Using Constrained Snakes for Feature Spotting in Off-line Cursive Script

A.W.Senior F. Fallside
Cambridge University Engineering Department
Trumpington Street
Cambridge, CB2 1PZ. England

Abstract

Studies in the psychology of reading indicate that reading probably involves recognising features which are present in letters, such as loops, turns and straight strokes. If this is the case it is likely that recognising these features will be a useful technique for the machine recognition of cursive script. This paper describes a new method of detecting the presence of these features in a cursive handwritten word. The method uses constrained snakes which adapt to fit the maxima in the distance transform of a word image while retaining their basic shape. When the snake has settled into a potential minimum, its goodness-of-fit is used to determine whether a match has been found. The features located by this method are passed on to a 'neural' network recogniser. Examples of the features recognised are shown and results for word recognition for this method, on a single-author database of scanned data with 825 word vocabulary are presented. These are followed by a conclusion and pointers to further work.

1 Introduction

Examining works on the psychology of reading and studies into handwriting recognition leads us to see that people probably recognise certain features in printed or handwritten words, and it is largely from these features that the words are identified [6, 11]. These features might include vertical strokes, turns, crossbars, hooks and loops, though exactly which are used in human perception is unknown. If the human recognition system uses them, it is likely to be these features which are preserved in handwriting — since the intention is to convey information as clearly as possible. Thus, although there is much random and personal variation in handwriting, we could expect the features which are important to recognition to be well preserved in writing and thus to provide useful invariants for a machine to use for handwriting recognition.

Several researchers have written computer handwriting readers which recognise such features [1, 2], but the problem of identifying them off-line has not been satisfactorily solved. This paper describes an alternative process for finding these features in cursive script so that they can be passed on to a recogniser in the form of an error propagation network which has already been developed to recognise handwriting coded as line segment information [12]. This line segment information might be considered to correspond to the information available from Hubel and Wiesel cells in the visual cortex since it codes rough position and angle information for short line features in the image and this new method might be thought of as a higher-level, less local representation.

The method chosen for identifying the higher-level features in this paper is 'snakes' — the active contour models proposed by Kass, Witkin and Terzopoulos [8]. These are described in the next section, followed by a description of the Principal Component Analysis method used to constrain the snakes to match particular features and an introduction to the overall recognition system which is described in more detail elsewhere [12]. Some examples of the features recognised by this method are shown, and future developments of the method are discussed. The complete system is able to identify features of a variety of types, despite distortions and a controlled amount of translation. Each feature is defined by the presentation of a small number of training examples and avoids the need to write rules describing feature appearances.

2 Snakes

Snakes are deformable splines (smooth curves) placed in a potential field which translate and deform to reduce their potential energy. Traditionally they have been used to find edges in grey level images, by according low potentials to areas of high contrast so that the snake seeks to match its contours to high contrast edges. We now describe in more detail the working of the snake models.

The shapes of the snakes are governed by cubic B-splines [10] like those of Cipolla and Blake [3]. A series of control points $\{\mathbf{p}_i : i = 0, \ldots, N-1\}$ is in a two-dimensional plane and the actual spline path generated is an interpolation of these points, each point $\mathbf{x}(s)$, $s \in [0, N-1]$ on the path being a weighted sum of the nearest control points' positions. B(s) is a polynomial function which determines how much weight is given to each control point.

$$\mathbf{x}(s) = \sum_{i=0}^{N-1} \mathbf{p}_i B(s+2-i) \tag{1}$$

$$B(s) = \begin{cases} \frac{1}{6}s^3 & 0 \leq s \leq 1 \\ \frac{2}{3} - \frac{1}{2}(s-2)^3 - (s-2)^2 & 1 < s \leq 2 \\ \frac{2}{3} + \frac{1}{2}(s-2)^3 - (s-2)^2 & 2 < s \leq 3 \\ \frac{1}{6}(4-s)^3 & 3 < s \leq 4 \\ 0 & elsewhere. \end{cases} \tag{2}$$

The spline shown in figure 1 has the minimum four control points. For more complex shapes, more control points can be added, but each point on the curve is only determined by the four nearest control points. The weighting polynomials ensure continuity and smoothness (C^2). The B-spline is forced to terminate at the end control points by generating a 'phantom' control point $\mathbf{p}_{-1} = 2\mathbf{p}_0 - \mathbf{p}_1$, and similarly for $\mathbf{p}_N$.

Given the positions of the control points we can now place the snake on an image and must determine how it moves upon that image. Typically we will define a potential function $-f(x,y)$ on the pixels $\{(x,y)\}$ where we want to match the snake to curves of high values in f. f might be intensity I, contrast $\mid \nabla I \mid^2$ or, as in our case, the distance transform $D(x,y)$ where D is the distance of any pixel to the nearest background pixel in the image, zero if the pixel is itself part of the background. Here the city-block metric $D = \mid \Delta x \mid + \mid \Delta y \mid$ has been used for simplicity of computation.

The spline curves are sampled at points s_k and from each sample point, the normal to the curve is searched for the minimum of the potential function $-f$ within a certain distance on either side. The displacement of the minimum is recorded for each sampling point, and these displacements are then added to the control points to move the snake towards the local maxima. Since each sample point is a weighted sum of the nearest four control points:

$$x(s_k) = B(s_k+2)\mathbf{p}_i + B(s_k+1)\mathbf{p}_{i+1} + B(s_k)\mathbf{p}_{i+2} + B(s_k-1)\mathbf{p}_{i+3} \text{ when } 1 \leq s_k \leq 2. \tag{3}$$

the displacement $\mathbf{d}$(s) is distributed among these control points:

$$\mathbf{p}_i(t+1) = \mathbf{p}_i(t) + \frac{1}{M}\sum_k B(s_k+2-i)\mathbf{d}(s_k) \tag{4}$$

where there are M samples per unit in s. The new control points define a spline which lies closer to the lines of local maxima, and by a few iterations, a good match will be found if one is present in the search area around the snake's initial position.

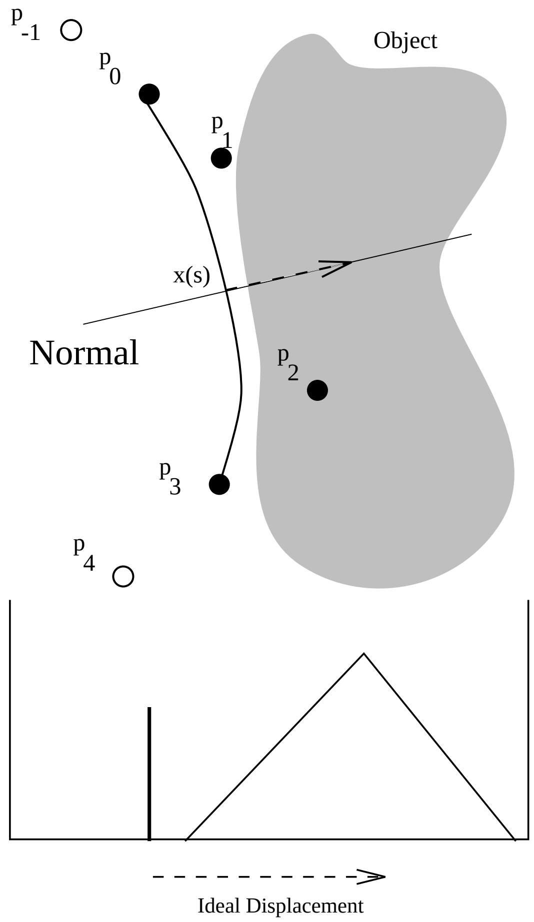


Figure 1: A snake with four control points and the distance transform along a normal.

As defined above, these snakes do not serve our purpose of feature recognition. They are very flexible, so any snake can adapt to fit a wide range of feature shapes, even collapsing to a point in some potential wells. To compensate for this, Kass, Witkin and Terzopoulos define an internal energy based on the integral of first and second derivatives along the snakes length, to penalise high curvature. Also, they are only able to find local features, whereas we need to find all occurrences of a feature in each word image. These two problems are tackled in the next two sections.

3 Point distribution models & constraints

Cootes & Taylor [4] describe 'Point Distribution Models' which they use as shape descriptors for various objects such as hearts in magnetic resonance images and resistors on images of circuit

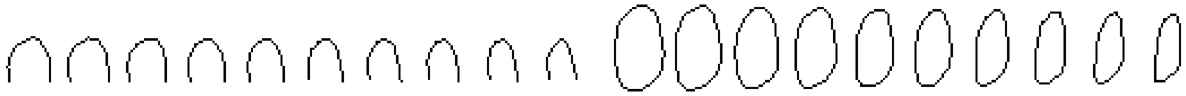


Figure 2: Snake models for *u* and *o* features showing the major mode of variation.

boards. A PDM is the covariance matrix of variation in the x and y coordinates of vertices of straight-line, snake-like models. In the examples they give, these points were placed by hand. If we place a fixed number n of points at important points (say inflections or points of high curvature) on a set of images of a particular object which we wish to model, after normalisation (subtracting the centroid and perhaps transforming to a fixed orientation and scale) we can calculate the covariance of the points in a $2n \times 2n$ matrix and accumulate these covariances across a series of images. With these statistics, we can perform Principal Component Analysis to determine the modes of variation in the system. This is carried out by diagonalisation of the covariance matrix. Each eigenvector shows a correlation in the variation of the point coordinates — a 'mode' of variation in which the points concerned have linearly related displacements. The eigenvalues give the extent of variation in the direction of the corresponding eigenvector, so the largest eigenvalue's eigenvector captures most of the variation in the model shape. These modes are strikingly demonstrated in Cootes, Taylor, Cooper & Graham's resistor model [5] where the first few modes correspond to natural physical parameters such as the position of the resistor on its wire, the bend of the wire, and the shape of the resistor body. Figure 2 shows the major modes of variation of two feature models.

Having determined these modes of variation, we can use them to constrain the variation of a snake. Having worked out the new position of a snake with no constraints, by the techniques of section 2, we can subtract the centroid of the model from the new control point coordinate vector. Transforming this difference into the coordinate frame of the principal components gives the deviation from the mean in each direction. Using the Mahalanobis distance $d^2(\mathbf{v}) = \sum_i \frac{v_i^2}{\lambda_i}$ we can work out how much the snake deviates from the model. This distance scales down variation along the principal axes, giving a measure of how many standard deviations the snake lies from the mean, assuming that deviations of snakes from the mean are distributed as a Gaussian ellipsoid. If the distance is too great, we can reduce it by scaling down all components, and discarding components in the directions of little variance. We then transform back to the original coordinates, add the centroid on again and have a new snake which is constrained to have a shape similar to those observed in the training set.

Work by Lanitis [9] has investigated the use of these models for whole character recognition for postcode reading. Here a model is produced for each of 36 alphanumeric characters and these models are matched to presegmented images of handwritten characters from a postcode database. Each model is compared with each image, and the best match is chosen. Lanitis achieved a 80.7% recognition rate on a database of postcode characters.

4 Combining snakes and PDMs for feature identification

In this work the ideas of splines and principal component analysis in the form of point distribution models have been linked together to form constrained B-spline models of features of handwritten letters.

We construct a model for each feature which we wish to recognise. In initial studies these features have been: '*n*' hump; '*u*' trough, which also models ligatures; '*i*' stroke (as found in '*u*' and '*n*'); '*t*' cross-stroke; ascender; descender and '*o*' shape. Each of these features can be modelled by a single spline, though other models such as '*x*' may be constructed from more than one. The model contains the mean position of each of the spline control points, and the permitted relative variations in these point positions. The preprocessor determines character size, so the coordinates are normalised to be independent of the size of writing.

Initially a seed model is generated by hand to describe the general characteristics of the feature:

- the number of points needed to model the feature;
- its topology (loop or line) and the interconnection of the points (whether they form an 'x', whether a loop has a tail or not);
- the position of the feature in a character — whether the feature is in an ascender or descender or the middle section of lower case letters.

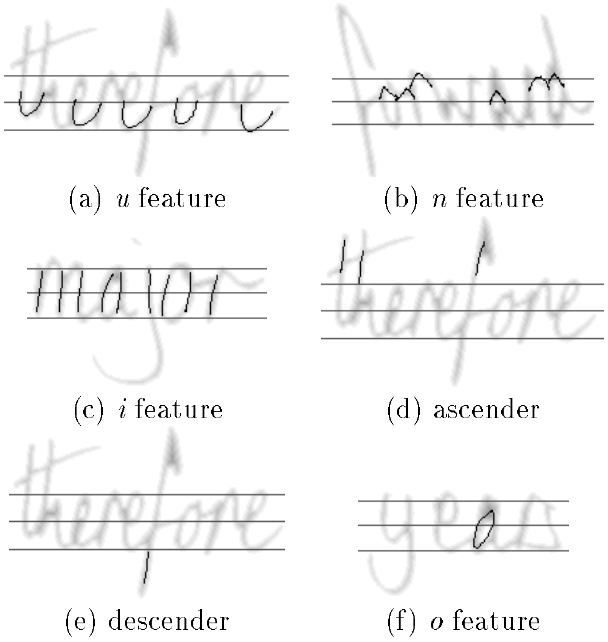
With a seed model for each of the features, we now match these models to instances of the features in images of handwritten words. Initially this can be by pointing out feature instances manually, and allowing the seed model to deform from the mean to match the stroke. When a good model has been found, this procedure can be automated so that the features in a word are found automatically. The automatic feature spotting is used both to train the models and subsequently to spot the features used in the recogniser.

In any word, a snake, whose shape is initially the mean shape for the model, is placed at the left edge of the word, and permitted to deform to match the distance transform potential, but with any deformation being constrained by the methods of section 3 to lie within κ standard deviations of the mean shape — so the shape will always be similar to shapes already taken by that feature before. (For κ, a value of 1 has been used here.) A best match given the constraints is found by iterating for a limited number of times or until the snake ceases to move. Then the degree of match between the snake and the image is determined. Should the snake move above or below the band where it is normally found (for instance '*t*' stroke feature matching the top of an '*r*') then it is rejected automatically.

The degree of match is defined as the sum of three components — the amount of deformation measured as before and the sum of the distance transform along the snake's length (sampled at a fixed number of points) plus an extra weight for all points which lie on the image (have a non-zero distance transform). The latter weights are divided by the curve length, and matches greater than a threshold are accepted as valid matches. (Values for the extra weight and threshold were empirically determined and are 7 and 10 respectively. The difference is dependent on the values of the distance transform and hence on the thickness of strokes and scan resolution.) This is in contrast to the measure of fit used by Lanitis, who adds two components — one the amount of data modelled by the snake and a penalty for the amount of data which the snake fails to model. This is to prevent, for example, an '*L*' model being matched to a '*B*'. If the unmodelled data were not taken into account, the '*L*' model might appear to match the '*B*' along its whole length. Since we are only trying to match a small part of each image at a time, such a measure would be inappropriate here.

When training, the snake corresponding to any valid match is allowed to deform without constraint so that it conforms exactly to the image shape, and then the point positions are normalised and the covariances are added into that feature's model. After each match, the snake is re-initialised to the mean and displaced to the right where the procedure is repeated until the whole word has been searched for that feature. In this way each feature is matched across the whole of each word in the training set. Figure 3 shows all the matches for the features used in a variety of words.

Having built up a library of feature models,



(a) *u* feature (b) *n* feature

(c) *i* feature (d) ascender

(e) descender (f) *o* feature

Figure 3: Different features found automatically in several words.

with their permitted variations, any word can be processed to mark the features by the above procedure. The preprocessing system already used for our handwriting recogniser (described in [13], see figure 4) is based on the position of line segments in the skeletonised word image, so tends to underplay larger and more complex features such as loops or strokes which seem to be significant in reading. The extra information determined by the snake method is incorporated by flagging the presence of the particular feature in the frame[1] in which it is found, providing an additional set of parameters at the input to the recurrent network in figure 4.

The recognition system used for this work is a recurrent network which estimates character probabilities for each of the input frames and a Viterbi decoder which integrates these probabilities across frames to determine the maximum likelihood word. This has already proved successful on a small vocabulary task [12, 13], but it is with the aim of improving performance on a larger vocabulary and with more than one writer that this feature-spotting technique has been added.

All the data is part of a single-author cursive script database collected by the authors. The database consists of 2000 word-images from an 825 word vocabulary, divided into training, vali-

[1] A frame is a vertical section through a word and is used to generate the inputs to the recogniser for one time-step.

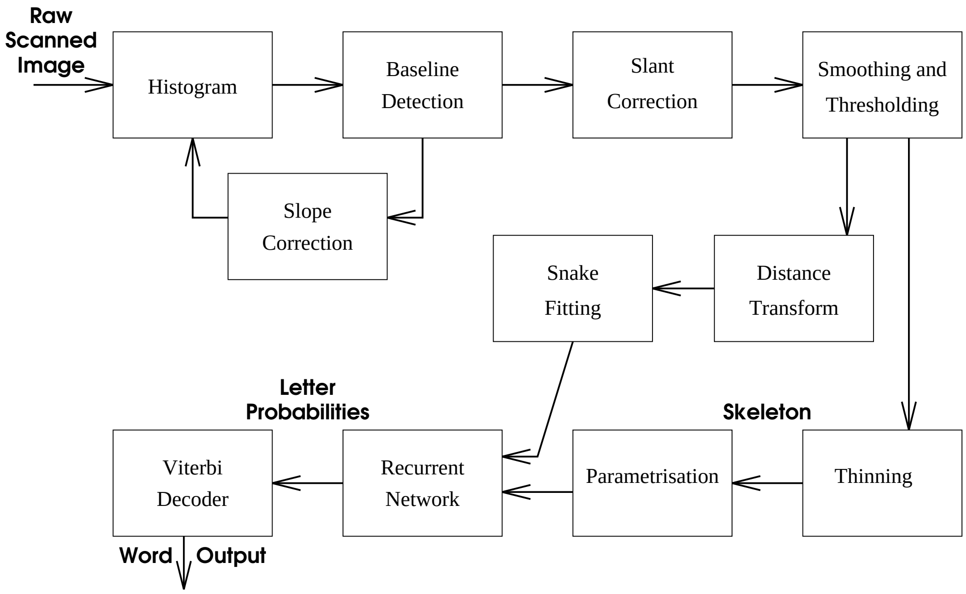


Figure 4: Schematic of the recognition system described in [12] showing the snake fitting modules.

dation and test sets. The original text is from the LOB [7] corpus of texts from newspapers. The images are scanned at 300dpi in 8 bits.

5 Results

The recogniser has been tested with the original preprocessing scheme, containing just skeleton information, and then again with the additional, snake-based feature spotting. This was found to reduce the errors by ten percent, giving a final accuracy of 78.7% words correct.

6 Conclusion

This paper has described work carried out on the application of constrained snake models to feature extraction for handwriting recognition, and has shown that snakes can be used to locate features which are useful to a recognition system identifying words. Most occurrences of the features are found, with few false alarms and the need for writing complex rules which may be scale dependent have been avoided. The feature spotting can be seen from the results to give improved recognition with an existing recogniser. Further work will investigate the use of whole-character models, both as part of the existing recogniser and as a separate technique, and apply the techniques discussed here to a multi-author database.

Acknowledgements

The authors would like to thank Roberto Cipolla, Mark Wright and Tony Robinson for their assistance in the research described in this paper.

References

[1] R.M. Božinović and S.N. Srihari. Off-line cursive word recognition. *IEEE PAMI*, 11(1):68–83, January 1989.

[2] M.K. Brown. *Cursive Word Script Recognition*. PhD thesis, University of Michigan, 1981.

[3] Roberto Cipolla and Andrew Blake. Surface shape from the deformation of apparent contours. *International Journal of Computer Vision*, 9(2):83–112, 1992.

[4] T.F. Cootes and C.J. Taylor. Active shape models — ‘smart snakes’. In Hogg and Boyle [5].

[5] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Training models of shape from sets of examples. In David Hogg and Roger Boyle, editors, *Proceedings of the British Machine Vision Conference*. Springer-Verlag, September 1992.

[6] Ralph Norman Haber and Lyn R. Haber. Visual components of the reading process. *Visible Language*, XV(2):147–182, 1981.

[7] S. Johansson, E. Atwell, R. Garside, and G. Leech. The tagged LOB corpus. Technical report, Norwegian Computing Centre for The Humanities, Bergen, 1986.

[8] Michael Kass, Andrew Witkin, and Demitri Terzopoulos. Snakes: Active contour models. In *Proc. 1st Inter. Conf. Computer Vision*, pages 259–268. IEEE, June 1987.

[9] Andreas Lanitis. Applications of point distribution models in handwritten optical character recognition and face recognition. Transfer report, Dept. of Medical Biophysics, University of Manchester, 1992.

[10] T. Pavlidis. Application of splines to shape description. In Arcelli, Cordella, and Sanitti di Baja, editors, *Visual Form*, pages 431–441. Plenum, 1992.

[11] Keith Rayner and Alexander Pollatsek. *The Psychology of Reading*. Prentice-Hall, 1989.

[12] A.W. Senior. Off-line handwriting recognition: A review and experiments. Technical Report TR105, Cambridge University Engineering Department, UK, December 1992. Available by anonymous ftp from svr-ftp.eng.cam.ac.uk.

[13] A.W. Senior and F. Fallside. Off-line handwriting recognition by recurrent error propagation networks. In *International Workshop on Frontiers in Handwriting Recognition 93*, CEDAR, SUNY Buffalo., May 1993.