Building Shape Models from Image Sequences using Piecewise Linear Approximation

Derek R. Magee and Roger D. Boyle
School of Computer Studies
University of Leeds
Leeds LS2 9JT, UK
drm@scs.leeds.ac.uk

Abstract

A method of extracting, classifying and modelling non-rigid shapes from an image sequence is presented. Shapes are approximated by polygons where the number of sides is related to the physical features of a shape class rather than any particular shape. A method of 'seeding' the polygonal approximation is given where 'seeds' are automatically extracted from a set of data. Multiple models are built using polygons with different numbers of sides to allow for feature occlusion. Principal component analysis (PCA) is performed on vector representations of the sides of the polygons which are normalised by the total perimeter. This removes the need for normalisation of scale and translation as required in the Point Distribution Model [16]. A 'fit score' metric is defined which gives an indication of how well a given shape fits a model.

1 Introduction

There has been much work carried out on the extraction and classification of non-rigid shapes from image sequences. In this paper a new method of classifying images automatically from an image sequence is proposed. This method is applied in the monitoring of cattle movement to identify a particular non-rigid object in a scene (a cow in our example) and gain knowledge about the way in which this object changes with time.

Kass *et al* [11] use an active shape model (snake) in which an energy function is minimised to extract an object from a scene. Terzopoulos and Szeliski [15] have made some refinements to this technique with the inclusion of Kalman filter techniques (Kalman snakes). Blake *et al* [3] have further refined the technique by adding 'templates' to this process to incorporate a search for a specific shape. In recent years the advantage of such model-based approaches has been shown where prior information about a class of shapes is used in extraction and classification. The point distribution model (PDM) described by Cootes *et al* [16] is a useful way of describing a class of non-rigid shapes where a shape class is described by a fixed number of points. A model is built from a group of shapes fitted with this fixed number of 'reference points' and principal component analysis (PCA) is used to extract 'modes of variation' from the data set. Members of a shape class are described with reference to an eigenspace where the axes are the eigenvectors produced from the principal component analysis. A 'valid' shape is constructed by taking an area

BMVC 1998 doi:10.5244/C.12.40

in this eigenspace centred around the mean shape (origin) and bounded by maximum deviations along each principal axes which are related to the eigenvalues produced by PCA and thus the variance within the data set. PDMs rely on accurate positioning of points and can only describe accurately shape classes which deform in a linear fashion which can result in invalid shapes being included in the model.

There have been a number of methods proposed for point fitting around a pre-extracted contour [1, 2] which involve finding a number of reference points (points of high curvature, extreme points etc) and adding extra points spaced evenly round the contour such that the contour may be approximated by a spline. Hill *et al* [8] have proposed a new scheme whereby a contour is approximated by an arbitrary number of straight lines as defined by the critical point detection algorithm described by Zhu and Chirlian [19]. In the comparison of two shapes the average number of points required to represent the shapes is used and points are placed such that the distance between corresponding points around the perimeter is the same proportion of the total perimeter for each shape.

The method proposed here extends this and seeks to find automatically the optimal number of straight lines required to approximate a given class of shapes from a set of examples to enable accurate models to be built. It is hypothesised that this is related to the physical nature of a particular class of shapes; in other words in an optimal representation of shape lines will represent actual physical features rather than arbitrary straight line sections. For a given shape class multiple models may be required as non-rigid objects may deform in ways which self-occlude physical features. As an extension to these ideas PCA is performed on the lines (as a normalised vector representation) rather than on absolute point positions which gives a scale and translation independent model. This results in the definition of multiple eigenspaces. A 'fit score' metric is defined for each model which is used to group members of the shape class according to the most appropriate model.

There have also been a number of developments on the original linear PDM which use a number of 'sub models' to classify a class of shape more precisely. Bregler and Omohundro [4] define a series of planes in the eigenspace whereas Heap and Hogg [7] define a series of hyper-ellipsoid bounded regions in the eigenspace. Sozou *et al* [14] try to solve the linearity problem of PDMs by constraining shapes to deform along polynomial paths in the eigenspace rather than simply along the axes. There has also been work on further eigen analysis of eigenspace to form a canonical space [9, 6] for which may be useful for classification purposes. These developments, however, restrict the model to a given eigenspace (i.e. a given number of reference points). In the real world we wish to classify classes of shapes that deform non-linearly such that features are occluded and a fixed number of reference points is not appropriate.

2 Initial Video Processing

The footage used is of cows walking in profile from a milking parlour. No attempt was made to constrain the background. For model building purposes the raw video input is pre-processed in two stages to produce a list of adjacent contour pixel points describing the outline.

Firstly 'background subtraction' is carried out to produce a binary silhouette as shown in Figure 2.1. A background image containing no moving objects is taken as a reference. Each colour component (red, green and blue) of this image is subtracted in a pixelwise

fashion from frames containing moving objects, the absolute value taken and the three colour difference components summed. The result of this is a difference image with only one colour component. Various image processing functions (blurring, median filtering, dilation and erosion etc.) may be applied to this image if required to clean up and smooth the image before it is thresholded to give a binary difference image as shown in Figure 2.1. (see [2]).

Figure 2.1: Background Subtracted Binary Image

The next stage is to extract the list of outline points. For model building purposes we are not interested in complex tracking algorithms so we simply select the area with the largest number of adjacent pixels as the object with which to build the model. The outline is extracted by starting at the top left extremity of the selected area and 'walking' round the edge in a 4-connective way (i.e. move up, left, down or right) until the initial point is reached again.

3 Fitting Lines to Shape Outlines

There has been much written on the approximation of continuous functions by straight lines [5, 13, 17, 18]. The general conclusion is that there is no 'correct' solution to such problems. Methods that have been employed to perform such tasks generally fall into two categories (i) Points of maximum curvature or (ii) Iterative end-point fits.

In (i) the continuous function is segmented by picking points of maximum curvature for the continuous function. Curvature is calculated as the differential of tangential angle [12] or less often as arc-chord distance [13]. This method works well for functions that are mainly made up of straight lines but fails when the function has long arcs of constant curvature. Hill and Taylor [1] use a system based on this method to fit points to the outlines of human hand and heart outlines but then go on to approximate the outline by a spline.

In method (ii) the continuous function is segmented using an iterative process. The function is first approximated by a single straight line with end points chosen arbitrarily. If the continuous function deviates from the approximation by more than a specified threshold the line is divided into two lines at the point of maximum deviation. This process continues until no point of interest on the continuous function deviates from the straight line approximation by more than the chosen threshold. This method works reasonably well for functions containing straight and curved segments but is sensitive to choice of initial points. It is also sensitive to 'wild' points and any noise in the continuous function may have a large effect on the final result.

The proposed method is a combination of the two methods with an additional iteration that results in a more robust solution to the problem. Initially the tangential angle of each point on a shape outline is calculated and from these angles curvature at each point is calculated by differentiation. The selection of this method is purely arbitrary and curvature

may equally well be calculated as arc-chord distance. Maxima of curvature are found and lines between successive maxima points used as a first estimate of the straight line approximation. It may be observed in Figure 3.3 that, due to perimeter roughness, there are many more of these points than is necessary to approximate accurately the perimeter and there are areas where there few points due to constant curvature These problems are solved by the following procedure:

To **Add Points** Repeat:

- For each straight line in the approximation calculate maximum perpendicular deviation of the contour. (see fig 3.1)

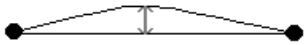

Figure 3.1: Maximum Perpendicular Deviation From Straight Line Approximation

- If the largest maximum deviation is greater than a specified fit threshold add an extra reference point on the contour at the point of maximum deviation replacing the straight line by two new lines.

Until: No new points need be added.

Repeat:

- **Remove Points** : Consider each reference point and the straight line between the two adjacent reference points. If the perpendicular distance between this line and the point is less than the specified fit threshold the point is unnecessary and may be removed. Points are removed in an order such that the point with minimum deviation from its respective line is removed. Points are removed until no distance is greater than the threshold. (see fig 3.2a)
- **Adjust Points** : Consider moving each reference point to each of the two adjacent shape perimeter points. If a move results in a lower average perpendicular deviation of the perimeter from the two lines adjacent to the point and the maximum deviation from both lines is not increased the move is a valid one. Points are moved such that the move that results in the greatest reduction in average deviation is carried out first. Points are adjusted until there are no more valid moves available. (see fig 3.2b)

Until: No further points need be removed or adjusted.

This method is robust in the fact that it produces approximations with similar numbers of lines for similar shapes (see fig 3.5). It is not prone to erroneous results caused by the fit being optimised locally as reported about methods described previously [5]
Note that the effect of the ‘Adjust points’ algorithm is limited and reasonably satisfactory results may be obtained without this stage if speed is an important issue. This gives a large increase in speed as the ‘Remove points’ stage must only be performed once rather than at least twice if the ‘Adjust points’ stage is included. The ‘Add points’ stage may also be removed although the speed advantage is less.

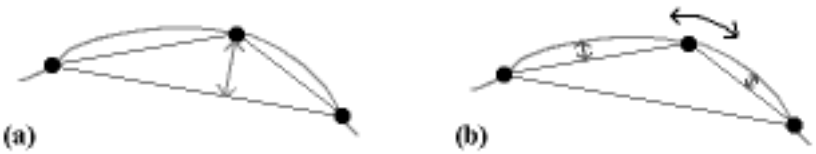

Figure 3.2: (a) Removing Unnecessary Reference Points, (b) Adjusting the Position of Reference Points

Figure 3.3: Maxima of Curvature

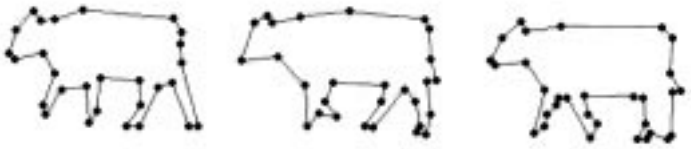

Figure 3.4: Straight Line Approximations to Outlines

4 Building Models from Outline Data Using Seeds

The method described in the previous section produces some aesthetically pleasing cow outlines but the number of lines required varies from shape to shape. Even fairly similar shapes may be represented by slightly different numbers of lines. This is a disadvantage when trying to build a concise model of shape using mathematical methods such as PDM's ([16] and [2]). These methods enforce a pre-determined number of points upon the shape regardless of the shape to be modelled. Using common sense a human subject will approximate groups of similar shapes with the same number of lines and different groups with different numbers of lines where these lines will correspond to actual physical features. This approximation is not only based on the outline of a particular shape but on prior knowledge of the object. An extension to the line fitting method proposed in the previous section includes prior knowledge about the shape we are trying to extract using of a 'seed' to start the line fitting process. A seed is created by hand placing reference points around the perimeter of a single shape belonging to the class to be modelled and converting these points into a series of vectors with an angle component and a magnitude component that is a proportion of the total distance around the perimeter of the shape.

The perimeters of the training shapes are divided up into sections with length proportional to the magnitude components of the seed vectors. In the system implemented the centre point of the largest seed vector is placed at the centre point of the longest line in

the straight line approximation calculated as in the previous section. It is obvious that this method will only work for shape classes where the longest line is a significant physical feature and a more complex method would be required to fit the initial guess if this were not the case however this works well for our example as a cows back is always its longest straight physical feature. Reference points are placed at the nearest maxima of curvature to each intersection of the sections calculated.

This first guess is improved using iterations minimising two criteria; the integral difference (area) between the straight line approximation and the actual shape contour and the difference between the seed vector angles and the straight line approximation angles. The integral error was used rather than average or maximum deviation from the approximation as it provides better information on the quality of the fit. It was observed that the two criteria often conflict and thus algorithms involving iterations that try to minimise both simultaneously are not well conditioned for all shapes. Several algorithms were tried but none converged reliably to a solution for any significant number of test shapes. Algorithms that minimise both criteria separately were designed and found to be much more robust converging to a solution for all test shapes. These algorithms are applied to the first guess sequentially with the two iterations performed several times until convergence of each of the iterations (for the test images used this was approximately three times). Obviously as the two criteria may conflict the results from the two iteration algorithms differ slightly at the final stage and as such the order that the two iterations are carried out is important. It was decided that the angle error minimisation would be carried out first followed by the integral error minimisation. This resulted in a better approximation as generally the angles of the test shapes were not identical to those of the seed shape. Either way round, however, acceptable results were produced. The algorithms are described below:

To **Minimise Angle**, Repeat :

1. For each reference point the two adjacent lines are examined. The absolute difference in angle between these lines and the corresponding lines in the seed is calculated and the two values summed. (see fig 4.1a)

2. For each reference point, moves to the two adjacent points of maximum curvature are considered and the sum of the angle differences calculated as in stage 1.

3. The point move that results in the biggest reduction in angle difference is performed.

Until: No move results in a reduction in angle difference.

To **Minimise Integral**, Repeat :

1. For each reference point the two adjacent lines are examined. The integral error (area) between these lines and the actual perimeter is calculated and summed. (see fig 4.1b)
2. For each reference point moves to the two adjacent points of maximum curvature are considered and the sum of the integral errors calculated as in stage 1).
3. The point move that results in the biggest reduction in total integral error without merging reference points is performed.

Until: No move results in a reduction in integral error.

- Note: Moves to adjacent maxima of curvature are considered in these iterations rather than moves to adjacent perimeter points to speed up the operation. This has little effect on the final result.

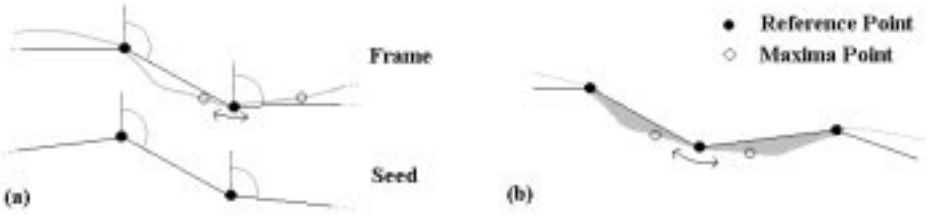

Figure 4.1: (a) Minimising Angle Difference between Data Frame and Seed, (b) Minimising Integral between Perimeter and Straight Line Approximation

For each frame a fit score is calculated using the formula:

$$Fit\,Score = \sum_{n=1}^{No.\,Lines} Integral\,Error_n \times Angle\,Difference_n \qquad (1)$$

For the data set used (a sequence of a cow walking) there are shape discontinuities related to the fact that when a cow's legs come together the number of lines required to represent the shape accurately falls. For this reason three 'seeds' were used, one with legs apart, one with front legs together and one with rear legs together. The training data was thus partitioned into three groups depending on which seed resulted in the lowest fit score for a particular frame. Frames that had a particularly high fit score (greater than a chosen number of standard deviations from the mean) were discarded for the first iteration. The remaining frame fits were converted into vectors and averaged to produce three new seeds. These seeds were applied to the entire data set in the same way as before. This process was iterated until the difference between the averaged output was not significantly different from the seed input. Figure 4.2 shows how the frames were grouped in the final iteration. This shows clearly the cyclic nature of cow walking behaviour and how the models may be used in tracking and behavioural studies. In the frame where no model fits satisfactorily the cow is in a transitionary state (front legs crossed). This suggests a fourth seed may be required. Eigen analysis was performed on the vectors of the three final groups of shapes and modes of variation are shown in fig 4.3.

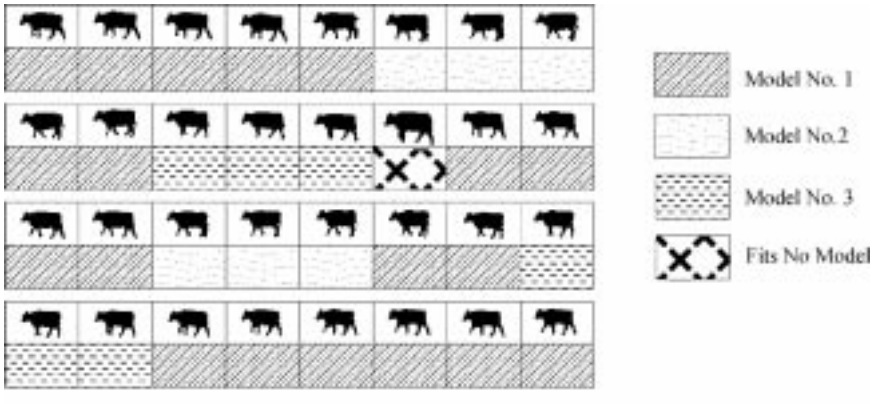

Figure 4.2: Grouping of Successive frames to Form Shape Models

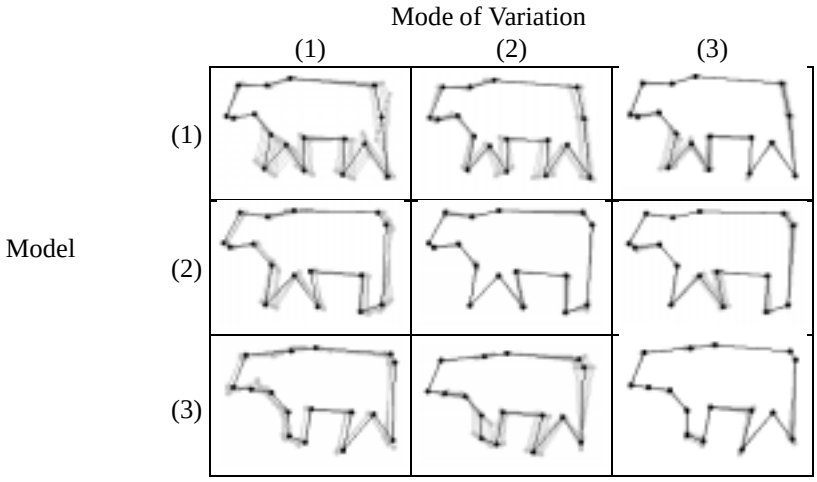

Figure 4.3: Average and Modes of Variation of the Three Shape Models

5 Automatically Extracting Seeds from Outline Data

The results from the 'hand seeded' method described previously are reasonably good but rely on human perception of the number of lines required to represent a given class of shapes and thus the method is not repeatable for an arbitrary shape class. Automation of the seeding process is also desirable if a large number of shape classes are to be extracted from a data set. The method devised uses straight line approximations as extracted in section 3 as initial seeds and a simplification stage to reduce the number of points necessary to describe a shape class. The straight line approximations used are calculated using the 'fit score' described in the previous section using the following method:

1. A similarity matrix is calculated by using each straight line approximation as a seed, fitting it to each other frame and calculating the 'fit score' (see equation 1).

2. The matrix is made symmetric by multiplying by its transpose to give a single similarity metric between pairs of frames.
3. Each row is summed to give a similarity score for each frame. The frame with the lowest score (i.e. the frame most similar to all other frames) is selected as the first seed frame.
4. The frame that is most different from the frame selected in 3) (i.e. the frame with the highest 'fit score' in the first seed frame row of the similarity matrix) is selected as the next seed frame.
5. Successive seed frames are selected as the frame that is most different from any previously selected frame. This is done by taking the rows in the similarity matrix for all previously selected frames and finding the lowest element in each column. The frame corresponding to the highest of these values is chosen as the next seed frame.

The number of seed frames used may be selected directly or on goodness of fit criteria. Goodness of fit criteria such as putting a lower limit on the similarity value at stage 5) are useful but should be used with caution as criteria may change from one shape class to another.

Once the seed frames are selected they are fitted to the data and grouped as in section 4. After each iteration the groups are converted into vectors and averaged as previously but then converted back to points using the average perimeter as a scaling factor. The number of these points is reduced using the 'remove points' algorithm described in section 3 in which points which are close to a the straight line between the adjacent two lines are removed. This simplified average is converted back into vector form and used to seed the next iteration. This algorithm can lead to the shape being over simplified. A modification to the algorithm may be incorporated to solve this problem whereby the 'remove points' algorithm is no longer performed at the point where the seed ceases to fit the data to a satisfactory degree as defined by the 'fit score'. The unmodified version resulting from the previous iteration is then used as the seed for the next stage. From now on the 'remove points' algorithm is no longer applied and the iteration works exactly as in section 4.

This method produces aesthetically pleasing results as can be seen in the models produced (see Figure 5.1) but there is room for refinement in two areas. Firstly fitting each frame to each other frame is computationally expensive. If speed were an issue it is obvious to see how this stage could be refined to use fewer comparisons by identifying similar shapes from the first few frame fits. Other refinements to this method could be implemented in the dynamics of the iteration. Over-simplification was observed to be a problem and the solution provided, although reasonably robust, is fairly basic. There is much scope for subtle refinement of this iteration, however this has not been investigated as the presented method was observed to perform satisfactorily for the purposes intended.

6 Discussion

The methods described in the previous sections produce some very satisfactory results. Class extraction is obviously not a new concept and this has been explored by Heap and Hogg [7] among others but the difference here is that the classes are extracted without reference to a specific eigen domain. This removes the need for inappropriate point fitting algorithms that fit a pre-defined number of reference points to a shape regardless of the

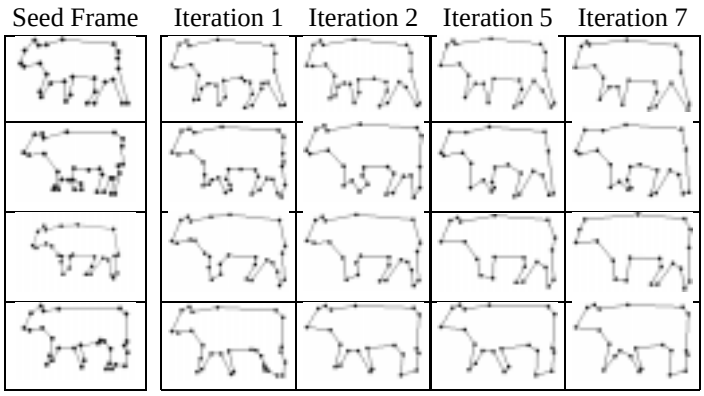

Figure 5.1: Mean Model Over Several Iterations of the Model Building Process

physical nature of that shape. In this method the number of points is extracted from the complete data set for a class of shapes rather than an individual shape. The vector representation of shapes used in the eigen analysis of the shape classes removes the need for complex algorithms to normalise shape scale and translation as used by Cootes *et al* [16] as the vector description is independent of scale or translation. No attempt is made to normalise rotation in this scheme as the subject of the study (cows) are not observed to rotate with respect to the ground plane and so fixing a camera with respect to the ground plane eliminates any rotational component in the input sequences. The models built by the methods described may be used for many purposes. The most obvious is classifying groups of shapes which may be anything from components in an assembly line to organs in a human body. Other applications include object tracking [2] and behaviour prediction [10].

References

[1] A.Hill and C.J. Taylor. Automatic landmark generation for point distribution models. In *Proc. British Machine Vision Conference*, volume 2, pages 429–438, 1994.

[2] A.M. Baumberg and D.C. Hogg. Learning flexible models from image sequences. In *European Conference on Computer Vision*, pages 299–308. Springer Verlag, 1994.

[3] A. Blake, R. Curwen, and A. Zisserman. A framework for spatiotemporal control in the tracking of visual contours. *International Journal of Computer Vision*, 11:127–145, 1993.

[4] C. Bregler and S. Omohundro. Surface learning with applications to lipreading. *Advances in neural information processing systems*, 6:43–50, 1994.

[5] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. Wiley-Interscience, 1973.

[6] K. Etemad and R. Chellappa. Discriminant analysis for recognition of human face images. In *First International Conference of the AVBPA*, pages 127–142, 1997.

[7] A. Heap and D.C. Hogg. Improving specificity in PDMs using a hierarchical approach. In *Proc. British Machine Vision Conference*, volume 1, pages 80–89, 1997.

[8] A. Hill, A.D. Brett, and C.J. Taylor. Automatic landmark identification using a new method of non-rigid correspondance. In *Proc. 15th International Conference on Information Processing in Medical Imaging*, pages 483–488, June 1997.

[9] P. Huang, C. Harris, and M. Nixon. Recognising humans by gait via parametric canonical space. In *ICSC Symposium on Engineering of Intelligent Systems*, 1998.

[10] N. Johnson, A. Galata, and D. Hogg. The Acquisition and Use of Interaction Behaviour Models. In *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 1998. To appear.

[11] M.Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. In *First International Conference on Computer Vision*, pages 259–268, 1989.

[12] P.J. Van Otterloo. *A Contour-Oriented Approach to Shape Analysis*. Prentice Hall, 1991.

[13] T.Y. Phillips and A. Rosenfeld. A method of curve partitioning using arc-chord distance. *Pattern Recogntion Letters*, 5:285–288, 1987.

[14] P.D. Sozou, T.F. Cootes, C.J. Taylor, and E.C. Di-Mauro. A non-linear generalisation of PDMs using polynomial regression. In *British Machine Vision Conference*, volume 2, pages 397–406. BMVA, September 1994.

[15] D. Terzopoulos and R. Szeliski. Tracking with Kalman snakes. *Active Vision*, pages 3–20, 1992.

[16] T.F.Cootes, C.J. Taylor, D.H. Cooper, and J.Graham. Training models of shape from sets of examples. In *Proc. British Machine Vision Conference*, pages 9–18, 1992.

[17] I. Tomek. Two algorithms for piecewise-linear continuous approximation of functions of one variable. *IEEE Transaction on Computers*, pages 445–448, April 1974.

[18] K. Wall and P.E. Danielsson. A fast sequential method for polygonal approximation of digitised curves. *Computer Vision, Graphics and Image Processing*, 28:220–227, April 1984.

[19] P. Zhu and P.M. Chirlian. On critical point detection of digital shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):737–748, 1998.