Active Shape Models

Andrew Chambers

November 2, 2012

Abstract

I have implemented Active Shape Models as described by Cootes et al [1]. These have been trained and tested using datasets representing human features. I have then applied these models to image search using an iterative algorithm which makes adjustments using image edge strength.

1 Introduction

Suppose we want to model the shape of an object; a trivial example being a rectangle. One way to model a rectangle is with 8 points. One on each corner and one in the centre of each edge. In order to generate a new example of a rectangle, we can simply arrange 8 new points to form a new shape. Of course this model breaks down because with 8 points I can generate a variety of shapes. In order to ensure that this model always produces a rectangle, we can apply constraints on the position of each point relative to its neighbouring points. Using Figure 1 as an example, two such rules could be that $P2$ must be equidistant from $P1$ and $P3$, and the vector through $P3$ and $P5$ must be orthogonal to the vector through $P5$ and $P7$.

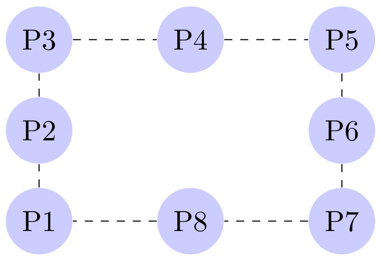


Figure 1: Simple example of a rectangle described with 8 points

Producing these rules is simple for a rectangle, however now consider the problem of modelling the shape of a human hand. We know that the hand can assume many poses yet still be hand shaped, so not only do we need to describe how each point lies in relation to other points, we need to describe how the position of points varies as the pose of a hand varies.

These shapes and the variation of these shapes can not be trivially described by a set of manually produced rules. We would like a method of automatically extracting these rules from example shapes.

A solution to this problem was proposed by Cootes et al [1]. These models are called Active Shape Models (ASMs). ASMs learn from a training set of shapes, and average this training data to create a mean shape along with a set of vectors which allow us to adjust the mean shape such that it still conforms to constraints learnt by the ASM. This allows us to generate a new example which is still recognisable as an instance of the shape that the ASM was trained to represent, but which was not present in the training set.

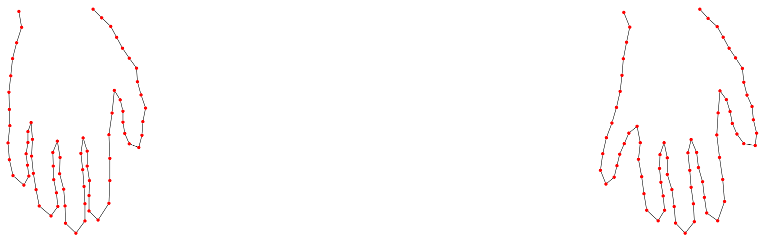


Figure 2: Two examples showing the annotation of a human hand

2 Active Shape Models

First, we define a shape as a vector of n points $\mathbf{x} = \{p_1 \cdots p_n\}$. These points are represented as Cartesian coordinates. For this implementation we concentrate on 2D coordinates, so some point p_k is represented $p_k = (x_k, y_k)$.

It is important to notice that since a shape can be represented as n points, it can also be represented as a vector $\mathbf{x} = \{x_1, y_1, x_2, y_2 \cdots x_n, y_n\}$ where the length of the vector is $2n$.

Using this representation of a shape, we produce a training set with which to train our ASM.

2.1 The Training Set

2.1.1 Annotation of Images for Training

We have a set of N training images which all contain an instance of the shape we wish to model. We generate a set of N shapes $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots \mathbf{x}_N\}$ by manually annotating each image such that p_1 in shape $\mathbf{x}_k$ corresponds exactly to the same part of the shape as p_1 in every other training image.

To do this we define "Landmark Points" for the shape. These points are chosen in such a way that it is possible to easily find this point on any training image. For instance on a human hand this could be the end of a finger or the base of a finger. Once we have established the location of the landmark points on an image, we then mark points at equal intervals along the edge of the shape between the landmark points. Figure 2 shows the result of this annotation on two separate images of human hands.

2.1.2 Aligning the Training Set

Since the shapes are captured from different images, there is no guarantee that the subject in any two images will be in the same position and orientation. We correct this before training the ASM using Procrustes Analysis.

First consider aligning a pair of shapes $\mathbf{x}_i$ and $\mathbf{x}_j$. If $M(s, \theta)[\mathbf{x}]$ is a rotation by θ and a scaling of s applied to the shape $\mathbf{x}$ then we wish to minimise the weighted sum

$$E_j = (\mathbf{x}_i - M(s_j, \theta_j)[\mathbf{x}_j] - \mathbf{t}_j)^T \boldsymbol{W} (\mathbf{x}_i - M(s_j, \theta_j)[\mathbf{x}_j] - \boldsymbol{t}_j), \tag{1}$$

where

$$\mathbf{M}(s, \theta) \begin{bmatrix} x_{jk} \\ y_{jk} \end{bmatrix} = \begin{pmatrix} (s\ cos\theta)x_{jk} - (s\ sin\theta)y_{jk} \\ (s\ sin\theta)x_{jk} + (s\ cos\theta)y_{jk} \end{pmatrix}, \tag{2}$$

$$\mathbf{t} = (t_x, t_y, \cdots, t_x, t_y)^T, \tag{3}$$

and $\mathbf{W}$ is a diagonal matrix of weights calculated for each point. For point p_k we calculate the weight $\mathbf{w}_k$ as the inverse of the average of the variance of the distance of p_k to all other points of all shapes. Thus a point which moves around less relative to other points has a higher weight.

More clearly, we define R_{kl} to be the distance of p_k to p_l, $V_{R_{kl}}$ to be the variance of this distance across all shapes, and

$$w_k = \left(\sum_{l=0}^{n-1} V_{R_{kl}} \right)^{-1} \tag{4}$$

Cootes [1] shows that if we write

$$a_x = s\ cos\theta \qquad a_y = s\ sin\theta, \tag{5}$$

Then using a least squares approach we obtain

$$\begin{pmatrix} X_2 & -Y_2 & W & 0 \\ Y_2 & X_2 & 0 & W \\ Z & 0 & X_2 & Y_2 \\ 0 & Z & -Y_2 & X_2 \end{pmatrix} \begin{pmatrix} a_x \\ a_y \\ t_x \\ t_y \end{pmatrix} = \begin{pmatrix} X_1 \\ Y_1 \\ C_1 \\ C_2 \end{pmatrix}, \tag{6}$$

where

$$X_i = \sum_{k=0}^{n-1} w_k x_{ik} \qquad Y_i = \sum_{k=0}^{n-1} w_k y_{ik} \tag{7}$$

$$Z = \sum_{k=0}^{n=1} w_k (x_{2k}^2 + y_{2k}^2) \qquad W = \sum_{k=0}^{n-1} w_k \tag{8}$$

$$C_1 = \sum_{k=0}^{n-1} w_k (x_{1k} x_{2k} + y_{1k} y_{2k}) \tag{9}$$

$$C_2 = \sum_{k=0}^{n-1} w_k (y_{1k} x_{2k} - x_{1k} y_{2k}) \tag{10}$$

Solving this results in a scaling, rotation and translation which will align shape $\mathbf{x}_i$ close to shape $\mathbf{x}_j$ giving precedence to points with a higher weighting.

Procrustes analysis applies this alignment to every shape in the training set using an iterative approach. The algorithm is shown in Algorithm 1

Algorithm 1 Procrustes algorithm

```
for all shapes $s$ in training_set after first shape do
    align $s$ to first shape
end for
repeat
    Calculate mean shape from aligned shapes
    Normalise orientation of mean shape
    Align all shapes to mean shape
until Convergence
```

Convergence is determined by comparing the difference between the adjustment required and the identity identity matrix.

2.2 Calculating a Mean Shape

We now have a training set of shapes $\mathbf{X}$ which are all aligned as closely as possible.

Given a set of N shapes each containing n points, we can then easily calculate an "Average" shape for the ASM as

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i \tag{11}$$

2.3 Capturing Shape Variation From the Training Set

Consider our vector representation of a shape $\mathbf{x}_i = \{x_1, y_1, x_2, y_2 \cdots x_n, y_n\}$. As shown, this vector is $2n$ elements long. We can therefore consider a $2n$ dimensional space and plot this shape as a point within that space. If we take our training set, we can think of this as a distribution of points within this $2n$ dimensional space, with our mean shape sat in the centre.

If we calculate the covariance matrix of this distribution, we obtain a concise description of how each point in the training set varies with all other points. Moreover, we can obtain eigenvectors and eigenvalues which give us the major and minor axis of this distribution. By sorting the eigenvalues by size, we have a list of the most significant variations across the training set.

In order to describe the whole training set, we require $2n$ eigenvectors, however we can describe a large proportion of the training set with the first t eigenvectors. We choose t such that the sum of the first t eigenvalues is greater than a significant proportion of the sum of eigenvalues over the whole training set.

Using these vectors, we can generate a new shape $\mathbf{x}$

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}\mathbf{b}, \tag{12}$$

Where $\mathbf{P}$ are the eigenvectors of the ASM, and $\mathbf{b}$ is a vector specifying the magnitude with which we move along each eigenvector from the mean.

Cootes [1] suggests that we only move along these eigenvectors of the order

$$-3\sqrt{\lambda_k} \leq b_k \leq 3\sqrt{\lambda_k}, \tag{13}$$

where λ_k is the eigenvalue of the kth eigenvector in order to maintain the integrity of the ASM.

3 Results

I train my implementation using two datasets gathered from Tim Cootes's website [2]. One dataset gives annotated points of a human hand, the other points of a human face.

I have provided images showing the variation along the first three eigenvectors learnt by both Active Shape Models.

3.1 Hand Model

Figure 3 shows the ASM simulating the fingers spreading on the hand. Figure 4 shows the thumb moving away from the other fingers, and Figure 5 shows the middle finger moving. It is interesting to observe that these movements correspond to the largest eigenvectors in the ASM and also to the most common movements we'd associate with a human hand.

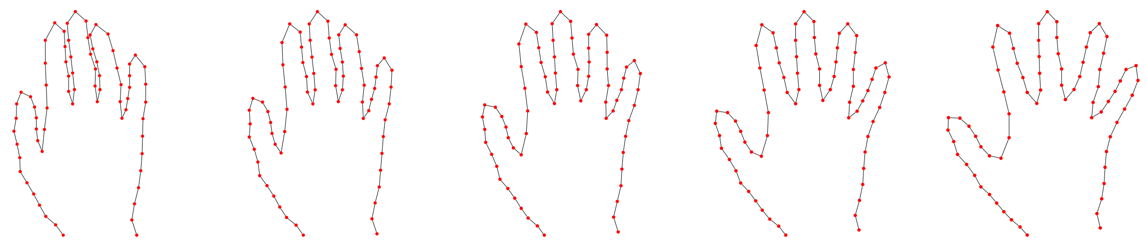


Figure 3: First mode of variation for the hand model

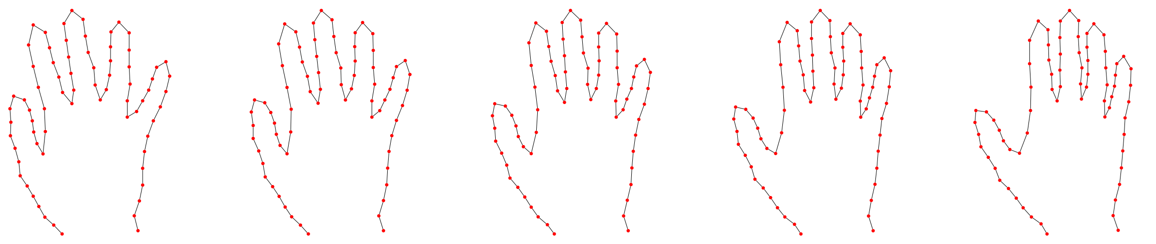


Figure 4: Second mode of variation for the hand model

3.2 Face Model

Figure 6 shows the head turning from side to side. Figure 7 shows the mouth opening and closing, while Figure 8 shows the mouth opening and closing while the head tilts upwards slightly.

4 ASMs for Image Search

We have captured a model of how the shape of particular object varies over a training dataset. We would like to use this model for image search, by attempting to fit it to a similar shape within a new image. Since our ASM can only deform in ways which the ASM has learnt for the object we are searching for, the process will result in an accurate description of the location of all points of the object within the image.

The method given uses the image gradient to find suitable adjustments to the ASM. When used iteratively, this method results in an ASM which moves closer to the correct shape and orientation of the object with each iteration.

4.1 Making Adjustments to the Model

The general algorithm for image search is given in Algorithm 2.

Algorithm 2 Algorithm for image search

```
repeat
    for all points $p$ in ASM do
        Find normal to point
        Search along normal for max edge strength
        Suggest new position for point as a function of edge strength along normal
    end for
    Align ASM to new shape given by suggested points
until Convergence
```

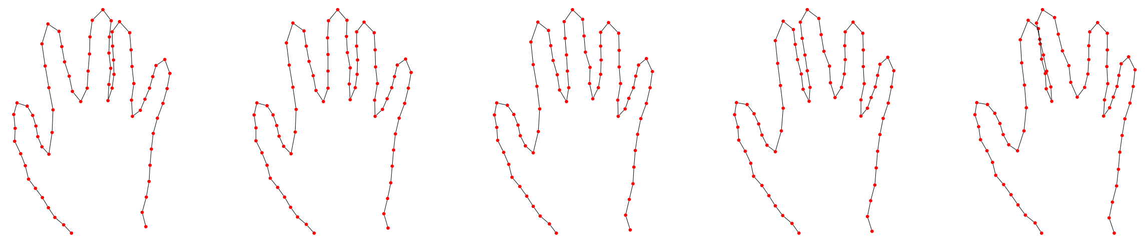


Figure 5: Third mode of variation for the hand model

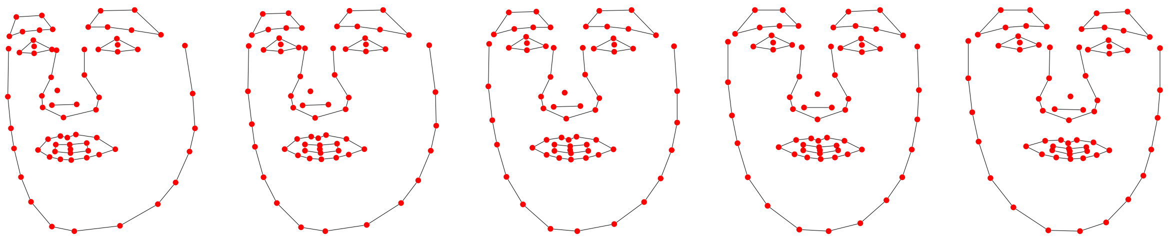


Figure 6: First mode of variation for the face model

Alignment is done in two steps. First the mean shape is aligned with the suggested shape using the same technique given in Section 2.1.2. The second stage of alignment requires adjustment to the ASM parameters using the calculated eigenvectors. Cootes [1] shows that the required adjustment along each eigenvector is calculated as

$$\mathbf{b} = \mathbf{P}^T d\mathbf{x}, \tag{14}$$

where $\mathbf{b}$ is the set of adjustments, $\mathbf{P}$ are the first t eigenvectors, and $d\mathbf{x}$ is the difference between the shape of the current model and the suggested shape.

Convergence is determined by the magnitude of $\mathbf{x}$.

4.2 Refinements to Algorithm

In practise, the algorithm struggles to converge for several reasons, and so we make slight adjustments.

4.2.1 Multi-resolution Search

Firstly, in any image, the object we are searching for will not be the only object in the image, meaning the ASM might get drawn to sharp edges which are nothing to do with that object. To reduce this effect, we do a multi-resolution search. We first search on a low resolution copy of the image. This makes large adjustments to the position of the ASM on the image. We slowly increase the resolution of the image that we perform the search on, which makes progressively finer adjustments to the ASM parameters.

4.2.2 Search Around Normal

Rather than just restrict our search to just the normal to each point, we widen our search a few pixels either side of the normal. This reduces the effect of noise on the search, which is particularly effective when doing a low resolution search. Figure 9 shows the gradient image of a hand along with the ASM plotted in red, the search area in grey, and the suggested position of the current point is shown in blue.

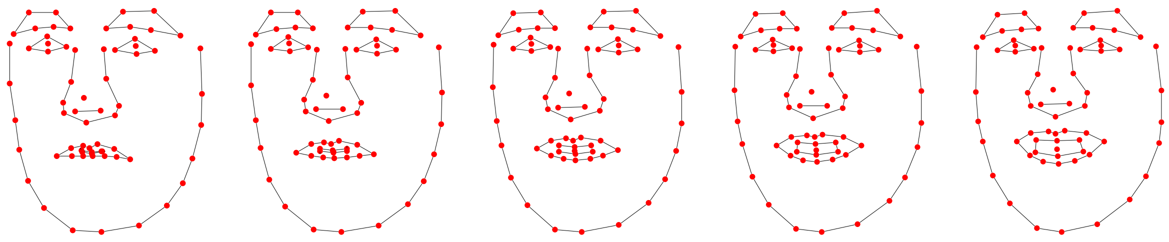


Figure 7: Second mode of variation for the face model

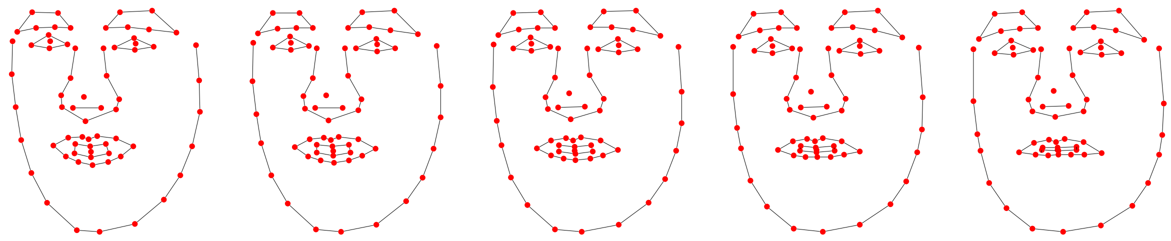


Figure 8: Third mode of variation for the face model

5 Results of Image Search

Figure 10 shows a sequence of images which shown the hand model converging on an image of a hand with a noisy background. We can see that after 10 iterations, the model has already started to converge on fine details. By 50 iterations, almost all points are fitted to the edge of the hand in the image. Some points do not converge on the correct position, the ends of the fingers is a good example of this. This is due to an inadequate training set. Since all images in the training set were of the same hand, the model does not allow for fingers as long as those in the test image.

The image search is very sensitive to starting position. This is helped by a multi-resolution search, but in order for the ASM to converge, a user must position it at roughly the correct position and orientation. If not, the ASM will become pulled to other features in the image and converge on something which is not the intended object.

6 Conclusions

I have implemented Active Shape Models as described by Cootes et al [1] and have constructed models of the human hand and face. Adjusting the mean models using the calculated eigenvectors produces believable variation of the shape. I implemented an image search algorithm using a method also proposed by Cootes which uses the edge strength of an image to make adjustments to the ASM. The method works well when the model is initially positioned close to the intended object in the image, but struggles to converge on the shape otherwise.

References

[1] T.F Cootes, C.J Taylor, D.H Cooper, J.Graham, *Active Shape Models - Their Training and Application*, Computer Vision and Image Understanding, Volume 61, pp. 38-59, 1995

[2] Tim Cootes, *Manchester Computer Science Personal Pages*, available at `http://personalpages.manchester.ac.uk/staff/timothy.f.cootes/`.

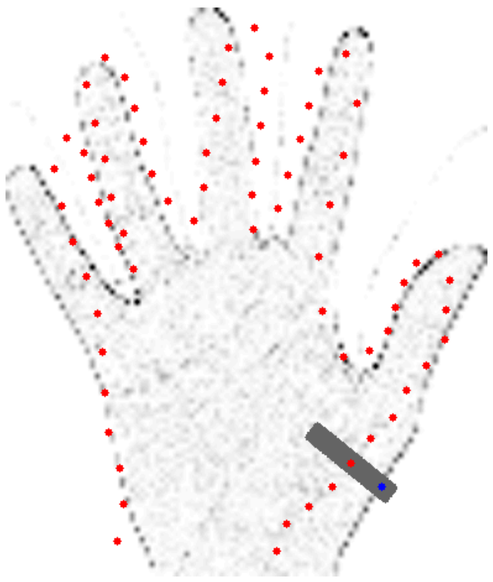
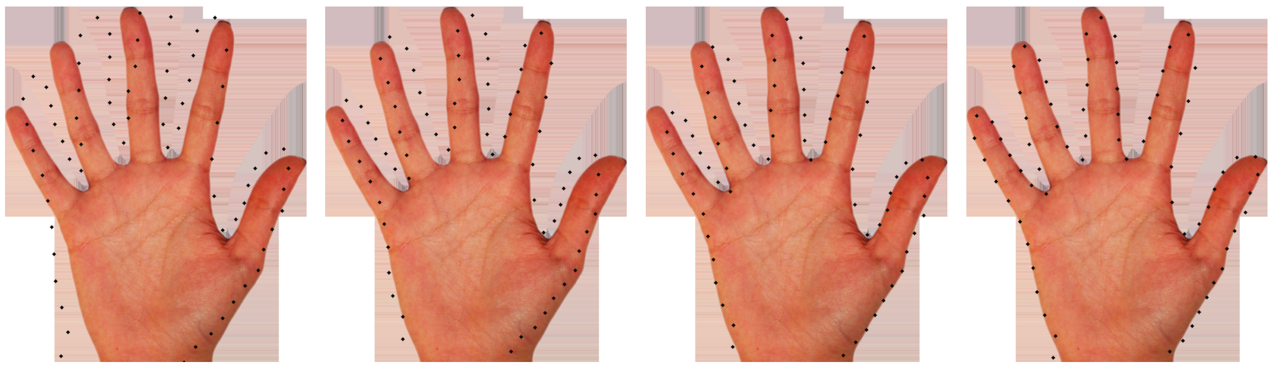


Figure 9: Showing the search area around a point and the suggested movement of the point



(a) 0 iterations

(b) 5 iterations

(c) 10 iterations

(d) 50 iterations

Figure 10: Showing an ASM converging on object during image search