An Introduction to Active Shape Models *

Tim Cootes

1 Introduction

Biomedical images usually contain complex objects, which will vary in appearance significantly from one image to another. Attempting to measure or detect the presence of particular structures in such images can be a daunting task. The inherent variability will thwart naive schemes. However, by using models which can cope with the variability it is possible to successfully analyse complex images.

Here we will consider a number of methods where the model represents the expected shape and local grey-level structure of a target object in an image.

Model-based methods make use of a prior model of what is expected in the image, and typically attempt to find the best match of the model to the data in a new image. Having matched the model, one can then make measurements or test whether the target is actually present.

This approach is a 'top-down' strategy, and differs significantly from 'bottom-up' methods. In the latter the image data is examined at a low level, looking for local structures such as edges or regions, which are assembled into groups in an attempt to identify objects of interest. Without a global model of what to expect, this approach is difficult and prone to failure.

A wide variety of model based approaches have been explored (see the review below). This chapter will concentrate on a statistical approach, in which a model is built from analysing the appearance of a set of labelled examples. Where structures vary in shape or texture, it is possible to learn what are plausible variations and what are not. A new image can be interpreted by finding the best plausible match of the model to the image data. The advantages of such a method are that

- It is widely applicable. The same algorithm can be applied to many different problems, merely by presenting different training examples.
- Expert knowledge can be captured in the system in the annotation of the training examples.

*appears as Chapter 7: "Model-Based Methods in Analysis of Biomedical Images" in "Image Processing and Analysis", Ed.R.Baldock and J.Graham,Oxford University Press, 2000, pp223-248.

- The models give a compact representation of allowable variation, but are specific enough not to allow arbitrary variation different from that seen in the training set.
- The system need make few prior assumptions about the nature of the objects being modelled, other than what it learns from the training set. (For instance, there are no boundary smoothness parameters to be set.)

The models described below require a user to be able to mark 'landmark' points on each of a set of training images in such a way that each landmark represents a distinguishable point present on every example image. For instance, when building a model of the appearance of an eye in a face image, good landmarks would be the corners of the eye, as these would be easy to identify and mark in each image. This constrains the sorts of applications to which the method can be applied - it requires that the topology of the object cannot change and that the object is not so amorphous that no distinct landmarks can be applied. Unfortunately this makes the method unsuitable in its current form for objects which exhibit large changes in shape, such as some types of cells or simple organisms.

The remainder of this chapter will briefly review other model based methods, describe how one form of statistical model can be built and tested, and how such models can be used to interpret objects in new images.

2 Background

The simplest model is to use a typical example as a 'golden image'. A correlation method can be used to match (or register) the golden image to a new image. If structures in the golden image have been labelled, this match then gives the approximate position of the structures in the new image. For instance, one can determine the approximate locations of many structures in an Magnetic Resonance (MR) image of a brain by registering a standard image, where the standard image has been suitably annotated by human experts. However, the variability of both shape and texture of most targets limits the precision of this method.

One approach to representing the variations observed in an image is to 'hand-craft' a model to solve the particular problem currently addressed. For instance Yuille *et al* [23] build up a model of a human eye using combinations of parameterised circles and arcs. Though this can be effective it is complicated, and a completely new solution is required for every application.

Staib and Duncan [22] represent the shapes of objects in medical images using fourier descriptors of closed curves. The choice of coefficients affects the curve complexity. Placing limits on each coefficient constrains the shape somewhat but not in a systematic way. It can be shown that such fourier models can be made directly equivalent to the statistical models described below, but are not as general. For instance, they cannot easily represent open boundaries.

Kass *et al* [15] introduced Active Contour Models (or 'snakes') which are energy minimising curves. In the original formulation the energy has an internal term which aims to impose smoothness on the curve, and an external term which encourages movement toward image features. They are particularly useful for locating the outline of general amorphous objects, such as some cells (see Chapter 3, secion 3.1 for the application of a snake to microscope images). However, since no model (other than smoothness) is imposed, they are not optimal for locating objects which have a known shape. As the constraints are weak, this can easily converge to incorrect solutions.

Alternative statistical approaches are described by Grenander et al [10] and Mardia et al [18]. These are, however, difficult to use in automated image interpretation. Goodall [9] and Bookstein [2] use statistical techniques for morphometric analysis, but do not address the problem of automated interpretation. Kirby and Sirovich [16] describe statistical modelling of grey-level appearance (particularly for face images) but do not address shape variability.

A more comprehensive survey of deformable models used in medical image analysis is given in [19].

3 Application

We demonstrate the method by applying it to two problems, that of locating features in images of the human face and that of locating the cartilage in MR images of a knee.

Images of the face can demonstrate a wide degree of variation in both shape and texture. Appearance variations are caused by differences between individuals, the deformation of an individual face due to changes in expression and speaking, and variations in the lighting. Typically one would like to locate the features of a face in order to perform further processing (see Figure 1). The ultimate aim may vary widey, from determining the identity or expression of the person to deciding in which direction they are looking [17].

When analysing the MR images of the knee (Figure 2), we wish to accurately locate the boundary of the cartilage in order to estimate its thickness and volume [21].

In both applications difficulties arise because of the complexity of the target images. Bottom-up approaches are unlikely to solve the problems successfully. Complex images give rise to many primitives, which must be linked together correctly to give a successful interpretation. The number of ways in which they can be connected explodes exponentially with the number of components. Thus a bottom-up approach can become inefficient for complex scenes. By building models from sets of examples we can apply the same algorithms to both applications. In each case the output is a set of model landmark points which best match the image data, together with the model parameters required to generate the points. The points or the parameters can then be used in further processing. For instance, in the face example the parameters can be used to estimate the person's expression or identity. In the knee example, we

Figure 1: Example face image annotated with landmarks

Figure 2: Example MR image of knee with cartilage outlined

can estimate the area of the cross-section of the cartilage from the points.

4 Theoretical Background

Here we describe the statistical models of shape and appearance used to represent objects in images. A model is trained from a set of images annotated by a human expert. By analysing the variations in shape and appearance over the training set, a model is built which can mimic this variation. To interpret a new image we must find the parameters which best match a model instance to the image. We describe an algorithm which can do this efficiently. Having fit the model to the image, the parameters or the model point positions can be used to classify or make measurements, or as an input to further processing.

4.1 Building Models

In order to locate a structure of interest, we must first build a model of it. To build a statistical model of appearance we require a set of annotated images of typical examples. We must first decide upon a suitable set of landmarks which describe the shape of the target and which can be found reliably on every training image.

4.1.1 Suitable Landmarks

Good choices for landmarks are points at clear corners of object boundaries, 'T' junctions between boundaries or easily located biological landmarks. However, there are rarely enough of such points to give more than a sparse desription of the shape of the target object. We augment this list with points along boundaries which are arranged to be equally spaced between well defined landmark points (Figure 3).

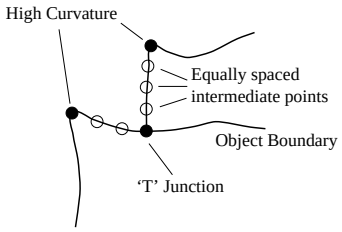

Figure 3: Good landmarks are points of high curvature or junctions. Intermediate points can be used to define boundary more precisely.

To represent the shape we must also record the *connectivity* defining how the landmarks are joined to form the boundaries in the image. This allows us to determine the direction of the boundary at a given point. Suppose the landmarks along a curve are labelled $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$.

For a 2-D image we can represent the n landmark points, $\{(x_i, y_i)\}$, for a single example as the $2n$ element vector, $\mathbf{x}$, where

$$\mathbf{x} = (x_1, \ldots, x_n, y_1, \ldots, y_n)^T \tag{1}$$

If we have s training examples, we generate s such vectors $\mathbf{x}_j$. Before we can perform statistical analysis on these vectors it is important that the shapes represented are in the same co-ordinate frame. The shape of an object is normally considered to be independent of the position, orientation and scale of that object. A square, when rotated, scaled and translated, remains a square.

Appendix 6 describes how to align a set of training shapes into a common co-ordinate frame. The approach is to translate, rotate and scale each shape so that the sum of distances of each shape to the mean ($D = \sum |\mathbf{x}_i - \bar{\mathbf{x}}|^2$) is minimised.

4.1.2 Statistical Models of Shape

Suppose now we have s sets of points $\mathbf{x}_i$ which are aligned into a common co-ordinate frame. These vectors form a distribution in the $2n$ dimensional space in which they live. If we can model this distribution, we can generate new examples, similar to those in the original training set, and we can examine new shapes to decide whether they are plausible examples.

To simplify the problem, we first wish to reduce the dimensionality of the data from $2n$ to something more manageable. An effective approach is to apply Principal Component Analysis (PCA) to the data. The data form a cloud of points in the $2n$-D space, though by aligning the points they lie in a $(2n-4)$-D manifold in this space. PCA computes the main axes of this cloud, allowing one to approximate any of the original points using a model with less than $2n$ parameters (See Appendix 6).

If we apply a PCA to the data, we can then approximate any of the training set, $\mathbf{x}$ using

$$\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{P}\mathbf{b} \tag{2}$$

where $\mathbf{P} = (\mathbf{p}_1 | \mathbf{p}_2 | \ldots | \mathbf{p}_t)$ contains t eigenvectors of the covariance matrix and $\mathbf{b}$ is a t dimensional vector given by

$$\mathbf{b} = \mathbf{P}^T(\mathbf{x} - \bar{\mathbf{x}}) \tag{3}$$

(See Appendix 6 for details).

The vector $\mathbf{b}$ defines a set of parameters of a deformable model. By varying the elements of $\mathbf{b}$ we can vary the shape, $\mathbf{x}$ using Equation 2. The variance of the i^{th} parameter, b_i, across the training set is given by λ_i. By applying limits of $\pm 3\sqrt{\lambda_i}$ to the parameter b_i we ensure that the shape generated is similar to those in the original training set.

We usually call the model variation corresponding to the i^{th} parameter, b_i, as the i^{th} *mode* of the model. The eigenvectors, $\mathbf{P}$, define a rotated co-ordinate

frame, aligned with the cloud of original shape vectors. The vector $\mathbf{b}$ defines points in this rotated frame.

4.1.3 Examples of Shape Models

Figure 4 shows example shapes from a training set of 300 labelled faces (see Figure 1 for an example image showing the landmarks). Each image is annotated with 133 landmarks. The shape model has 36 parameters, and can explain 98% of the variance in the landmark positions in the training set. Figure 5 shows the effect of varying the first three shape parameters in turn between ± 3 standard deviations from the mean value, leaving all other parameters at zero. These 'modes' explain global variation due to 3D pose changes, which cause movement of all the landmark points relative to one another. Less significant modes cause smaller, more local changes. The modes obtained are often similar to those a human would choose if designing a parameterised model, for instance shaking and nodding the head, or changing expression. However, they are derived directly from the statistics of a training set and will not always separate shape variation in an obvious manner.

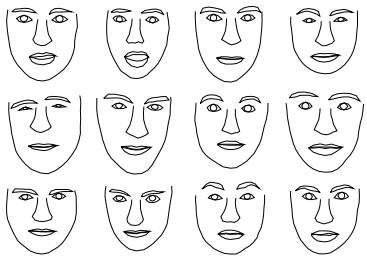

Figure 4: Example shapes from training set of faces

Figure 6 shows examples from a training set of 33 knee cartilage boundaries. (See Figure 2 for an example of the corresponding images). Each is annotated with 42 landmarks. When we apply the alignment and PCA as described above, we generate a model with 15 modes. Figure 7 shows the effect of varying the first three shape parameters in turn between ± 3 standard deviations from the mean value, leaving all other parameters at zero.

Thus in the case of faces the use of the PCA reduced the dimension of the shape vectors from 266 to 36. In the case of the knees, the dimension went from 84 to 15.

4.1.4 Fitting a Model to New Points

A particular value of the shape vector, $\mathbf{b}$, corresponds to a point in the rotated space described by $\mathbf{P}$. It therefore corresponds to an example model. This can

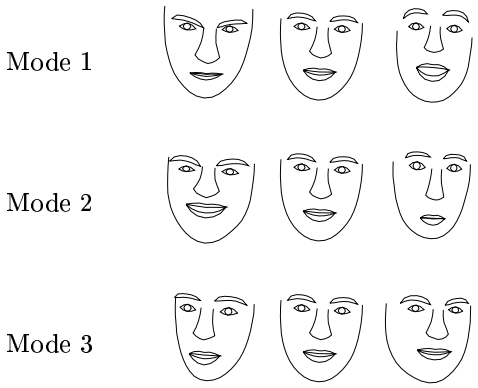

Figure 5: Effect of varying each of first three face model shape parameters in turn between ±3 s.d.

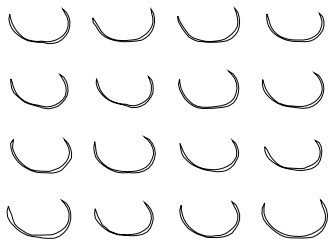

Figure 6: Example shapes from training set of knee cartilages

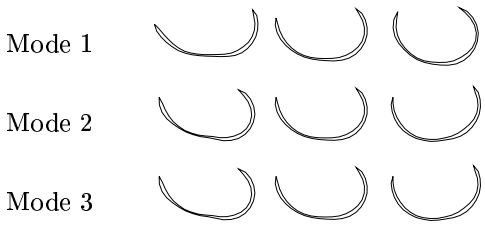

Figure 7: Effect of varying each of first three cartilage model shape parameters in turn between ±3 s.d.

be turned into an example shape using the tranformation from the model co-ordinate frame to the image coordinate frame. Typically this will be a euclidean transformation defining the position, (X_t, Y_t), orientation, θ, and scale, s, of the model in the image.

The positions of the model points in the image, $\mathbf{X}$, are then given by

$$\mathbf{X} = T_{X_t, Y_t, s, \theta}(\bar{\mathbf{x}} + \mathbf{Pb}) \tag{4}$$

Where the function $T_{X_t, Y_t, s, \theta}$ performs a rotation by θ, a scaling by s and a translation by (X_t, Y_t). For instance, if applied to a single point (x, y),

$$T_{X_t, Y_t, s, \theta} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} X_t \\ Y_t \end{pmatrix} + \begin{pmatrix} s\cos\theta & -s\sin\theta \\ s\sin\theta & s\cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \tag{5}$$

Suppose now we wish to find the best pose (translation, scale and rotation) and shape parameters to match a model instance $\mathbf{X}$ to a new set of image points, $\mathbf{Y}$. Minimising the sum of square distances between corresponding model and image points is equivalent to minimising the expression

$$|\mathbf{Y} - T_{X_t, Y_t, s, \theta}(\bar{\mathbf{x}} + \mathbf{Pb})|^2 \tag{6}$$

A simple iterative approach to achieving this is as follows:

Protocol 1: Matching model points to target points

1. Initialise the shape parameters, $\mathbf{b}$, to zero (the mean shape).
2. Generate the model point positions using $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Pb}$
3. Find the pose parameters (X_t, Y_t, s, θ) which best align the model points $\mathbf{x}$ to the current found points $\mathbf{Y}$ (See Appendix 6).
4. Project $\mathbf{Y}$ into the model co-ordinate frame by inverting the transformation T:
$$\mathbf{y} = T^{-1}_{X_t, Y_t, s, \theta}(\mathbf{Y}) \tag{7}$$
5. Project $\mathbf{y}$ into the tangent plane to $\bar{\mathbf{x}}$ by scaling: $\mathbf{y}' = \mathbf{y}/(\mathbf{y}.\bar{\mathbf{x}})$.
6. Update the model parameters to match to $\mathbf{y}'$
$$\mathbf{b} = \mathbf{P}^T(\mathbf{y}' - \bar{\mathbf{x}}) \tag{8}$$
7. If not converged, return to step 2.

Convergence is declared when applying an iteration produces no significant change in the pose or shape parameters. This approach usually converges in a few iterations.

4.1.5 Testing How Well the Model Generalises

The shape models described use linear combinations of the shapes seen in a training set. In order to be able to generate new versions of the shape to match to image data, the training set must exhibit all the variation expected in the class of shapes being modelled. If it does not, the model will be over-constrained and will not be able to match to some types of new example. For instance, a model trained only on squares will not generalise to rectangles.

One approach to estimating how well the model will perform is to use 'leave-one-out' experiments (see Chapter 4). Given a training set of s examples, build a model from all but one, then fit the model to the example missed out and record the error (for instance using (6)). Repeat this, missing out each of the s examples in turn. If the error is unacceptably large for any example, more training examples are probably required. However, small errors for all examples only mean that there is more than one example for each type of shape variation, not that all types are properly covered (though it is an encouraging sign).

Equation (6) gives the sum of square errors over all points, and may average out large errors on one or two individual points. It is often wise to calculate the error for each point and ensure that the maximum error on any point is sufficiently small.

4.1.6 Choice of Number of Modes

Section 4.1.2 suggested that during training the number of modes, t, could be chosen so as to explain a given proportion (eg 98%) of the variance exhibited in the training set.

An alternative approach is to choose enough modes that the model can approximate any training example to within a given accuracy. For instance, we may wish that the best approximation to an example has every point within one pixel of the corresponding example points.

To achieve this we build models with increasing numbers of modes, testing the ability of each to represent the training set. We choose the first model which passes our desired criteria.

Additional confidence can be obtained by performing this test in a miss-one-out manner. We choose the smallest t for the full model such that models built with t modes from all but any one example can approximate the missing example sufficiently well.

4.2 Image Interpretation with Models

4.2.1 Overview

To interpret an image using a model, we must find the set of parameters which best match the model to the image. This set of parameters defines the shape and position of the target object in an image, and can be used for further processing, such as to make measurements or to classify the object.

There are several approaches which could be taken to matching a model instance to an image, but all can be thought of as optimising a cost function. For a set of model parameters, $\mathbf{c}$, we can generate an instance of the model projected into the image. We can compare this hypothesis with the target image, to get a fit function $F(\mathbf{c})$. The best set of parameters to interpret the object in the image is then the set which optimises this measure. For instance, if $F(\mathbf{c})$ is an error measure, which tends to zero for a perfect match, we would like to choose parameters, $\mathbf{c}$, which minimise the error measure.

Thus, in theory all we have to do is to choose a suitable fit function, and use a general purpose optimiser to find the minimum. There are many approaches to optimisation which can be used, for instance simplex, Powells Method [20] or Genetic Algorithms [8]. The minimum is defined only by the choice of function, the model and the image, and is independent of which optimisation method is used to find it. However, in practice, care must be taken to choose a function which can be optimised rapidly and robustly, and an optimisation method to match.

4.2.2 Choice of Fit Function

Ideally we would like to choose a fit function which represents the probability that the model parameters describe the target image object, $P(\mathbf{c}|\mathbf{I})$ (where $\mathbf{I}$ represents the image). We then choose the parameters which maximise this probability.

In the case of the shape models described above, the parameters we can vary are the shape parameters, $\mathbf{b}$, and the pose parameters X_t, Y_t, s, θ.

The form of the fit measure, however, is harder to determine. If we assume that the shape model represents boundaries and strong edges of the object, a useful measure is the distance between a given model point and the nearest strong edge in the image (Figure 8).

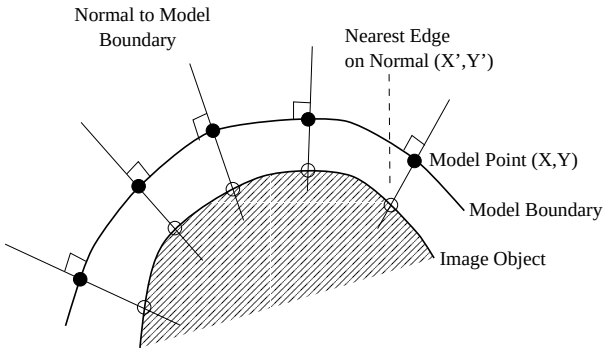

Figure 8: An error measure can be derived from the distance between model points and strongest nearby edges.

If the model point positions are given in the vector $\mathbf{X}$, and the nearest edge

points to each model point are $\mathbf{X}'$, then an error measure is

$$F(\mathbf{b}, X_t, Y_t, s, \theta) = |\mathbf{X}' - \mathbf{X}|^2 \tag{9}$$

Alternatively, rather than looking for the best nearby edges, one can search for structure nearby which is most similar to that occuring at the given model point in the training images (see below).

It should be noted that this fit measure relies upon the target points, $\mathbf{X}'$, begin the correct points. If some are incorrect, due to clutter or failure of the edge/feature detectors, Equation (9) will not be a true measure of the quality of fit.

An alternative approach is to sample the image around the current model points, and determine how well the image samples match models derived from the training set. This approach was taken by Haslam *et al* [11].

A related approach is to model the full appearance of the object, including the internal texture. The quality of fit of such an appearance model can be assessed by measuring the difference between the target image and a synthetic image generated from the model. This is beyond the scope of this chapter, but is described in detail in [4].

4.2.3 Optimising the Model Fit

Given no initial knowledge of where the target object lies in an image, finding the parameters which optimise the fit is a difficult general optimisation problem. This can be tackled with general global optimisation techniques, such as Genetic Algorithms or Simulated Annealing [12].

If, however, we have an initial approximation to the correct solution (we know roughly where the target object is in an image, due to prior processing), we can use local optimisation techniques such as Powell's method or Simplex. A good overview of practical numeric optimisation is given by Press *et al* [20].

However, we can take advantage of the form of the fit function to locate the optimum rapidly. We derive an algorithm which amounts to a directed search of the parameter space - the Active Shape Model.

4.3 Active Shape Models

Given a rough starting approximation, an instance of a model can be fit to an image. By choosing a set of shape parameters, $\mathbf{b}$ for the model we define the shape of the object in an object-centred co-ordinate frame. We can create an instance $\mathbf{X}$ of the model in the image frame by defining the position, orientation and scale, using Equation 4.

An iterative approach to improving the fit of the instance, $\mathbf{X}$, to an image proceeds as follows:

Protocol 2: Active Shape Model Algorithm

1. Examine a region of the image around each point $\mathbf{X}_i$ to find the best nearby match for the point $\mathbf{X}_i'$
2. Update the parameters $(X_t, Y_t, s, \theta, \mathbf{b})$ to best fit the new found points $\mathbf{X}$
3. Apply constraints to the parameters, $\mathbf{b}$, to ensure plausible shapes (eg limit so $|b_i| < 3\sqrt{\lambda_i}$).
4. Repeat until convergence.

In practise we look along profiles normal to the model boundary through each model point (Figure 9). If we expect the model boundary to correspond to an edge, we can simply locate the strongest edge (including orientation if known) along the profile. The position of this gives the new suggested location for the model point.

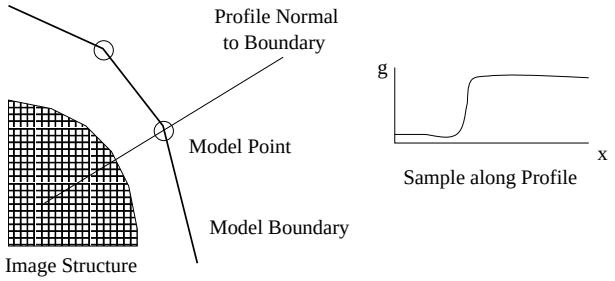

Figure 9: At each model point sample along a profile normal to the boundary

However, model points are not always placed on the strongest edge in the locality - they may represent a weaker secondary edge or some other image structure. The best approach is to learn from the training set what to look for in the target image. This is done by sampling along the profile normal to the boundary in the training set, and building a statistical model of the grey-level structure.

Suppose for a given point we sample along a profile k pixels either side of the model point in the i^{th} training image. We have $2k+1$ samples which can be put in a vector $\mathbf{g}_i$. To reduce the effects of global intensity changes we sample the derivative along the profile, rather than the absolute grey-level values. We then normalise the sample by dividing through by the sum of absolute element values,

$$\mathbf{g}_i \rightarrow \frac{1}{\sum_j |g_{ij}|}\mathbf{g}_i \tag{10}$$

We repeat this for each training image, to get a set of normalised samples $\{\mathbf{g}_i\}$ for the given model point. We assume that these are distributed as a

multivariate gaussian, and estimate their mean $\bar{\mathbf{g}}$ and covariance $\mathbf{S}_g$. This gives a statistical model for the grey-level profile about the point. This is repeated for every model point, giving one grey-level model for each point.

The quality of fit of a new sample, $\mathbf{g}_s$, to the model is given by

$$f(\mathbf{g}_s) = (\mathbf{g}_s - \bar{\mathbf{g}})^T \mathbf{S}_g^{-1} (\mathbf{g}_s - \bar{\mathbf{g}}) \tag{11}$$

This is the Mahalanobis distance of the sample from the model mean, and is linearly related to the log of the probability that $\mathbf{g}_s$ is drawn from the distribution. Minimising $f(\mathbf{g}_s)$ is equivalent to maximising the probability that $\mathbf{g}_s$ comes from the distribution.

During search we sample a profile m pixels either side of the current point ($m > k$). We then test the quality of fit of the corresponding grey-level model at each of the $2(m-k)+1$ possible positions along the sample (Figure 10) and choose the one which gives the best match (lowest value of $f(\mathbf{g}_s)$).

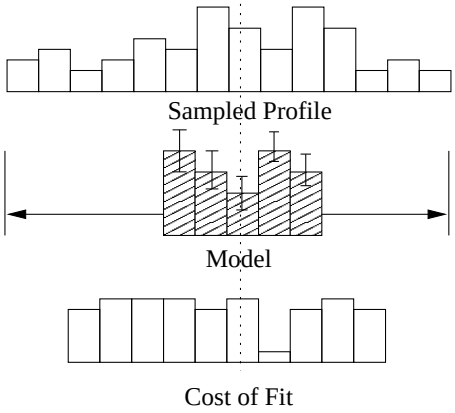

Figure 10: Search along sampled profile to find best fit of grey-level model

This is repeated for every model point, giving a suggested new position for each point. We then apply one iteration of the algorithm given in (4.1.4) to update the current pose and shape parameters to best match the model to the new points.

4.3.1 Multi-Resolution Active Shape Models

To improve the efficiency and robustness of the algorithm, it is implementeded in a multi-resolution framework. This involves first searching for the object in a coarse image, then refining the location in a series of finer resolution images. This leads to a faster algorithm, and one which is less likely to get stuck on the wrong image structure.

For each training and test image, a gaussian image pyramid is built [3]. The base image (level 0) is the original image. The next image (level 1) is formed

by smoothing the original then subsampling to obtain an image with half the number of pixels in each dimension. Subsequent levels are formed by further smoothing and sub-sampling (Figure 11).

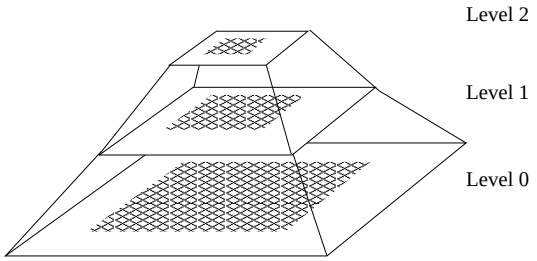

Figure 11: A gaussian image pyramid is formed by repeated smoothing and sub-sampling

During training we build statistical models of the grey-levels along normal profiles through each point, at each level of the gaussian pyramid. We usually use the same number of pixels in each profile model, regardless of level. Since the pixels at level L are 2^L times the size of those of the original image, the models at the coarser levels represent more of the image (Figure 12). Similarly, during search we need only search a few pixels, (n_s), either side of the current point position at each level. At coarse levels this will allow quite large movements, and the model should converge to a good solution. At the finer resolution we need only modify this solution by small amounts.

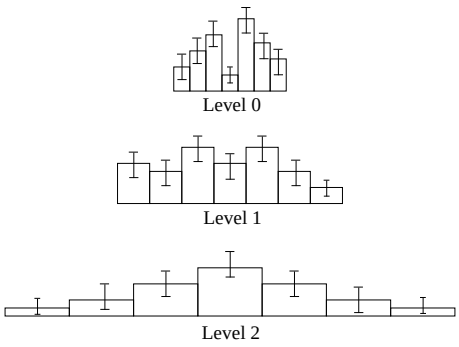

Figure 12: Statistical models of grey-level profiles represent the same number of pixels at each level

When searching at a given resolution level, we need a method of determining when to change to a finer resolution, or to stop the search. This is done by recording the number of times that the best found pixel along a search profile is within the central 50% of the profile (*ie* the best point is within $n_s/2$ pixels

of the current point). When a sufficient number (eg $\geq 90\%$) of the points are so found, the algorithm is declared to have converged at that resolution. The current model is projected into the next image and run to convergence again. When convergence is reached on the finest resolution, the search is stopped.

To summarise, the full MRASM search algorithm is as follows:

Protocol 3: Multi-Resolution Search Algorithm

---

1. Set $L = L_{max}$

2. While $L \geq 0$

(a) Compute model point positions in image at level L.

(b) Search at n_s points on profile either side each current point

(c) Update pose and shape parameters to fit model to new points

(d) Return to (2a) unless more than p_{close} of the points are found close to the current position, or N_{max} iterations have been applied at this resolution.

(e) If $L > 0$ then $L \rightarrow (L - 1)$

3. Final result is given by the parameters after convergence at level 0.

The model building process only requires the choice of three parameters:

Model Parameters	
n	Number of model points
t	Number of modes to use
k	Number of pixels either side of point to represent in grey-model

The number of points is dependent on the complexity of the object, and the accuracy with which one wishes to represent any boundaries. The number of modes should be chosen that a sufficient amount of object variation can be captured (see 4.1.5 above). The number of pixels to represent in each local grey model will depend on the width of the boundary structure (however, using between 3 and 7 pixels either side has given good results for many applications).

The search algorithm has four parameters:

Search Parameters (Suggested default)	
L_{max}	Coarsest level of gaussian pyramid to search
n_s	Number of sample points either side of current point (2)
N_{max}	Maximum number of iterations allowed at each level (5)
p_{close}	Desired proportion of points found within $n_s/2$ of current position (0.9)

The levels of the gaussian pyramid to seach will depend on the size of the object in the image.

4.3.2 Examples of Search

Figure 13 demonstrates using the ASM to locate the features of a face. The model instance is placed near the centre of the image and a coarse to fine search performed. The search starts at level 3 (1/8 the resolution in x and y compared to the original image). Large movements are made in the first few iterations, getting the position and scale roughly correct. As the search progresses to finer resolutions more subtle adjustments are made. The final convergence (after a total of 18 iterations) gives a good match to the target image. In this case at most 5 iterations were allowed at each resolution, and the algorithm converges in much less than a second (on a 200MHz PC).

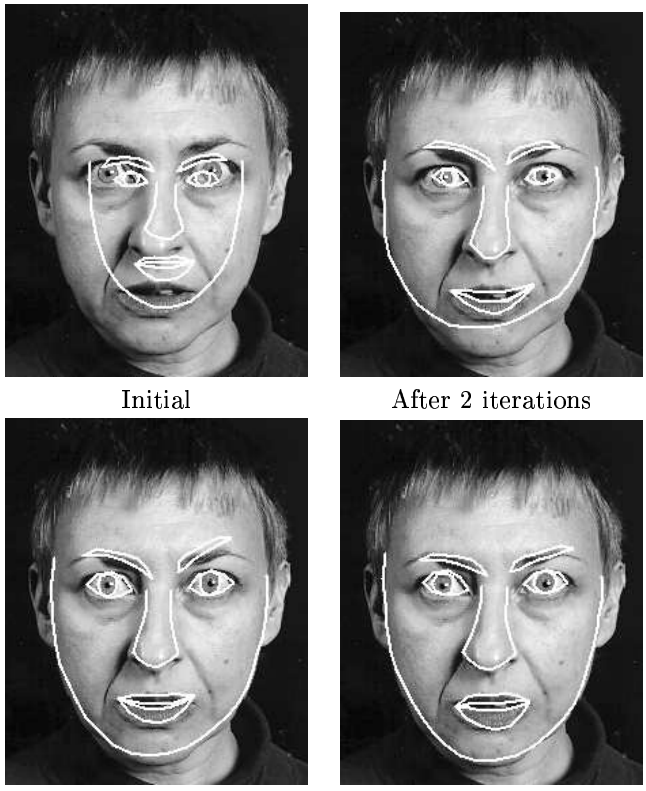

Figure 13: Search using Active Shape Model of a face

Figure 14 demonstrates how the ASM can fail if the starting position is too far from the target. Since it is only searching along profiles around the current

position, it cannot correct for large displacements from the correct position. It will either diverge to infinity, or converge to an incorrect solution, doing its best to match the local image data. In the case shown it has been able to locate half the face, but the other side is too far away.

Initial | After 2 iterations | After 20 Iterations

Figure 14: Search using Active Shape Model of a face, given a poor starting point. The ASM is a local method, and may fail to locate an acceptable result if initialised too far from the target

Figure 15 demonstrates using the ASM of the cartilage to locate the structure in a new image. In this case the search starts at level 2, samples at 2 points either side of the current point and allows at most 5 iterations per level. A detailed description of the application of such a model is given by Solloway *et. al.* in [21].

5 Discussion

Active Shape Models allow rapid location of the boundary of objects with similar shapes to those in a training set, assuming we know roughly where the object is in the image. They are particularly useful for:

- Objects with well defined shape (eg bones, organs, faces etc)
- Cases where we wish to classify objects by shape/appearance
- Cases where a representative set of examples is available
- Cases where we have a good guess as to where the target is in the image

However, they are not necessarily appropriate for

- Objects with widely varying shapes (eg amorphous things, trees, long wiggly worms etc)
- Problems involving counting large numbers of small things

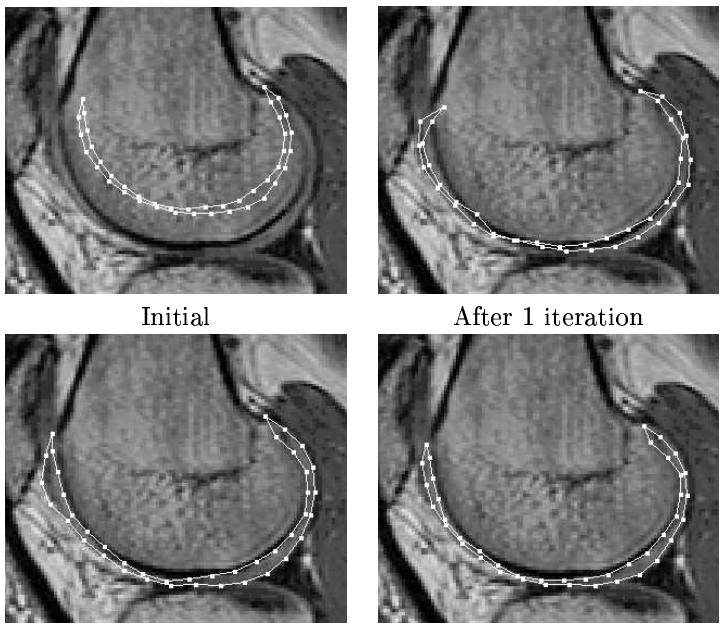

Figure 15: Search using ASM of cartilage on an MR image of the knee

- Problems in which position/size/orientation of targets is not known approximately (or cannot be easily estimated).

In addition, it should be noted that the accuracy to which they can locate a boundary is constrained by the model. The model can only deform in ways observed in the training set. If the object in an image exhibits a particular type of deformation not present in the training set, the model will not fit to it. This is true of fine deformations as well as coarse ones. For instance, the model will usually constrain boundaries to be smooth, if only smooth examples have been seen. Thus if a boundary in an image is rough, the model will remain smooth, and will not necessarily fit well. However, using enough training examples can usually overcome this problem.

One of the main drawbacks of the approach is the amount of labelled training examples required to build a good model. These can be very time consuming to generate. However, a 'bootstrap' approach can be adopted. We first annotate a single representative image with landmarks, and build a model from this. This will have a fixed shape, but will be allowed to scale, rotate and translate. We then use the ASM algorithm to match the model to a new image, and edit the points which do not fit well. We can then build a model from the two labelled examples we have, and use it to locate the points in the third. This process is repeated, incrementally building a model, until the ASM finds new examples

sufficiently accurately every time, so needs no more training.

Both the shape models and the search algorithms can be extended to 3D. The landmark points become 3D points, the shape vectors become $3n$ dimensional for n points. Although the statistical machinery is identical, a 3D alignment algorithm must be used (see [13]). Of course, annotating 3D images with landmarks is difficult, and more points are required than for a 2D object. In addition, the definition of surfaces and 3D topology is more complex than that required for 2D boundaries. However, 3D models which represent shape deformation can be successfully used to locate structures in 3D datasets such as MR images (for instance [14]).

The ASM is well suited to tracking objects through image sequences. In the simplest form the full ASM search can be applied to the first image to locate the target. Assuming the object does not move by large amounts between frames, the shape for one frame can be used as the starting point for the search in the next, and only a few iterations will be required to lock on. More advanced techniques would involve applying a Kalman filter to predict the motion [1][7].

The shape models described above assume a simple gaussian model for the distribution of the shape parameters, $\mathbf{b}$. A more general approach is to use a mixture of gaussians. We need to ensure that the model generates plausible shapes. Using a single gaussian we can simply constrain the parameters with a bounding box or hyper-ellipse. With a mixture of gaussians we must arrange that the probability density for the current set of parameters is above a suitable threshold. Where it isn't, we can use gradient ascent to find the nearest point in parameter space which does give a plausible shape [5]. However, in practice, unless large non-gaussian shape variations are observed and the target images are noisy or cluttered, using a single gaussian approximation works perfectly well.

To summarise, by training statistical models of shape from sets of labelled examples we can represent both the mean shape of a class of objects and the common modes of shape variation. To locate similar objects in new images we can use the Active Shape Model algorithm which, given a reasonable starting point, can match the model to the image very quickly.

6 Implementation

Though the core mathematics of the models described above are relatively simple, a great deal of machinery is required to actually implement a flexible system. This could easily be done by a competent programmer.

However, implementations of the software are already available.

The simplest way to experiment is to obtain the MatLab package implementing the Active Shape Models, available from Visual Automation Ltd. This provides an application which allows users to annotate training images, to build models and to use those models to search new images. In addition the package allows limited programming via a MatLab interface.

A C++ software library, co-written by the author, is also available from

Visual Automation Ltd. This allows new applications incorporating the ASMs to be written.

See `http://www.wiau.man.ac.uk/VAL` for details of both of the above.

It is intended that a free (C++) software package implementing ASMs will be provided for the Image Understanding Environment (a free computer vision library of software). For more details and the latest status, see `http://s20c.smb.man.ac.uk/services/IUE/IUE_gate.html`.

In practice the algorithms work well on mid-range PC (200 Mhz). Search will usually take less than a second for models containing up to a few hundred points.

Details of other implementations will be posted on `http://www.wiau.man.ac.uk`

Acknowledgements

Dr Cootes is grateful to the EPSRC for his Advanced Fellowship Grant, and to his colleagues at the Wolfson Image Analysis Unit for their support. The MR cartilage images were provided by Dr C.E. Hutchinson and his team, and were annotated by Dr S.Solloway. The face images were annotated by G. Edwards, Dr A. Lanitis and other members of the Unit.

Appendix A Aligning the Training Set

There is considerable literature [9, 6] on methods of aligning shapes into a common co-ordinate frame, the most popular approach being Procrustes Analysis [9]. This aligns each shape so that the sum of distances of each shape to the mean ($D = \sum |\mathbf{x}_i - \bar{\mathbf{x}}|^2$) is minimised. It is poorly defined unless constraints are placed on the alignment of at least one of the shapes. Typically one would ensure the shapes are centred on the origin, have a mean scale of unity and some fixed but arbitrary orientation.

Though analytic solutions exist to the alignment of a set [6], a simple iterative approach is as follows:

Protocol 4: Aligning a Set of Shapes

1. Translate each example so that its centre of gravity is at the origin.
2. Choose one example as an initial estimate of the mean shape and scale so that $|\bar{\mathbf{x}}| = \sqrt{\bar{x}_1^2 + \bar{y}_1^2 + \bar{x}_2^2 \ldots} = 1$.
3. Record the first estimate as $\bar{\mathbf{x}}_0$ to define the default orientation.
4. Align all the shapes with the current estimate of the mean shape.
5. Re-estimate the mean from aligned shapes.

6. Apply constraints on scale and orientation to the current estimate of the mean by aligning it with $\bar{\mathbf{x}}_0$ and scaling so that $|\bar{\mathbf{x}}| = 1$.

7. If not converged, return to **4**.

   (Convergence is declared if the estimate of the mean does not change significantly after an iteration)

The operations allowed during the alignment will affect the shape of the final distribution. A common approach is to centre each shape on the origin, then to rotate and scale each shape into the *tangent space* to the mean so as to minimise D. The tangent space to $\mathbf{x}_t$ is the hyperplane of vectors normal to $\mathbf{x}_t$, passing through $\mathbf{x}_t$. ie All the vectors $\mathbf{x}$ such that $(\mathbf{x}_t - \mathbf{x}).\mathbf{x}_t = 0$, or $\mathbf{x}.\mathbf{x}_t = 1$ if $|\mathbf{x}_t| = 1$.

To align two shapes, $\mathbf{x}_1$ and $\mathbf{x}_2$, each centred on the origin, we choose a scale, s, and rotation, θ, so as to minimise $|T_{s,\theta}(\mathbf{x}_1) - \mathbf{x}_2|^2$, the sum of square distances between points on shape $\mathbf{x}_2$ and those on the scaled and rotated version of shape $\mathbf{x}_1$. Appendix 6 gives the optimal solution.

The simplest way to find the optimal point in the tangent plane is to first align the current shape, $\mathbf{x}$, with the mean, allowing scaling and rotation, then project into the tangent space by scaling $\mathbf{x}$ by $1/(\mathbf{x}.\bar{\mathbf{x}})$.

Since we normalise the scale and orientation of the mean at each step, the mean of the shapes projected into the tangent space, $\bar{\mathbf{x}}$, may not be equal to the (normalised) vector defining the tangent space, $\mathbf{x}_t$. We must retain $\mathbf{x}_t$ so that when new shapes are studied, they can be projected into the same tangent space as the original data.

Different approaches to alignment can produce different distributions of the aligned shapes. We wish to keep the distribution compact and keep any non-linearities to a minimum, so recommend using the tangent space approach.

Appendix B Principal Component Analysis

Principal Component Analysis (PCA) allows us to find the major axes of a cloud of points in a high dimensional space. This is useful, as we can then approximate the position of any of the points using a small number of parameters.

Given a set of vectors $\{\mathbf{x}_i\}$, we apply PCA as follows.

Protocol 5: Principal Component Analysis

1. Compute the mean of the data,

$$\bar{\mathbf{x}} = \frac{1}{s}\sum_{i=1}^{s}\mathbf{x}_i \tag{12}$$

2. Compute the covariance of the data,

$$\mathbf{S} = \frac{1}{s-1}\sum_{i=1}^{s}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \tag{13}$$

3. Compute the eigenvectors, $\mathbf{p}_i$, and corresponding eigenvalues, λ_i, of $\mathbf{S}$ (sorted so that $\lambda_i \geq \lambda_{i+1}$). (When there are fewer samples than dimensions in the vectors, there are quick methods of computing these eigenvectors - see Appendix 6.)

4. Each eigenvalue gives the variance of the data about the mean in the direction of the corresponding eigenvector. Compute the total variance from $V_T = \sum_t \lambda_i$

5. Choose the first t largest eigenvalues such that

$$\sum_{i=1}^{t}\lambda_i \geq f_v V_T \tag{14}$$

where f_v defines the proportion of the total variation one wishes to explain (for instance, 0.98 for 98%). (See also section 4.1.6 below)

Given the eigenvectors $\{\mathbf{p}_i\}$, we can approximate any of the training set, $\mathbf{x}$ using

$$\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{P}\mathbf{b} \tag{15}$$

where $\mathbf{P} = (\mathbf{p}_1|\mathbf{p}_2|\ldots|\mathbf{p}_t)$ contains t eigenvectors and $\mathbf{b}$ is a t dimensional vector given by

$$\mathbf{b} = \mathbf{P}^T(\mathbf{x} - \bar{\mathbf{x}}) \tag{16}$$

The eigenvectors $\mathbf{p}_i$ essentially define a rotated co-ordinate frame (centred on the mean) in the original $2n$-D space. The parameters $\mathbf{b}$ are the most significant co-ordinates of the shapes in this rotated frame.

For instance, Figure 16 shows the principal axes of a 2D distribution of vectors. In this case any of the points can be approximated by the nearest point on the principal axis through the mean. $\mathbf{x} \approx \mathbf{x}' = \bar{\mathbf{x}} + b\mathbf{p}$ where b is the distance along the axis from the mean of the closest approach to $\mathbf{x}$.

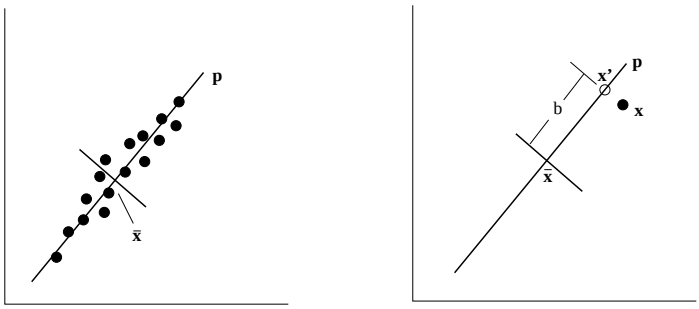

Figure 16: Applying a PCA to a set of 2D vectors. $\mathbf{p}$ is the principal axis. Any point $\mathbf{x}$ can be approximated by the nearest point on the line,$\mathbf{x}'$ (see text).

Appendix C Applying a PCA when there are fewer samples than dimensions

Suppose we wish to apply a PCA to s n-D vectors, $\mathbf{x}_i$, where $s < n$. The covariance matrix is $n \times n$, which may be very large. However, we can calculate its eigenvectors and eigenvalues from a smaller $s \times s$ matrix derived from the data. Because the time taken for an eigenvector decomposition goes as the cube of the size of the matrix, this can give considerable savings.

Subract the mean from each data vector and put them into the matrix $\mathbf{D}$

$$\mathbf{D} = ((\mathbf{x}_1 - \bar{\mathbf{x}})|\ldots|(\mathbf{x}_s - \bar{\mathbf{x}})) \tag{17}$$

The covariance matrix can be written

$$\mathbf{S} = \frac{1}{s}\mathbf{D}\mathbf{D}^T \tag{18}$$

Let $\mathbf{T}$ be the $s \times s$ matrix

$$\mathbf{T} = \frac{1}{s}\mathbf{D}^T\mathbf{D} \tag{19}$$

Let $\mathbf{e}_i$ be the s eigenvectors of $\mathbf{T}$ with corresponding eigenvalues λ_i, sorted into descending order. It can be shown that the s vectors $\mathbf{D}\mathbf{e}_i$ are all eigenvectors of $\mathbf{S}$ with corresponding eigenvalues λ_i, and that all remaining eigenvectors of $\mathbf{S}$ have zero eigenvalues. Note that $\mathbf{D}\mathbf{e}_i$ is not necessarily of unit length so may require normalising.

Appendix D Aligning Two Shapes

Suppose we have two shapes, $\mathbf{x}_1$ and $\mathbf{x}_2$, centred on the origin ($\mathbf{x}_1.\mathbf{1} = \mathbf{x}_2.\mathbf{1} = 0$). We wish to scale and rotate $\mathbf{x}_1$ by (s, θ) so as to minimise $|s\mathbf{A}\mathbf{x}_1 - \mathbf{x}_2|$, where

A performs a rotation of a shape $\mathbf{x}$ by θ. Let

$$a = (\mathbf{x}_1.\mathbf{x}_2)/|\mathbf{x}_1|^2 \tag{20}$$

$$b = \left(\sum_{i=1}^{n}(x_{1i}y_{2i} - y_{1i}x_{2i})\right)/|\mathbf{x}_1|^2 \tag{21}$$

Then $s^2 = a^2 + b^2$ and $\theta = \tan^{-1}(b/a)$. If the shapes do not have centroids on the origin, the optimal translation is chosen to match their centroids, the scaling and rotation chosen as above.

References

[1] A. Baumberg and D. Hogg. Learning flexible models from image sequences. In J.-O. Eklundh, editor, 3^{nd} *European Conference on Computer Vision*, volume 1, pages 299–308. Springer-Verlag, Berlin, 1994.

[2] F. L. Bookstein. Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(6):567–585, 1989.

[3] P. Burt. The pyramid as a structure for efficient computation. In A.Rosenfeld, editor, *Multi-Resolution Image Processing and Analysis*, pages 6–37. Springer-Verlag, Berlin, 1984.

[4] T. F. Cootes, G. J. Edwards, and C. J. Taylor. Active appearance models. In H.Burkhardt and B. Neumann, editors, 5^{th} *European Conference on Computer Vision*, volume 2, pages 484–498. Springer, Berlin, 1998.

[5] T. F. Cootes and C. J. Taylor. A mixture model for representing shape variation. In A. Clarke, editor, 8^{th} *British Machine Vison Conference*, pages 110–119. BMVA Press, Essex, Sept. 1997.

[6] I. Dryden and K. V. Mardia. *The Statistical Analysis of Shape.* Wiley, London, 1998.

[7] G. J. Edwards, C. J. Taylor, and T. F. Cootes. Learning to identify and track faces in image sequences. In 8^{th} *British Machine Vison Conference*, pages 130–139, Colchester, UK, 1997.

[8] D. E. Goldberg. *Genetic Algorithms in Search, Optimisation and Machine Learning.* Addison-Wesley, Wokingham,UK, 1989.

[9] C. Goodall. Procrustes methods in the statistical analysis of shape. *Journal of the Royal Statistical Society B*, 53(2):285–339, 1991.

[10] U. Grenander and M. Miller. Representations of knowledge in complex systems. *Journal of the Royal Statistical Society B*, 56:249–603, 1993.

[11] J. Haslam, C. J. Taylor, and T. F. Cootes. A probabalistic fitness measure for deformable template models. In E. Hancock, editor, 5^{th} *British Machine Vison Conference*, pages 33–42, York, England, Sept. 1994. BMVA Press, Sheffield.

[12] A. Hill, T. F. Cootes, and C. J. Taylor. A generic system for image interpretation using flexible templates. In D. Hogg and R. Boyle, editors, 3^{rd} *British Machine Vision Conference*, pages 276–285. Springer-Verlag, London, Sept. 1992.

[13] A. Hill, T. F. Cootes, and C. J. Taylor. Active shape models and the shape approximation problem. *Image and Vision Computing*, 14(8):601–607, Aug. 1996.

[14] A. Hill, T. F. Cootes, C. J. Taylor, and K. Lindley. Medical image interpretation: A generic approach using deformable templates. *Journal of Medical Informatics*, 19(1):47–59, 1994.

[15] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In 1^{st} *International Conference on Computer Vision*, pages 259–268, London, June 1987.

[16] M. Kirby and L. Sirovich. Appliction of the karhumen-loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(1):103–108, 1990.

[17] A. Lanitis, C. J. Taylor, and T. F. Cootes. Automatic interpretation and coding of face images using flexible models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):743–756, 1997.

[18] K. V. Mardia, J. T. Kent, and A. N. Walder. Statistical shape models in image analysis. In E. Keramidas, editor, *Computer Science and Statistics:* 23^{rd} *INTERFACE Symposium*, pages 550–557. Interface Foundation, Fairfax Station, 1991.

[19] T. McInerney and D. Terzopoulos. Deformable models in medical image analysis: a survey. *Medical Image Analysis*, 1(2):91–108, 1996.

[20] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C (2nd Edition)*. Cambridge University Press, 1992.

[21] S. Solloway, C. Hutchinson, J. Waterton, and C. J. Taylor. Quantification of articular cartilage from MR images using active shape models. In B. Buxton and R. Cipolla, editors, 4^{th} *European Conference on Computer Vision*, volume 2, pages 400–411, Cambridge, England, April 1996. Springer-Verlag.

[22] L. H. Staib and J. S. Duncan. Boundary finding with parametrically deformable models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(11):1061–1075, 1992.

[23] A. L. Yuille, D. S. Cohen, and P. Hallinan. Feature extraction from faces using deformable templates. *International Journal of Computer Vision*, 8(2):99–112, 1992.