Advanced Computer Vision Lecture Slides.

Neil A. Thacker.

Last updated
25 / 7 / 2008
Appart from any fair dealing for the purpose of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents act 1988, this publication may only be reproduced, stored or transmitted, in any form or by any means, with prior permission in writing of the publishers, or in the case of reprographic reproduction in accordance with the terms of the licences issued by the Copyright Licensing Agency. Enquiries concerning reproduction outside those terms should be sent to the publishers.

Copyright ©University of Manchester 2008

The use of registered names, trademarks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant laws and regulations and therefore free for general use.
The publisher makes no representation, express or implied, with regard to the accuracy of the information in this document and cannot accept any legal responsibility for any error or omission that may be made.

Given the length of this document and the sparse nature of the material we suggest printing with at least 4 slides to a page.
Preface

This document contains the lecture material I have used on advanced computer vision MSc courses over the last decade at the University of Manchester. It is at a level deemed suitable for students with a mathematically based first degree. The course is run with a web based introduction which is intended to bring all students up to the same (undergraduate) level of knowledge regarding basic ideas and terminology before commencing lectures. The set of 50 minute lectures is then intended to provide the level of knowledge required so that students can begin to apply a minimum level of critical thinking to the material they encounter when undertaking computer vision oriented literature reviews.

The quantity of published work and range of applications for computer vision is immense. An attempt to cover even the most popular algorithms in the field would generate far more material than could be covered in a single course module. Therefore, in order to focus on a specific set of approaches we need some basic criteria. This focus, and the intellectual material which we would say justifies the description of the course as ‘advanced’, stems mainly from the interpretation of algorithmic data analysis as a statistical process. In this lecture course efforts are made to explain how to relate techniques and algorithms directly to probability theory and the benefits of this approach. The difference between closed form algebraic solutions (vision as inverse optics and geometry) and statistical estimation (using quantitative probability) is identified at the outset. The lecture notes are illustrated throughout with examples from our own published work and other material available from our web pages (http://www.tina-vision.net). Many of the lectures have associated tutorials, which have been written with the experience gained from many years of research.

I have used LaTeX in portrait format for these lectures, partly for reasons of maintenance. I also like being able to put a 5 line proof on a slide when I need to, and I feel the layout is better suited to creating readable handouts than (for example) powerpoint in landscape. The layout of the slides is less about what they look like, and more to do with being a record of the main ideas that students can use as the basis for notes and revision. I have also avoided using movies in my lectures for the same reasons, preferring instead to run separate live software demonstrations when they are appropriate.

As this material is also used to train new PhD students and researchers much of it is intended to identify the limitations associated with particular methods, so that avenues for future work might be more easily identified. The presentation style is therefore often necessarily critical, with attempts made to identify key requirements and theoretical principles. Individual comments are fleshed out in detail during the lectures with appropriate arguments. I expect, and generally get, quite a few difficult questions, particularly from those students who have already been on undergraduate computer vision courses. I believe that the topic of computer vision is on a cusp, with methodological changes on the way which will mean that it can no longer be described as an “immature discipline” (wikipedia), with many real world applications soon to be followed by uses in medicine, general science and cognitive modelling. Those students wishing to follow these ideas further might be interested in the three volume series available from our web site and summarised in tina memo 2008-001.

Students from our own courses should be aware that lectures covering wavelets, colour and image registration are not included in this volume, as they are not yet in a suitable electronic form. The TINA based practicals associated with this course can be found in tina memo 2005-005.

Neil Thacker, Senior Lecturer, University of Manchester, 2008.
Table of Contents

Part 1 : Images and Processing
Images and Deconvolution ................................................. 1
Image Warping ................................................................. 12

Part 2 : Statistical Methods
Statistics and Error Propagation ......................................... 24
Stability of Image Processing Algorithms ............................... 38
Pattern Recognition .......................................................... 52
Region Based Image Segmentation ...................................... 63

Part 3 : Feature Based Methods
Edge Based Vision ............................................................. 75
Hough Shape Detectors ....................................................... 89
Corner Detection ............................................................... 101
Advanced Linear Filters ..................................................... 113

Part 3 : Structure and Measurement
Shape From Stereo, Motion and Shading ............................... 125
Model Based Vision ........................................................... 137
Stereo Geometry and Calibration ......................................... 149
Part 4 : Recognition and Intelligence

Completeness of Recognition Algorithms. ........................................ 164

Probabilistic Relaxation Labelling. ............................................. 178

Statistical Foundations of Algorithmic Research. ......................... 190

Visual Intelligence. .................................................................... 203
Images and Deconvolution.

N. A. Thacker

- Image Storage
- Image Representation
- Neighbourhood Processing
- Convolution.
- Common Convolution Kernels.
- Spatial Frequency.
- Convolution Theorem.
- Statistical Solutions vs Analytical Solutions.
Image Representation and Storage.

Images are BIG so not ASCI

Stored on DISC as a direct copy of memory pieces

• binary bit eg: 0 or 1
• byte 8 bits, eg: 01010011 (char)
  The top bit may be the sign, ie -127 to 128, or not, ie: 0 to 256.
• 2 bytes (short), can also be signed.
• 4 bytes is a word, this can represent either an integer 0-
  \( (2^{32} -1) \)
  (again signed or unsigned) or floating point
  MINFLOA T to MAXFLOA T (IEEE represenation).
• others (double precision etc.)

Other factors

• byte alignment. (words can only start on even address
  boundaries)
• Big-endian (SUN, VAX) and Little Endian (PC, Trans-
  puter).
• colour
Simple Images are

- Header
- Data Block (memory copy).

Using a representation for the data block which can be any of the above.

Examples: raw, raster, pgm (encodes possibilities and is portable)

More complicated formats are compressed either

- losslessly, eg: bitmap, tiff (lossless option), gzip.
- or with losses, eg: gif, JPEG, MPEG.
Image Representation.

Images can be considered in a variety of equivalent representations;

- Array of pixels: $I(i, j)$.
- Array of vectors: $v_{i,j} = (r, g, b)$., eg, colour.
- Function of two continuous variables: $f(x, y)$
- Landscape.
- Image histogram (no spatial information).
- Spatial frequency transform.

Which of these we choose and the accuracy required for representation (short, integer, floating point), will depend upon what we are trying to achieve.
Neighbourhood Processing.

Point processes replace individual array elements with a processed version of the original value.

\[ I'(i, j) = f[I(i, j)] \]

examples; addition, subtraction, trigonometric functions, simple thresholding.

Neighbourhood processes replace the individual array elements using a value computed from a local region

\[ I'(i, j) = f[I(i, j), I(i-1, j), I(i, j-1), I(i+1, j), I(i, j+1), \ldots] \]

Though this set of data is generally close to the computed location, potentially this region can be the whole image.

examples; rank filtering, convolution.

Rank filtering is a process where each greylevel is replaced with the ranking of this value in comparison to some number of its neighbours.

Convolution is a linear weighted sum of neighbouring values.
Convolution.

In 1D $I(i)$ with a kernel $g(a)$ we have

$$I(i) = \sum_a g(a)I(i - a)$$

Normally this is done subject to a normalisation of the kernel

$$\sum_a g(a) = 1$$

In 2D $I(i, j)$ with a kernel $g(a, b)$ we have

$$I'(i, j) = \sum_a \sum_b g(a, b)I(i - a, j - b)$$

or for a continuous case

$$f'(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} g(a, b)f(x - a, y - b)da \, db$$

also written as

$$f'(x, y) = g \otimes f(x, y)$$
Common Convolution Kernels.

- Gaussian
  \[ g(a, b) = k \exp\left[-\frac{(a^2 + b^2)}{2\sigma^2}\right] \]
  - smoothing (noise) filter
- Difference of Gaussian (DoG)
  - notch filter
- Laplacian (or Laplacian of Gaussian)
  - 2nd Derivative filter, similar to DoG.
- Tangential (anisotropic) Filter
  - smooth along direction perpendicular to local slope

\[ I' = \frac{a+b}{2} \]
Common Convolution Kernels II.

Original Image

Gaussian

Difference of Gaussian
Spatial Frequency.

Any curve $f(x)$ can be represented as a set of sine and cosine curves $F(u)$.

The Fourier transform is defined as

$$f(x) = k_1 \int_{-\infty}^{\infty} F(u) \exp[2\pi ixu] du$$

with

$$F(u) = k_2 \int_{-\infty}^{\infty} f(x) \exp[-2\pi ixu] dx$$

We can also take the Fourier transform of 2D functions $F(u, v)$

For images $f(x, y)$ we call $F(u, v)$ spatial frequency.
Convolution Theorem.

Multiplication on the frequency domain is convolution in the spatial domain

\[ F'(u, v) = F(u, v)G(u, v) = f'(x, y) = g \otimes f(x, y) \]

This is frequency domain filtering and can have computational advantages in some cases (large kernels).

In theory it looks possible to undo the effects of a convolution, using de-convolution

\[ F(u, v) = F'(u, v)/G(u, v) \]

This concept, together with many other related ideas presented in many image processing textbooks is vastly over-simplistic.
There are many things wrong with the idea that spatial frequency, computed using a Fourier transform is of any practical value in the interpretation of images.

One such error is that simple de-convolution doesn’t really work.

In the case of de-convolution this error can be traced back to the fact that reconstructing the original image in practical situations should be considered an estimation task. Estimation tasks must take account of measurement error and involve the use of probability and statistics.

The practical algorithms we must actually use to perform de-convolution do not look as simple as the analytic analysis on the previous slide would imply.

In this course we will try to explain the true complexity of such problems and the approaches which can be expected to solve them.
Image Warping.

N. A. Thacker

• The Need for Image Warping
• Forward Transforms.
• Backward Transforms.
• Polynomial Interpolation.
• Quadratic Masks.
• Shifted Linear Filtering.
The Need for Image Warping.

Vision algorithms often require the image to be presented in a particular co-ordinate system which may be different to that produced by the imaging system.

There are two basic approaches; **forward** and **backward** transforms

- Forward transforms are quick but generally used for display purposes only because of drop out.

- Backward transforms are better but require either an assumption for the local image surface model or combination with a filtering stage.
Forward Transform.

General definition:

\[ I(g, h) = I(f'(x, y), f''(x, y)) \]

Simply set the I(g,h) pixel by looping over x and y.

example 1: the radial polar transform:

\[ \theta = \text{atan}(x, y) \]
\[ r = \sqrt{x^2 + y^2} \]

Log polar maps \((\log(r), \theta)\) are closely related to physiological models of retina in the primate, cat and owl.

They thus occur quite frequently in the machine vision literature. Some people have even tried building hardware.

Relationship between cartesian and radial/polar images.
example 2: face modelling

This process is performed by a set of independent affine warps defined within triangular cells of "control points".

Face warping by interpolation within control point triangles.
Backward Transform.

General definition:

\[ I(x, y) = I(f'(g, h), f''(g, h)) \]

Construct the output image by looping over \( g \) and \( h \) and interpolating an image grey level at the calculated \( x \) and \( y \).

example 1: the radial polar transform:

\[
\begin{align*}
x &= r \sin(\theta) \\
y &= r \cos(\theta)
\end{align*}
\]

example 2: Radial Distortion

- The most commonly used model for an optical system in machine vision is the pinhole camera (not a lens!).
- For optical systems this is frequently inaccurate due to distortion effects.
- Inverse distortion projection is a problem and generally must be iteratively computed.
Polynomial Interpolation.

Simple 1D interpolation from 3 points can be done with a quadratic.

\[ I(x) = a + bx + cx^2 \]

assuming that the image values are sampled from a regular grid of equispaced points at \( x = -1, 0 \) and 1.

\[ a = I(0), \quad b = I(1) - I(-1), \quad c = I(-1) - 2I(0) + I(1) \]

We can attempt to interpolate in both directions this way but the result is different depending on which axis we compute first.

The coefficients of a 2D quadratic

\[ P(x, y) = a + bx + cy + dx^2 + exy + fy^2 \]

can be computed from the solution of a least squares cost function.

\[ E = \sum_m \sum_n (P(m, n) - I(i + m, j + n))^2 \]
Quadratic Masks.

The set of required polynomial parameters can be obtained from a set of differential constraint equations.

\[ \frac{\partial E}{\partial a} = \sum_m \sum_n 2(P(m,n) - I(i+m,j+n)) = 0 \quad (1) \]

\[ \frac{\partial E}{\partial b} = \sum_m \sum_n m(P(m,n) - I(i+m,j+n)) = 0 \quad (2) \]

\[ \frac{\partial E}{\partial c} = \sum_m \sum n(P(m,n) - I(i+m,j+n)) = 0 \quad (3) \]

\[ \frac{\partial E}{\partial d} = \sum_m \sum_n m^2(P(m,n) - I(i+m,j+n)) = 0 \quad (4) \]

\[ \frac{\partial E}{\partial e} = \sum_m \sum mn(P(m,n) - I(i+m,j+n)) = 0 \quad (5) \]

\[ \frac{\partial E}{\partial f} = \sum_m \sum n^2(P(m,n) - I(i+m,j+n)) = 0 \quad (6) \]
These can be solved as a set of simultaneous linear equations

\[ \text{from (1)} \quad 9a + 6d + 6f = I \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad (7) \]

\[ \text{from (2)} \quad b = \frac{I}{6} \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \]

\[ \text{from (3)} \quad c = \frac{I}{6} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \]

\[ \text{from (4)} \quad 6a + 6d + 4f = I \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix} \quad (8) \]

\[ \text{from (5)} \quad e = \frac{I}{4} \begin{bmatrix} -1 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & -1 \end{bmatrix} \]
\[
\begin{align*}
\text{from (6) } & \quad 6a + 4d + 6f = I \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \\
\text{from (7) (8) (9) } & \quad d = \frac{I}{6} \begin{bmatrix} 1 & -2 & 1 \\ 1 & -2 & 1 \\ 1 & -2 & 1 \end{bmatrix} \\
\text{also } & \quad f = \frac{I}{6} \begin{bmatrix} 1 & 1 & 1 \\ -2 & -2 & -2 \\ 1 & 1 & 1 \end{bmatrix} \\
\text{and finally } & \quad a = \frac{I}{9} \begin{bmatrix} -1 & 2 & -1 \\ 2 & 5 & 2 \\ -1 & 2 & -1 \end{bmatrix}
\end{align*}
\]

• This is fine if the image is known to be composed of quadratic peaks but otherwise more complex models need to be used.
• This raises the problem of selecting the correct model!

<table>
<thead>
<tr>
<th>Symmetric Masks Parameters</th>
<th>Polynomial Factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>□</td>
<td>1</td>
</tr>
<tr>
<td>□</td>
<td>1, y</td>
</tr>
<tr>
<td>□</td>
<td>1, xy</td>
</tr>
<tr>
<td>□</td>
<td>1, xy, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, y, xy</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, y, yx, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
<tr>
<td>□</td>
<td>x, xy, x^2, y, yx</td>
</tr>
</tbody>
</table>

• In some cases (if the imaging system integrates each pixel over a rectangular area) the correct model to use is the sinc function (see TINA Memo 1999-005).
Shifted Linear Filtering.

If the next process in the algorithm is to be a linear filter then this can often be combined with the warping process.

The process attempts to estimate the result we would have had if the filter had been positioned in the correct place over the input image.

This technique requires no surface model assumption.

Likely to be slow, but can be supported on specialised hardware (see TINA Memo 1994-002).
**Problem:** Try to derive a simple linear warping algorithm for mapping one triangle into another, such as was described in the face warping example.
Statistics and Error Propagation.

N. A. Thacker

- Methodology
- Basic Definitions.
- Bayes Theorem.
- Maximum Likelihood.
- Common Probability Equations.
- Covariance Estimation and Error Propagation.
- Data Orthogonalisation.

Methodology.

- Vision algorithms must deliver information with which to make practical decisions regarding interpreting the data present in an image.

- Probability is the only self-consistent computational framework for data analysis.

- Probability theory must form the basis of all statistical analysis processes.

- The most direct form of information regarding an hypothesis is the posterior (often conditional) probability.

- The most effective/robust algorithms will be those that match most closely the statistical properties of the data.

- There are several common models for statistical data analysis all of which can be related at some stage to the principle of maximum likelihood.

- An algorithm which takes correct account of all of the data will yield an optimal result.
Basic Definitions of Probability.

- \( P(A) \) probability of event A.
- \( P(\bar{A}) = 1 - P(A) \) probability of non-event A.
- \( P(A, B) \) probability of simultaneous events A and B.
- \( P(A + B) \) joint probability of events A or B.
- \( P(A|B) \) probability of event A given event B.
- \( P(A|B, C) \) probability of event A given events B and C.
- \( P(A|B + C) \) probability of event A given events B or C.
- \( P(A = B) \) probability of equivalence of events A and B.
- \( p(x) \) probability density (not a probability).
- \( P(x_1 < x < x_2) \) probability that \( x \) is between \( x_1 \) and \( x_2 \)
  \( = \int_{x_1}^{x_2} p(x) \, dx \).

Warning: Expressions relating probabilities do not reveal the assumptions with which these results were derived.
Bayes Theorem.

The basic foundation of probability theory follows from the following intuitive definition of conditional probability.

\[ P(A, B) = P(A|B)P(B) \]

In this definition events \( A \) and \( B \) are simultaneous and have no (explicit) temporal order we can write

\[ P(A, B) = P(B, A) = P(B|A)P(A) \]

This leads us to a common form of Bayes Theory, the equation:

\[ P(B|A) = P(A|B)P(B)/P(A) \]

which allows us to compute the probability of one event in terms of observations of another and knowledge of joint distributions.
Maximum Likelihood

Starting with Bayes theorem we can extend the joint probability equation to three and more events

\[ P(A, B, C) = P(A|B, C)P(B, C) \]

\[ P(A, B, C') = P(A|B, C')P(B|C')P(C') \]

For \( n \) events with probabilities computed assuming a particular interpretation of the data (for example a model \( Y \))

\[ P(X_0, X_1, X_2...X_n|Y)P(Y) = \]

\[ P(X_0|X_1, X_2...X_nY)P(X_1|X_2...X_nY)......P(X_n|Y)P(Y) \]

- Maximum Likelihood statistics involves the identification of the event \( Y \) which maximises such a probability. In the absence of any other information the prior probability \( P(Y) \) is assumed to be constant for all \( Y \).

- Even if the events were simple binary variables there are clearly an exponential number of possible values for even the first term in \( P(X|Y) \) requiring a prohibitive amount of data storage.

- In the case where each observed event is independent of all others we can write.

\[ P(X|Y) = P(X_0|Y)P(X_1|Y)P(X_2|Y)...P(X_n|Y) \]
Dealing with Binary Evidence.

If we make the assumption that the event $X_i$ is binary with probability $P(X_i)$ then we can construct the probability of observing a particular binary vector $X$ as

$$P(X) = \prod_i P(X_i)^{X_i} P(\tilde{X}_i)^{\tilde{X}_i}$$

or

$$P(X) = \prod_i (P(X_i)^{X_i}(1 - P(X_i))^{(1-X_i)}$$

The log likelihood function is therefore

$$\log(P) = \sum_i X_i \log(P(X_i)) + (1 - X_i) \log(1 - P(X_i))$$

This quantity can be or directly evaluated in order to form a statistical decision regarding the likely generator of $X$. This is therefore a useful equation for methods of statistical pattern recognition.

eg:

$$X = (0, 1, 0, ..., 1)$$

and

$$P(X) = (0.1, 0.2, 0.05, ..., 0.9)$$
Dealing with Data Distributions.

- The generation process for a histogram, making an entry at random according to a fixed probability, is described by the Poisson distribution.

The probability of observing a particular number of entries $h_i$ for an expected probability of $p_i$ is given by

$$P(h_i) = e^{p_i} \frac{p_i^{h_i}}{h_i!}$$

- For large expected numbers of entries this distribution approximates a Gaussian with $\sigma = \sqrt{h_i}$.

- The limit of a frequency distribution for an infinite number of samples and bins of infinitesimal width defines a probability density distribution.

These two facts allow us to see that the standard $\chi^2$ statistic is appropriate for comparing two frequency distributions $h_i$ and $j_i$ for large measures.

$$-2 \log(P) = \chi^2 = \sum_i (h_i - j_i)^2 / (h_i + j_i)$$

ie:

$$e^{-\log(P)} = \prod_i e^{-\chi^2_i / 2}$$
Dealing with Functions.

If we now define the variation of the observed measurements $X_i$ about the generating function with some random error, the probability $P(X_0|X_1X_2...X_NaY_0)$ will be equivalent to $P(X_0|aY_0)$.

Choosing Gaussian random errors with a standard deviation of $\sigma_i$ gives

$$P(X_i) = A_i \exp\left(\frac{-(X_i - f(a, Y_i))^2}{2\sigma_i^2}\right)$$

where $A_i$ is a normalization constant. We can now construct the maximum likelihood function

$$P(X) = \prod_i A_i \exp\left(\frac{-(X_i - f(a, Y_i))^2}{2\sigma_i^2}\right)$$

which leads to the $\chi^2$ definition of log likelihood

$$\log(P) = -\frac{1}{2} \sum_i \frac{(X_i - f(Y_i))^2}{\sigma_i^2} + \text{const}$$

- This expression can be maximized as a function of the parameters $a$ and this process is generally called a least squares fit.

- Least squares fits are susceptible to fliers (outliers).

- The correct way to deal with these leads to the methods of robust statistics.
Covariance Estimation.

For locally linear fit functions \( f \) we can approximate the variation in a \( \chi^2 \) metric about the minimum value as a quadratic. We will examine the two dimensional case first, for example:

\[
z = a + bx + cy + dxy + ex^2 + fy^2
\]

This can be written as

\[
\chi^2 = \chi^2_0 + \Delta X^T C_x^{-1} \Delta X \quad \text{with} \quad \Delta X = (x, y)
\]

where \( C_x^{-1} \) is defined as the inverse covariance matrix

\[
C_x^{-1} = \begin{bmatrix}
u & v \\
w & s
\end{bmatrix}
\]

Comparing with the above quadratic equation we get

\[
\chi^2 = \chi^2_0 + x^2u + yxw + xyv + y^2s
\]

where

\[
a = \chi^2_0, \quad b = 0, \quad c = 0, \quad d = w + v, \quad e = u, \quad f = s
\]

Notice that the \( b \) and \( c \) coefficients are zero as required if the \( \chi^2 \) is at the minimum.
Starting from the χ² definition using the same notation as previously.

\[ \chi^2 = \frac{1}{2} \sum_i \frac{(X_i - f(y_i, a))^2}{\sigma_i^2} \]

We can compute the first and second order derivatives as follows:

\[ \frac{\partial \chi^2}{\partial a_n} = \sum_i \frac{(X_i - f(y_i, a))}{\sigma_i^2} \frac{\partial f}{\partial a_n} \]

\[ \frac{\partial^2 \chi^2}{\partial a_n \partial a_m} = \sum_i \left( \frac{\partial f}{\partial a_n} \frac{\partial f}{\partial a_m} \right) \frac{\partial^2 f}{\partial a_n \partial a_m} \]

The second term in this equation is expected to be negligible giving

\[ = \sum_i \left( \frac{\partial f}{\partial a_n} \frac{\partial f}{\partial a_m} \right) \frac{\partial^2 f}{\partial a_n \partial a_m} \]

The following quantities are often defined.

\[ \beta_n = \frac{1}{2} \frac{\partial \chi^2}{\partial a_n} \]

\[ \alpha_{nm} = \frac{1}{2} \frac{\partial^2 \chi^2}{\partial a_n \partial a_m} \]

As these derivatives must correspond to the first coefficients in a polynomial (Taylor) expansion of the χ² function then,

\[ C = \alpha^{-1} \]

And the expected change in χ² for a small change in model parameters can be written as

\[ \Delta \chi^2 = \Delta a^T \alpha \Delta a \]
Optimal Combination.

Given two estimates of a set of parameters $a_1$ and $a_2$ and their covariances we can combine the two sets of data as follows

$$a_T = \alpha_T^{-1}(\alpha_1 a_1 + \alpha_2 a_2)$$

with

$$\alpha_T^{-1} = (\alpha_1 + \alpha_2)^{-1}$$

The simplest version of this approach is for the combination of two measurements $q_1, q_2$ with error estimates $\sigma_1, \sigma_2$, such that;

$$q_T = \frac{\sigma_1^2 + \sigma_2^2}{\sigma_1^2 \sigma_2^2}(q_1/\sigma_1^2 + q_2/\sigma_2^2)$$

* This method combines the data in the least squares sense, that is the approximation to the $\chi^2$ stored in the covariance matrices has been combined directly to give the minimum of the quadratic form.

* The method can be rewritten slightly giving

$$a_T = a_1 + \alpha_T^{-1} \alpha_2 \Delta a$$

where $\Delta a = a_2 - a_1$.

* In this form the method is directly comparable to the information filter form of the Kalman filter.
**Error Propagation.**

In order to use a piece of information $f(X)$ derived from a set of measures $X$ we must have information regarding its likely variation.

If $X$ has been obtained using a measurement system then we must be able to quantify measurement accuracy.

Then

$$(\Delta f)^2(X) = \nabla f^T C_X \nabla f$$

example 1: the Poisson distribution $s$

$$t = \sqrt{s}$$

then we can show, using a simplified form of error propagation for one parameter, that the expected variance on $t$ is given by

$$\Delta t = \frac{\partial t}{\partial s} \Delta s$$

$$= \frac{-1}{2}$$

Thus the distribution of the square-root of a random variable drawn from a Poisson distribution with large mean will be constant.
Data Orthogonalization.

Under any practical circumstance the data delivered by a system may be correlated. It is then that we may need to preprocess the data to remove these correlations. This process is often called PRINCIPAL COMPONENT ANALYSIS.

We can define the correlation matrix

\[ R = \sum_j (X_j - X_m) \otimes (X_j - X_m) \]

where \( X_j \) is an individual measurement vector from a data set and \( X_m \) is the mean vector for that set.

It can be shown that orthogonal (linearly independent) axes correspond to the eigenvectors \( V_k \) of the matrix \( R \). Solution of the eigenvector equation

\[ RV_k = \lambda_k V_k \]

The method known as Singular Value Decomposition (SVD) approximates a matrix by a set of orthogonal vectors \( W_l \) and singular values \( w_l \).

\[ R = \sum_l \frac{1}{w_l^2} W_l \otimes W_l \]

If we multiply both sides of the equation by one of these vectors \( W_k \)

\[ RW_k = \sum_l \frac{1}{w_l^2} W_l \otimes W_l.W_k \]

we see that the singular vectors satisfy the eigenvector equation with

\[ \lambda_k = \frac{1}{w_k^2} \]
The definition of Likelihood as presented in this lecture uses probabilities, ie:

$$\log(L) = - \sum_i \log(P(X_i))$$  \hspace{1cm} (1)

where $X_i$ is a quantised measurement.

The definition of likelihood found in many publications is actually in terms of probability densities, ie:

$$\log(L) = - \sum_i \log(p(x_i))$$  \hspace{1cm} (2)

where $x_i$ is a real valued measurement and $p(x)$ is a probability density.

Using the definition

$$P(X_i) = \int_{x-\delta}^{x+\delta} p(x) \, dx$$

explain by reference to sampled distributions how (1) is invariant to non-linear redefinition of the measurement ($y = f(x)$) while (2) is not.

What are the implications of this finding?
Stability of Image Processing Algorithms.

N. A. Thacker

- The Importance of Stability.
- Error Propagation.
- Image Arithmetic.
- Linear Filters.
- Histogram Equalisation.
- Monte-Carlo Techniques.
- Generating Random Variates.
- Generating Distributions.

see TINA memo 2001-007, Performance Characterisation in Computer Vision.
The Importance of Stability.

In simple image processing the requirements of an image processing algorithm may be purely to enhance the image for viewing.

But; the aim of advanced image processing to produce an image that makes certain information explicit in the resulting image values for automated data extraction.

eg: edge strength maps.

Generally, high values located over features of interest. The process which determines a good algorithm is its behaviour in the presence of noise, in particular does the resulting image give results which really can be interpreted purely on the basis of output value.

ie: is a high value genuine or just a product of the propagated noise.

In this lecture we will cover two ways of assessing algorithms: Error Propagation and Monte-Carlo techniques.
Alternativey, every algorithm should provide detailed error estimates and make appropriate use of them in subsequent stages.
Error Propagation.

General Approach for Error Propagation (Recap).

\[ \Delta f^2(X) = \nabla f^T C_X \nabla f \]

where \( \nabla f \) is a vector of derivatives

\[ \nabla f = \left( \frac{\partial f}{\partial X_1}, \frac{\partial f}{\partial X_2}, \frac{\partial f}{\partial X_3}, \ldots \right) \]

and \( \Delta f(X) \) is the standard deviation on the computed measure.

If we apply this to image processing assuming that images have uniform random noise then we can simplify this expression to

\[ \Delta f_{xy}^2(I) = \sum_{nm} \sigma_{nm}^2 \left( \frac{\partial f_{xy}}{\partial I_{nm}} \right)^2 \]

ie: the contribution to the output from each independent variance involved in the calculation is added in quadrature.
Image Arithmetic.

We can drop the $xy$ subscript as it is not needed.

Addition:

$$O = I_1 + I_2$$

$$\Delta O^2 = \sigma_1^2 + \sigma_2^2$$

Division:

$$O = \frac{I_1}{I_2}$$

$$\Delta O^2 = \frac{\sigma_1^2}{I_2^2} + \frac{I_1^2 \sigma_2^2}{I_2^4}$$

Multiplication:

$$O = I_1 . I_2$$

$$\Delta O^2 = I_2^2 \sigma_1^2 + I_1^2 \sigma_2^2$$

Square-root:

$$O = \sqrt(I_1)$$

$$\Delta O^2 = \frac{\sigma_1^2}{I_1}$$

Logarithm:

$$O = \log(I_1)$$

$$\Delta O^2 = \frac{\sigma_1^2}{I_1^2}$$
Polynomial Term:

\[ O = I^n_1 \]
\[ \Delta O^2 = (nI^n_1 - 1)^2\sigma^2_1 \]

Square-root of Sum of Squares:

\[ O = \sqrt{I^2_1 + I^2_2} \]
\[ \Delta O^2 = \frac{I^2_1\sigma^2_1 + I^2_2\sigma^2_2}{I^2_1 + I^2_2} \]

Notice that some of these results are independent of the image data. Thus these algorithms preserve uniform random noise in the output image.

Such techniques form the basis of the most useful building blocks for image processing algorithms.

Some however, (most notably multiplication and division) produce a result which is data dependent, thus each output pixel will have different noise characteristics. This complicates the process of algorithmic design.
For Linear Filters we initially have to re-introduce the spatial subscript for the input and output images $I$ and $O$.

$$O_{xy} = \sum_{nm} h_{nm} I_{x+n, y+m}$$

where $h_{nm}$ are the linear co-efficients.

Error propagation gives:

$$\Delta O_{xy}^2 = \sum_{nm} (h_{nm} \sigma_{x+n, y+m})^2$$

for uniform errors this can be rewritten as

$$\Delta O_{xy}^2 = \sigma^2 \sum_{nm} (h_{nm})^2 = K \sigma^2$$

Thus linear filters produce outputs that have uniform errors.

Unlike image arithmetic, although the errors are uniform they are no-longer independent because the same data is used in the calculation of the output image pixels. Thus care has to be taken when applying further processing.

For the case of applying a second linear filter this is not a problem as all sequences of linear filter operations can be replaced by a combined linear filter operation, thus the original derivation holds.
Histogram Equalisation.

For this algorithm we have a small problem as the differential of the processing process is not well defined.

If however we take the limiting case of the algorithm for a continuous signal then the output image can be defined as:

\[ O_{xy} = \frac{\int_{0}^{I_{xy}} f \, dI}{\int_{0}^{\infty} f \, dI} \]

where \( f \) is the frequency distribution of the grey levels (ie the histogram.

This can now be differentiated giving

\[ \frac{\partial O_{xy}}{\partial I_{xy}} = K \, f_{I_{xy}} \]

ie: the derivative is proportional to the frequency of occurrence of grey level value \( I_{xy} \) and the expected variance is:

\[ \Delta O_{xy}^2 = K \, \sigma_{xy}^2 f_{I_{xy}}^2 \]

Clearly this will not be uniform across the image, nor would it be in the quantized definition of the algorithm.

Thus although histogram equalisation is a popular process for displaying results (to make better use of the dynamic range available in the display) it should generally be avoided as part of a Machine Vision algorithm.
Monte-Carlo Techniques.

Differential propagation techniques are inappropriate when:

- Input errors are large compared to the range of linearity of the function.
- Input distribution is non-Gaussian.

The most general technique for algorithm analysis which is still applicable under these circumstances is known as the Monte-Carlo technique.

This technique takes values from the expected input distribution and accumulates the statistical response of the output distribution.

The technique requires simply a method of generating random numbers from the expected input distribution and the algorithm itself.
Monte-Carlo Techniques II.

Non-Linear Function. Non-Gaussian Errors.

Monte-Carlo Technique.

Algorithm $f(x,y)$

Monte-Carlo Technique.
Generating Random Variates.

In order to generate a random number from a chosen distribution we generally start by generating a uniform (flat) random variate (between 0 and 1) and then compute from this a value from the required distribution.

Uniform random variates are generally generated using Linear Congruential Generation.

\[ rand(t + 1) = (a \, rand(t) + c) \mod(m) \]

this simply means that every random number in the sequence can be generated using the number before it.

The number generated is clearly not truly random and is referred to as psuedo-random.

However, this does not matter provided that the random number generator does not correlate with the programme that will use the data.
Generation of a uniform random variate generally looks something like the following

```c
void srand(unsigned int seed);
int rand(void);
.
.
.
x = rand()/(RAND_MAX+1.0);
```

The LCG series will eventually repeat itself and low order bits are generally strongly correlated.

If $a$ and $c$ are properly chosen the sequence will be indistinguishable from a true random number and will repeat itself with a period $m$.

**WARNING:** Implementations are botched particularly in the ANSI standard which only gives $m = 32767$. See “Numerical Recipies” for further details.
Generating Distributions.

The general technique for generating a variate from the distribution $f(x)$ using $x$ is to solve for $y$ in

$$x = \frac{\int_{-\infty}^{y_0} f(y) \, dy}{\int_{-\infty}^{\infty} f(y) \, dy}$$

ie: $x$ is used to locate a variable some fraction of the way through the integrated distribution.

eg: a Gaussian distribution leads to the BOX MULLER method

$$y_1 = \sqrt{-2\ln(x_1)} \cos(2\pi x_2)$$
$$y_2 = \sqrt{-2\ln(x_1)} \sin(2\pi x_2)$$

which generates two Gaussian random deviates $y_1$ and $y_2$ for every two input deviates $x_1$ and $x_2$.

Armed with distribution generators we can generate many alternative images for statistical testing from only a few examples of image data. Examples:

- drop out noise in images.
- bit inaccuracy and thermal noise.
- feature location and orientation accuracy.
- stereo matching reliability. etc...
Problems:

• What happens to the least significant bit in LCG’s.
• Try to derive the Box-Muller Gaussian deviate generator.
Pattern Recognition.

N. A. Thacker

• Hypercube Classifier

• Probability and Probability Density

• Nearest Neighbour Classifiers

• Parametric Classifier

• Linear Discriminant Analysis

• Mean Shift Algorithm

• Training Classifiers

• Testing Classifiers

• Advanced Classifiers
Hypercube Classifier.

A simple approach
eg: binary image arithmetic;
\[(G(i, j) > g_1) \ast (G(i, j) < g_2) \ast (H(i, j) > h_1) \ast (H(i, j) < h_2)\]

Simple but does not compactly describe the true distribution of data.

Is there a theoretical optimal approach?
Probability and Probability Density.

We can define a probability density \( p(x) \) as a function of one or more variables.

To compute a probability we must integrate over an interval

\[
P(x_1 < x < x_2) = \int_{x_1}^{x_2} p(x) \, dx
\]

for slowly varying densities

\[
P(x_1 < x < x_2) = p((x_1 + x_2)/2)(x_2 - x_1)
\]

or alternatively for small intervals \((x_2 - x_1) = \delta_x\)

\[
P(x) = p(x)\delta_x
\]

Probability densities are not generally interchangable with probabilities.

This is important!
A decision based upon the classification probability eg:

\[ P(a|g, h) = \frac{P(g, h|a)}{P(g, h|a) + P(g, h|b) + P(g, h|c)} \]

\approx \frac{p(g, h|a)}{p(g, h|a) + p(g, h|b) + p(g, h|c)}

will minimise the classification error.

In general

\[ P(\omega_m|X) = \frac{P(X|\omega_m)}{\sum_n P(X|\omega_n)} \]

This best case performance is called the **Bayes Error Rate**.

eg: K Nearest Neighbours, Expectation Maximisation, discriminant analysis, Support Vecotor Machines (SVM).
Nearest Neighbour Classifiers.

Given sample data we can make an estimate $P'(\omega_m|x)$ of the classification probability $P(\omega_m|x)$ by counting the relative number of samples in an interval or region.

In this example $P'(b|g, h) = 4/(4+6) = 0.4$.

The nearest neighbour classifier does not define a region but a fixed number of nearest samples $N$.

Then $P'(b|g, h) = n_b/N$, ie: the fraction of the required class.

In the limit $N \to \infty$ \hspace{1cm} $P'(\omega_m|x) \to P(\omega_m|x)$.
Parametric Bayes Classifier.

If class distributions are assumed to be Normal (Gaussian)

\[ p(x|\omega_m) = k_1 \exp\left[-\frac{1}{2}(x - x_m)^T C_m^{-1}(x - x_m)\right] \]

where \( C_m \) is the covariance matrix of class \( m \) and \( D_m \) is the mahalanobis to \( x_m \).

We can re-write the classification rule as

if \( k_1 \exp[-D_m] > k_1 \exp[-D_n] \) class is \( m \)

or

if \( D_m < D_n \) class is \( m \)
Linear Discriminant Classifier.

For a two class problem we can imagine a boundary curve between two probability distributions which passes through all points where $P(\omega_m|x) = 0.5$

We can therefore approximate the decision process using a linear classification rule, eg;

$$G(x) = k_0 + \sum_i k_i x_i$$

where we define the parameters of $G$ so that

- $G = 0$ is the decision hyperplane.
- $G > 0$ corresponds to the required class
- $G < 0$ is not the required class

Support Vector Machines extend this idea to non-linear decision boundaries.
Mean Shift Algorithm.

For cases where we have no a-priori definitions of classes we can attempt to **cluster** the data into high density regions.

One approach involves taking small steps in the direction of the local mean.

This process converges on the local maximal densities, but also defines the decision boundaries between two clusters at $P(\omega_m|x) = 0.5$.

The up-hill direction can be learned on a training data set and then then applied (see Tina Memo :2001-015).
Training Classifiers.

Training a classifier requires a representative data set from which to estimate the classifier parameters.

Non-parametric techniques, such as KNN and Mean Shift, require no explicit parameter estimation.

For parametric techniques there are two approaches, which are based upon the characteristics of the chosen method.

- Methods which model probability densities can simply concentrate on estimating the parameters which best describe the observed data, these estimation techniques are based upon Likelihood.
  For an accurate density model this will implicitly also produce the optimal decision.

- Methods which require only the decision boundary can be trained directly to minimise the classification error.
Testing Classifiers.

What we really care about is how a classifier will work on new data.

For finite quantities of training data performance of the classifier will be better on the training data set than any other sample from the same system.
This subtle point has an effect if we need to make a choice between alternative classifiers, or similar classifiers with different numbers of parameters.
The pattern recognition and neural network communities define the term generalisation to mean the performance of a classifier on data different to that used for training.
Obtaining meaningful estimates of generalisation are difficult when there is a limited data sample, dividing the data into two sets (one to train, one to test) means we only have half of the available data for training.
A popular alternative is to train multiple times, while withholding one item of data (leave one out) for testing. The performance of the method is then estimated by averaging over all tests.
Leave one out testing is pessimistic in its predictions of performance.
You can get a better estimate of the true limit of performance of a classifier by taking the average between leave one out result and the result from the full training set.
**Other Advanced Classifiers.**

- **Neural Networks**
  - This field is so broadly defined that it includes just about every classifier ever invented.
  - Popular approaches, such as Multi-Layer Perceptrons, can be difficult to train, particularly on large problems.
  - Execute quickly.
  - Currently “out of fashion”.

- **Support Vector Machines**
  - Easier to train, but more restricted in their application to problems which require only one decision boundary.
  - Execute at about the same speed as a neural network.
  - Currently “in fashion”.

- **Fuzzy Logic**
  - "Kind of works", probability theory done badly.
  - To be avoided.

There is no reason to believe that any specific approach is necessarily always going to be the best. Generally all of the principled approaches work about as well as each other for an equivalent number of model parameters.

This view of things changes if we want to build learning (intelligent) systems (see: Tina Memo 2006-008).
Region Based Image Segmentation.

N. A. Thacker

- Standard Approaches.
- Colour Segmentation.
- Multi-spectral Segmentation.
- Texture Segmentation.
Standard Approaches.

We can attempt to extract structure using image features, such as edges (see “Edge Based Vision” lecture), but we can also attempt to extract connected regions.

Knowledge of the region gives us knowledge of the its boundaries.

Although we might expect that boundaries might generally be consistent with edges, some data (i.e., textured regions) do not have this property.

The following techniques can be seen as primitive pattern recognition:

- Thresholding (part of hypercube classifier)
- Adaptive thresholding (parameter estimation)
Other techniques found in textbooks differ from these by making use of regional dependencies;

- Binary morphology (adjacent pixels have similar labels $\rightarrow$ connectivity)
- Skeletonisation
- Relaxation labelling (looks like probability but isn’t)
- Split and merge
- Optimisation (adjacent pixels have similar labels $\rightarrow$ cost function)

$$C = \alpha \sum_{ij} |L(i, j) - L(i-1, j)| + |L(i, j) - L(i, j-1)|$$

$$+ \beta \sum_{ij} |L(i, j) - D(i, j)|$$

choose $L$’s which minimise $C$ and look like data $D$.

eg: **simulated annealing**

Stochasite optimisation of $C$ involves changing to the new set of labels $L_t$ if a random variable ($0 < x < 1$) is less than

$$P(L_t) = exp[-(C_t - C_{t-1})/T]$$

Where $T$ is the ‘temperature’.

Useful for avoiding local minima.
The above techniques do not make explicit use of probability theory.

We want techniques which are based upon probability theory so that we have a chance of identifying an optimal solution, ie:

\[ C = -\log[P(I|Model)] \]

We then need to ensure that all terms in the cost function are applicable to the data.

This requires us to consider the distributions found in real world samples.
Colour Segmentation.

Colour images are obtained with 3 components, red, green and blue \((r, g, b)\). This 3D data has one dimension which generally we don’t care much about for recognition purposes.

This is an example of recognition invariance.

These components can be rotated into various spaces, Intensity Saturation and Hue (ISH), YUV etc. to make the colour explicit from the intensity.

These two components can then be treated as inputs to a pattern recognition system in order to group colour regions.
A good way of segmenting arbitrary colour images is to use a variant of the Mean Shift algorithm.

First loop over all of the data identifying statistically distinguishable examples (knot points).

Then link these knot points to identify the direction of increasing density.

Track the input data along linked paths to identify common labels.
Labels are generated according to data distributions; ie: sampled probability densities. (See: Tina Memo 2001-015)
Colour Segmentation III.

Real data

Image of data labels
Multi-Spectral Segmentation.

Each grey-level can be used as one component of the measured data vector $\mathbf{x} = (x_1, x_2, x_3, x_4)$.
Pure and partial volume models (simulation).

Real Data $x_1$ vs $x_2$ and $x_3$ vs $x_4$

Parameters for the density model are estimated using **Expectation Optimisation** (EM), which involves iteration of the following steps:

- use an initial estimate of probability classifications $P(\omega_m|x)$ to generate a new estimate of the parameters ($x_m$ and $C_m$).
- use the new parameters to re-estimate $P(\omega_m|x)$.

There is a proof that this process will converge on a local optima of the Likelihood of the $N$ data vectors $x_n$.

$$L = \sum_n \sum_m p(x_n|\omega_m)$$
Multi-Spectral Segmentation III.

Most probable tissue volume estimates for $\omega_1 - \omega_6$.

(See: Tina memo 2003-007)
Texture Segmentation.

We have already seen how the concept of spatial frequency is not particularly useful when based directly on a 2D Fourier transform. In particular;

- An image is not periodic.
- Fourier amplitudes are global parameters which do not localise well.
- Fourier based algorithms often do not work well for estimation tasks (non-statistical).
- Periodic structures, even when present, need not have the lattice frequencies present in a Fourier transform.

Non periodic signals also occur in temporal data analysis, here the non periodicity and localisation problem is solved by windowing (eg: Hanning window).

The sine curve is multiplied by a function which approaches zero at the edges.
Texture Segmentation II.

For 2D signals if we select the window function as a 2D Gaussian $G(\sigma, x, y)$ we can construct windowed wavelets which can be computed at each location in the image to give a local representation, eg:

$$H_s(u) = G(\sigma_u, x, y) \sin(2\pi ux)$$

$$H_c(u) = G(\sigma_u, x, y) \cos(2\pi ux)$$

Gaussian windowed sine curves are called Gabor filters and can be oriented along any direction in the image plane $\theta$.

We can construct a set of Gabor responses at any location in an image for a range of spatial scales and orientations.

$$A(u, \theta) = \sqrt{H_s(u, \theta)^2 + H_c(u, \theta)^2}$$

for use in a pattern recognition system

$$\mathbf{x} = (A(u, \theta_1), (A(u, \theta_2), (A(u, \theta_3), ..)$$

This representation describes the local texture information present at each point.

This representation has problems which will be discussed in later lectures.
Edge-Based Vision.

N. A. Thacker

• Marr’s theory.

• Edge Detection.

• Scale Space.

• Statistical Thresholding.

• Information Based Features.

• SIFT.

• The Primate Retina.

• Model Matching.
Marr’s Theory of Vision.

Edge correspond to physical events (diffeomorphic equivalence)

Levels of representation;

• feature detection → raw ‘primal’ sketch’ (edges etc.)
• edge grouping → full ‘primal sketch’ (curves)
• surface extraction → ‘2.5D sketch’
• model matching → 3D model

This theory matches pre-conceptions of how vision might work, and has consequently been highly influential, but in the 2 decades since it’s publication it has not proved possible to build a reliable system in this way.

Attempts to represent and extract volumetric representations have included the use of;

• skeletonisation (Gonzales and Wintz)
• smooth local symmetries (Brady and Asada)
• generalised cylinders (Pentland)
• medial axis transform
**example: Medial Axis Transform.**

A medial axis transform defines all the maximally inscribed circles (tangent to at least 2 boundary points). This can be obtained using a distance transform (ie: a morphological filter).

- Axis based representations are inherently very sensitive to factors such as noise and occlusion.
- The early vision stages provide only sparse data which is unreliable without some ‘top down’ constraints.
- The later stages require a method to represent all possible 3D objects with parameters computed from the available evidence.

These problems have led to a more pragmatic approach, which focuses on detectable features which are useful for specific visual tasks (ie: edges and corners) rather than presupposing the necessary extraction of complete multi-layered representations.
Second Derivative Edge Filters.

eg: Laplacian, etc (mathematical definition).

\[
\begin{pmatrix}
  -1 & -1 & -1 \\
  -1 & 8 & -1 \\
  -1 & -1 & -1 \\
\end{pmatrix}
\]

zero response in uniform region

Located at zero crossing.

Edge detectors often enhance noise, \(\rightarrow\) Marr-Hildreth (Laplacian of a Gaussian + thinning).
First Derivative Edge Filters.

eg: Roberts, Sobel, Canny etc (noise tolerant).

\[
\begin{array}{ccc}
-1 & 0 & 1 \\
-2 & 0 & 2 \\
-1 & 0 & 1 \\
\end{array}
\]

\[
\begin{array}{ccc}
1 & 2 & 1 \\
0 & 0 & 0 \\
-1 & -2 & -1 \\
\end{array}
\]

- Edge Strength \( g = (f_x^2 + f_y^2)^{1/2} \)
- Orientation \( n = \nabla f / |\nabla f| \)
- Located at ridge in \( g \) (ie: \( g \) greater than at least 6 neighbours).
Scale Space.

What is a real edge?

- Edges exist at many scales.
- Important edges persist across scales.
- Gaussian filters define optimally compact features.

eg;

Canny; (based on principle of optimal noise filtering and feature localisation)

- Convolve with Gaussian (Canny’s filter slightly different)
- Compute edge strength (eg: sum square gradient).
- Connect edges along the local ridge maxima (eg: edge strength > 6 neighbours) above a threshold (non-maximal suppression).
- Hysteresis thresholding (see below).
- Define edges only at locations which are stable across scale.

Edges detected above threshold at multiple scales, but what is the threshold?
Treat edge strength thresholding as a null hypothesis test. We can do this because the process of edge detection using convolution is stable in the presence of noise and generates equivalent noise responses for images with uniform independent random noise. (See lecture on Algorithmic Stability.)

Probability that the observed feature response is due to noise.

Edge are extended structures, so this approach can be improved by taking account of local context.

Probability that an edge will be connected to another edge. **Hysteresis thresholding** (high and low thresholds);

- Any pixel $> high\ threshold$ is an edge.
- Any pixel $> low\ threshold$ and connected to an edge is an edge.
- possibly iterate

This will detect 95% of anything that you might have expected to detect.

Detected edges can be fitted to lines or curves (eg: least squares), or used for structure detection (eg: Hough Transform).
Scale Invariant Feature Transform (SIFT).

With this approach the evidence for a feature is analysed across subsamples at a range of scales to find the maximal response.

The selected response is expected to be scale invariant.

The region around that location is then used to construct a histogram of local image orientation.

The histograms are matched to a database determine correspondence.

Evidence is accumulated to generate a hypothesis for a specific object (eg: voting).
The Primate Retina.

The structure of the retina is quite complex (rods and cones). Psychophysical interpretation of perceived resolution varies as a radial function. Uniform in the centre and decreasing at the edge.

The following model can be used to explain how a scale invariant detection system might be constructed.

![Decreasing spatial resolution](image)

Visual representations supported at a range of scales and when matching the best response is selected.

Benefits;

- Stored object/feature representation does not need to be scale invariant.
- Can have fewer scaled versions of object stored in memory.
**Information Based Features.**

The features we detect always have associated practical uses;

- localisation,
- identification of regions for pattern matching,
- measurement of shape structure.

Why not choose our features to optimise the utility for these tasks → **Fisher Information** \( (1/variance) \)

- Best localisation given at maxima of the derivative of local auto-correlation (eg: Harris Corner detection).
- Most significant grey level changes given at maxima in local variance (eg: difference of Gaussian).
- Most informative angular measurement given at maxima in conventional edge strength.

For this and other statistical constraints see: TINA memo 2007-001.
Interpretation Trees.

A set of $M$ model features ($m_0, m_1, m_2...$) is matched to a set of $S$ scene features ($s_0, s_1, s_2...$) (Grimson 87). The algorithm proceeds from a single match by applying consistency constraints to prune the tree. Initial constraints are termed *unary*, later constraints are termed *binary*.

- The tree size is of order $M^S$, ie: a possible ‘combinatorial explosion’.
- This is made significantly worse if there is no 1:1 correspondence possible between scene and model.
- The search space can be further limited using geometric constraints (angle, distance).

If we use geometric relationships at the unary level, then we do not need an interpretation tree and can avoid the combinatorial problems. (see: Pairwise Geometric Histograms)
3D Model Matching.

Wire-frame matching proceeds as follows;

• Fit edge strings to extract 2D primitives.

• Project into 3D using stereo (ie: left and right images, see later lectures).

• Identify a maximal ‘clique’ of 3D features with the same geometric configuration as features in a 3D wire-frame model.

• Compute the 3D transformation (pose) which takes the model into the image and verify.

TINA (www.tina-vision.net)
Similar to Marr, but no surfaces!
Bottom-up feature based solution of object detection is important, but Marr’s theory has never achieved the performance required for practical systems (his book still provides a good exposition of the difficulties involved in understanding human perception).

Modern approaches are generally view and appearance based (concentrating on edges) and abandon the 2.5D and 3D stages. First derivative filters and second derivative filters are mathematically equivalent, but the former make for more reliable algorithms.

Scale space issues and the effects of noise are both important. The process of detection is best understood in terms of statistical thresholding.

More recently scale based detection has been re-invented in the guise of the SIFT detector, which takes a maximum response across scale as well as location.

These ideas can also be applied to non-connected features (ie: corners). Interpretation trees are replaced by matching using relational geometry.
Questions.

1. Make a list of all the invariance characteristics we may wish a shape recognition system to possess.
To what extent does SIFT achieve these invariances?

2. The convolution of an image with a Gaussian $G \otimes I$ can be considered as a weighted mean $\bar{I} = \sum_{ij} G_{ij} I_{ij}$
Show that a weighted variance calculation $\sum_{ij} G_{ij} (\bar{I} - I_{ij})^2$ can be written as a difference of two convolutions of $I$ and $I^2$.
What is such a measure quantifying, and what might this be useful for?

3. Assuming equal levels of independent noise on the spatial derivatives of an image ($u = \frac{\partial I}{\partial x}$ and $v = \frac{\partial I}{\partial y}$) apply error propagation to
\[ \theta = \text{atan}[u/v] \]
to get the variance on $\theta$.
What is the relationship of this result to the Canny algorithm?
Hough Shape Detectors.

N. A. Thacker

- The Hough Transform
- Lines Using the Hough Transform
- Circles Using the Hough Transform
- Statistical Origins
Hough Transform.

Originally used in particle physics to identify charged track trajectories in a magnetic field.

The centre of the target is known and tracks follow circular paths.

Each data point corresponds to a unique particle momentum (proportional to the radius of curvature).
General Principle.

Accumulate votes for each possible interpretation of the data and select the model which accounts for most data points.

The main strength of this method is that it will cope with very noisy data.

example: finding the scaling ratio between two images (A and B).

\[ A_{ij} = \alpha B_{ij} + noise \]

\[ \alpha_{ij} = \frac{A_{ij}}{B_{ij}} \]
Lines can be parameterised in one of two ways;

\[ y = mx + c \]

\[ \rho = x \cos(\theta) + y \sin(\theta) \]

Both of these are a function defined by two constants.

Line based Hough transforms thus require two dimensional accumulator arrays.
Line Detection.
Circle Detection 1.

\[(x - cx)^2 + (y - cy)^2 = r^2\]

if the radius is known then;

\begin{itemize}
  \item a 3-dimensional accumulator needed for arbitrary circle.
  \item entries are made on the surface of a cone.
  \item high dimensionality not practical; 512^2 image needs a 512^3 Hough array!
\end{itemize}
Circle Detection 2.

\((x - cx)^2 + (y - cy)^2 = r^2\)

if the tangent at each point is known then;

The radius can be determined using a second 1 D transform once the circle centre is known.

Breaking the problem down into manageable stages reduces the dimensionality and storage.
Ellipse Detection.

use parameter space reduction

\[
\frac{(x - cx)^2}{a^2} + \frac{(y - cy)^2}{b^2} = 1
\]

- pick a pair of points \( P1 \) \( P2 \)
- measure local tangents
- compute intersection of tangent \( T \)
- compute midpoint \( M \)
- centre \( C \) must lie on the line \( TM \)
Tuple Hough Transforms.

We can construct a Hough transform for any parametric problems, and increment the array based either on a single data point or any number of points (a tuple) up to the number of dimensions of the model (cw. RANSAC).

For a line we can take pairs of points to estimate a unique set of parameters.

For a circle we can take 3 points etc.

We can even apply these ideas to 3D reconstructed data (pose space).

Hough transforms can be understood in terms of the workings of the algorithm but;

- What happens when they don’t work?
  (ie: entries do not make a unified peak)

- Are they theoretically valid?
Deriving Hough Transforms from Probability.

We can relate the accumulation process to estimation of a log probability.

\[-\log P(x|\text{model}) = - \sum_i \log P(x_i|\text{model})\]

For lines formed from pairs of points \((x_i, x_j)\)

\[
\sum_i \sum_j -\log P(x_i, x_j|\text{model})
= \left[- \sum_i \log P(x_i|\text{model})\right] \left[ - \sum_j \log P(x_j|\text{model})\right]
= \left[ -\log P(x|\text{model})\right]^2
\]

Hough transforms are likelihood!

For poorly constrained parameters \(\log P(x_j|\text{model})\)
is poorly localised, for well constrained parameters it is well localised.
Other Ideas.

- Generalised Hough Transform; any shaped object can be located.

- You must know what shape you are looking for!

- Hough transforms can be directly related to methods in robust statistics.
Questions.

Give three reasons why Hough transforms cannot be considered as a general solution for the recognition of edge based descriptions of shape?

Can you derive the accumulation entries necessary for a 3 point tuple circle Hough detector from Likelihood?

How might you estimate the 3D location, or pose, of an object in stereo data using a Hough transform?
Corner detection.

N. A. Thacker

• Edge Detection Recap.
• What is a Corner Detector?
• What are corners good for?
• Approaches to Corner Detection.
• Derivative Based Techniques.
• Median Filter Approaches.
• Auto-Correlation based approaches.
• Evaluating Corner Detectors.
• Summary.
Edge Detection Recap.

Edges have ”diffeomorphic equivalence” with the real world. That is edges on the whole correspond to real 3D structure both in terms of presence and location.

Edges are very useful for scene segmentation and interpretation but correspond to only the simplest derivative based operation that we can apply to images.

Are there any other derivative based operators which have the same property?
What is a Corner Detector?

Having located a high derivative we know with some certainty its location in a direction perpendicular to the edge but there is less certainty in a direction along the edge. This is known as the aperture problem.

We cannot establish unique correspondence between two sets of edge strings without more information with which to disambiguate the possibilities.

We might imagine that if we could work with derivatives in both possible orthogonal directions we might do better.

This is basically the idea behind corner detection.
What are corners good for?.

Edges are good for defining object shape and scene structure but generally not so good for correspondance and 2D positioning.

Corners are good for feature tracking and calibration.

The definition of a corner in machine vision is not simply a geometric artefact.

It should be interpreted as anything that can be well located in 2D within the image.
The simplest approach to corner detection is template matching.

A simple grey level model is generated which approximates to the average shape of the required corner in the image and it is matched using a correlation based technique.

eg: look for a minima in

\[ C(x, y) = \sum_{n} \sum_{m} (I(x + n, y + m) - T(n, m))^2 \]

where \( T(n, m) \) is a corner template.

This approach has limitations in that the orientation of the corner needs to be taken into account and this can require considerable quantity of processing.

Training a neural network to associate particular image distributions with ”cornerness” would be equivalent.
Derivative Based Techniques.

As we have said, the idea of edge detection is based on the idea of first derivatives of the image. The there are two natural extensions to this, both based on examining second derivative or "curvature" behavior.

We start by defining a local image patch as:
\[ I(x, y) = I(0, 0) + I_x x + I_y y + I_{xx} x^2/2 + I_{xy} xy + I_{yy} y^2/2 + h o t \]
where \( I_n \) and \( I_{nm} \) are the first and second derivatives.

We can now define the Hessian Matrix
\[ J = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix} \]

now for some rotation of the image \( I(x', y') \) about the origin there is a matrix such that the off diagonal terms are zero.
\[ J = \begin{bmatrix} k_1 & 0 \\ 0 & k_2 \end{bmatrix} \]

where \( k_1 \) and \( k_2 \) are the principle curvatures.

These are related to the first and second derivatives by
\[ I_{xx} + I_{yy} = k_1 + k_2 \]
and
\[ Det(J) = I_{xx}I_{yy} - I_{xy}^2 = k_1 k_2 \]

The basic corner detector then defines corner strength as
\[ C = k g = k (I_{x}^2 + I_{y}^2)^{0.5} \]
Such a measure is computed at each pixel in the image. Corner detection then involved locating peaks in such a corner strength image above some corner strength threshold.
The other approach (which builds directly upon edge detection) simply involves looking for places with a high curvature (in the more normal sense) along edge strings.

Unfortunately these techniques are hampered because edge detectors quite often give a low response at corners.

The other problem with these approaches to corner detection is the behavior in the presence of noise.

The curvature computed from second derivatives has nasty noise properties which can be seen from error propagation.

In particular regions of the image with high first derivative have the error in the corner strength boosted.
Median Filter Approaches.

The basic technique for this approach to corner detection involves computing the corner strength image as

\[ C = Med(I) - I \]

This approach has been shown (quite surprisingly) to mathematically almost equivalent to the differential approach.

It therefore must suffer from the same noise problems.
Auto-Correlation based approaches.

Given that we wish to define a corner as a position in the image which can be well located in 2D an alternative approach is to use what Moravec called an interest operator.

This approach computes the minimum change auto-correlation for any direction of shift of the local image patch about each point in the image.

Because this technique is based on auto-correlation, which is a linear operation (see previous lecture), the behaviour in the presence of noise is expected to be better.

The main problems with the technique are the use of a non-radially symmetric window and the need to quantize the auto-correlation directions.

A refinement of this technique due to Harris and Stephens computes the expected minimum auto-correlation from the smoothed derivatives and can be shown to have relatively good noise properties.

However, because the technique is based on a radially weighted auto-correlation measure the detected corners are often systematically shifted away from the geometric corner.
Evaluating Corner Detectors.

The most important criteria which determine the successful use of a corner detector are

- quantity of detected noise (false features)
- repeatability (how often can the same corner be detected)
- location and orientation accuracy

The first and second quantities are a function of the required corner strength threshold, but can typically be found to be 2 % and 80 % respectively for practical situations.

Thus corner detectors are not perfect and any algorithm using them must be designed to deal with missing and outlier data.

The typical corner location accuracy is 0.3 pixels. The typical orientation accuracy is 0.2 radians. Both of these can be estimated under specific image analysis conditions using Monte=Carlo approaches.
Summary.

Corners provide a good way of accurately locating a small number of interesting points in an image.

However, they can be noisy and unreliable.

Designing better corner detectors is still an open research question.
Advanced Linear Filters.

N. A. Thacker

- FIR and IIR Filters.
- Fourier Domain Relationship.
- Exponential Filter.
- Deriche’s Filter.
- Central Limit.
- Stabilising Deconvolution.
- Removing Gibbs Oscillations.
- Optimal (Wiener) Filtering.
- Stability of (Wiener) Filtering.
FIR and IIR Filters.

We have already seen that linear filters can be used as stable components within an image processing algorithm, but large filters may require excessive computation.

One solution is to convolve in the Fourier domain, another is called the Recursive Filter.

The most general linear filter takes the form of a sequence of input pixels \(i_n\) and output pixels \(o_n\)

\[
o_n = \sum_{k=0}^{M} c_k i_{n-k} + \sum_{j=1}^{N} d_j o_{n-j}
\]

IIR (Infinite Impulse Response) if \(N! = 0\) then FIR (Finite Impulse Response).

Notice: the technique operates only along direction and 2D filters have to be constructed by a second pass in the \(y\) direction. Thus 2D kernal must be decomposable.
Fourier Domain Relationship.

Every FIR or IIR filter has an equivalent impulse response function (convolution kernal).

\[ o_n = \sum_k h_k i_{n-k} \]

It would be easy to pick arbitrary values for \( c_k \) and \( d_j \) but how can we do this so that they correspond to a particular convolution kernal \( h_k \)?

If we take the Fourier transform of both sides of our recursive filter we get

\[ O_f = X_f \sum_{k=0}^{M} c_k \exp(-2\pi i k f \Delta) + O_f \sum_{j=1}^{N} d_j \exp(-2\pi i j f \Delta) \]

where \( \Delta \) is the sampling interval.

Therefore

\[ O_f = \frac{\sum_{k=0}^{M} c_k \exp(-2\pi i k f \Delta)}{1 - \sum_{j=1}^{N} d_j \exp(-2\pi i j f \Delta)} X_f \]

thus from the convolution theorem

\[ H_f = \frac{\sum_{k=0}^{M} c_k \exp(-2\pi i k f \Delta)}{1 - \sum_{j=1}^{N} d_j \exp(-2\pi i j f \Delta)} \]
We split the response function \( h_n = s \exp(-\alpha |n|) \) into two parts corresponding to positive and negative \( n \).

\[
h_n = h_n^+ + h_n^-
\]

now using a well known expansion

\[
H_f^+ = s \sum_{n}^{\infty} \exp(-\alpha n) z^{-n} = \frac{s}{1 - \exp(-\alpha) z^{-1}}
\]

where

\[
z^{-n} = \exp(-2\pi i n f \Delta)
\]

similarly

\[
H_f^- = \frac{s}{1 - \exp(-\alpha) z}
\]

From these equations we can deduce the two corresponding recurrence relations

\[
o_n^+ = s \ i_n + \exp(-\alpha) o_{n-1}^+
\]

and

\[
o_n^- = s \ i_n + \exp(-\alpha) o_{n+1}^+
\]

with

\[
o_n = o_n^+ + o_n^-
\]

Thus a two sided 1D exponential filter can be constructed with only two multiplications and three additions per pixel.
Deriche’s Filter.

\[ h_n = s \, n \, \exp(-\alpha |n|) \]

Again split the response function \( h_n = s \, n \, \exp(-\alpha |n|) \) into two parts corresponding to positive and negative \( n \).

\[ h_n = h_n^+ + h_n^- \]

now using the derivative of the same expansion we get

\[ H_f^+ = s \, \sum_{n=0}^\infty \, n \, \exp(-\alpha n) \, z^{-n} = \frac{s \, \exp(-\alpha) \, z^{-1}}{1 - \exp(-\alpha) \, z^{-1}} \]

and

\[ H_f^- = \frac{s \, \exp(-\alpha) \, z}{1 - \exp(-\alpha) \, z} \]

From these equations we can deduce the two corresponding recurrence relations

\[ o_n^+ = i_{n-1} + b_1 o_{n-1}^+ - b_2 o_{n-2}^+ \]

and

\[ o_n^- = i_{n+1} + b_1 o_{n+1}^- - b_2 o_{n+2}^- \]

with

\[ o_n = a \left( o_n^+ + o_n^- \right) \]

with: \( a = s \, \exp(-\alpha) \), \( b_1 = 2 \, \exp(-\alpha) \) and \( b_2 = \exp(-2\alpha) \).

ie: each pixel can be computed with five multiplications and five additions for any value of \( \alpha \).
The above 1D convolutions can be used to produce an effective 2D kernal by processing in alternate directions.

The resulting filter looks somewhat strange but we can make it look more like a conventional Gaussian kernel if we repeat the process (**Central Limit Theorem**).

Any set of identical pertubations on a measured value will ultimately converge on a Gaussian distribution.

Repeated convolution with the same kernel will ultimately produce a result which is equivalent to convolving with a Gaussian.

The final thing to remember about linear filters is that because any set of filters can be represented as a single linear filter sometimes we can improve the efficiency of algorithms by splitting the process into smaller sized stages.
Stabilising Deconvolution.

Linear convolution is such a common process that we often need to reverse the process.

Given the output image and the convolution kernel we need to regenerate the input image. eg: deblurring.

Direct deconvolution is rarely stable due to problems associated with error propagation in the Fourier domain.

\[ I = \frac{O}{R} \]

where \( R \) is the Fourier transform of the convolution kernel.

However there are two methods available which can increase the stability of the deconvolution process:

- symmetrisation
- Wiener Filtering
Removing Gibbs Oscillations.

Any discontinuity (such as an edge) in an image must be represented by a large set of Fourier coefficients.

As Fourier transforms assume cyclic data a direct transform of an arbitrary image is likely to have discontinuities at the perimeter.

These will be a source of Gibb’s oscillations.

These can be removed by forming a symmetrised version of the image.

The resulting data is actually a Discrete Cosine Transform (DCT).
Optimal (Wiener) Filtering.

We will consider this process in the 1D frequency and time domain, but the results generalise directly to 2D images.

We start by defining the generation of an image \( s \) from the convolution of a noise free image \( u \) with a known kernel \( r \). Which is given in the Fourier domain as

\[
S(f) = R(f)U(f)
\]

We can model the observed image \( i \) as signal plus an unknown random noise component \( n \).

\[
i(t) = s(t) + n(t)
\]

Our task is to find the optimal filter which when applied to the measured signal will recover the original noise free image \( u \) most accurately.

ie:

\[
U'(f) = I(f)\Phi(f)/R(f)
\]

This is achieved by minimising the squared error between the noise free and reconstructed image and makes use of Parseval’s Theorem.

\[
\int_{-\infty}^{\infty} (u(t) - u'(t))^2 dt = \int_{-\infty}^{\infty} (U(f) - U'(f))^2 df
\]

substituting in the above equations

\[
\int_{-\infty}^{\infty} \left( \frac{S(f) + N(f)}{R(f)} \right) \Phi(f) - \frac{S(f)}{R(f)} \right)^2 df
\]
Using the fact that the signal and noise must be uncorrelated (and therefore their integrated cross product must be zero) we get

$$\int_{-\infty}^{\infty} (R(f)^{-2}(S(f)^2(1 - \Phi(f))^2 + N(f)^2\Phi(f)^2)df$$

This must be a minimum for all valued of $f$.

Differentiating with respect to $\Phi(f)$ and setting the result to zero gives us

$$\Phi(f) = S(f)^2/(S(f)^2 + N(f)^2)$$

The optimal filter is now generated by estimating $N(f)$ and subtracting this from $I(f)$ so that

$$\Phi'(f) = (I(f)^2 - N'(f)^2)/I(f)^2$$

if $\Phi'(f) < 0$ otherwise $\Phi'(f) = 0$

For images we already know from error propagation that $N(f)$ is expected to be constant and we can compute this from the noise level in the input image $i(t)$.
Stability of (Wiener) Filtering.

Unfortunately the Wiener filter is not as stable in image processing applications as in other domains. The problem arises because the deconvolution kernel often has a large number of very small values $\delta$ say.

If the noise estimation process is not entirely correct we may have the following situation for a single component $f$

$$I = \delta U + N' + n$$

then the optimal filter gives

$$U' = \frac{I(I^2 - N'^2)}{I^2 \delta}$$

$$= \frac{I(\delta U + N' + n)^2 - N'^2)}{I^2 \delta}$$

which in the limit of small $\delta$ gives

$$U' \approx \frac{I 2N' n}{I^2 \delta}$$

As the image content is completely independent of the filter coefficients it is quite likely that somewhere in the Fourier domain we will generate very large coefficients!

If we increase $N'$ sufficiently to remove $n$ entirely we may also be severely damaging the reconstructed image.

Wiener filters thus do not overcome all of the problems and have to be used with care. Often the most stable way of doing deconvolution is an iterative approach in the spatial domain.
Exercises.

• Plot the shape of the first 10 coefficients \((h_n)\) of the Deriche filter. Given that the main use of the Deriche filter is in image smoothing do any of these values seem strange?

• Derive the equivalent linear filter mask for successive vertical and horizontal difference operators. Which approach would require the most computational effort.
Shape From Stereo, Motion and Shading.

N. A. Thacker

- Top-Down and Bottom-Up.
- Stereo Analysis.
- Geometry.
- Correspondence.
- Stereo Algorithms.
- Motion Analysis.
- Shape from Shading.
- Optical Flow.
- Conclusions.
Marr’s theory postulated the complete extraction of 3D surface information from images. This idea is still highly popular in the psychophysics community. In computer vision, there are several topics which attempt this, generally referred to as “Shape From..” eg:

- Sparse Stereo.
- Motion.
- Dense Stereo.
- Shading.
- Optical Flow.

Depending upon the task, some of these require additional data to that present in the image, ie: they are ill-posed.
For this simple geometry we can compute the distance to any object from the apparent locations in the two projected images.

\[ Z = \frac{fI}{(x_l - x_r + \text{const})} \]
Given knowledge of the camera geometry, any image plane location can be projected into an equivalent parallel geometry.

The match constraint is a non-horizontal line (epi-polar).
Correspondence.

The epi-polar constraint is not enough to allow unique identification of correspondence.

Algorithms generally use local image similarity, continuity, smoothness, and disparity gradient (see below).
Disparity Gradient.

The continuous nature of real surfaces leads to a constraint on the allowed disparity gradient.

This constraint was exploited in the PMF algorithm as a weak form of surface smoothness.

Candidate matches can be iteratively selected on the basis of disparity gradient.
Sparse and Dense Stereo Algorithms.

Edge based matching assumes only that detected edges correspond to real 3D structure (e.g., occluding boundaries).

Regional, matching strategies assume local surface smoothness (e.g., stretch or shear).

Away from edges there is no quantitative information, and surfaces must be extrapolated/interpolated.

An honest error model would give very large errors away from edges and corners.

See TINA memo 1994-001.
Motion Analysis.

Well located 2D features (ie: corners) can be tracked between images in order to compute camera to object motion.

Multiple moving object require some careful attention (eg: Hough Transform).

This process is generally well posed and can be used to calibrate stereo cameras from arbitrary scenes (cw. vertical disparity effects in psychophysics). (TINA memo 1991-002)

The calculation of structure from a pair of images from a single moving camera is the same as computation of structure from stereo, but with a smaller baseline.
Shape From Shading.

Let \( z = z(x, y) \) be a depth map with illuminant direction

\[
L = (L_x, L_y, L_z) = (\cos\tau \sin\sigma, \sin\tau \sin\sigma, \cos\sigma)
\]

ie: a single distant light source with \( \sigma \) slope and \( \tau \) tilt.

Putting: \( p = \frac{\partial z(x, y)}{\partial x} \) \( q = \frac{\partial z(x, y)}{\partial y} \)

\[
I(x, y) = \frac{p \cos\tau \sin\sigma + q \sin\tau \sin\sigma + \cos\sigma}{(p^2 + q^2 + 1)^{1/2}}
\]

\[
\approx \text{const} + p \cos\tau \sin\sigma + q \sin\tau \sin\sigma
\]

Taking Fourier transforms;

\[
F_z = m_z(f, \theta) \exp[i(\phi_z(f, \theta) + \pi/2)]
\]

\[
F_p = 2\pi \cos\theta f m_z(f, \theta) \exp[i(\phi_z(f, \theta) + \pi/2)]
\]

\[
F_q = 2\pi \sin\theta f m_z(f, \theta) \exp[i(\phi_z(f, \theta) + \pi/2)]
\]

and ignoring the constant term

\[
F_I(f, \theta) \approx 2\pi \sin\sigma f m_z(f, \theta)\exp[i(\phi_z(f, \theta) + \pi/2)]
\times (\cos\theta \cos\tau + \sin\theta \sin\tau)
\]

ie:

\[
F_I(f, \theta) \approx F_z(f, \theta) 2\pi \sin\sigma f (\cos\theta \cos\tau + \sin\theta \sin\tau)
\]

The image is related to the depth map by a **convolution**.

Extended light sources are linear combinations so shape from shading can be solved by deconvolution.

(see; Alex Pentland, Neural Computation: 1,2, 208-217, 1989)
Optical Flow.

Motion in 1D

\[ -\frac{df}{dt} = u \frac{df}{dx} \]
\[ u = \frac{dx}{dt} \]

Motion in 2D

\[ -\frac{df}{dt} = u \frac{df}{dx} + v \frac{df}{dy} \]
\[ u = \frac{dx}{dt} \quad v = \frac{dy}{dt} \]

The motion of objects in a scene can be analysed at non-feature locations, but this depends upon:

- Illumination (points a and b).
- Local image structure (points c and d) (cw. aperture problem).

Generally this is an ill-posed problem which requires prior knowledge of the scene.

For more details see TINA memo 2004-012.
Conclusions.

Extraction of 3D structure using feature based techniques is generally well posed.

eg:

- Feature based stereo.
- Corner tracking.

Extraction of 3D structure using grey level data is generally ill-posed.

- Dense stereo. (requires knowledge of surfaces)
- Shape from shading. (requires knowledge of illumination)
- Optical Flow. (requires knowledge of surfaces, and illumination)

This is why a purely bottom up approach to the extraction of dense 3D data has never been achieved and there exists no working computational model of Marr’s theory.

The human vision system displays many optical illusions which are a direct consequence of the use of top-down constraints.
Questions.

What are the assumptions made regarding image formation which need to be met for shape from shading to work?

Can you think of a data set which might fit these requirements?

What might be wrong with taking a deconvolution approach to estimating shape from shading?
Model-Based Vision.

N. A. Thacker

- Motivation.
- Projected 3-D models.
- Non-Rigidity.
- Statistical Shape Modelling.
- Active Shape Models.
- Active Appearance Models.
- Conclusions.
Motivation.

- Need to ‘understand’ images - requires a high level model.
- Exploit expert knowledge - ‘top down information’.
- Complexity - we need a way of describing confusing structures.
- Noisy/missing data - robustness.
- Interpretation of image evidence - eg: segmentation.
Model-Based Approach.

We wish to use prior knowledge of an interpretation task in order to;

- better recover/locate shapes,
- quantify spatial relationships,
- describe grey-scale patterns (i.e.: image patches).

We can do this if the model is generative (i.e.: complete);

- label components,
- interpretation by synthesis.

To do this properly we need a model to describe every possible variation of every possible view of every possible object.

In general we will need to be able to select each model for any part of an arbitrary scene.

In restricted applications there may be only 1 model (face recognition, medical images).
Projected 3-D Models.

Given knowledge of the camera geometry, a wire frame model can be projected into an image.

Standard wire-frame models do not account for the apparent shifts of features due to illumination.

see Tina Memo: 2007-011, A Methodology for Constructing View-Dependent Wireframe Models.
Non-Rigidity.

- **Variability** - illumination, object deformation.
- **Generalisation** - represent all possible examples.
- **Specificity** - only represent ‘legal’ examples.

Possible approaches;

- **Active Contours** (Snakes) -
  
  minimise internal ‘energy’ $E(s)$ (eg: $(\partial \theta / \partial s)^2$)
  
  maximise external ‘potential’ $V_{ij}$ (eg: $(\partial I / \partial x)^2 + (\partial I / \partial y)^2$)
  
  satisfy constraints $F(s) = 0$ along the string ($0 < s < S$)
  
  (ie: a poorly defined Likelihood with weak physics analogy).

see Tina Memo 2005-006.

$$C = \alpha \sum_{s} S V_{ij}(s) - \beta \sum_{s} S E(s) - \gamma F(s)$$

General purpose, little prior knowledge, poorly defined parameters, can vary in ‘illegal’ ways.
• **Active shape** - minimise distance of model $f$ (see linear model below) to nearest edges $e$ (ie: a better Likelihood approximation.)

$$C = \sum_s (e(s) - f(s))^2$$

Specific knowledge of variability, parameters of $f$ estimated from example data, less likely to vary in ‘illegal’ ways.

Assume a linear correlation between positions of key points.

Model this behaviour in a set of pre-aligned training data using Principle Component Analysis (PCA) (ie: parameters determined via training).

Shape model: $x_i = x_{\text{mean}} + P b_i$

$P = \text{Linear Modes, } b_i = \text{Shape Vector.}$
Localisation is achieved by iterative minimisation of a cost function.

- A grey-level profile is defined at each point perpendicular to the boundary.
- Match score defined according to Likelihood of matching profile at projected location.

Problems with arbitrary backgrounds.
Active Appearance Modelling.

Aim: Model both the shape and texture

Numerous applications in:
- face recognition,
- face tracking,
- lip reading,
- medical image interpretation.

Active Appearance Models: Example 1.

Modelling a slice of an MR volume of the human brain.

Use in segmentation:
Active Appearance Models: Example 2.

3D MR image of the knee.

Use in measurement of cartilage thickness:
Conclusions.

Appearance modelling is a powerful tool for solving many vision applications.
It breaks with the philosophy of extracting image features as the basis for early vision.
In general they are not a solution to the problem of object recognition.
They are a solution to the localisation of specific objects (ie: similar in capabilities to Hough Transforms).
Researchers have not yet answered the key questions which would allow these techniques to be used as components of general purpose vision systems.
General use would require the ability to learn models in arbitrary environments.
The need to identify key points in training data, in order to build models, has been an important research question for almost a decade.
Questions.

Can you think of a real world example where the shape or grey-level appearance of an object cannot be modelled as a linear function?

Why is the modelling boundary profiles less of a problem for medical images such as MR?

Why can’t we train an appearance model on images of normal subjects and then expect it to work in pathology?
Stereo Geometry and Calibration.

N. A. Thacker

- The Camera Model.
- General Notes Regarding Calibration.
- Homogenous Notation.
- Standard Camera Model.
- The Essential Matrix.
- The Fundamental Matrix.
- The Determining the Fundamental Matrix.
- Reconstruction.
- Summary.
The Camera Model.

The basic set of parameters which define a camera model for machine vision can be placed into **extrinsic** and **intrinsic** groups.

The intrinsic group includes camera specific parameters such as; focal length, image centers, aspect ratios, (radial distortion terms). There can be a variable number of these, depending upon the complexity of the imaging system.

The extrinsic group specify the transformation between the camera model and a specific world geometry and includes the 6 degrees of freedom required to transform between these two frames.

In order to produce a complete stereo camera calibration, for the construction of full 3D metric information, two sets of these parameters are required. One for the left and the other for the right camera.

Over the years many researchers have found that the accurate determination of these parameters can be a problem and much research has recently been devoted to obtaining alternative descriptions to describe stereo camera geometry. Some researchers even claim that useful stereo can be performed in a way which entirely avoids the need to calibrate.
General Notes Regarding Calibration.

The Stereo Camera Model

The freedom for camera model specification, cost function definition and numerical implementation is immense.

- The camera model must be specified with a minimum number of parameters which describe the important degrees of freedom.

- There are at least three ways of representing the left-to-right camera coordinate rotation matrix.

- There is probably very little to distinguish between the performance of these methods.
• If the cameras display radial distortion effects these may well not be visible to the casual observer but will weaken the accuracy of the epi-polar constraint and completely bias any resulting 3D measurement.

![Diagram showing True R(T) vs Observed R(x)]

• The cost function must be defined in the image plane as the errors between back projected positions for points. This is the only domain in which measurement errors can be expected to be uniform.

• If the data for calibration is to be obtained from any automatic matching process then the cost function must be in the form of a robust statistic.
In order to follow the derivation of these new techniques we will need the following **homogenous** notation

- the image point \((u, v)\):
  \[
  \mathbf{x} = \begin{bmatrix}
  x_1 \\
  x_2 \\
  x_3
  \end{bmatrix}
  \]
  where \(u = x_1/x_3\) and \(v = x_2/x_3\) (note: only the ratio is important)

- the image line
  \[
  \mathbf{l} = \begin{bmatrix}
  l_1 \\
  l_2 \\
  l_3
  \end{bmatrix}
  \]
  where \(l_1u + l_2v + l_3 = 0\) (note: again only the ratio is important)

- point on a line: \(\mathbf{l} \cdot \mathbf{x} = \mathbf{l}^T \mathbf{x} = 0\) (or \(\mathbf{x}^T \mathbf{l} = 0\)).

- two points define a line: \(\mathbf{l} = \mathbf{p} \times \mathbf{q}\)

- two lines define a point: \(\mathbf{x} = \mathbf{l} \times \mathbf{m}\)
Standard Camera Model.

The simplest camera model (pinhole) is written as

\[
\begin{pmatrix}
 u \\
 v \\
 1
\end{pmatrix} = \begin{pmatrix}
 f k u & 0 & u_0 \\
 0 & -f k v & v_0 \\
 0 & 0 & 1
\end{pmatrix} \begin{pmatrix}
 x \\
 y \\
 f
\end{pmatrix} = C \begin{pmatrix}
 x \\
 y \\
 f
\end{pmatrix}
\]

The vector \((x, y, f)\) is obtained from perspective geometry

\[
\begin{pmatrix}
 x \\
 y \\
 f
\end{pmatrix} = f/Z \begin{pmatrix}
 X \\
 Y \\
 Z
\end{pmatrix} = \begin{pmatrix}
 X \\
 Y \\
 Z
\end{pmatrix} = P \begin{pmatrix}
 X \\
 Y \\
 Z
\end{pmatrix}
\]

Finally a point in the external world co-ordinate system \(X_w\) transforms to the camera frame using

\[
X = RX_w + T
\]

which can be written in homogenous coordinates as

\[
\begin{pmatrix}
 X \\
 Y \\
 Z \\
 1
\end{pmatrix} = \begin{pmatrix}
 R & T \\
 0^T & 1
\end{pmatrix} \begin{pmatrix}
 X_w \\
 Y_w \\
 Z_w \\
 1
\end{pmatrix} = H \begin{pmatrix}
 X_w \\
 Y_w \\
 Z_w \\
 1
\end{pmatrix}
\]

Thus the full process of projecting a point in 3D onto the image plane can be written as a single matrix product

\[
\begin{pmatrix}
 u \\
 v \\
 1
\end{pmatrix} = CPH \begin{pmatrix}
 X_w \\
 Y_w \\
 Z_w \\
 1
\end{pmatrix}
\]

where \(CPH\) is the 3x4 projection matrix.
The Essential Matrix.

The two camera co-ordinates $X$ and $X'$ for the same 3D point can be related by the between camera transformation

$$X' = RX + T$$

Taking the vector product with $T$, followed by the scalar product with $X'$

$$X'.(T \times RX) = 0$$

which expresses that $OX O'X'$ and $OO'$ are co-planar.

This is simply the epi-polar constraint which can now be written as

$$X'^T E X = 0$$

where $E$ is known as the essential matrix.
For image points this needs extending slightly to

\[(Z'Z/f'f)(u'^T C'^{-T} E C^{-1} u) = 0\]

or equivalently

\[u'^T C'^{-T} E C^{-1} u = 0\]

constructing this expression via the calibrated camera route requires the determination of at least 14 free parameters (6 in E, and 4 in each of the C’s).
The Fundamental Matrix.

There is an alternative way of obtaining the epi-polar constraint.

If \( u \) and \( u' \) are our corresponding image points, then

\[
\begin{vmatrix}
u' & v' & 1 \\
u & 1
\end{vmatrix}
\begin{vmatrix}
F \\
u \\
v \\
1
\end{vmatrix} = 0
\]

ie: \( u'^T F u = 0 \) where \( F \) is the fundamental matrix.

Using homogenous notation

- epi-polar lines:
  \( l' = F u \)
  is the epi-polar line corresponding to \( u \), since \( u'^T l' = 0 \)
  \( l = F^T u' \)
  is the epi-polar line corresponding to \( u' \), since \( l^T u = 0 \)

- epi-poles:
  \( F e = 0 \) and \( F^T e' = 0 \)

Clearly

\[
F = C'^{-T} E C^{-1}
\]
Determining the Fundamental Matrix.

$F$ has 9 elements but the determinant of the matrix must be zero and there is an arbitrary normalisation, so the total number of degrees of freedom is 7.

Thus $F$ can be determined from 8 or more points, in the simplest case of no noise we can use a matrix inversion to solve for the coefficients of the linear equation.

Each point correspondence $uu'$ gives us one constraint on $F$. $$\begin{vmatrix} u' & v' & 1 \\ f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{vmatrix} = 0$$

This can be re-arranged as $Mf = 0$ where $M$ is an $n \times 9$ measurement matrix and $f$ is the fundamental matrix represented as a 9 vector.

$$\begin{vmatrix} u'u_1 & u'_v1 & u'_1 & v'u_1 & v'_1v_1 & v'_1 & u_1 & v_1 & 1 \\ \cdot & \cdot & \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot & \cdot & \cdot & \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ u'nu_n & u'_nv_n & u'_nu_n & v'u_nv_n & v'_nv_n & v'_nu_n & u_n & v_n & 1 \\ \end{vmatrix} = 0$$

This can be solved directly by matrix inversion to determine $f$, but this is not a good idea!
Solving such systems of linear equations in this way makes the implicit assumption that the expected variance on the constraint function is equal for each data point.

Simple error propagation on the linear matrix shows that this is not the case, in general the expected variance on the constraint function is dependant on the coordinate system chosen for image plane locations $u$.

Simply solving for the matrix $f$ as it stands can lead to solutions where one or two outlying data points completely dominate the fit.

One solution to this is to condition the matrix, which basically involves applying an affine transform to the image data to give a zero mean and unity variance to the image plane coordinates, though simple this is not statistically valid.

A better solution would be to rescale the contribution to the constraint function according to the estimated variance. This is called a variational technique and is the correct maximum likelihood formulation for the problem. However, such a method requires an iterative approach as the required rescaling is solution dependant.

Reconstruction.

For calibrated reconstruction the image data is generally transformed to a **rectified** parallel camera geometry before calculation of depth with the simple formula (see figure)

\[ Z = \frac{f I}{(x_l - x_r)} \]

Calibrated Stereo Reconstruction.
Reconstruction from uncalibrated stereo is slightly more complicated.

First construct a default (sensible) projection matrix for each camera $C$ and $C'$. Then compute an estimated essential matrix.

$$E_e = C'FC$$

Now compute a nominal rotation and translation that would be consistent with identity perspective matrices (there may be several, any one will do)

$$(T_e \times R_e)X = EX$$

3D structure is then computed from intersection of rays projected back from the image. In homogenous coordinates this can be written as

$$X_{ie} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} + \lambda_i \begin{bmatrix} x_i \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} -R_e^{-1}T_e \\ 1 \end{bmatrix} + \mu_i \begin{bmatrix} R_e^{-1}x_i \\ 0 \end{bmatrix}$$

These are solved for $\lambda_i$ and $\mu_i$ so that $X_{ie}$ can be computed.
Uncalibrated coordinates and real coordinates are related by

\[ X_i = GX_{ie} \]

where \( G \) is an unknown 4x4 linear projection.

Structure is preserved but not absolute measurements.

For example: we can take any three points and determine if a fourth is above or below the plane defined by the points. The answer will be the same regardless of \( G \).

Such information is useful for feedback control of a robot or obstacle detection but clearly much less useful for metric measurement.
Summary.

Camera calibration is a non-trivial problem, models with small numbers of parameters may be easier to determine.

For establishing epi-polar geometry for stereo matching the minimum parameter model is the fundamental matrix.

This representation makes explicit the fact that certain parameters in the standard stereo camera configuration are correlated.

The other degree’s of freedom are needed for metric depth calculation.

If the camera is not well represented by a pin-hole model (eg: due to radial distortion) then the methodology will not work.

Direct solution of the linear constraint equation is not statistically valid, neither is simple conditioning (though the latter can be made to “work”).

The correct approach is the variational method which is iterative.

In practice we would not expect to obtain noise free correspondances and mis-matches will force the use of a robust statistical approach. (Which will again need to iterative).

Possible to use the RANSAC algorithm on 8 tuple combinations of data points.

Completeness of Recognition Algorithms.

N. A. Thacker

- Evaluating Algorithms.
- Optimal Interpretation Algorithms.
- Completeness: Fourier.
- Completeness: Moments.
- Completeness: Pairwise Geometric Histograms.
- Summary.
Evaluating Algorithms.

Evaluating machine vision algorithms properly is difficult and time consuming. The situation isn’t too bad if we are attempting to make simple measurements.

- object location
- length
- area
- distance etc.

as the statistics for these processes can be well modelled and controlled.

However, for more generic processes such as

- segmentation
- object recognition

the issue is more problematic as the result can quite easily be data set dependent.
So how do we evaluate such algorithms?

During design we have rules of thumb for what constitutes good practice. These are insufficient for quantitative evaluation.

We may even attempt to define a standard data set.

But everyones application will require a new data set, forcing re-implementation of other peoples algorithms in order to obtain a comparison.

It is therefore crucial that other methods for quantifying algorithmic performance are sought.
Optimal Interpretation Algorithms.

For the specific task of scene interpretation it may not be necessary to evaluate algorithms empirically at all, if we can define theoretical measures which only need confirmation on small data sets.

One such measure is the idea of Completeness.

All scene interpretation algorithms fall into a two stage scheme

- representation
- recognition

For scene interpretation tasks, completeness is the property that the representation chosen for the algorithm is invertible. ie: it is possible to reconstruct the original data (in all important respects including required invariances) from the representation parameters.
Further if the recognition stage of the algorithm can then be shown to probabilistically (Bayes) correct decisions based on this data then the whole scheme can be said to be **optimal**.

This may have to be defined under a restrictive set of assumptions which define the **scope** of the method and may also have temporal dependence.

Ignoring implementation and speed issues, the best image interpretation schemes will be those that are complete and optimal (with the largest scope of application).

Few algorithms in the literature have this property.

The issue of building good pattern recognition systems is an ongoing research topic.

For this lecture we will concentrate on completeness.
Completeness: Fourier.

This simple FT in $x$ and $y$ is not scale or rotation invariant.

Fourier Descriptors require a complete boundary.

Alternatively, a curve is plotted as tangential orientation against arc length $\Psi(s)$ and converted to a variable $\Psi^*(t)$ which measures the deviation from circularity.

$$
\Psi^*(t) = \Psi(Lt/(2\pi)) + t \quad t = 2\pi s/L
$$

This periodic contour can then be represented as a Fourier series.

$$
\Psi^*(t) = \mu_0 + \sum_{k=1}^{\infty} A_k \cos(kt - \alpha_k)
$$

The boundary is now uniquely represented by the infinite series of Fourier coefficients, $A_k$ and $\alpha_k$.

Attempting to introduce rotation invariance to this by keeping only the amplitude components $A_k$ destroys the completeness.

This is a general feature of most invariant algorithms, the process of obtaining the required invariance characteristics introduces representational ambiguity.
Completeness: Moments.

After Fourier descriptors, the other common complex shape descriptor in the literature is **Moment Descriptors**.

Ignoring for the moment the main difficulties

- pre-processing the image to obtain suitable data
- defining an accurate centroid.

Moment Descriptors work with pre−segmented blobs.

The regular moment of a shape in an M by N binary image is defined as:

\[
 u_{pq} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} i^p j^q f(i, j)
\]  

(1)

Where \( f(x, y) \) is the intensity of the pixel (either 1 or 0) at the coordinates \( (x, y) \) and \( p + q \) is said to be the order of the moment.
Measurements are taken relative to the shapes centroid \((x', y')\) to remove translational variability.

The coordinates of the centroid are determined using the equation above:

\[
i_0 = \frac{u_{10}}{u_{00}} \quad \text{and} \quad j_0 = \frac{u_{01}}{u_{00}}
\]  

(2)

Relative moments are then calculated using the equation for central moments which is defined as:

\[
u_{pq} = \frac{N-1}{N} \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} \left( i - i_0 \right)^p \left( j - j_0 \right)^q f(i, j)
\]  

(3)

We can also compute a set of rotation invariant moment measures.

\[
M_1 = (u_{20} + u_{02}), \quad M_2 = (u_{20} - u_{02})^2 + 4u_{11}^2
\]

\[
M_3 = (u_{30} - 3u_{12})^2 + (3u_{21} - u_{03})^2, \quad M_4 = (u_{30} + u_{12})^2 + (u_{21} + u_{03})^2
\]

\[
M_5 = (u_{30} - 3u_{12})(u_{30} + u_{12})((u_{30} + u_{12})^2 - 3(u_{21} + u_{03})^2)
\]

\[+ (3u_{21} - u_{03})(u_{21} + u_{03})(3(u_{30} + u_{12})^2 - (u_{21} + u_{03})^2)\]

\[
M_6 = (u_{20} - u_{02})((u_{30} + u_{12})^2 - (u_{21} + u_{03})^2)
\]

\[+ 4u_{11}(u_{30} + 3u_{12})(u_{21} + u_{03})\]

\[
M_7 = (3u_{21} - u_{03})(u_{30} + u_{12})((u_{30} + u_{12})^2 - 3(u_{21} + u_{03})^2)
\]

\[+ (u_{30} - 3u_{12})(u_{21} + u_{03})(3(u_{30} + u_{12})^2 - (u_{21} + u_{03})^2)\]
We can also recompute the original moment descriptors from
the invariant quantities, and from them the original shape. So
the rotational invariant equations are complete.

However, errors do not propagate well through the moment
calculations and successively higher terms become increasingly
unstable.

We are limited to the practical number of terms that we can
actually use for recognition.

In practice moment descriptors are not actually complete.
Completeness: Pairwise Geometric Histograms.

Unlike the previous two representation schemes PGH’s have been designed to directly encode local shape information.

Pairwise Geometric Histograms work with disjoint edge segments.

PGH’s encode local orientation and distance information between edges detected on an object in a way that provides rotation and translation invariance.

They are robust and do not require prior segmentation of the object from the scene.

They have unlimited scope for arbitrary shape representation and encode the expected errors on shape description directly so that there are no problems with error propagation.

Recognition can be performed using a simple nearest neighbour strategy.

Scale invariance can be obtained by interpolating matching responses across a range of scales (c.w. SIFT).
**PGH Inverse.**

It is not immediately obvious how we might get from the set of PGH’s describing an object back to an unambiguous shape. If we take the projection of a PGH onto the angle axis we will obtain a 1D histogram which is the same for all line fragments apart from a shift due to the line orientation.

Relative line orientation within the object can be recovered.

A PGH can be considered as a projection along the direction of the line through the area of the object onto a projection axis.

Thus the set of pairwise histograms provides a complete set of projections for various line orientations through the object analogous to a 2D image reconstruction process such as is commonly found in medical image processing applications.
A reconstruction process can thus be performed which reconstructs the volume around the object as a set of edge orientation specific density images.

These can then be combined to regenerate the original shape.

Reconstruction accuracy is a function of the blurring used to encode measurement error.
Summary.

Image interpretation has two stages: representation and recognition.

If a representation is invertable then it is also complete.

Often, obtaining the required image plane invariance characteristics from basic representation introduces ambiguity and destroys completeness.

Both Fourier and Moment descriptors are found lacking as a complete shape based representation.

PGH’s are complete and can form the basis of an optimal shape based recognition system.
Questions.

• What other limitations does a Fourier based representation of shape possess?
• Would it be possible to create a scale invariant recognition scheme from the PGH’s?
Probabilistic Relaxation Labelling.

N. A. Thacker

- Data Fusion.
- Probability Recap.
- Combining Classifications.
- Nearest Neighbour Classifier.
- Comparison with Relaxation Schemes.
- Incremental Optimal Updating.
- Summary.
Data Fusion.

A common problem that arises in automatic scene interpretation is the process of data fusion.

This generally occurs at the point we wish to combine the results of several local analyses into a global (or at least less local) one.

There are two simple examples of this;
- combining results from per-pixel classifiers.
- combining results from feature groupings.

What extra information regarding C at j can be obtained from local behaviour?

Both of these problems can be approached in an analogous fashion via the use of probability theory.
Probability Recap.

The Bayes Classifier.

Choose the class $C_i$ for which

$$P(C_i|a) = \frac{P(a|C_i)P(C_i)}{\sum_j P(a|C_j)P(C_j)}$$

is a maximum.

The Bayes Classifier is Bayes optimal but gives us no indication of how likely the classification is to be right or wrong.

This would require maintaining the class probabilities $P(C_i|a)$. The predicted number of errors is simply $1 - P(C_i|a)$.

ie: $P = 0.2$ means that the classification will be wrong $1-0.2$ of the time.

An honest classifier is one for which the practical performance meets this prediction.

Combining Classifications.

We want a way of combining classification results ie:

\[ P(C|P(C|A), P(C|B)) \]

to give an optimal (and honest) classification result.

This will only be true if

\[ P(C|P(C|A), P(C|B)) = P(C|A, B) \]

This is called class conditional independance or CCI.

It can be viewed as preservation of information and will be true if there is enough information in the conditional probabilities to uniquely reconstruct the original data

(ie: invertable, clearly related to completeness).

This will clearly be dependent upon the chosen representation for the data.
Assuming CCI we set as our goal the determination of $f$ such that:

$$
P(C | P(C|a), P(C|b), ...) = f(P(C|a), P(C|b), ...)
$$

for just one selected class $C = C_j$. (clearly $P(C_i|...)$ and all of the other class conditional probabilities also exist).

ie: we are looking for a function ($f$) which will map per-pixel probabilities into honest classification probabilities.

Alternatively, for region based data fusion of data around the $i,j$th pixel we simply change the nomenclature and write

$$
P'_{ij} = f(P_{ij}, P_{(j-1)i}, P_{j,(i-1)}...) \text{ ie : the neighbourhood}
$$

We could train a system to achieve this using a look-up table (nearest neighbour classifier).
Nearest Neighbour Classifier.

Normally assign a class based on the classes of example training vectors which are in some sense “nearest”.

In this case estimate the probability that a previously unseen probability vector is consistent with a class label by counting the relative number of training examples with this class in the local neighbourhood.

The technique is likely to be very slow, unless you have a massively parallel processor available.

Neural Networks can be considered as a natural way to reduce computation during use. Simply train the NN to approximate KNN output.

see TINA memo 1992-001, Combining the Opinions of Several Early Vision Modules Using a Multi-Layered Perceptron.

We would call this trained probabilistic relaxation labelling.
**Example: Medical Image Segmentation.**

MR image showing grey matter, white matter and CSF.

![Example image](image1.png)

Single pixel classification of white matter.

![Example classification](image2.png)

Regional classification using local context.

![Example classification](image3.png)
Comparison with Relaxation Schemes.

The technique looks very similar to relaxation schemes based on Markov assumptions.

The standard approach to relaxation labelling involves picking an update function $f$ and repeatedly applying it until there is convergence (ie: things stop changing).

If the procedure converges to 0’s and 1’s (ie: it’s definite about the answer) yet is seen to make labelling errors then the classifier is definitely not honest!. The theory it is based upon must be flawed.

For example, if the Markov assumptions are invalid then the optimal update function required at the second iteration would need to be different to the first and the output would be biased.

In practice both of these can be tested.
**Incremental Optimal Updating.**

Generally, data sets do not have the properties of Markov random fields and if we wish to apply probabilistic relaxation for more than one iteration we will need different update functions for each step.

We can easily envisage generating such a function from a set of suitable data.

The process of computing a different update function for each iterative step is called **Incremental Optimal Updating**.

To implement such a scheme in practice requires us to convert a set of example mapping data into the correct mapping function. This can be achieved directly by use of a neural network (as described above).

Currently Neural Networks do not provide automatic solutions thus this area of data fusion is still a very important research area.

IOU generally tends to converge after a few iterations with no subsequent improvement in classification.
Example: IOU on Synthetic Data.

(a) Per-pixel classification  (b) First stage IOU.

(c) 4th stage IOU  (d) Ground truth.

Figures taken from;
I. Poole, Optimal Probabilistic Relaxation Labelling, BMVC 90.
Honest classifiers produce output which is self-consistent.

If data is class conditional independent then using the conditional probabilities for data fusion is equivalent to using the data.

Generally speaking Markov update schemes are not valid, and represent one of many examples of the mis-application of physics theories in computer vision.

Incremental optimal updating is valid, but constructing a practical system (eg: with NN’s) is non-trivial.

All region based update schemes are likely to have difficulties at boundaries.
Questions.

IOU and TPR both require a mechanism to compare a vector of classification probabilities \((P_1, P_2, \ldots P_N)\) to another such vector, in order to select similar configurations. State three ways you might achieve this?

Can you justify the use of any of these in terms of probability/statistics?

See TINA memo 2004-005;
Statistical Foundations of Algorithmic Research.

N. A. Thacker

- Algorithmic Failings.
- Maximum Likelihood - Revisited.
- Model Selection.
- Non-Gaussian Errors.
- Non-Independent Measurements.
- Inverse Statistic Identification.
Algorithmic Stability.

Algorithmic research is involved with matching computational approaches to statistical behaviour.

The most successful algorithms will be those that are based on assumptions which are most closely matched to the true statistical behaviour of the data.

Most modern machine vision research involves developing new approaches which overcome or avoid the failings of standard techniques.

This lecture will explain the statistical motivation behind such common buzzwords as,

- Robust Statistics,
- Neural Networks
- GA’s.
The most common approach for algorithm development is based on the idea of MAXIMUM LIKELIHOOD, which is derived from the joint probability:

\[ P(Y|X) = (\Pi_i P(X_i|Y)) P(Y) \]

Least squares (as we have seen) is derived from Probability theory on the assumption of independent Gaussian errors and that the prior probability of the model \( P(Y) \) can be ignored. such that:

\[ \log(P(X|Y)) = \sum_i \log(P(X_i|Y)) = -\sum_i (X_i - f(i, Y))^2/\sigma_i^2 \]

The best choice for \( Y \) is the one which maximises this likelihood. There are several key failings of such an approach when used as the basis for machine vision algorithms.

Much research is thus directed (sometimes unknowingly) to overcoming these limitations.

Understanding what problems are being addressed and how is fundamental to making use of the results from other peoples research.
**Model Selection.**

The first problem associated with Maximum Likelihood is that it cannot be used for model selection. ie: Can model Z be selected in preference to model Y?

This is unfortunate as it forms the very basis of any scene interpretation task (segmentation, labelling, model fitting etc.)!

eg: If we use least squares directly as an indicator of model performance we find that the models continue to improve with the addition of new model terms (parameters).
Ultimately when the number of unknown is equal to the number of data points we get $\log(P) = 0$.

There are several candidate solutions:

- The first involves assuming that the problem is due to the lack of knowledge of the prior probability $P(Y)$ for the model. An attempt is made to estimate the prior likelihood of the model based on **entropy** measures (generally defined using Shannons Theory).

ie: choose the simplest model.
• The second assumes that the error is due to a bias in the log-likelihood itself, which can be estimated and subtracted off:

\[ AIC = -log(P) - m \]

where \( m \) = the number of independent degrees of freedom.

• Thirdly, we can use a technique known as cross validation. ie: check the model selected on the basis of prediction ability.

The first approach has no well defined methodology and it can be considered to be a licence to print papers.

It is generally accepted that the last two approaches are ultimately statistically equivalent.
Non-Gaussian Errors.

Machine Vision is full of data that cannot be assumed to be from a Gaussian distribution.

There are two forms of problem:

- The error distribution may be relatively compact but badly skewed.
- There may be outliers caused by data “contamination”.

The general technique for coping with the first problem is to transform the data to remove skewing.

eg: 

\[ D_x = f(x) \]

so we seek a function \( g \) which will give us

\[ D_g(x) = \text{const} \]

using error propagation

\[ D_g(x) = D_x \frac{dg}{dx} = f(f) \frac{dg}{dx} = \text{const} \]

ie integrate the reciprocal of the error dependence:

\[ g = \int \frac{\text{const}}{f(x)} dx \]
example. Stereo data.

\[ z = \frac{f I}{(x_l - x_r)} \]

errors in \( Pos(x, y, z) \) are badly skewed.

Attempting a LSF with these measures directly (eg for model location) is unstable due to large errors for large \( z \).

However, errors on **disparity space** \( Pos(x, y, 1/(\sqrt{2}z)) \) are uniform and can be used for fitting.

The technique can be considered as applying the inverse of error propagation (such as in image processing) in order to work back to a uniform distribution.
Dealing with Outliers.

This area of algorithm design is generally referred to as Robust Statistics. The simplest technique involves limiting the contribution of any data point to the total LSF ie:

\[ -\log(P) = \sum_i \min((X_i - f(i, Y))^2/\sigma^2_i, 9.0) \]

The choice of 9.0 as the limit on the contribution is approximate and may depend on the problem.

This technique is not particularly good for methods which use derivatives during optimisation, as it introduces discontinuities which can introduce local minima.

Alternative involve replacing the Gaussian with a continuous distribution with long tails.

The most common of these is the double sided exponential.

\[ -\log(P) = \sum_i |(X_i - f(i, Y))/\sigma_i| \]

This is adequate for most applications.
More complex techniques which attempt to model slightly more realistic distributions can be found in the literature eg: Caucy distribution

\[ P(X_i|Y) = \frac{1}{1 + (X_i - f(i, Y))/\sigma_i^2/2} \]

so that our log probability is now

\[-\log(P) = \sum_i \log(1 + 1/2(X_i - f(i, Y))/\sigma_i^2) \]

The price we pay for this is that, unlike standard least squares, such cost functions can rarely (never?) be optimised by direct solution so we have to use iterative techniques which take more time.
Non-Independent Measurements.

What is correlation and independence?

Once again Machine Vision regularly generates data sets that are correlated.

The most frequent method used to overcome this problem is data orthogonalisation (via PCA).

Once the data has been orthogonalised (by rotation) we can once again use Maximum Likelihood.

\[
P(x',y') = P(x')P(y')
\]

Principle Component Rotation in 2D.

Sometimes the correlations in the data set are not linear and cannot be removed by a rotation. Here we would want to use what we might call ”non-linear” PCA.

One approach to this is to use a Neural Network.
In general, if we wish to combine multiple pieces of information to obtain a combined objective (data fusion) we would use probability theory and probably combine them in the form of a $\chi^2$.

Sometimes we have what is known as non-commensurate variables. This means that we do not know how to relate them or what their correlations might be and there is no way of finding out.

The only method for dealing with this problem currently is based on the concept of the **Pareto Front**.

![Diagram of the Pareto Front](image)

This has formed the basis for Multi-Objective Genetic Algorithms (MOGA’s).
Summary.

All Machine Vision research involves the use of algorithms which embody algorithmic assumptions.

New or novel techniques generally cope with problems which have a different scope of application to the conventional techniques.

Robust statistics avoids problems due to outliers.

Neural Networks and PCA avoid problems due to non-independence.

MOGA’s solve the problem of non-commensurate objectives.

In general, when you read a paper you should look for the underlying statistical model (if it is not described directly) and try to compare this against what you may expect to work for the data set involved.

You can call this process Inverse Statistic Identification.
Problems:

• A common technique for image compression starts by matching image blocks on the basis of an absolute sum of grey level values. What are the statistical assumptions made in this algorithm? Why would you expect this approach to be reliable?

• Compute the expected error on $z$ for a stereo algorithm using error propagation. Show that $Pos(x, y, 1/(\sqrt{2}z))$ has uniform errors.
Visual Intelligence.

N. A. Thacker

• Old Definitions.
• New Definitions.
• Sequential Reasoning.
• Consciousness.
• Minimal Intelligence Architecture.
• Problem Solving using a State Map.
• Generalised problem Solving.
• Conclusions.

see Tina Memo: 2001-006, What is Intelligence?
Old Definitions.

- Oxford English Dictionary:
  Intelligence is a process of understanding and reasoning (*entirely relational*).

- Animal Behaviorists:
  The ability to solve problems outside the habituated environment (*humans are not special*).

- Computer Scientists (Turing Test):
  The ability to mimic human behaviours and interactions (*humans are special*).

It is generally a bad idea to redefine things, as if we are not careful we can trivialise our goals, any redefinition must be done with the aim of identifying genuinely useful capabilities.

But standard definitions tell you how intelligence can be recognised, not what it is doing.

What would the structure of a working intelligent system look like?

What do we think an intelligent system should do?

We require a definition of intelligence which addresses these questions while identifying genuinely useful capabilities.
New Definitions.

Basic terminology;
• state: a measurably distinct set of internal and external circumstances.

• representation: a derived description of the environment sufficient to identify specific circumstances (states).

• manipulation: the modification of representations by the system in order to generate a different state (including physical actions).

Key concepts;
• reasoning: a process which identifies particular sequences of manipulations required (or likely) to achieve a particular state.

• intelligence: the ability to learn from available sensor data without external supervision and modify decision processes (reasoning or manipulation) in order to improve the probability of achieving a future state.

• understanding: a set of learned manipulations which enable reasoning on a particular set of representations.

Armed with these definitions we can attempt to imagine what an intelligent system might look like and assess its likely capabilities.
Sequential Reasoning

With the above definitions we can consider ‘thinking’ as the processes of the manipulation of the extrinsic (sensory originated) and intrinsic (consciously formed) representations in order to form new representations.

Notice that this particular definition of intelligence includes aspects of direct exploration and learning for those circumstances where the system does not know what else to do.

It also deliberately includes a probabilistic definition of prediction which forewarns of an inherently statistical approach to decision making and system modification (learning).

It would be useful to be able to describe this process mathematically and the need for manipulation implies something more than conditional notation (ie: \( P(A|B) \) used in probability theory).

For a unique manipulation \( M \) we need \( P(A|M|B) \) (ie: the probability that state \( A \) will result from state \( B \) following action \( M \)). This looks like Turing’s universal processor.

Clearly building such a system requires methods for defining (learning) relevant \( A \)’s, \( B \)’s, and \( M \)’s.
Consciousness.

• sub-conscious (1).
  The use of frequently updated *representations* to generate an interrupt mechanism to stop current task processing and attend to other circumstances.

• sub-conscious (2).
  The use of previous *reasoning* or knowledge given at construction for semi-automatic (sub-conscious(1)) execution.

• consciousness.
  The process of continually monitoring the incoming sensor data (and therefore changing *representation*) to ensure that a planned *manipulation* task is being executed correctly.

• concentration.
  The ability to attend to a task with a relative minimum of *sub-conscious(1)* interrupts.

• self-awareness.
  The ability to *understand* the systems own *sub-conscious(2)* factors.

• self-consciousness.
  The ability to predict the consequences of *manipulations* on the system as well as the environment.
A Minimal Intelligence Architecture.

- **sensors**: examples include auditory and vision.
- **representation**: examples include speech, object recognition, location, 3D measurement.
- **interrupt generator**: take the current *representation* and generate an interrupt when it matches specific *states* which have been specified a-priori.
- **memory**: previous examples of sensory *representation* and associations stored in a manner suitable for planning tasks.
- **goal generator**: take the current analysis of environment and system and select target *states*.
- **task scheduler**: generate a *reasoned* list of actions and allowable interrupts storing generic information for future use.
- **controller**: take the current list of tasks and execute them using feedback from the current *representation*. Feed execution timing back to the task scheduler.
- **actuators**: the physical robot may require; wheels, robot arms, steerable vision system.
Problem Solving Using a State Map.
The basic idea is as follows:

1. The current *state* (i.e., visual location) and target *states* are identified.

2. Starting from the current *state* connections representing allowable *manipulations* are followed to nearby *states*. The cost of arriving at this location is computed (as for example the minimum iteration number to arrive at that *state*).

3. The process is iterated by following connections to newly reachable *states*.

4. The process ceases when we reach one or more target *states*.

5. The map is then traversed in reverse from the target selecting the minimum cost *state* at each step (dotted curve).

6. The next suggested physical *manipulation* is the minimum cost link from the start which is executed in a closed visual control loop using the computed visual *representation*.

The algorithm can be said to be optimising a function of the form

$$Q = \sum_j C(S_i|M_j|S_k)$$

Where a chain of *manipulations* $M_j$ are determined which link the current *state* to any targets.

If necessary, the cost function can be related to probability of success (e.g., $C(S_i|M_j|S_k) = -\log(P(S_i|M_j|S_k)))$. 
Examples.

Colour Segmentation. (Tina Memo 2001-015)

Path Planning. (Tina Memo 1999-002)
Generalised Serial Problem Solving.

Many problems do not come with a fixed *state* map on which to do planning.

To solve such problems we need the equivalent of mental exploration.

The process would look like the following:

1. Use the result of previous attempted *manipulations* to train a *state* transition predictor.

2. Use the *state* transition predictor to grow new *states* from the current set, comparing them with known locations to ensure that they are distinguishable and storing information such as costs and interrupt levels.

3. Iterate the process of *state* generation until a target location is reached with some required cost.

4. Having generated a map, retrace the minimum cost solution as in the serial problem solver algorithm.
Conclusions.

We cannot answer questions like; Will computers ever be intelligent?, without first defining intelligence.

The standard definitions of intelligence have done little to advance the field of artificial intelligence.

PARRY, ELIZA, chess playing robots, Brookes’ subsumption architecture, are designed to mimic (Turing) complex behaviour rather than have general utility.

Many useful capabilities could be supported with a serial problem solver within an architecture which looks very similar to a conventional computer (ie: memory, processors and interrupts).
Questions.

Can you think of an existing real world application for a serial problem solver?

Are there any differences between this solution and what you would need for a vision system?

Do you think that the parallel structure of the human brain offers computational advantages for intelligence, over and above the capabilities of a digital serial computer?