Part 1 : Algorithmic Performance Evaluation.

P. Courtney and N. A. Thacker

Structure

- The Need for Algorithmic Testing.
- Conventional Approach to Algorithmic Testing.
- Alternative Approaches to Algorithmic Testing.
- Good practice vs bad practice.
- More Images.
- Software Standardisation.
- Summary.

The Need for Algorithmic Testing.

What about other fields of engineering ?

Algorithmic testing is necessary:

- to develop new approaches.
- to compare the performance of alternative algorithms.
- to validate incremental modifications to algorithms.
- for algorithmic system integration
  - ROBUSTNESS
- an algorithm can only be used if it has been evaluated. (when algorithms are used which haven't been evaluated it's called research!).

The Conventional Approach to Algorithmic Testing.

A simplification of the standard approach to algorithm design implementation and testing is:

1. have an idea.

2. implement in software.

3. test on a small number (1-4) of images.

4. publish with test results as proof of concept.

5. go back to 1

Tuning parameters: Good vs bad practice

- What you see is not what you get.

Although this results in the rapid publication of ideas it is a bad approach because:

- the algorithm is treated as a black box.
- displaying results as images rarely conveys any statistically useful measure of performance.
- a small quantity of images is rarely a conclusive demonstration of robustness.
- the method does not expose the assumptions underlying the algorithm or the limitation on application domain.
- the method does not help comparative analysis.
- if the idea really was any good, the work will need to be done again.

Ref: Foerstner

Alternative Approaces to Algorithmic Testing.

It is now accepted by the majority of the machine vision community that a more rigerous methodology for testing is required (Ref: Haralick).

This methodology needs to be grounded in statistics, both identifying and validating the statistical assumptions used in the algorithm and in the evaluation process itself.

Due to the diversity of the subject area, several approaches are probably going to be needed.

Alternative Approaces to Algorithmic Testing.

Alternatives are:

- test on more images.
- analytic evaluation eg:
  - error propagation.
  - Monte-Carlo.
- algorithmic modelling.
- software standardisation.

More Images.

Example of image registration.

(Ref: West and Fitzpatrick)

- Definition of metrics.

- Trials.

- Results: all about the same.

Some other successes:

- optical character recognition for US census

- face recognition for US Army (Ref: Phillips)

More Images.

BUT:

- how do we define the standard test set?
  - not too easy
  - not too hard
- logistical problems with acquiring the data.
  - cost
  - errors in the data
- robust algorithms require huge quantities of data.
  - 99% reliability means 1% error rate
  - 1% error rate means hundreds of test images

(Ref: Guyon on test set size)

More Images.

- a black box evaluation has limited prediction capabilities.
- testing time grows exponentially with the number of tuning parameters.
- the timescales for evaluation are large, software harnesses would help.
- Good vs bad practice: what you see is not what you get.
- it's very early days.

Software Standards.

What standards ?

- Existing tools eg manufacturer's libraries
- IUE (image understanding environment)
- DICOM
- Baron and Fleet's optical flow code
  - ftp.csd.uwo.ca/pub/vision
- open source eg TINA
  - TINA: www.niac.man.ac.uk/TINA

Conclusions.

- Rigorous approaches to testing exist but the main obstacle may be psychological (Ref: Foerstner).

- The biggest problems associated with evaluation are associated with the time taken to perform the analysis. This precludes rapid publication rates.

- Evaluation work does not have the same apparent standing in the field as, for example, new ideas. (but this is changing)

- Unless algorithms are evaluated, in a manner that can be used to predict the capabilities of a technique on an arbitrary data set, it is unlikely be successfuly reimplemented and used.

- The subject cannot advance without a scientific methodology, which it will not have without an acknowledged system for evaluation, characterisation and the ability to reimplement algorithms.

References.

- W. Foerstner, 10 Pros and Cons Against Performance Characterisation of Vision Algorithms,
- R. M. Haralick, Covariance Propagation in Computer Vision,
- J. West, J.M. Fitzpatrick, et al, Comparison and Evaluation of Retrospective Intermodality Brain Image Resotration,
- P.J. Phillips, H. Moon, S.A. Rizvi and P.J. Rauss, The FERET Evaluation Methodology for Face-Recognition Algorithms
- I. Guyon, J. Makhoul, R. Schwartz, and V. Vapnik, What size test set gives good error rate estimates?

Part 2: Statistics and Error Propagation.

N. A. Thacker and P. Courtney.

- Methodology.
- Basic Definitions.
- Bayes Theorem.
- Maximum Likelihood.
- Common Probability Equations.
- Covariance Estimation and Error Propagation.
- Examples: Stereo and Co-registration.

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.
- $P(\tilde{A}) = 1 - P(A)$ probability of non-event A.
- $P(AB)$ 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.
- $P(A|BC)$ 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.

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.

$$P(AB) = P(A|B)P(B)$$

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

$$P(AB) = P(BA) = P(B|A)P(A)$$

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

$$P(ABC) = P(A|BC)P(BC)$$

$$P(ABC) = P(A|BC)P(B|C)P(C)$$

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

$$P(X_0 X_1 X_2 ... X_n|Y)P(Y) =$$

$$P(X_0|X_1 X_2 ... X_n Y)P(X_1|X_2 ... X_n Y) ...... P(X_n|Y)P(Y)$$

- 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 $P(XY)$ requiring a prohibitive amount of data storage.

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

  $$P(X_n|Y) = P(X_0|Y)P(X_1|Y)P(X_2|Y)...P(X_n|Y)$$

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

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

or

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

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

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

- For large expected numbers of entries this distribution approximates a Gaussian with

  $$\sigma = \sqrt{p_i}$$

- 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 χ^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:

$$e^{-log(P)}\ =\ \Pi^i e^{-\chi_i^2/2}$$

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 σ_i gives

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

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

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

which leads to the χ^2 definition of log likelihood

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

- 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 χ^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

$$\chi^2 = \chi_0^2 + \Delta X^T C_x^{-1} \Delta X \quad with \quad \Delta X = (x - x_0, y - y_0)$$

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

$$C_x^{-1} = \begin{vmatrix} u & v \\ w & s \end{vmatrix}$$

Comparing with the above quadratic equation we get

$$\chi^2 = \chi_0^2 + x^2 u + yxw + xyv + sy^2$$

where

$$a = \chi_0^2, b = 0, c = 0, d = w + v, e = u, f = s$$

Notice that the b and c coefficients are zero as required if the χ^2 is at the minimum. (Ref : Haralick)

Starting from the χ^2 definition using the same notation as previously.

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

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

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

$$\frac{\partial^2 \chi^2}{\partial a_n \partial a_m} = \sum_i^N \frac{1}{\sigma_i^2}(\frac{\partial f}{\partial a_n}\frac{\partial f}{\partial a_m} - (X_i - f(y_i, a))\frac{\partial^2 f}{\partial a_n \partial a_m})$$

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

$$= \sum_i^N \frac{1}{\sigma_i^2}(\frac{\partial f}{\partial a_n}\frac{\partial f}{\partial a_m})$$

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 χ^2 function then,

$$C = \alpha^{-1}$$

And the expected change in χ^2 for a small change in model parameters can be written as

$$\Delta\chi^2 = \Delta a^T \alpha \Delta a$$

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.

example 2: Stereo Measurement (Demo Stereo)

Using rectified images, the distance, Z between the feature and the camera plane can be found with the equation:

$$Z = \frac{fI}{x_1 - x_2}$$

where:
f is the focal length of the lenses
I is the inter-occular seperation
x_1 and x_2 are positions of the features on the epipolars
We can determine the sensitivity of Z with changes in x_1 and x_2 thus,

$$\Delta Z^2 = \left(\frac{\delta Z}{\delta x_1}\Delta x_1\right)^2 + \left(\frac{\delta Z}{\delta x_1}\Delta x_2\right)^2$$

where,

$$\frac{\delta Z}{\delta x_1} = -\frac{fI}{(x_1 - x_2)^2} \quad \text{and} \quad \frac{\delta Z}{\delta x_2} = \frac{fI}{(x_1 - x_2)^2}$$

Δx is the feature position error in the image and can be assumed to be equal in each image, so

$$\Delta x_1 = \Delta x_2 = \Delta x$$

Solving for ΔZ yields the result,

$$\boxed{\Delta Z = \frac{\sqrt{2} f I \Delta x}{(x_1 - x_2)^2} \quad \text{or w.r.t. } Z, \quad \Delta Z = \frac{\sqrt{2} Z^2 \Delta x}{f I}}$$

(Demo Matcher) Ref: Fisher.

example 3: Medical Image Co-registration.

The work of West et al. illustrates one of only a few examples of a co-ordinated attempt to compare algorithms.

The work involved getting numerous groups to co-register test data sets while a central cite collated the results.

While this is an important piece of work it has two key failings;

- The choice of data sets is specific and finite.

- Results cannot be extended to other data sets.

In this case the use of covariance matricies may have allowed the validation and reliability of algorithms capable of prediciting their own accuracy on any data.

Part 3: Image Processing Stability.

N. A. Thacker.

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

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.

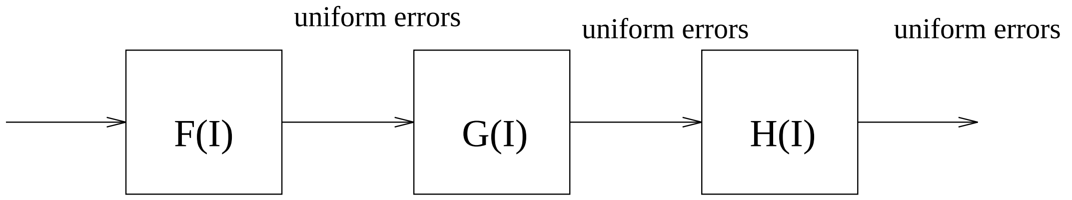
eg: edge strength maps.

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.

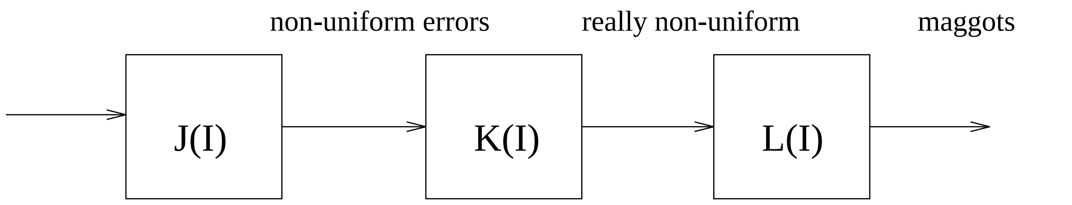
ie: is a high value genuine or just a product of the propagated noise.

In this lecture we will cover two ways of assessing algorithms: **Error Propagation** and **Monte-Carlo** techniques.

Designing Stable Feature Detectors.



Propegation through stable algorithm.



Propegation through un-stable algorithm.

Error Propagation.

General Approach for Error Propagation (Recap).

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

where ∇f is a vector of derivatives

$$\nabla f \ = \ (\frac{\partial f}{\partial X_1}, \frac{\partial f}{\partial X_2}, \frac{\partial f}{\partial X_3}, ...)$$

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

$$\Delta f_{xy}^2(I) = \sum_{nm} \sigma_{nm}^2 (\frac{\partial f_{xy}}{\partial I_{nm}})^2$$

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

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 techniques 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.

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:

$$O = I_1 \; / \; I_2$$

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

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:

$$O = I_1^n$$

$$\Delta O^2 = (nI_1^{n-1})^2\sigma_1^2$$

Square-root of Sum of Squares:

$$O = \sqrt{I_1^2 + I_2^2}$$

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

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.

Linear Filters.

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

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

eg: $\mathbf{h} = (-1, 0, 1)$ gives $\Delta O^2 = 2\sigma^2$

Thus linear filters produce outputs that have uniform errors.

Unlike image arithmetic, although the errors are unform 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:

$$O_{xy} \;=\; \int_0^{I_{xy}} f dI / \int_0^{\infty} f dI$$

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.

Edge Detection.

Edge detection is a combination of operations and the simplest approach to testing is likely to be Monte-Carlo.

Canny was designed to combine optimal noise suppression with location accuracy, but does this account for its stability?

The sequence of processing involves;

- convolution with the noise filter (eg: $\otimes$ Gaussian)
- calculation of spatial derivatives (eg: $\otimes$ (-1, 0, 1))
- calculation of edge strength (eg: $\sqrt{(\nabla_x^2 + \nabla_y^2)}$)
- thresholding and peak finding

The final stage will be reliable provided that we have stability after the first three image processing steps. (Demo: Edges)

Feature Detection Reliability.

Generally, when locating features, we are interested in a limited set of performance characteristics.

- Position and orientation accuracy
- Detection reliability
- False Detection rate

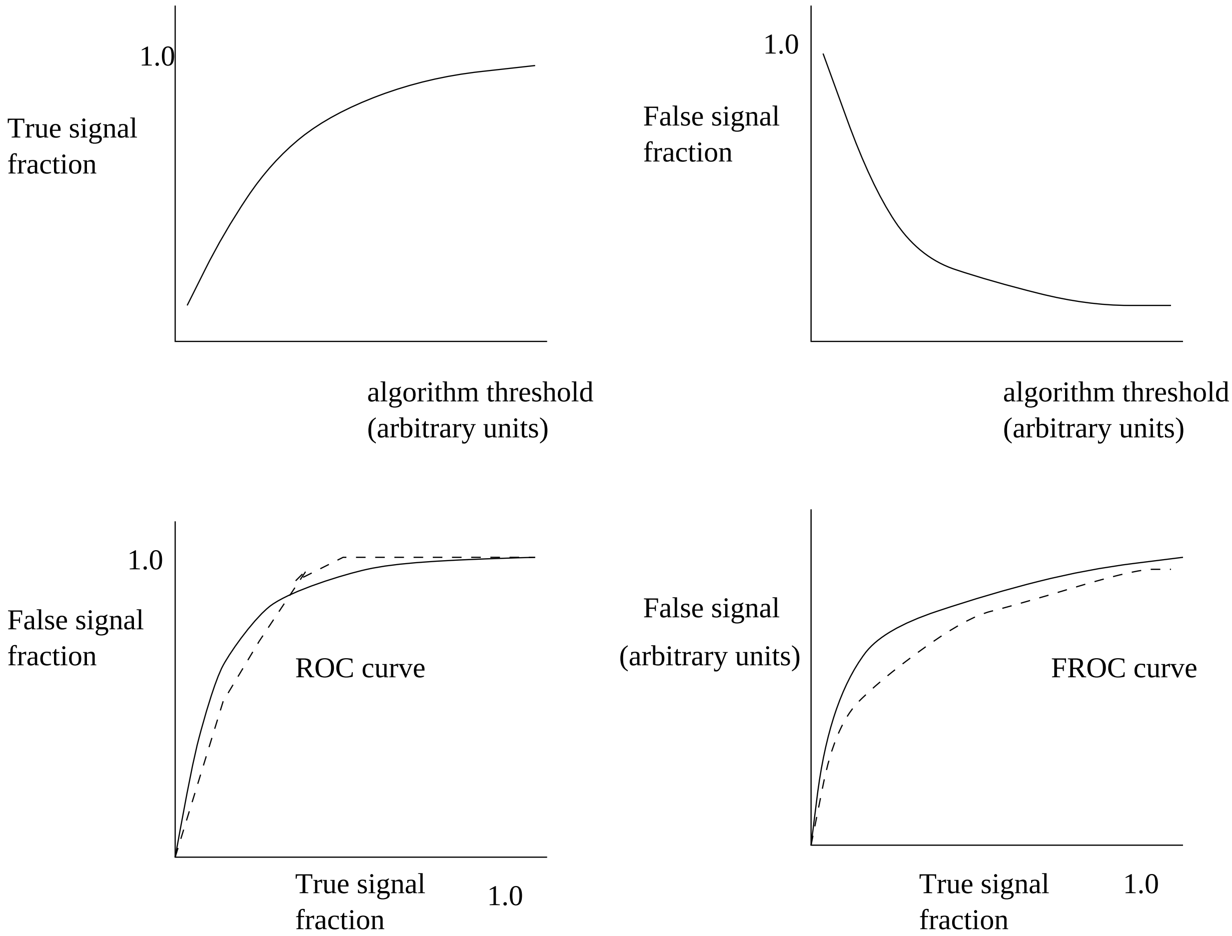
The first of these can be performed using a Monte-Carlo repeatability experiment.

The last two require a gold standard against which to make a comparison.

In addition, most feature detection algorithms have a sensitivity threshold (which corresponds to the probability level of the null hypothesis). The best value will be data dependent.

The way to deal with this is to produce curves which describe the detection and false detection rates as a function of threshold or even better ROC curves.

Reciever Operator Curves.



Part 4: Evaluating Correspondence Algorithms.

P. Courtney, N. A. Thacker

- Algorithmic Modelling.

- Demo of corner matcher.

- Feature Matching and use of heuristics.

- Failure modes:
  - Not getting correct matches.
  - Getting incorrect matches.

- Algorithm Analysis.

Algorithmic Modelling.

Statistical models of data vary with image test sets.

- N images.
- D degrees of freedom.

Perhaps trying to model the data is insufficient

Try modelling the algorithm and sampling the relevant statistical distributions (Ref: Thacker).

Algorithmic Modelling.

Advantages:

- statistical distributions and assumptions which determine the outcome of the algorithm are made explicit and observed.

- performance may even be predicted for an individual image (assuming known or measurable properties such as contrast, noise etc)

- algorithmic models allow the analysis of the data- independent properties of the algorithm.

- provides an independent estimate of expected performance for people wishing to develop their own implementation of the algorithm.

The method requires a completely new approach to algorithm development, new stages of an algorithm are only added once the effects on performance can be modelled.

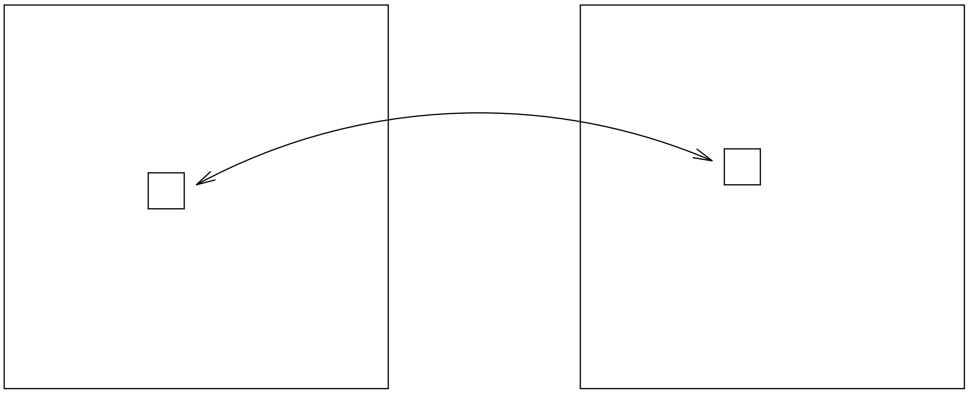
Demo.

Evaluation of a matching algorithm

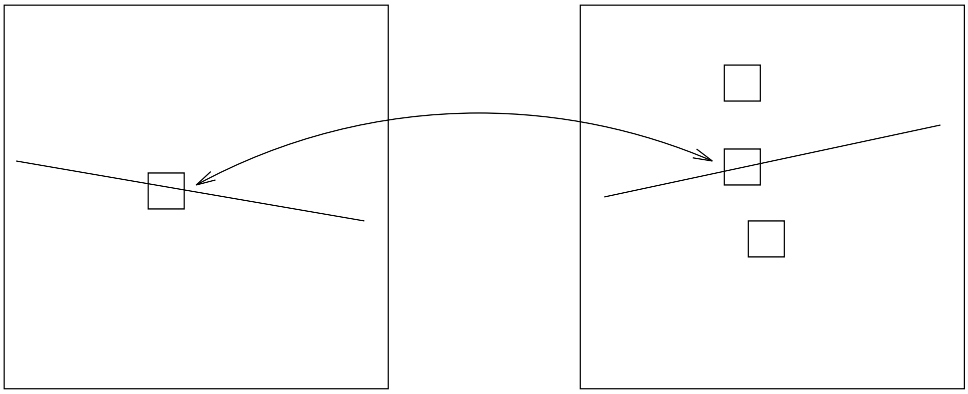
- image pair of textured face mask
- corner detection
- corner matching left and right
- discussion of matching performance

Feature Matching: Heuristics.

(a) local image similarity (eg image correlation).

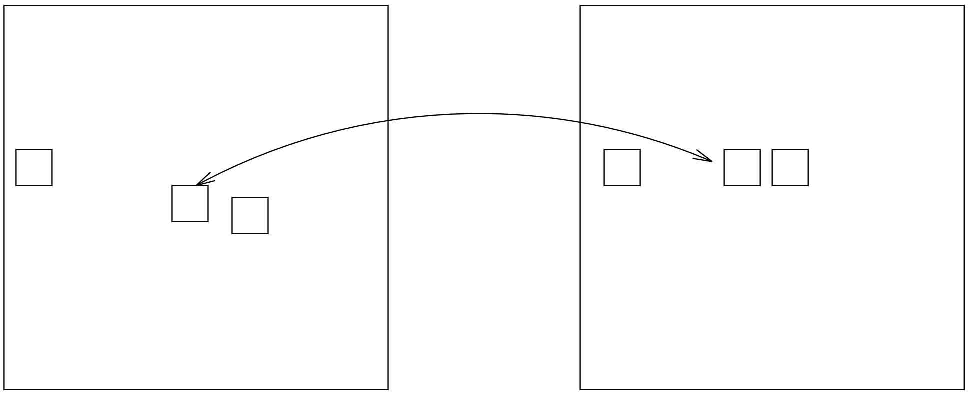


(b) restricted search strategies (eg stereo epi-polars).

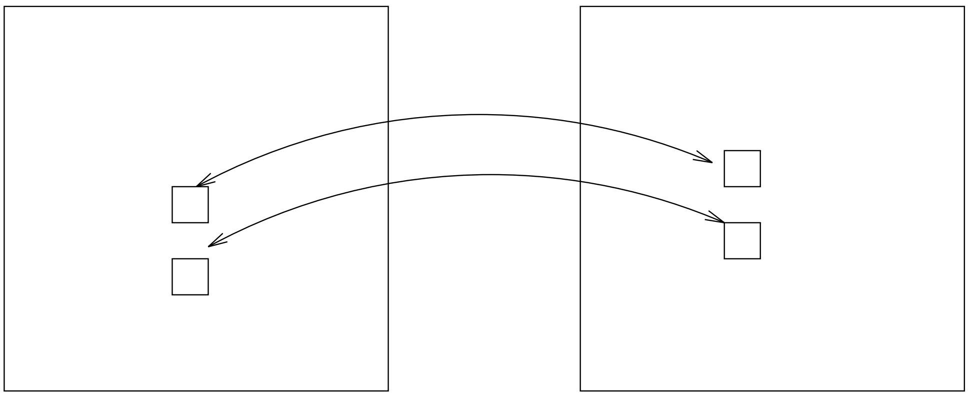


Feature Matching: Heuristics (cont).

(c) disparity gradient (or smoothness) constraints.



(d) one to one matching.



(e) reliability of detection.

Steps to the Algorithm

(a) Construct a list of possible matches a limited search area A.

(b) Order the list according to a cross correlation measure c.

(c) Select good matches on the basis of:

- a threshold ρ on the minimum acceptable cross correlation $c < \rho$.

- a threshold ω on the ratio of absolute corner strengths $\frac{(c_1 - c_2)^2}{(c_1 + c_2)^2} < \omega$.

- reliability of the best candidate match c_m compared to the next best match c_n on the basis of a uniqueness parameter δ : eg: $c_m - c_n < \delta$.

- the same best candidate match must be obtained when matching from image 1 to image 2 and image 2 to image 1 (enforces 1:1 match).

GETTING AN INCORRECT MATCH

The algorithm can be modelled as a limited candidate match selection followed by matching on the basis of a single cross-correlation similarity measure.

The important distributions for algorithm modelling are the cross-correlation values for known correct and incorrect matches.

By considering each detection configuration in turn we can estimate the probability of getting an incorrect match.

- Case (a):

$$P_m^a = 2n_u P_d^2 (1 - P_d)^2 P_I(\rho)$$

$$P_I(x) = \int_x^1 P_N(a) da$$

- Case (b):

$$P_m^b = 4n_u P_d^3 (1 - P_d) P_n(\delta, \rho)$$

$$P_n(\delta, \rho) = \int_\rho^{1-\delta} P_S(x) \int_x^{1-\delta} P_N(a - \delta) da dx$$

- Case (c):

$$P_m^c = 2n_p P_d^4 P_n(\delta, \rho) P_k(\delta, \rho)$$

REJECTING A CORRECT MATCH

- Case (a):

$$P_r^a = P_d^2(1 - P_d)^2 P_J(\rho)$$

$$P_J(\rho) = \int_0^\rho P_S(x)dx$$

- Case (b):

$$P_r^b = 2P_d^2(1 - P_d)^2(P_J(\rho) + n_u P_l(\delta, \rho))$$

$$P_l(\delta, \rho) = \int_\rho^1 P_S(x) \int_x^{1+\delta} P_N(a + \delta)dadx$$

- Case (c):

$$P_r^c = P_d^4(P_J(\rho) + 2n_p P_l(\delta, \rho) - n_p^2 P_l(\delta, \rho)^2)$$

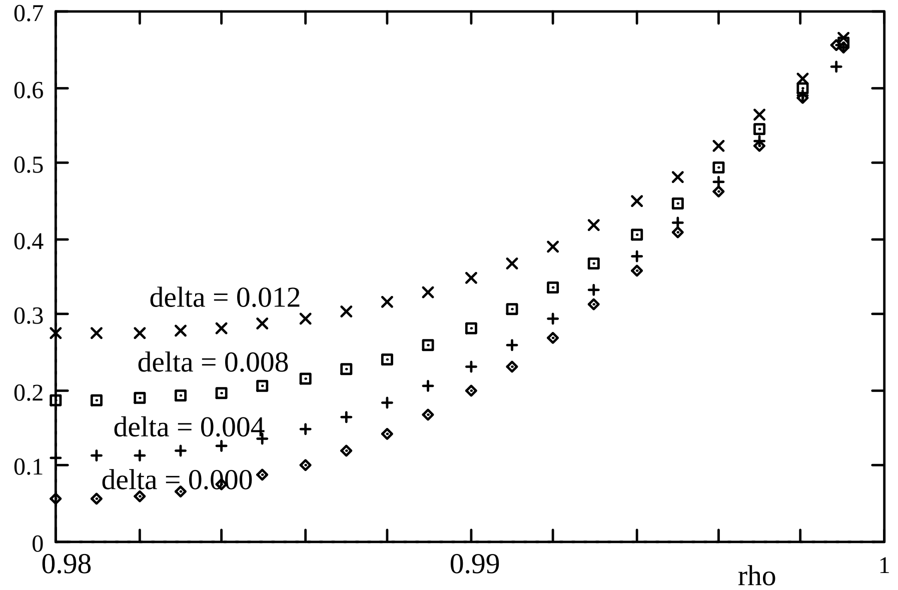
Total correct and incorrect matches.

The total number of matches is given by:

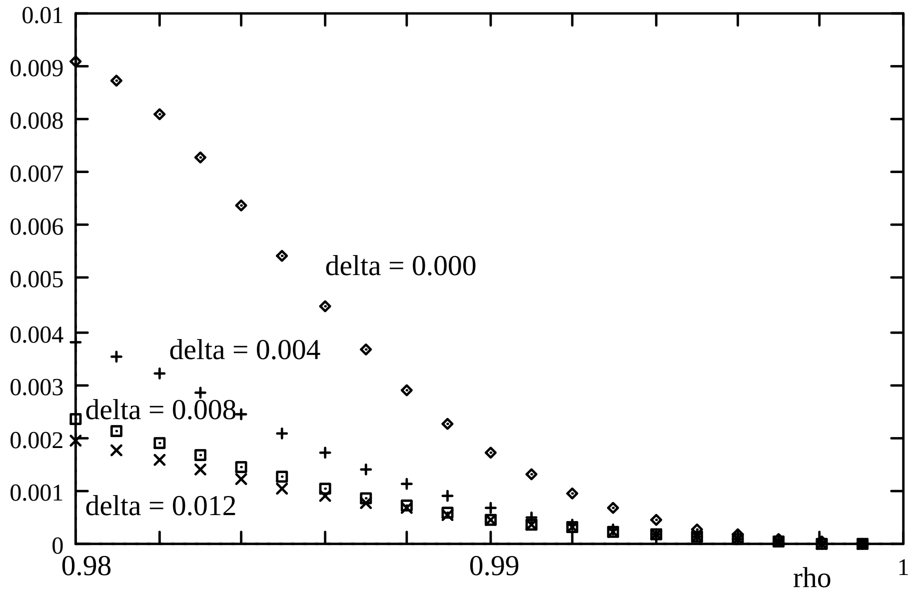
$$P_m^T = P_m^a + P_m^b + P_m^c$$

$$P_r^T = P_r^a + P_r^b + P_r^c$$

Quantity of Rejected Data



Quantity of Mismatched Data



ALGORITHM ANALYSIS

1) All terms in P_m^T are proportional to the mean number of candidate matches, thus we would expect the total number of mismatches to vary proportionately with the search area A.

2) We expect type (a) mismatches to be a very small fraction of the total number of mismatches. The only way to remove these is to increase the minimum required cross correlation value ρ.

3) We expect type (b) and (c) mismatches to be of roughly equal importance and both are reduced considerably by use of the uniqueness parameter δ at the cost of only marginal reduction in the overall number of matches.

ALGORITHM ANALYSIS

4) There is no improvement obtained by increasing δ beyond a value of $1-\rho$ as at this point all mismatches of type (b) have already been rejected.

5) There is no set of parameters which give an optimal signal to noise ratio, this value keeps on rising with increasing ρ. There are however optimal values of ρ and δ corresponding to the minimum noise obtainable for a required proportion of signal. For example using the above model for the data the minimum noise obtainable at a signal level of 60% is 0.2% at parameter values of $\rho = 0.985$ and $\delta = 0.0032$.

Part 5: More Advanced Statistical Foundations.

P.Courtney and N. A. Thacker

- Maximum Likelihood - Revisited.
- 3 cases:
  - Non-Gaussian Errors.
  - Dealing with Outliers.
  - Non-Independent Measurements.
- Demo of camera calibration.

Maximum Likelihood - Revisited.

The most common approach for algorithm development is based on the idea of MAXIMUM LIKELIHOOD, which is derived from the joint probability:

$$P(Y\mathbf{X}) \;=\; (\Pi_i P(X_i|Y))P(Y)$$

Least squares (as we have seen) is derived from Probability theory on the assumption of indepedent Gaussian errors and that the prior probability of the model $P(Y)$ can be ignored.

such that:

$$\begin{aligned} log(P(\mathbf{X}|Y)) \;&=\; \sum_i \; log(P(X_i|Y)) \\ &=\; -\sum_i (X_i - f(i,Y))^2/\sigma_i^2 \end{aligned}$$

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.

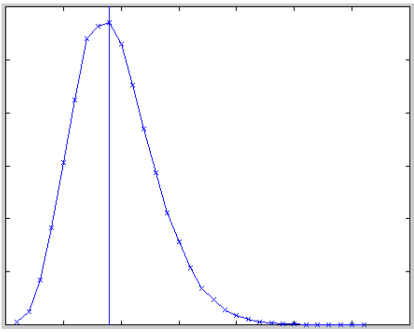
(Ref: Stevens - links Hough transform and maximum likelihood)

Non-Gaussian Errors.

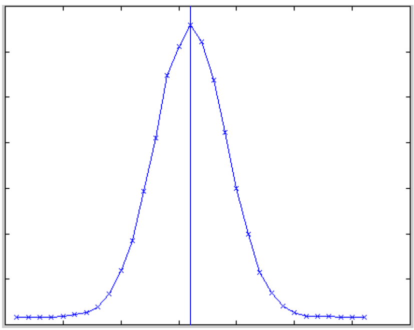
Machine Vision is full of data that cannot be assumed to be from a Gaussian distribution.

There are two forms of the problem:

- The error distribution may be relatively compact but badly skewed.



- There may be outliers caused by data "contamination".



Non-Gaussian Errors.

The general technique for coping with the first problem is to transform the data to remove skewing.

eg:

$$\Delta_x \;=\; f(x)$$

so we seek a function g which will give us

$$\Delta_g(x) \;=\; const$$

using error propagation

$$\Delta_g(x) \;=\; \Delta_x \; dg/dx \;=\; f(x) \; dg/dx \;=\; const$$

ie: integrate the reciprocal of the error dependence:

$$g = \int \frac{const}{f(x)} dx$$

Non-Gaussian Error Example: Stereo data

$$z = fI/(x_l - x_r)$$

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:

$$-log(P) \;=\; \sum_i \; min((X_i - f(i,Y))^2/\sigma_i^2, 9.0)$$

The choice of 9.0 (3 σ) 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.

Demo: Camera Calibration.

- corner detection and matching
- optimisation to camera model
- outlier rejection
  - binomial plot
- rerun optimisation
  - 1900 matches rises to 2200

Dealing with Outliers.

Alternative involves 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.

Dealing with Outliers (cont).

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

$$-log(P) \;=\; \sum_i log(1 \;+\; 1/2(X_i \;-\; f(i,Y))/\sigma_i)^2)$$

These are continuous, so we can use derivative methods for optimisation.

However, the price we pay 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 are slower.

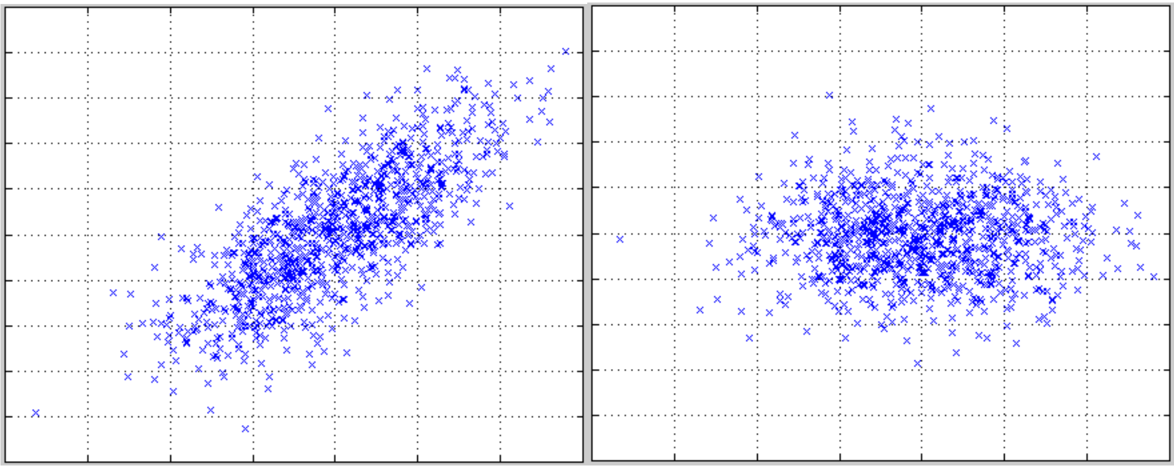
Non-Independent Measurements.

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

$$R = \sum_i (X_j - X_m) \otimes (X_j - X_m)$$

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



Non-Independent Measurements (cont).

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

$$\lambda_k = \frac{1}{w_k^2}$$

Identifying Correlations.

Correlation produces systematic changes in one paramater due to changes in another.

This can be visualised by producing a scatter-plot of the two variables $f(x, y)$.

In general for any two variables to be un-correlated knowledge of one must give no information regarding the other.

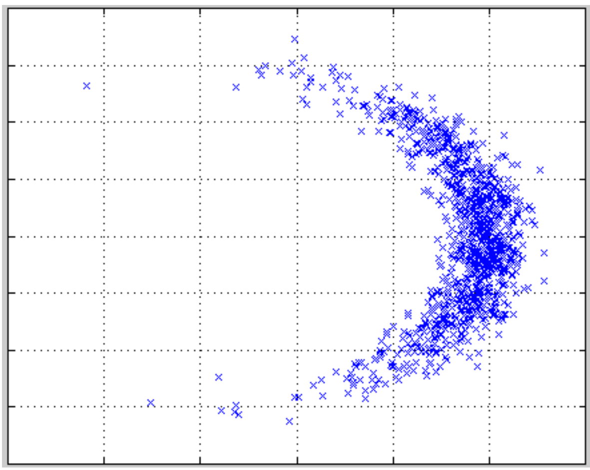
In terms of the scatter plot this means that the structure seen must be entirely modelable in terms of the outer-product of the two marginal distributions.

$$f(x, y) \;=\; f(x) \otimes f(y)$$

ie: decomposable.

Identifying Correlations (cont).

Principal component analysis works by rotating the axes of the space to align along the axes of major variance of the data. This may not necessarily de-correlate the data.



There exist techniques for decorrelating non-linear relationship but the methods are often quite difficult to use. (Ref: Sozou and Cootes using a bottleneck neural network)

Part 6 : Evaluating Representation.

N. A. Thacker and P.Courtney.

- Evaluating Algorithms.
- Optimal Interpretation Algorithms.
- Completeness in Shape Recognition: Fourier, Moments and Pairwise Geometric Histograms.
- Completeness in Texture recognition: Gabor, Wavelets.

Evaluating Algorithms.

Probels such as object recognition will require enourmous data sets of a wide variety of examples, but which?

Can define theoretical measures which only need confirmation on small data sets.

One example is separability (see later) (Ref : Maybank) Another 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 invertable. 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).

Completeness: Shape Representation.

Fourier Descriptors.

One example of a complete algorithm is the Fourier descriptor of an object boundary. The existence of an inverse process for this makes the technique automatically complete.

However, this simple representation in x and y is not scale or rotation invariant.

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 circulatiry.

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

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 α_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 **representaional ambiguity**.

Moment Descriptors.

Appart from Fourier descriptors, the other most 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.

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

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

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:

$$i = \frac{u_{10}}{u_{00}} \qquad \text{and} \qquad j = \frac{u_{01}}{u_{00}} \tag{2}$$

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

$$u_{pq} = \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} (i - i)^p (j - j)^q f(i, j) \tag{3}$$

The basic moment equations are complete (again there is an inverse).

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

$$M_1 = (u_{20} + u_{02})$$

$$M_2 = (u_{20} - u_{02})^2 + 4u_{11}^2$$

$$M_3 = (u_{30} - 3u_{12})^2 + (3u_{21} - u_{03})^2$$

$$M_4 = (u_{30} + u_{12})^2 + (u_{21} + u_{03})^2$$

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

$$+ (3u_{21} - u_{03})(u_{21} + u_{03})(3(u_{30} + u_{12})^2 - (u_{21} + u_{03})^2)$$

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

$$+4u_{11}(u_{30} + 3u_{12})(u_{21} + u_{03})$$

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

$$- (u_{30} - 3u_{12})(u_{21} + u_{03})(3(u_{30} + u_{12})^2 - (u_{21} + u_{03})^2)$$

We can also recompute the original moment descriptors from the invariant quantities, so the rotational invariant equations are still complete.

However, errors do not propegate well through the moment calculations and successively higher terms become increasingly unstable. Thus we are limited to the practical number of terms that we can actually use for recognition.

In practice moment descriptors are not actually complete.

Pairwise Geometric Histograms.

Unlike the previous two represenatation schemes PGH's have been designed to directly encode local shape information. (Demo: pairwise)

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

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

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

Scale invariance can be obtained by interpolating matching responses across a range of scales.

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 appart 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.

Finally, recognition can be performed using a simple nearest neighbour strategy based on the histograms which is gauranteed to be optimal

Reconstructed Data.

Completeness: Texture Representation.

Gabor Filters.

We know that the most compact function in both

the spatial and frequency domain is the Gaussian. Can we therefore think of a way of performing a frequency analysis (sinusoidal convolution) with a Gaussian dependancy.

The simplest idea would be to multiply the Gaussian and sinusoidal functions to give a spatially located but frequency tuned convolution kernal. This is an example of the Gabor filter.

Gabor filters have several free parameters to adjust the scale of the Gaussian and sine components. They can also be oriented in 2D within the image plane.

They form a large possible set of image representations, too large! Which ones should we use for classification, segmentation etc.

The Gabor filter does not form a complete set of filters, nor are they orthogonal. Thus it is not possible to perform an inverse.

Discrete Wavelet.

Like the Fourier transform, the wavelet transform has a discrete (and therefore programable) form.

$$\psi_{mn}(t) \;=\; {a_0}^{-m/2}\;\psi(\frac{t - nb_0}{a_0^m})$$

$$S_{mn} \;=\; \int_{-\infty}^{\infty} \psi'_{mn}(t) S(t)\;dt$$

$$s_t \;=\; K_\psi \;\sum_m \sum_n \; S_{mn}\psi_{mn}(t)$$

Generally $a_0 \;=\; 2^{1/v}$ where $v \;=\;$ voices/octave.

Any application using the Fourier Transform can be formulated using wavelets to give more accurately localised temporal and frequency information.

The existance of an inverse implies that the set of wavelet transforms for an image region can be used as a complete representation.

Part 7: Pattern Recognition and Neural Networks

N. A. Thacker and P. Courtney

- Pattern Space Separability.
- Honest Classifiers.
- Neural Network Training Criteria.
- Statistical Testing.
- Akaike Information Criteria.

Seperability of the Pattern Space.

All pattern recognition systems make explicit assumptions regarding the expected distribution of the data.

The minimum assumption we can make is that the data we have is sampled from the class distributions with some measurement error.

Construction of a Parzen Classifier using the expected measurement distribution ($G(D_i)$) for each data point gives a minimum assumption Bayes classifier.

$$P(C_n|D_i) = \frac{\sum_{i_n}^{I_n} G_i(D_i)}{\sum_n \sum_{i_n} G_i(D_i)}$$

This can be used to construct a classification matrix for the space which reflects data separability.

(Demo: xgobi)

Exclusion of the classified point from the model gives a **cross-validated** estimate of performance.

Disease	Norm.	F.T.D.	Vas.D.	Alz.
$Normal$	7	2	8	1
$Fronto-Temporal$	5	21	3	7
$Vascular$	3	2	13	4
$Alzheimers$	1	3	6	28

Table 1. Disease (rows) vs classification (columns) for a cross-validated Parzen classifier.

Averaging leave-one-out and leave-all-in results gives an estimate of the performance which would result with infinite statistics with the same prior ratios. (Ref: Cox)

Disease	Norm.	F.T.D.	Vas.D.	Alz.
$Normal$	12.0	1.0	4.5	0.5
$Fronto-Temporal$	2.5	28	2	3.5
$Vascular$	1.5	1	17.5	2
$Alzheimers$	0.5	1.5	3	33

Table 2. Disease (rows) vs predicted classification (columns) for unlimited statistics.

Honest Classifiers.

Any classification system which attempts to estimate conditional probabilities of classification $P(C|D)$ should produce uniform frequency distributions when tested on the data it is intended for.

This can be used as the basis for a simple test.

The **Honest** Classsifier will produce errors $1-P(C|D)$ of the time for a forced decision based on $P(C|D)$.

Only honest classifiers are Bayes optimal.

This result has been used to show that Markov update schemes when used for regional labelling are not optimal. (Ref: Poole)

Neural Network Training Criteria.

Under certain conditions artificial neural networks can be shown to estimate Bayesian conditional probabilities.

- The most common optimisation function is the least squares (Gaussian based) error criteria which summed over the entire data set for output k gives.

$$E_k \;=\; \sum_n (o(I_n) - t_{nk})^2$$

  where t_{nk} is the nth training output and o is the output from the network for a particular input I_n.

- Provided that we are training with data which defines a 1-from-K coding of the output (ie classification) we can partition the error measure across the K classes according to their relative conditional probabilities $p(C_k|I)$ so that

$$E_k \;=\; \sum_n \sum_k (o(I_n) - t_{nk})^2 \; p(C_k|I_n)$$

expanding the brackets

$$E_k = \sum_n (o^2(I_n) - 2o(I_n) \sum_k t_{nk} \; p(C_k|I_n)$$

$$+ \sum_k t_n^2 \; p(C_k|I_n))$$

$$= \sum_n (o^2(I_n) - 2o(I_n) < t_k|I_n > + < t_k^2|I_n >)$$

where $< a|I >$ is the expectation operator of a at I_n. By completing the square

$$E_k = \sum_n (o(I_n) - < t_n|I_n >)^2 + \sum_n var(t_k|I_n)$$

The last term is purely data dependent so training minimises the first term which is clearly a minimum when $o(I_n) = < t_k|I_n >$.

For a 1-from-K coding of the output and in the limit of an infinite number of samples $< t_k|I_n > = P(C_k|I_n)$.

- Another cost function often defined for network optimisation is the cross-entropy function

  $$E_k = -\sum_n t_{nk} log(o(I_n)) + (1 - t_{nk}) log(1 - o(I_n))$$

  This is motivated by the assumption that desired outputs t_{nk} are independent, binary, random variables and the required output network response represents the conditional probability that these variables would be one.

  The proof of this follows as above with the introduction of the partion of the classification state over the one and zero cases eventually giving

  $$\begin{aligned} E_k &= -\sum_n < t_k | I_n > log(o(I_n)) \\ &+ (1 - < t_k | I_n >) log(1 - o(I_n)) \end{aligned}$$

  which when differentiated with respect to the desired output shows that this function is minimised again when

  $$o(I_n) = < t_k | I_n >$$

Statistical Testing.

- A practical problem in understanding network performance.

- The final cost function value after training provides only a best case estimate of performance.

- Increasing the complexity of the network will always improve the ability of the network to map the training data.

- The ability of the network to provide accurate outputs for unseen data may reduce. The bias-variance dilemma.

- The common solution to this problem is known as 'jack-knifing' or the 'leave-one-out' strategy.

Akaike Information Criteria.

The probabilistic form of the χ^2 is written as follows;

$$\chi^2 \ = \ -2\sum_{i=1}^{N} log(p(x_i,\theta))$$

The limit of the bias is estimated directly as ;

$$q \ = \ <2\sum_{i=1}^{N} log(p(x_i,\theta))> - <2\sum_{i=1}^{N} log(p(x_i))>$$

where $p(x_i)$ is the true probability from the correct model and $< X >$ denotes the expectation operation.

We can expand this about the true solution θ_0 as;

$$q \ = \ <2\sum_{i=1}^{N} [log(p(x_i,\theta_0)) + (\theta-\theta_0)\partial log(p(x_i,\theta_0))/\partial\theta$$

$$+\frac{1}{2}(\theta-\theta_0)^T H(x_i,\theta_0)(\theta-\theta_0) + h.o.t] >$$

$$- <2\sum_{i=1}^{N} log(p(x_i))>$$

where $H(x_i, \theta_0)$ is the Hessian of the log probability for a single data point.

The second term has an expectation value of zero and excluding the higher orders the remaining terms can be re-written as;

$$q' \;=\; < 2\sum_{i=1}^{N} log(p(x_i, \theta_0)) - 2\sum_{i=1}^{N} log(p(x_i)) >$$

$$+ < \sum_{i=1}^{N} (\theta - \theta_0)^T H(x_i, \theta_0)(\theta - \theta_0) >$$

The first term is $2n$ independent estimates of the Kullback-Liebler distance which is expected to be zero. The second term can be re-written using the matrix $trace$ identity such that

$$q' \;=\; 2nL_{KL}(p, p_{\theta_0})$$

$$+ \; trace(< \sum_{i=1}^{N} H(x_i, \theta_0) >< (\theta - \theta_0)(\theta - \theta_0)^T >)$$

For a well determined system we would expect the trace of the product of these matricies to be the rank of the parameter covariance.

This is simply the number of model parameters k and leads to the standard form of the AIC measure used for model selection

$$AIC \ = \ \chi^2 \ + \ k$$

Summary.

Testing with images is neccesary but not always sufficient.

Conventional Statistical methods for computing covariances can be used to qualify the results of algorithms based upon likelihood statistics.

Robust Statistics will probably be needed for all practical problems, but covariances can still be computed.

Error propegation can be used to assess the effects of noise and guide the design of stable algorithms.

Monte-Carlo techniques can be used when all other methods fail.

Theoretical requirements of algorithms such as scope, optimality and completeness can guide the design of good algorithms.

Pattern classification techniques require representative test data sets and embody the fundamental problem of model selection.

References.

1. Algorithmic Performance Evaluation

   - W. Foerstner, 10 Pros and Cons Against Performance Characterisation of Vision Algorithms, Proc. ECCV Workshop on Performance Characteristics of Vision Algorithms, Cambridge, UK, April 1996.
   - R. M. Haralick, Covariance Propagation in Computer Vision, Proc. ECCV Workshop on Performance Characteristics of Vision Algorithms, Cambridge, UK, April 1996.
   - J. West, J.M. Fitzpatrick, et al, Comparison and Evaluation of Retrospective Intermodality Brain Image Restoration, J Computer Assisted Tomography, 1996.
   - P.J. Phillips et. al., The FERET Evaluation Methodology for Face-Recognition Algorithms, 2000
   - I. Guyon, J. Makhoul, R. Schwartz, and V. Vapnik, What size test set gives good error rate estimates?, IEEE Transactions PAMI, Jan 1998, pp.52-64.

2. Statistics and Error Propagation

   - Lorusso, A., Eggert, D.W., and Fisher, R.B., A Comparison of Four Algorithms for Estimating 3-D Rigid Transmformation, British Machine Vision Conference, Birmingham, UK, September 1995.

3. Image Processing Stability

   - E.P. Lyvers and O.R. Mitchell, Precision edge contrast and orientation estimation, IEEE Transactions PAMI, 10 (6), November 1988, pp. 927-937 (not included).

4. Evaluating Correspondence Algorithms

   - P. Courtney, N. Thacker and A. Clark, Algorithmic Modelling for Performance Evaluation, Proceedings of the ECCV Workshop on Performance Characteristics of Vision Algorithms, Cambridge, UK, April 1996.

5. More Advanced Statistical Foundations

   - R.S. Stephens, A Probabilistic Approach to the Hough Transform, British Machine Vision Conference BMVC90, 1990.
   - P.D. Sozou, T.F Cootes, C.J. Taylor and E.C. Di Mauro, Non-linear Point Distribution Modelling using a Multi-layer Perceptron, British Machine Vision Conference Birmingham, UK, September 1995.

6. Evaluating Representation

   - S.J. Maybank, Probabilistic Analysis of the Application of the Cross Ratio to Model Based Vision, Intl. J. Computer Vision, 16, 1995, pp.5-33 (not included).
   - N.A.Thacker, P.A.Riocreux, and R.B.Yates, 'Assessing the Completeness Properties of Pairwise Geometric Histograms", Image and Vision Computing, 13, 5, 423-429, 1995. (not included)

7. Pattern Recognition and Neural Networks

   - J. Piper, Variability and Bias in Experimentally Measured Classifier Error Rates. Pattern Recognition Letters, 13, 685-692, 1992. (not included)
   - I. Poole, Optimal Probabilistic Relaxation Labeling, British Machine Vision Conference BMVC90, 1990.

For fun: Barry: An Autonomous Train-spotter.

Demos: **URL:** www.niac.man.ac.uk/Tina.html