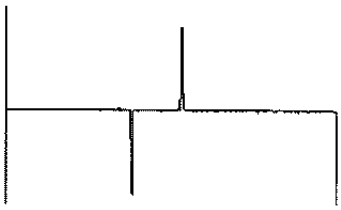
- Edges demarcate the boundaries of objects, or of material properties.
- Objects have parts, and these are typically joined with edges.

- The three-dimensional distribution of objects in a scene usually generates occlusions of some objects by other objects, and these form occlusion edges which reveal the geometry of the scene.
- Edges can be generated in more abstract domains than luminance. For example, if some image property such as colour, or a textural signature, or stereoscopic depth, suddenly changes, it forms a highly informative "edge" in that domain.
- Velocity fields, containing information about the trajectories of objects, can be organized and understood by the movements of edges. (The motions of objects in space generates velocity discontinuities at their edges.)
- The central problem of stereoscopic 3D depth vision is the "correspondence problem:" matching up corresponding regions of two images from spatially displaced cameras. Aligning edges is a very effective way to solve the correspondence problem. The same principle applies to measuring velocities (for image frames displaced in time, rather than displaced in space) by tracking edges to align corresponding regions and infer velocity (ratio of object displacement to temporal interval).

In summary, DISCONTINUITIES = INFORMATION.



Original profile



Mask = [-1 1]

Mask = [1 -2 1]

An intuitive way to find edges is to compute the derivative of a (1D) signal, as this will be large where the luminance is changing rapidly. Since image arrays

are discrete, we must use the *finite difference* representation of a derivative, and this is implemented by a *convolution*: If our (1D) luminance array is L[n] (sequence of pixels, index n), then the first-order finite difference operator (h[0],h[1])=(-1, 1) when convolved with L[n] would generate an output which is large in amplitude only where L[n] has edges (see previous figure).

However, note an important disadvantage of this approach: "rightward edges" (say, from dark to bright) generate the opposite sign from "leftward edges" (say, from bright to dark). We would prefer to generate the same detection signal regardless of the *polarity* of the edge.

A solution is to convolve the discrete luminance data L[n] instead with the *second finite difference operator*, defined as (h[-1],h[0],h[1])=(1,-2, 1) and look for the *zero-crossings* of this operator. These correspond to peaks or troughs of the first finite difference operator that we considered above, and thus they reveal the edges, regardless of their polarity. Similarly for (-1,2,-1).

In the two-dimensional case, we have the choice of using *directional derivative* operators, or *non-directional* ones. An example of a directional operator is one which integrates (sums) pixels in one direction, but differentiates (differences) them in the perpendicular direction. Clearly, such an operator will detect edges only in a specific orientation; - namely the orientation along which the integration was done. A example of such a directional edge detector is the following 3 x 3 array:

-1	2	-1
-1	2	-1
-1	2	-1

In comparison, an *isotropic* operator such as the Laplacian (sum of second derivatives in two perpendicular orientations) has no preferred direction; that is the meaning of isotropy. It will detect edges in all orientations. The next picture illustrates such an effect. A discrete approximation to the Laplacian operator ∇^2 in just a 3 x 3 array is:

-1	-2	-1
-2	12	-2
-1	-2	-1

Notice how each of these simple 3 x 3 operators sums to zero when all of their elements are combined together. These types of operators (of which there are obviously numerous other examples, differing in array sizes as well as element composition) are called *filters*, because of their spectral consequences

for favouring some spatial frequency bands and orientations at the expense of others. Their zero-sum property means that they are insensitive to the overall brightness value of a scene, as we would desire: they have "no DC term." (Their Fourier transform is equal to zero at the origin.) They also may, or may not, have a certain preferred, or characteristic *direction*; a certain phase or *symmetry* (even or odd); and a certain *scale*, defined by the spacing between changes of sign in the elements in (larger) arrays.

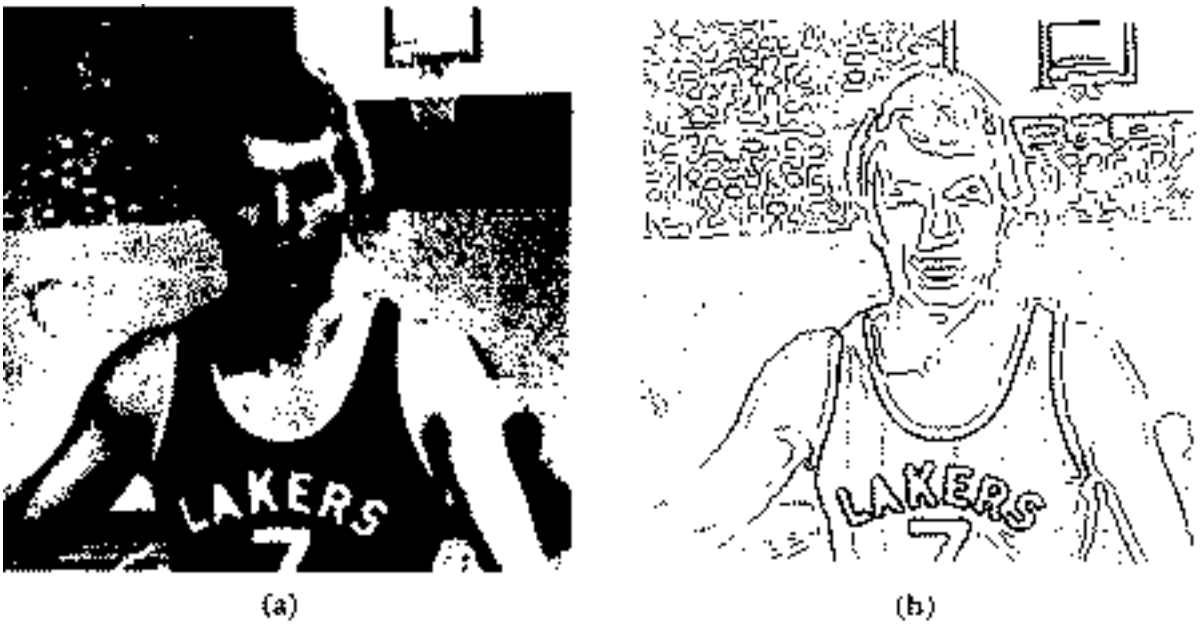


Figure: Illustration of edge-detection by convolution with an isotropic Laplacian operator, and marking the zero-crossings of the result of the convolution.

Edges in images are defined at <u>different scales:</u> some transitions in brightness are gradual, others very crisp. More importantly, at different scales of analysis, different edge structure emerges.

Example: an image of a leopard that has been low-pass filtered (or analyzed at a coarse scale) has edge outlines corresponding to the overall form of its body. At a somewhat finer scale of analysis, image structure may be dominated by the contours of its "spots." At a still finer scale, the relevant edge structure arises from the texture of its fur.

In summary, <u>non-redundant structure</u> exists in images at different scales of analysis (or if you prefer, in different frequency bands).

The basic recipe for extracting edge information from images is to use a multi-scale family of image filters (convolution kernels). A wide variety of these are in standard use, differing in terms such as:

- isotropic (circularly symmetric), or anisotropic (directional)
- self-similar (dilates of each other), or not self-similar
- separable (expressible as product of two 1D functions), or not
- degree of conjoint uncertainty in the information resolved
- size of support (number of "taps," or pixels, in the kernel)
- preferred non-linear outputs (zero-crossings; phasor moduli; energy)
- theoretical foundations (e.g. Logan's Theorem)

5.1 The Laplacian $\nabla^2 G_\sigma(x,y) * I(x,y)$ and its zero-crossings. Logan's Theorem.

One highly influential idea due to Marr (1981), that is frequently exploited for edge detection in machine vision systems, is to convolve the image with a multi-scale family of isotropic (non-directional) blurred 2nd-derivative filters, and to retain only their output zero-crossings. These correspond well to the edges in the image, at each chosen scale.

One primary motivation for doing this comes from Logan's Theorem (1977) concerning the "richness" of Laplacian zero-crossings for band-limited signals. What Logan proved (albeit only in the 1D case) is that subject to two constraints, the zero-crossings alone suffice to represent the signal completely (i.e. it could be perfectly recovered from just its zeros, up to a scale factor).

This is a truly remarkable result. Consider the fact that a signal is continuous and dense, but in any finite interval it will have only a finite (countable) number of zero-crossings (e.g., 7). How can those 7 points completely determine what the signal does everywhere else within this finite interval??

The two constraints are:

1. The signal must be strictly bandlimited to one octave, or less. This means that its highest frequency component must be no more than twice its lowest frequency component.
   (This constraint is much more powerful than it may appear.)
2. The signal must have no complex zeros in common with its Hilbert Transform. This effectively excludes purely amplitude-modulated signals. For example, a pure sinewave whose amplitude is merely modulated will have exactly the same zero-crossings as the unmodulated sinusoid, so their zero-crossings would not distinguish between them. Thus AM signals cannot be represented by zero-crossings.

The $\nabla^2 G_\sigma(x,y)$ filter kernel that is convolved with the image serves to bandpass-filter it. In the 2D Fourier domain, as we have seen, the spectral consequence of the Laplacian operator $\nabla^2 \equiv \left(\frac{d^2}{dx^2} + \frac{d^2}{dy^2}\right)$ is to multiply the image spectrum by a paraboloid: $(\mu^2 + \nu^2)$. Clearly this emphasizes the high frequencies at the expense of the low frequencies, and eliminates the DC component entirely (hence the output is centered around a mean of zero).

Blurring the Laplacian by a Gaussian $G_\sigma(x,y)$ of scale σ, simply limits the high-frequency components. The 2DFT of a Gaussian is also a Gaussian, with reciprocal dimension (by the Similarity Theorem discussed earlier). The scale parameter σ determines where the high-frequency cut-off occurs.

The resulting bandwidth of a $\nabla^2 G_\sigma(x,y)$ filter is about 1.3 octaves, regardless of what value for scale parameter σ is used. Note that this doesn't *quite* satisfy the first constraint of Logan's Theorem.

Note also that by commutativity of linear operators, the order in which these steps are applied to the image $I(x,y)$ doesn't matter. First computing the Laplacian of the image, and then blurring the result with the Gaussian, is equivalent to first convolving the image with the Gaussian and then computing the Laplacian of the result:

$$\nabla^2 \left[G_\sigma(x,y) * I(x,y)\right] = G_\sigma(x,y) * \nabla^2 I(x,y) \tag{13}$$

Moreover, both of these sequences are equivalent to just convolving the image with a single filter kernel, namely the Laplacian of a Gaussian: $\left[\nabla^2 G_\sigma(x,y)\right] * I(x,y)$. Clearly this is the preferred implementation, since it just involves a single convolution.

Some open theoretical issues in this approach are:

1. It is not clear how to generalize the constraint of one-octave bandlimiting to the case of 2D signals (images). E.g. should their 2DFT be confined to an annulus in the Fourier plane, whose outer radius is twice its inner radius?; or to four squares in the four quadrants of the Fourier plane that satisfy the one-octave constraint on each frequency axis? The first method doesn't work, and clearly the second filter is no longer isotropic!

2. Whereas the zeros of a 1D signal (soundwave) are denumerable [countable], those of a 2D signal (image) are not. Rather, they form "snakes" that are continuous contours in the plane.

3. As a practical matter, the $\nabla^2 G_\sigma(x,y) * I(x,y)$ approach to edge extraction

tends to be very noise-sensitive. Many spurious edge contours appear that shouldn't be there. This defect inspired the development of more sophisticated non-linear edge detectors, such as Canny's, which estimates the local image signal-to-noise ratio (SNR) to adaptively optimize its local bandwidth. This, however, is very computationally expensive.

4. Finally, strong claims were originally made that $\nabla^2 G_\sigma(x,y) * I(x,y)$ edge-detecting filters describe how human vision works. In particular, the receptive field profiles of retinal ganglion cells were said to have this form. However, counterexamples reveal several visual tasks that humans are able to perform, effortlessly and pre-attentively, which we could not perform if our visual systems functioned in this way.

6 Scale-space; multi-resolution. Wavelets as visual primitives.

Images contain information at multiple scales of analysis, so detecting visual features (such as edges) must be done across a range of different scales.

- An interesting property of edges as defined by the zero-crossings of multi-scale operators whose scale is determined by convolution with a Gaussian, is that as the Gaussian is made coarser (larger), new edges (new zero-crossings) can never appear. They can only merge and thus become fewer in number. This property is called causality. It is also sometimes called 'monotonicity,' or 'the evolution property,' or 'nice scaling behaviour.'

- One reason why causality is important is that it ensures that features detected at a coarse scale of analysis were not spuriously created by the blurring process (convolution with a low-pass filter) which is the normal way to create a multi-scale image pyramid using a hierarchy of increasing kernel sizes. One would like to know that image features detected at a certain scale are "grounded" in image detail at the finest resolution.

- For purposes of edge detection at multiple scales, a plot showing the evolution of zero-crossings in the image after convolution with a linear operator, as a function of the scale of the operator which sets the scale (i.e. the width of the Gaussian), is called scale-space.

- Scale-space has a dimensionality that is one greater than the dimensionality of the signal. Thus a 1D waveform projects into a 2D scale-space. An image projects into a 3D scale space, with its zero-crossings (edges) forming surfaces that evolve as the scale of the Gaussian changes. The scale of the Gaussian, usually denoted by σ, creates the added dimension.

- A mapping of the edges in an image (its zero-crossings after such filtering operations, evolving with operator scale) is called a scale-space fingerprint.

Several theorems exist called "fingerprint theorems" showing that the Gaussian blurring operator uniquely possesses the property of causality. In this respect, it is a preferred edge detector when combined with a bandpass or differentiating kernel such as the Laplacian.

- However, other non-linear operators have advantageous properties, such as reduced noise-sensitivity and greater application for extracting features that are more complicated (and more useful) than mere edges.

6.1 2D Gabor "Logons;" Quadrature Pair Wavelets

The family of filters which uniquely achieve the lowest possible conjoint uncertainty (i.e. minimal dispersion, or variance) in both the space domain and the Fourier domain are the complex exponentials multiplied by Gaussians. These are sometimes known as Gabor wavelets, or "logons." In one dimension:

$$f(x) = \exp(-i\mu_0(x - x_0)) \exp(-(x - x_0)^2/\alpha^2)$$

This is a Gaussian localized at position x_0, complex modulated at frequency μ_0, and with size or spread constant α. It is noteworthy that such wavelets have Fourier Transforms $F(\mu)$ with exactly the same functional form, but with their parameters merely interchanged or inverted:

$$F(\mu) = \exp(-ix_0(\mu - \mu_0)) \exp(-(\mu - \mu_0)^2\alpha^2)$$

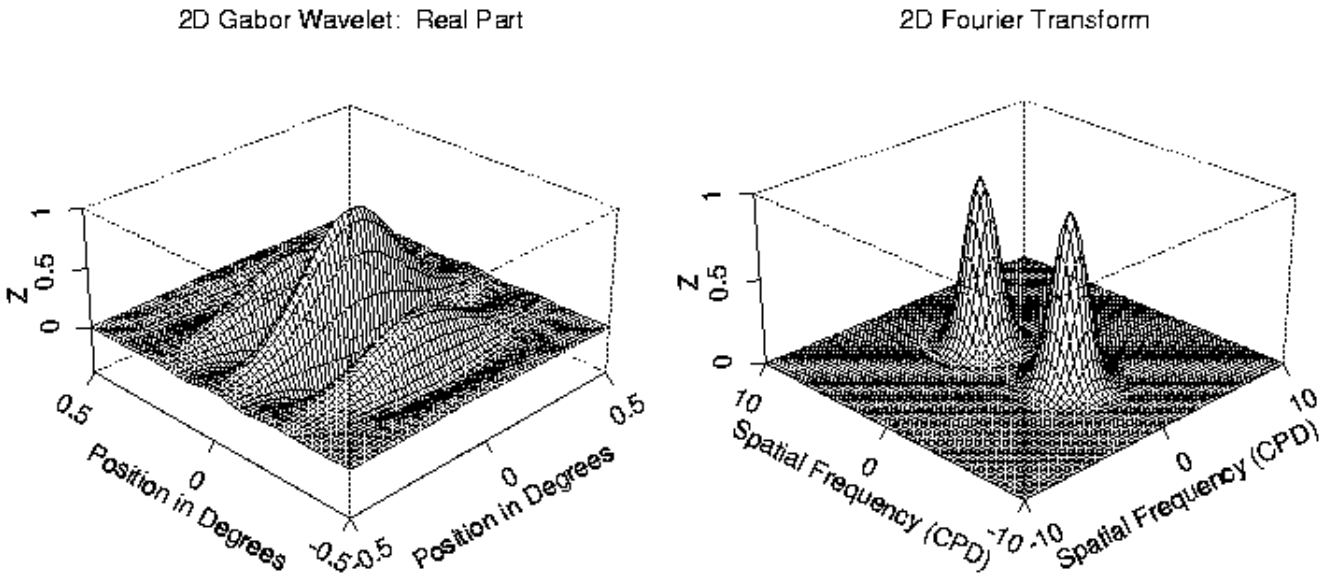
Note that for the case of a wavelet $f(x)$ centered on the origin so $x_0 = 0$, its Fourier Transform $F(\mu)$ is simply a Gaussian centered on the modulation frequency $\mu = \mu_0$, and whose width is $1/\alpha$, the reciprocal of the wavelet's space constant. This shows that it acts as a bandpass filter, passing only those frequencies that are within about $\pm\frac{1}{\alpha}$ of the wavelet's modulation frequency μ_0.

Dennis Gabor (1946) named these wavelets "logons" from the Greek word for information, or order: logōs. Because of the optimality of such wavelets under the Uncertainty Principle, Gabor proposed using them as an expansion basis to represent signals. In particular, he wanted them to be used in broadcast telecommunications for encoding continuous-time information. He called them the "elementary functions" for a signal. Unfortunately, because such functions are mutually non-orthogonal, it is very difficult to obtain the actual coefficients to be used with the elementary functions in order to expand a given signal in this basis. (Gabor himself could not solve this problem, although he went on to invent holography and to win the Nobel Prize in Physics in 1974.)

When a family of such Gabor functions are parameterized to be self-similar, i.e. they are dilates and translates of each other so that they all have a common

template ("mother" and "daughter"), then they constitute a (non-orthogonal) *wavelet basis*. Today it is known that infinite classes of wavelets exist which can be used as the expansion basis for signals. Because of the self-similarity property, this amounts to representing or analyzing a signal at different scales. This general field of investigation is called *multi-resolution analysis*, and we have already encountered its importance for extracting edge features.

6.2 Generalization of Wavelet Logons to 2D for Image Analysis



An effective method for extracting, representing, and analyzing image structure is the computation of the 2D Gabor wavelet coefficients for the image. This family of 2D filters were originally proposed as a framework for understanding the orientation-selective and spatial-frequency-selective receptive field properties of neurones in the brain's visual cortex, as well as being useful operators for practical image analysis problems. These 2D filters are conjointly optimal in extracting the maximum possible information both about the orientation and modulation of image structure ("what"), simultaneously with information about 2D position ("where"). The 2D Gabor filter family uniquely achieves the theoretical lower bound on joint uncertainty over these four variables in the Uncertainty Principle when it is suitably generalized.

These properties are particularly useful for texture analysis because of the 2D spectral specificity of texture as well as its variation with 2D spatial position. These wavelets are also used for motion detection, stereoscopic vision, and many sorts of visual pattern recognition such as face recognition. A large and growing literature now exists on the efficient use of this non-orthogonal expansion basis and its applications.

Two-dimensional Gabor wavelets have the functional form:

$$f(x,y) = e^{-\left[(x-x_0)^2/\alpha^2+(y-y_0)^2/\beta^2\right]}e^{-i[u_0(x-x_0)+v_0(y-y_0)]}$$

where (x_0, y_0) specify position in the image, (α, β) specify effective width and length, and (u_0, v_0) specify modulation, which has spatial frequency $\omega_0 = \sqrt{u_0^2 + v_0^2}$ and direction $\theta_0 = \arctan(v_0/u_0)$. (A further degree-of-freedom not included above is the relative orientation of the elliptic Gaussian envelope, which creates cross-terms in xy.) The 2D Fourier transform $F(u, v)$ of a 2D Gabor wavelet has exactly the same functional form, with parameters just interchanged or inverted:

$$F(u,v) = e^{-\left[(u-u_0)^2\alpha^2+(v-v_0)^2\beta^2\right]}e^{-i[x_0(u-u_0)+y_0(v-v_0)]}$$

The real part of one member of the 2D Gabor filter family, centered at the origin $(x_0, y_0) = (0, 0)$ and with unity aspect ratio $\beta/\alpha = 1$ is shown in the Figure, together with its 2D Fourier transform $F(u, v)$.

By appropriately parameterizing them for dilation, rotation, and translation, 2D Gabor wavelets can form a complete self-similar (but non-orthogonal) expansion basis for images. If we take $\Psi(x, y)$ to be some chosen generic 2D Gabor wavelet, then we can generate from this one member a complete self-similar family of 2D wavelets through the generating function

$$\Psi_{mpq\theta}(x,y) = 2^{-2m}\Psi(x',y')$$

where the substituted variables (x', y') incorporate dilations in size by 2^{-m}, translations in position (p, q), and rotations through orientation θ:

$$x' = 2^{-m}[x\cos(\theta) + y\sin(\theta)] - p$$
$$y' = 2^{-m}[-x\sin(\theta) + y\cos(\theta)] - q$$

It is noteworthy that as consequences of the similarity theorem, shift theorem, and modulation theorem of 2D Fourier analysis, together with the rotation isomorphism of the 2D Fourier transform, all of these effects of the generating function applied to a 2D Gabor mother wavelet $\Psi(x, y) = f(x, y)$ have corresponding identical or reciprocal effects on its 2D Fourier transform $F(u, v)$. These properties of self-similarity can be exploited when constructing efficient, compact, multi-scale codes for image structure.

The completeness of 2D Gabor wavelets as an expansion basis for any image can be illustrated by reconstruction of a facial image, in stages. Note how efficiently the facial features, such as the eyes and mouth, are represented using only a handful of the wavelets. Later we will see how this can be exploited both for automatic feature localization, and for face recognition.

Reconstruction of Lena: 25, 100, 500, and 10,000 Two-Dimensional Gabor Wavelets

6.3 Unification of Domains

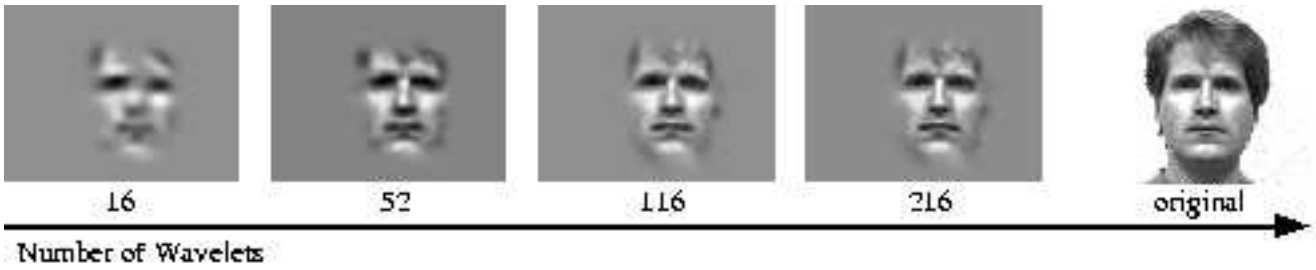
Until now we have viewed "the image domain" and "the Fourier domain" as very different domains of visual representation. But now we can see that the "Gabor domain" of representation actually embraces and unifies both of these other two domains. How?

In the wavelet equations above, the scale constant α (and β in the 2D case) actually builds a continuous bridge between the two domains. If the scale constant is set very large, then the Gaussian term becomes just 1 and so the expansion basis reduces to the familiar Fourier basis. If instead the scale constant is made very small, then the Gaussian term shrinks to a discrete delta function (1 only at the location $x = x_0$, and 0 elsewhere), so the expansion basis implements pure space-domain sampling: a pixel-by-pixel image domain

representation. This allows us to build a continuous deformation between the two domains when representing, analyzing, and recognizing image structure, merely by changing a single scaling parameter in this remarkable, unifying, expansion basis.

A "philosophical" comment about 2D Gabor wavelets.

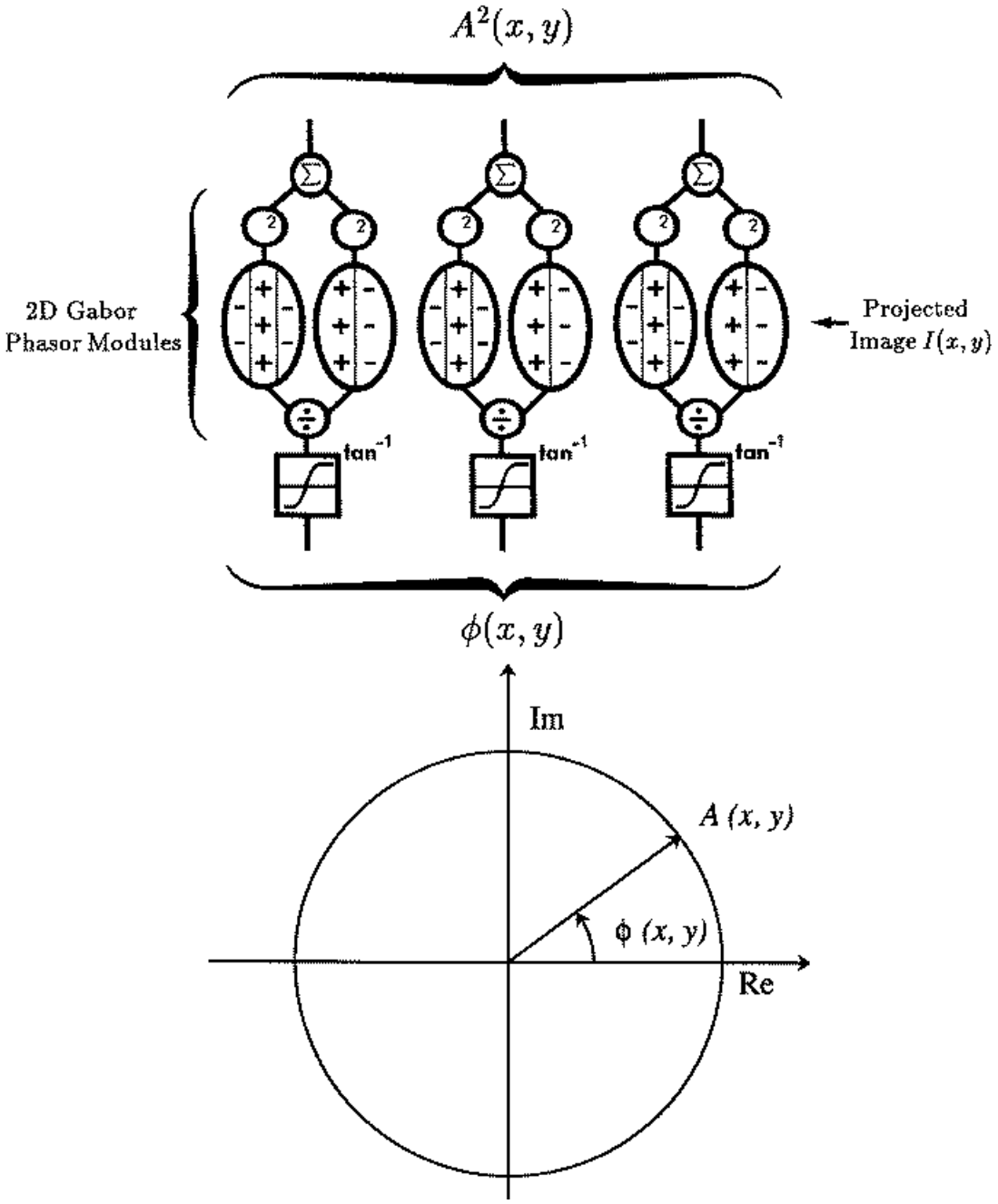
Aristotle defined vision as "knowing what is where." We have noted the optimality (conjoint uncertainty minimization) property of 2D Gabor wavelets in the two domains for extracting structural ("what") and positional ("where") information. Thus if we share Aristotle's goal for vision, then we cannot do better than to base computer vision representations upon these wavelets. Perhaps this is why mammalian visual systems appear to have evolved their use; the receptive field profiles of isolated neurones in the brain's visual cortex, as determined by the spatial distribution of excitatory and inhibitory inputs to each so-called "simple cell," can be well-described as quadrature-paired 2D Gabor wavelets. At the present time, this is basically the standard model for how the brain's visual cortex represents the information in the retinal image.



6.4 Detection of Facial Features by Quadrature Gabor Wavelet Energy

One illustration of a practical application of such image operators is in the automatic localization of facial features. Interestingly, most facial features themselves can be described by only a handful of wavelets, since such features are after all just localized undulations having certain positions, orientations, spatial frequencies, and phases. By taking the modulus (sum of the squares of the real and imaginary parts) of a facial image after convolving it with complex-valued 2D Gabor wavelets, key facial features (eyes and mouth) are readily detected; we may call this a Quadrature Demodulator Neural Neural. This capability is illustrated in the next two Figures.

Quadrature Demodulator Network



Neural Network for Image Analysis. The above neurobiologically-inspired network performs image demodulation using 2D Gabor wavelets, in order to find salient features in the image that have a characteristic orientation and scale or frequency composition. The operation of the biphasic receptive fields (representing even- and odd-symmetric visual cortical neurons) is described by:

$$g(x,y) = \int_\alpha \int_\beta e^{-\left((x-\alpha)^2+(y-\beta)^2\right)/\sigma^2} \cos(\omega(x-\alpha))\ I(\alpha,\beta)\ d\alpha\ d\beta$$

$$h(x,y) = \int_\alpha \int_\beta e^{-\left((x-\alpha)^2+(y-\beta)^2\right)/\sigma^2} \sin(\omega(x-\alpha))\ I(\alpha,\beta)\ d\alpha\ d\beta$$

and the demodulated output at the top of the network resembles that of the brain's "complex cells" which combine inputs from the quadrature simple cells as:

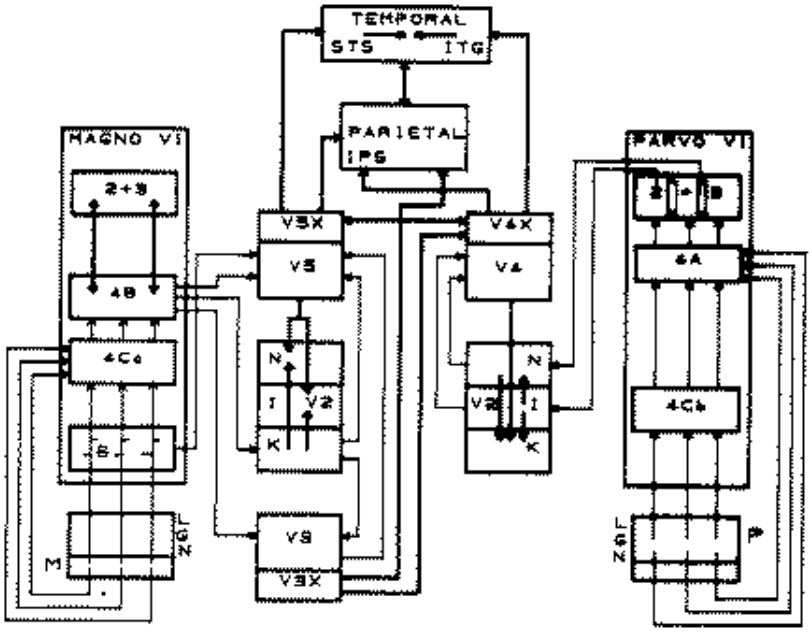
$$A^2(x,y) = g^2(x,y) + h^2(x,y)$$

Illustration of Facial Feature Detection by Quadrature Filter Energy. Left panel: original image. Right panel (clockwise from top left): the real part after 2D Gabor wavelet convolution; the imaginary part; the modulus energy; and this energy superimposed on the original (faint) image, illustrating successful feature localization.

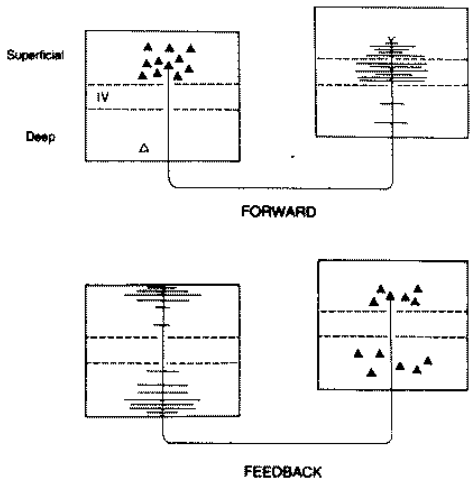
These operations can be performed by the **Quadrature Demodulator Network** shown in the previous Figure.

7 Higher brain visual mechanisms; streaming; reciprocal feedback

Besides the primary visual cortex in the occipital lobe, there are at least 30 further visual areas distributed across the parietal and temporal cortices of the brain. Many of these are specialised for particular kinds of visual processing, including colour (V4), motion (MT), stereo (Area 18), and facial and other form processing areas. There is a pronounced functional streaming, or division of labour, for form, colour, and motion processing; some neuroscientists have proposed a fundamental division into "two visual systems" along lines such as magno and parvo (fast/slow) or even conscious and unconscious vision.

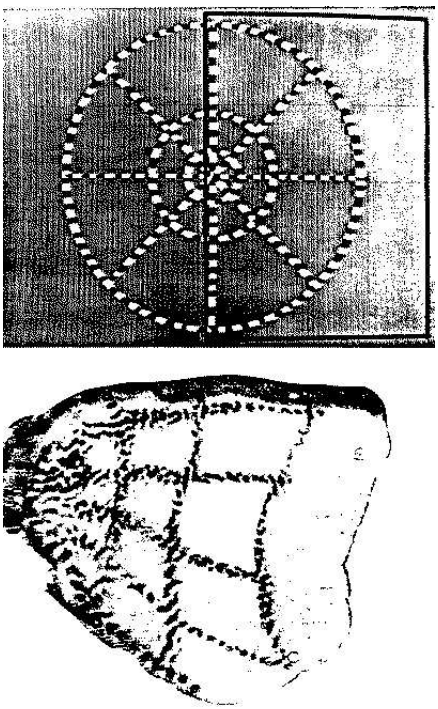


The existence of so many distinct visual areas in the brain almost begs the question of "how the visual world gets put back together again." An intriguing aspect of this architecture is the pattern of reciprocating feedback connections.



In general there are pairwise reciprocating connections between visual areas from the deep layers in one area to superficial layers in another area, whose deep layers in turn project back to the superficial layers of the first. Just as noted earlier with the massive feedback projections from primary visual cortex back down to the LGN (where it meets afferent signals ascending from the eyes), these reciprocating projection pathways are perhaps suggestive of a kind of "hypothesis generation and testing" iterative strategy for understanding the visual environment and the objects that populate it.

The fovea tends to be represented in all visual areas, and the mapping from the retina is retinotopic (meaning that adjacent points in the retinal image usually project to adjacent points in a given cortical map); but typically there is a highly pronounced geometrical distortion. In part this reflects a great over-representation of the fovea, which is called cortical magnification factor. In the foveal projection to primary visual cortex, about 6mm of neural tissue is devoted to 1 degree of visual angle, whereas in the periphery, 1mm of neural tissue handles about 6 degrees. It has been proposed that the geometrical distortion in visual mapping actually serves a specific mathematical role, that of achieving pattern representations that are invariant to rotation and dilation because of log-polar projection. Crudely speaking, this converts a polar grid (whose concentric circles have geometrically-increasing radii) into a cartesian grid with a nearly uniform lattice. Thus changes in object distance (hence image size) become just translations along one axis, while rotations become just translations along the other axis, thereby facilitating pattern recognition.



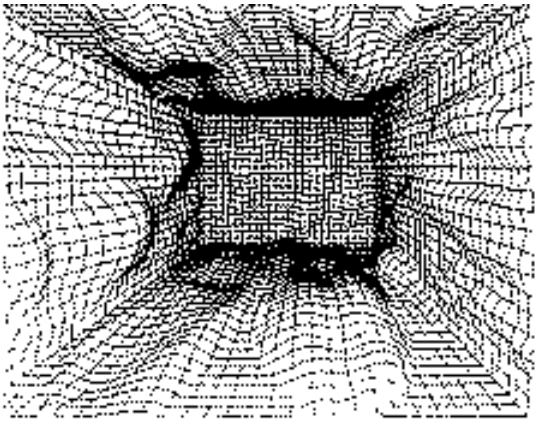
8 Texture, colour, stereo, and motion descriptors. Disambiguation.

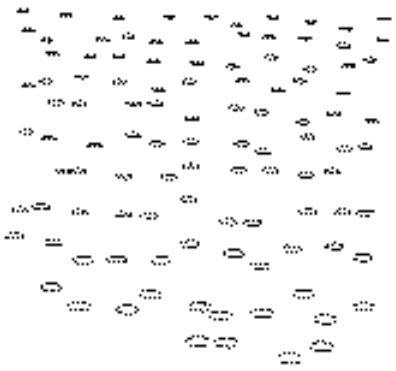
Many seemingly disparate tasks in computer vision actually share a common formal structure: to convert ill-posed, insoluble problems of inference from raw data, into well-posed problems in which we can compute object properties disambiguated from the image-formation processes which confound the raw luminance data itself.

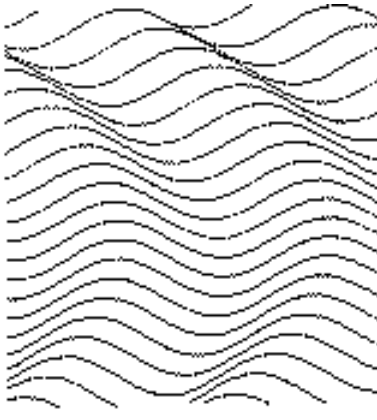
One obvious aspect of this issue is the fact that images are 2D projections of 3D data which could, in principle, arise equally well from many different constellations of worlds and objects. A more subtle aspect is the fact that the information received as an image is the compound product of several factors that are difficult to disambiguate: (1) the nature, geometry, and wavelength composition of the illuminant(s); (2) properties of the objects imaged, such as: spectral reflectances; surface shape; surface albedo; surface texture; geometry, motion, and rotation angle; and (3) properties of the camera (or viewer), such as (i) geometry and viewing angle; (ii) spectral sensitivity; (iii) prior knowledge, assumptions, and expectations. The aim of this lecture is to study how these many factors can be disambiguated and even exploited, in order to try to make objective inferences about object and world properties from these ambiguous and confounded image properties.

8.1 Texture information.

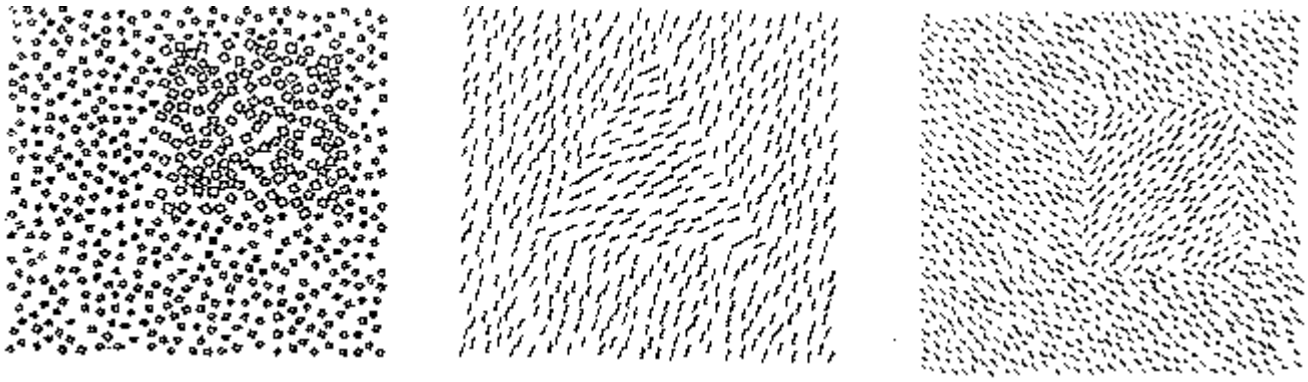
Most surfaces are covered with texture of one sort or another. Texture can serve not only as a helpful identifying feature, but more importantly as a cue to surface shape because of the foreshortening it undergoes as it follows the shape of the object if one can assume that it has some uniform statistics along the surface itself. The following patterns illustrate the inference of surface slant and of 3D surface shape from texture cues when they are combined with the assumption of texture uniformity on the surface itself:







Texture is also a useful cue to image segmentation by parsing the image into local regions which are relatively homogeneous in their textural properties. Here are some illustrations:



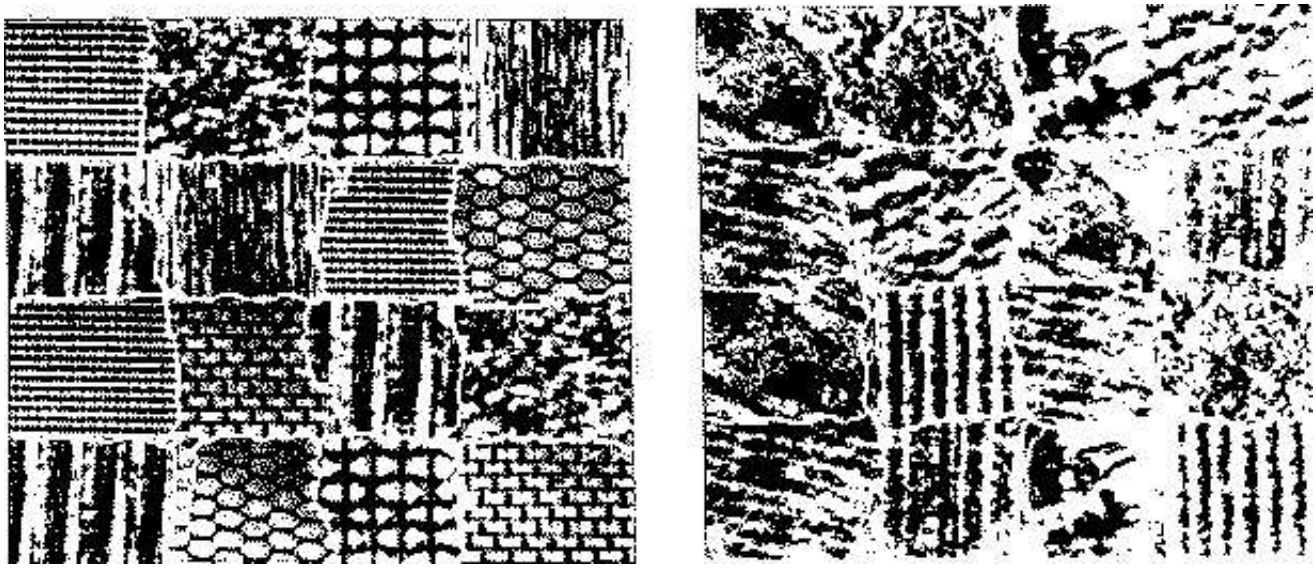
How can one measure something as ill-defined as a "textural signature?" What is texture, anyway?

As implied by the root of the word, which links it to textiles, texture is defined by the existence of certain statistical *correlations* across the image. These can be almost anything, from quasi-periodic undulations as one might see in water ripples or in woven fabrics, to repetitive but highly punctate features. Many natural scenes, such as woodlands, grasslands, mountain ranges and other terrains, have such properties which give them a distinctive identifying visual signature. The unifying notion in all of these examples is *quasi-periodicity*, or repetitiveness, of some features.

The detection of quasi-periodicity is best done by Fourier methods. There are deep and multi-faceted links between many topics in statistics (such as time-series analysis, correlation, moments) and Fourier analysis. These links arise from the fact that the eigenfunctions of the Fourier transform, complex exponentials (sinusoids in quadrature), are of course periodic but also have a specific scale (frequency) and direction (wavefront). Thus they excel in detecting the existence of a correlation distance and direction, and in estimating the relative "power" represented in various components of quasi-periodic correlated structures.

Unfortunately, these eigenfunctions are globally defined, but we wish to use local regional information as a basis for texture-based image segmentation. Hence the ideal solution is to "window" the sinusoids so that they analyze the image characteristics only within a local region and thus extract the spectral statistics as a function that varies with location. The optimal set of windowing functions are bivariate Gaussians, since their joint spatial/spectral localization

is greater than that of any other function. The product of complex exponentials times bivariate Gaussians are called *2D Gabor wavelets*, and these form a complete basis for image analysis and representation. The pictures below illustrate successful segmentation of collages of textured natural scenes, as well as of textured artificial objects, using such 2D wavelets for local spectral analysis to infer and measure their textural discriminators.



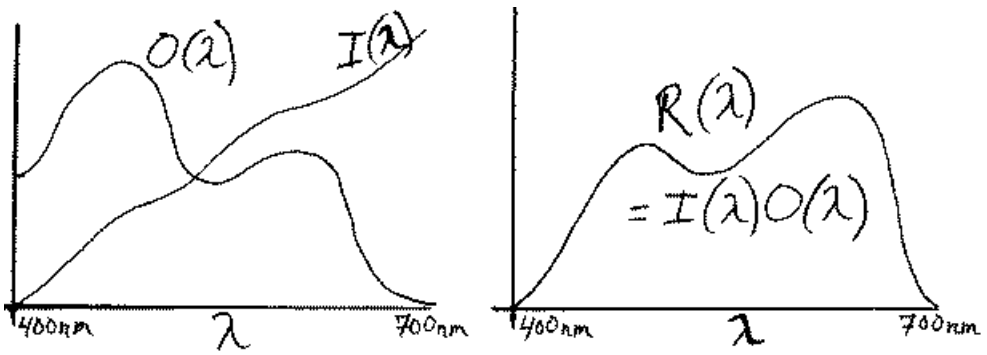
8.2 Colour information.

Colour is a nearly ubiquitous property of surfaces. Just like texture, it can serve both in object identification and in scene segmentation. But the fundamental difficulty in using the wavelength composition of images to infer the colour properties ("spectral reflectances") of objects, is the fact that the wavelengths received depend as much upon the *illuminant* as upon the spectral reflectances of the surface that is scattering back the light. When a yellow banana is illuminated in bluish light, the image that it forms obviously has a very different wavelength composition than when it is illuminated in reddish light. The central mystery of human colour perception is the fact that the banana still appears yellow ("colour constancy"). In computer vision, how can we possibly achieve this same vital capability of inferring an inherent underlying *object property* from a confounded (i.e., illuminant-wavelength dependent) set of *image properties*?

To give the problem a slightly more formal presentation:

- Let $I(\lambda)$ represent the wavelength composition of the illuminant (i.e. the amount of energy it contains as a function of wavelength λ, across the visible spectrum from about 400 nanometers to 700 nm).
- Let $O(\lambda)$ represent the inherent spectral reflectance of the object at a particular point: the fraction of incident light that is scattered back from its surface there, as a function of the incident light's wavelength λ.
- Let $R(\lambda)$ represent the actual wavelength mixture received by the camera at the corresponding point in the image of the scene.

Clearly, $R(\lambda) = I(\lambda)O(\lambda)$. The problem is that we wish to infer the "object colour" (its spectral reflectance as a function of wavelength, $O(\lambda)$), but we only know $R(\lambda)$, the actual wavelength mixture received by our sensor. So unless we can measure $I(\lambda)$ directly, how could this problem of inferring $O(\lambda)$ from $R(\lambda)$ possibly be solved?



One simple idea that has been proposed is to try actually to measure $I(\lambda)$ directly, by searching for highly specular (shiny, metallic, glassy) regions in an image where the reflected light might be a fairly faithful copy of $I(\lambda)$. This might be a glint from someone's glasses or from a shiny doorknob. Then at all other points in the image we need only to divide the $R(\lambda)$ we receive there by our other specular "measurement" of $I(\lambda)$, and we can then compute the desired $O(\lambda)$ across the image.

Clearly, this method has several weakness: (1) there may be no specular surfaces in the image; (2) those that there are may themselves affect somewhat the wavelength composition that they reflect (e.g. metals which have a brassy colour); and (3) the method is neither robust nor stable, since global inferences about scene interpretation depend critically upon uncertain measurements at (what may be just) a single tiny point in the image.

A more stable and interesting approach was developed by Dr E Land, founder of Polaroid, and is called the Retinex because he regarded it as modelled after

biological visual systems (RETINa + cortEX). Land's critical observation was that (contrary to almost universal popular belief), the colour perceived in an area of a scene is *not* determined by the wavelength composition of light received from that area (!). A simple experiment proves this: illuminate a scene, such as a bowl of fruit containing (say) a yellow banana, a red tomato and a green pepper, with three different narrowband light sources, each of which contains a different wavelength (say red, green, or blue) and with adjustable intensities. (No other light sources are present.)

The first observation is that even under drastic changes in the intensities of each of the three illuminators, the objects maintain exactly their normal colours. Obviously the wavelength mixture reaching the eye from each object is drastically changing, in proportion to the illuminators, but there are no changes in perceived colours. The phenomenon does not depend upon knowing the natural colours for objects identifiable by (say) their shape; a collage of patches of coloured paper cut into random shapes, forming a *mondrian,* produces exactly the same effect.

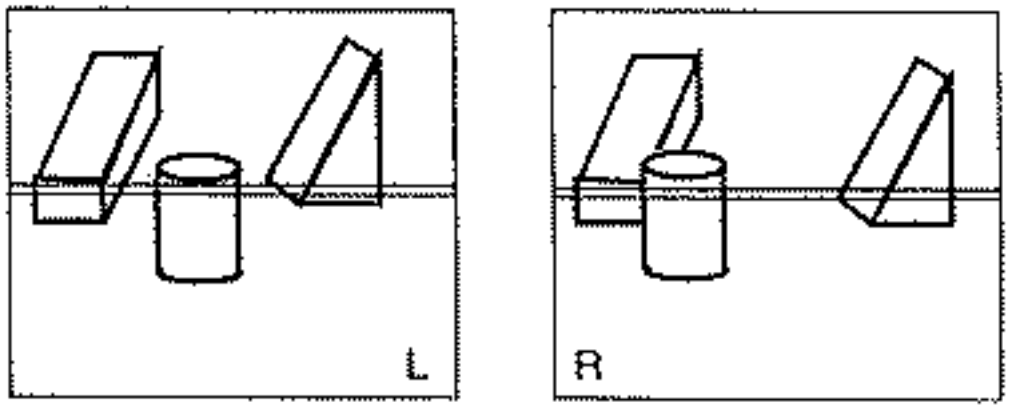
The second observation is that even when the wavelength composition of light reflected from each object is exactly the same (i.e. the three light sources are adjusted separately for each object to ensure that the light reflected in the three wavebands as measured by a spectral photometer is exactly the same for each of the objects individually), they *still* retain their natural colours. The banana still looks yellow, the tomato still looks red, and the pepper still looks green, even when each one is sending *identical* wavelength "messages" to your eyes. This is rather miraculous.

The Retinex algorithm attempts to account for this remarkable biological phenomenon, and to provide a means to achieve similar *colour constancy* in computer vision systems so that they may "discount the illuminant" and infer the spectral reflectance properties of objects, independent of the composition of their illumination. Only a cursory description of Retinex will be given here.

The key idea is that *the colours of objects or areas in a scene are determined by their surrounding spatial context.* A complex sequence of ratios computed across all the boundaries of objects (or areas) enables the illuminant to be algebraically discounted in the sense shown in the previous Figure, so that object spectral reflectances $O(\lambda)$ which is what we perceive as their colour, can be infered from the available retinal measurements $R(\lambda)$ without explicitly knowing $I(\lambda)$.

8.3 Stereo information

Important information about depth can be obtained from the use of two (or more) cameras, in the same way that humans achieve stereoscopic depth vision by virtue of having two eyes. Objects in front or behind of the point in space at which the two optical axes intersect (as determined by the angle between them, which is controlled by camera movements or eye movements), will project into different relative parts of the two images. This is called *stereoscopic disparity.*



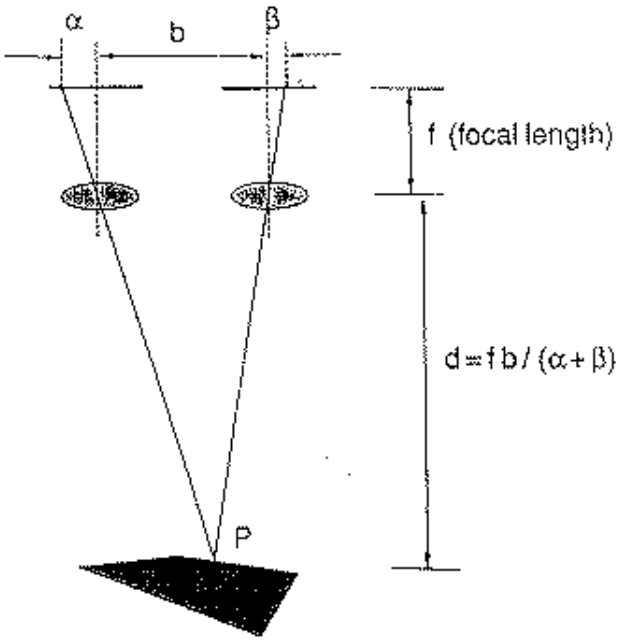
This "error signal" becomes greater in proportion to the distance of the object in front or behind the point of fixation, and so it can be calibrated to obtain a depth cue. It also becomes greater with increased spacing between the two eyes or cameras, since that is the "base of triangulation." (That is why the German Army in WWI introduced V-shaped binocular "trench periscopes" to increase stereoscopic visual acuity, for breaking camouflage by increasing the *effective* spacing between the viewer's two eyes to almost a meter.)

The essence of making use of such stereoscopic disparity cues is the need to solve the Correspondence Problem. In order to infer that the cylinder is in a different position relative to the background objects in the two frames shown, it is first necessary to detect the correspondence of the background objects in the two frames, or at least of their edges. This puts the two frames "into registration," so that the disparity of the foreground object can be detected.

Unfortunately, current algorithms for solving the Correspondence Problem tend to require very large searches for matching features under a large number of possible permutations. It is difficult to know which set of features in the two frames to select for comparison in evaluating the degree of alignment, when trying to find that relative registration which generates maximum correlation between the two background scenes.

One helpful approach here is to use a "multi-scale image pyramid," which steers the search in a coarse-to-fine fashion to maximize its efficiency. In initially sparsely sampled (coarsely blurred and under-sampled) images, the permutation-matching space of possible corresponding points is greatly attenuated compared with full-resolution images.

After an adequate alignment match is found for low-resolution (blurred) copies of the image pair, the process repeats on somewhat higher resolution (less blurred) copies of the image pair but over a search space that has been greatly curtailed by having first found the coarse-scale solution. Such "pyramid" processes usually increment in one-octave steps (factors of two in improved resolution), from coarse to fine, spanning a total of perhaps four or five levels before the final solution is determined to within single-pixel precision.



Once the Correspondence Problem has thereby been solved, the inference of depth from object disparity in the two image frames is then just a matter of triangulation and "look-up" from a calibration table which includes information about the spacing between the two cameras (or eyes) and their focal lengths. (See the above simplifying diagram, for the case that the two cameras' optical axes are parallel and hence converged at infinity.) Specifically, if the two cameras have focal length f and the optical centres of their lenses (remember the trench periscopes!) are separated by a distance b, and the disparity in the projections of some object point onto the two images (in opposite directions relative to their optical axis) is α in one image and β in the other image, then the distance d to the object in front of the two lenses is simply:

$$\boxed{d = fb/(\alpha + \beta)}$$

8.4 Motion information

Only a few vision applications actually involve just static image frames. That is basically vision "off-line;"– but the essence of an effective visual capability must be for real-time use in a dynamic environment. This requires the ability to detect and measure motion, and to be able thereby to draw inferences quickly (such as time-to-collision).

In a formal sense, the problem of computing motion information from an image sequence is very similar to that of computing stereo information.

- For stereo vision, we need to solve the Correspondence Problem for two images simultaneous in time but acquired with a spatial displacement.
- For motion vision, we need to solve the Correspondence Problem for two images coincident in space but acquired with a temporal displacement.
- The object's spatial "disparity" that can be measured in the two image frames once their backgrounds have been aligned, can be calibrated to reveal motion information when compared with the time interval, or depth information when compared with the binocular spatial interval.

Among the challenging requirements of motion detection and inference are:

1. Need to infer 3D object trajectories from 2D image motion information.
2. Need to make *local* measurements of velocity, which may differ in different image regions in complex scenes with many moving objects. Thus, a velocity vector field needs to be assigned over an image.
3. Need to disambiguate object motion from contour motion, so that we can measure the velocity of an object regardless of its *form*.
4. Need to measure velocities regardless of the size of the viewing aperture in space and in time (the spatial and temporal integration windows).
5. It may be necessary to assign more than one velocity vector to any given local image region (as occurs in "motion transparency")
6. We may need to detect a *coherent* overall motion pattern across many small objects or regions separated from each other in space.

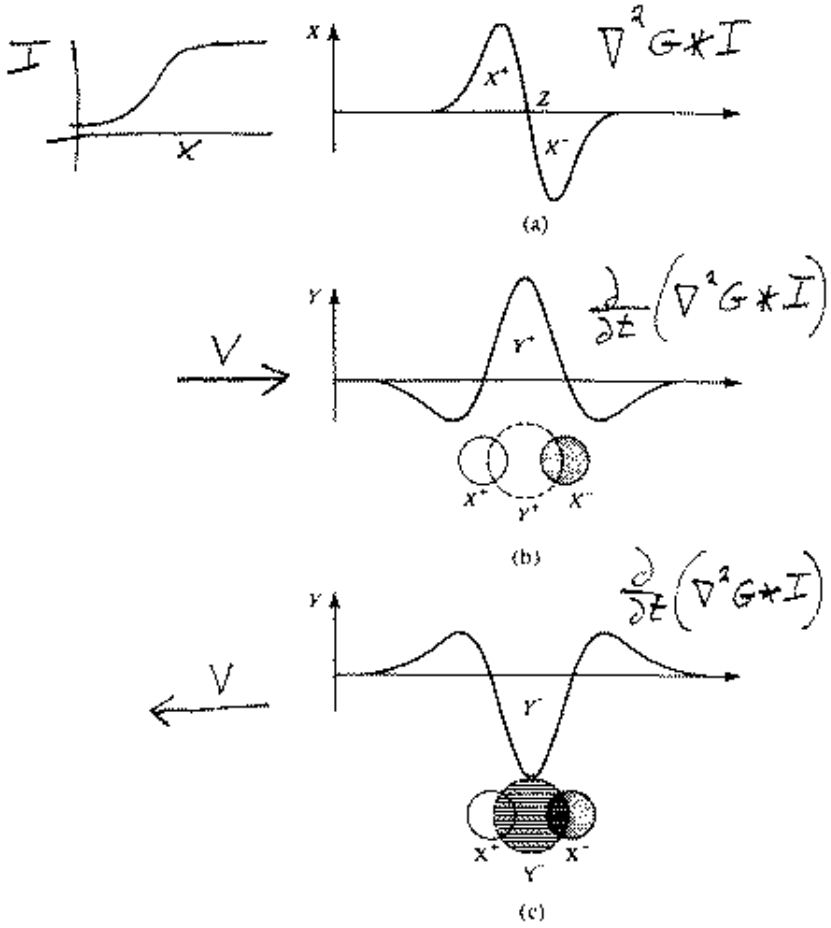
The major classes of models and approaches to motion detection are largely inspired by detailed neurobiological studies of motion processing both in the invertebrate eye and in mammalian retina and cortex. Diverse mathematical frameworks have been proposed, but the main classes of models are:

INTENSITY GRADIENT MODELS .

Assume that the local time-derivative in image intensities at a point, across many image frames, is related to the local spatial gradient in image intensities because of object velocity $\vec{v}$:

$$-\frac{\partial I(x,y,t)}{\partial t} = \vec{v} \cdot \vec{\nabla} I(x,y,t)$$

Then the ratio of the local image time-derivative to the spatial gradient is an estimate of the local image velocity (in the direction of the gradient).



DYNAMIC ZERO-CROSSING MODELS .

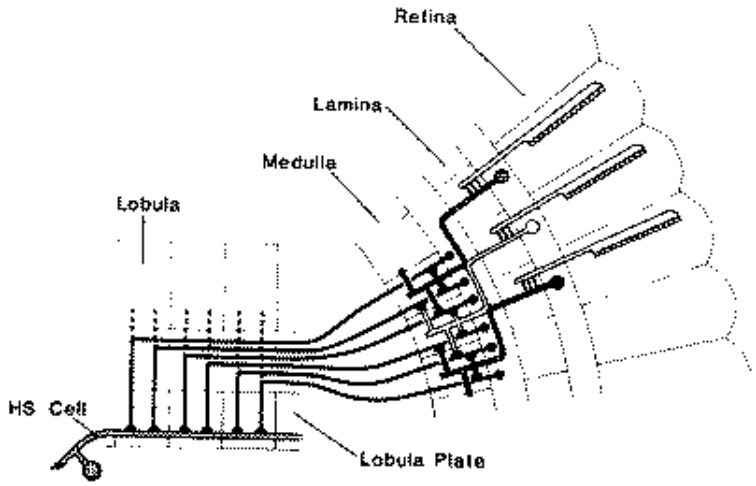
Measure image velocity by first finding the edges and contours of objects (using the zero-crossings of a blurred Laplacian operator!), and then take the time-derivative of the Laplacian-Gaussian-convolved image:

$$-\frac{\partial}{\partial t}\left[\nabla^2 G_\sigma(x,y) * I(x,y,t)\right]$$

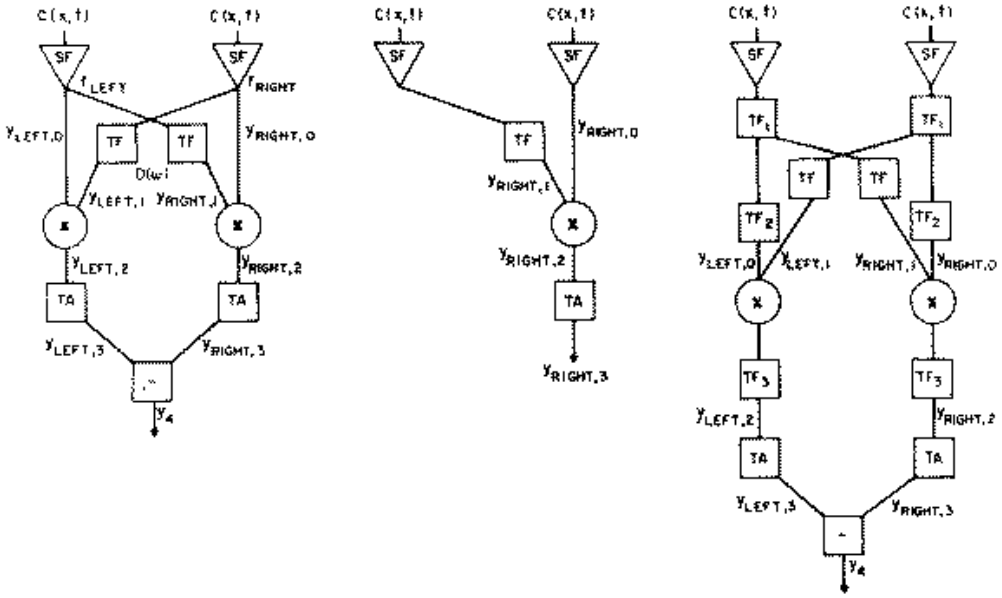
in the vicinity of a Laplacian zero-crossing. The amplitude of the result is an estimate of speed, and the sign of this quantity determines the direction of motion relative to the normal to the contour.

SPATIO-TEMPORAL CORRELATION MODELS .

Image motion is detected by observing a *correlation* of the local image signal $I(x, y, t)$ across an interval of space and and after an interval of time τ. Finding the pair of these intervals which maximizes the correlation between $I(x, y, t)$ and $I(x - v_x\tau, y - v_y\tau, t - \tau)$ determines the two components of image velocity v_x and v_y that we desire to know.



Detailed studies of fly neural mechanisms (above) for motion detection and visual tracking led to elaborated correlation-based motion models:



SPATIO-TEMPORAL SPECTRAL MODELS .

It is possible to detect and measure image motion purely by *Fourier* means. This approach exploits the fact that motion creates a covariance in the spatial and temporal *spectra* of the time-varying image $I(x, y, t)$, whose three-dimensional (spatio-temporal) Fourier transform is defined:

$$F(\omega_x, \omega_y, \omega_t) = \int_X \int_Y \int_T I(x, y, t) e^{-i(\omega_x x + \omega_y y + \omega_t t)} dx dy dt$$

In other words, rigid image motion has a 3-D spectral consequence: the local 3-D spatio-temporal spectrum, rather than filling up 3-space $(\omega_x, \omega_y, \omega_t)$, collapses onto a 2-D inclined plane which includes the origin. Motion detection then occurs just by filtering the image sequence in space and in time, and observing that tuned spatio-temporal filters whose center frequencies are **co-planar** in this 3-space are activated together. This is a consequence of the **SPECTRAL CO-PLANARITY THEOREM**:

Theorem: Translational image motion of velocity $\vec{\mathbf{v}}$ has a 3-D spatio-temporal Fourier spectrum that is non-zero only on an inclined plane through the origin of frequency-space. Spherical coordinates of the unit normal to this spectral plane correspond to the speed and direction of motion.

Let $I(x, y, t)$ be a continuous image in space and time.

Let $F(\omega_x, \omega_y, \omega_t)$ be its 3-D spatio-temporal Fourier transform:

$$F(\omega_x, \omega_y, \omega_t) = \int_X \int_Y \int_T I(x, y, t) e^{-i(\omega_x x + \omega_y y + \omega_t t)} dx dy dt.$$

Let $\vec{\mathbf{v}} = (v_x, v_y)$ be the local image velocity.

Uniform motion $\vec{\mathbf{v}}$ implies that for all time shifts t_o,

$$I(x, y, t) = I(x - v_x t_o, y - v_y t_o, t - t_o).$$

Taking the 3-D spatio-temporal Fourier transform of both sides, and applying the shift theorem, gives

$$F(\omega_x, \omega_y, \omega_t) = e^{-i(\omega_x v_x t_o + \omega_y v_y t_o + \omega_t t_o)} F(\omega_x, \omega_y, \omega_t).$$

The above equation can only be true if $F(\omega_x, \omega_y, \omega_t) = 0$ everywhere the exponential term doesn't equal 1.

This means $F(\omega_x, \omega_y, \omega_t)$ is non-zero only on the 3-D spectral plane

$$\boxed{\omega_x v_x + \omega_y v_y + \omega_t = 0} \qquad \textbf{Q.E.D.}$$

The spherical coordinates $(\theta, \phi, 1)$

$$\phi = \tan^{-1}\left(\omega_t / \sqrt{\omega_x^2 + \omega_y^2}\right)$$

$$\theta = \tan^{-1}\left(\omega_y / \omega_x\right)$$

of the inclined spectral plane's unit normal are determined by $\vec{\mathbf{v}}$ and correspond to the *speed* (ϕ) and *direction* (θ) of motion:

$$\phi = \sqrt{v_x^2 + v_y^2}$$

$$\theta = \tan^{-1}\left(v_y / v_x\right)$$

(Notes TO BE CONTINUED)