Human Shape Recognition from Snakes using Neural Networks

Ken Tabb, Stella George, Rod Adams, Neil Davey
e-mail: *K.J.Tabb@herts.ac.uk, S.George@searchspace.com, R.G.Adams@herts.ac.uk, N.Davey@herts.ac.uk*

Department of Computer Science, Faculty of Engineering & Information Sciences,
University of Hertfordshire, England AL10 9AB

Abstract

This paper documents experiments which have been carried out with several neural network systems designed to categorise pedestrian shapes from non-pedestrian shapes. Active Contour models ('Snakes') [1] have been used to obtain contours of pedestrians as they move around the visual field. Neural networks have then been trained on representations of these relaxed snakes. The neural network systems developed can successfully discriminate these contours based upon whether they are 'pedestrian' in shape or not. Results are presented along with a discussion of some of the system's possible applications.

Keywords: Snake, Pedestrian, Neural Network, Shape Classification.

1.0 Introduction

This paper outlines a mechanism for detecting and categorising objects in images. A method that combines the use of snakes [1] and a neural network for categorisation is used. The method discussed is part of a larger system designed to track moving pedestrians, a problem that has been the subject of much research [2, 3, 4]. Clearly the first task of such a system is to identify human shapes under a variety of conditions, such as if the camera perspective on that human is changing, or if the human is being partially or completely occluded by other objects (static, moving or deformable) in the visual field. The method we propose is shown to be capable of representing, and then correctly classifying, over 90% of unseen shapes as either human or non-human.

The paper is divided into 5 sections. Section 2 discusses the use of Active Contour models for detecting pedestrians, and focusses particularly on Fast Snakes [5]. A description of the footage used by the snakes to abstract human shapes is also given. Section 3 introduces a method to represent a contour for use in neural networks. Experiments with different neural network architectures to categorise contours as being 'pedestrian' or 'non-pedestrian' are documented in section 4, along with a discussion of the results of these experiments. Finally section 5 summarizes the main issues raised in this work.

2.0 Identifying Pedestrians with Fast Snakes

Over a decade ago a computer vision technique called Active Contour models, or more commonly 'snakes', was developed [1] to detect and track target objects in an image or series of images. A wide variety of objects can be detected in images by adapting the energy function being minimized [6]. Since their introduction, countless improvements have been made to the original model [5, 7, 8] to make it more computationally efficient.

Whilst many of these improvements have succeeded in making the model more suitable for particular tasks, one of the model's key weaknesses still remains; snakes cannot categorise shapes, so have no 'knowledge' of the object they are detecting [9]. This makes the snake technique less suited for tracking target objects in complex environments, where it is often unknown what other objects may enter and disrupt the visual field. Moreover, without a mechanism of identifying what is being tracked in the image, the technique has limited appeal to artificial intelligence applications.

2.1 Fast Snakes

The Active Contours used in this research were Fast Snakes [5], as they offer a number of advantages for object tracking. A summary of these features follows, whilst a more exhaustive list can be found in [9]. Fast snakes do not require the user to provide corrective guidance to the snake; indeed no external energy is used at all. Fast Snakes space their control points equally along the snake, without explicitly contracting or expanding the snake. The original model's 'shrink-wrapping' [5] effect can continue to shrink a snake even when it is already situated on the target contour, pulling it back off the target object. Fast Snakes allow corners to form at certain points

by providing 'personalised' energy functions for control points. In the original model, the user-defined parameters are identical for all control points, resulting in a tendency to cut the corners off the target contour, as they are aiming to achieve global smoothness. To move a control point, Fast Snakes determine whether the control point would have lower energy if it were located elsewhere within its neighbourhood. By considering every location in the immediate neighbourhood, rather than jumping the control point from one location in the image to another, Fast Snakes overcome the original model's limitation in so much as strong local edges cannot be overlooked.

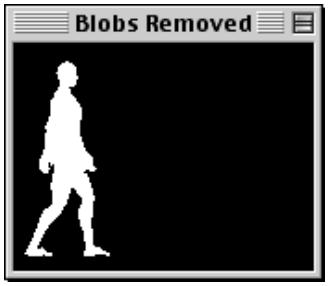

Figure 1: Simulated pedestrian footage generated from 3D modelling and animating software. Inverse kinematics ensures the model's movement is realistic.

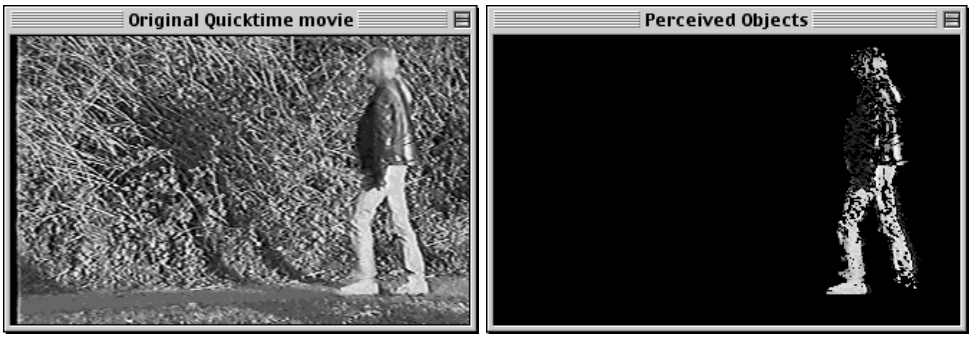

Figure 2: [Left] Real world pedestrian footage. [Right] The pedestrian is extracted using a combination of image preprocessing techniques, making the Active Contour's task easier.

2.2 Pedestrian Footage

The pedestrian footage used by the Active Contours consisted of greyscale movie files of pedestrians walking across the visual field. Pedestrian models simulated in a 3D modelling package were chosen over and above real world footage, as more control over pedestrian-specific parameters was needed, for example their height, weight and direction of movement; this variety was needed to supply the neural networks with a wide range of pedestrians. Inverse kinematics ensured that the simulated pedestrian moves realistically around the visual field. Using the 3D software, movies containing only one pedestrian and no other misleading edges were generated (Figure 1), making the task of detecting the pedestrian easier for the snakes. Real footage could have been used, but this would require additional image preprocessing to aid the Active Contour model (Figure 2), and was unnecessary for the purpose of generating training data for the neural network.

3.0 Representing Relaxed Contours for use in Neural Network Systems

An active contour is stored as a vector of (x,y) coordinates, each (x,y) coordinate reflecting the position of a different control point on the contour's spline. For such a vector to be used as input for a neural network, the pairs of x and y values must be seen to be related; unless the vector is reconstructed into coordinates, the list of numbers lose their meaning. Finding a representation which keeps the control point information together as coordinates is therefore fundamental to the success of the neural network.

3.1 Axis Crossover Representation

The contour representation used in this project was the axis crossover representation. This representation

assumes that the contour being represented is closed (i.e. that it forms a loop), which is a reasonable assumption to make for pedestrian contours. The contour's centrepoint is calculated, and from that centrepoint a number of axes are projected outwards at definable intervals. For example if four axes are required, they are projected at 0°, 90°, 180° and 270° from the contour's centre, where 0° is vertical (Figure 3a). In theory the intervals need not be equal, although in this project they are. The distance from where the axis crosses the contour's boundary to the contour's centrepoint is then stored in a vector. The vector (whose length equals the number of axes being projected) forms the contour representation. In cases where an axis crosses more than one part of the contour's boundary (Figure 3b), the distance to the most remote edge is stored; again this is not a necessary rule of the representation, but one which has been adopted for this project for consistency across vectors.

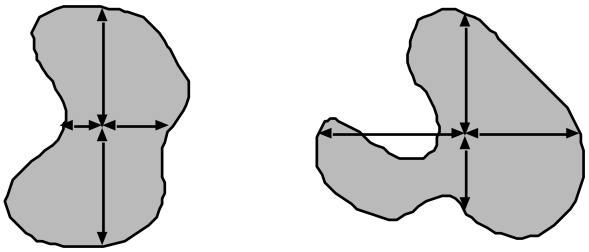

Figure 3: The axis crossover representation. 3a [Left]: The axis crossover being applied to a closed contour using four axes; the lengths of the four arrows form the contour's axis crossover representation. 3b [Right]: In situations where an axis crosses over the contour's edge more than once, the longest distance is used.

4.0 Experiments & Results with Neural Network Categorisation Tasks

It was necessary to test whether or not a neural network could distinguish one group of crossover vectors (pedestrians) from other groups of crossover vectors (non-pedestrians). The vectors were tested with simple hidden layer backpropagation networks as the task, at this stage at least, was a categorisation of the vectors.

The axis crossover representation allows for different numbers of axes to be used in the contour representation. Having fewer axes simplifies the neural network's task, however enough axes need to be used that the pedestrian qualities of the contours are encapsulated in the vectors, so that they can be classified differently from the non-pedestrian vectors. It was decided to test several different numbers of axes used in the representations, which in turn meant testing several neural networks, each with as many input units as there were axes in the representations. It was hoped that these experiments would identify the optimal number of axes to use in representing the particular class of contours relevant to this project. Initially neural networks with a single output unit were experimented on (section 4.1), as only two outputs were needed: 'pedestrian' and 'non-pedestrian'. Following these experiments, double output unit networks were used (sections 4.2 and 4.3) so that its categorisation confidence values could be analysed (section 4.4). In all experiments, network output was allowed some lenience; an output of 0 - 0.2 was classed '0', and an output of 0.8 - 1 was classed '1'.

4.1 Single output unit network experiments

The networks were trained with a range of different hidden layers, to allow the network with the optimal generalisation skills to be identified. The training set contained 150 pedestrian vectors and 150 non-pedestrian vectors. Training was stopped when the network had reached 15% or less error. It was then tested with 10 unseen pedestrian vectors and 10 unseen non-pedestrian vectors. The non-pedestrian vectors were of indoor objects, for example lampshades and teapots.

Figure 4 (upper left diagram) shows the average results of the various single output unit networks when tested with the 20 unseen vectors (10 pedestrian and 10 non-pedestrian); each network had been trained and tested 10 times using different initial weight matrices. Of note is that the network which used the most complex contour representation being tested (24-axis) failed to learn the task adequately, irrespective of the size of its hidden layer. It can be seen from the graph that networks trained on 16-axis vectors were most successful at classification.

4.2 Double output unit network experiments with indoor non-pedestrian data

The same experiments, training and test data described in section 4.1 were repeated with a double output unit architecture.

The results can be seen in Figure 4 (upper right diagram). Although all of the networks learnt the task, their general performance was worse than the single output unit networks. As with the single output unit results, networks trained with 16-axis vectors were most successful (although the optimal number of hidden units differed from the single unit networks).

4.3 Double output unit network experiments with outdoor non-pedestrian data

At this stage in the experimentation the axis crossover representation had been shown to be sufficiently descriptive of pedestrian objects. Nevertheless the project involves outdoor pedestrian scenes, so it was necessary to use more relevant non-pedestrian objects, such as cars, streetlights and traffic lights; the 150 indoor object vectors used in training were changed for 150 outdoor object vectors accordingly, as were the 10 unseen non-pedestrian vectors used in the test set. Apart from the different non-pedestrian data, the experiments were the same as for section 4.2.

The results of the double output unit networks can be seen in Figure 4 (lower left diagram). The networks are able to classify pedestrian vectors more accurately than non-pedestrian vectors. Interestingly the best scores were still obtained by those networks which had used 16 axes in their crossover representations.

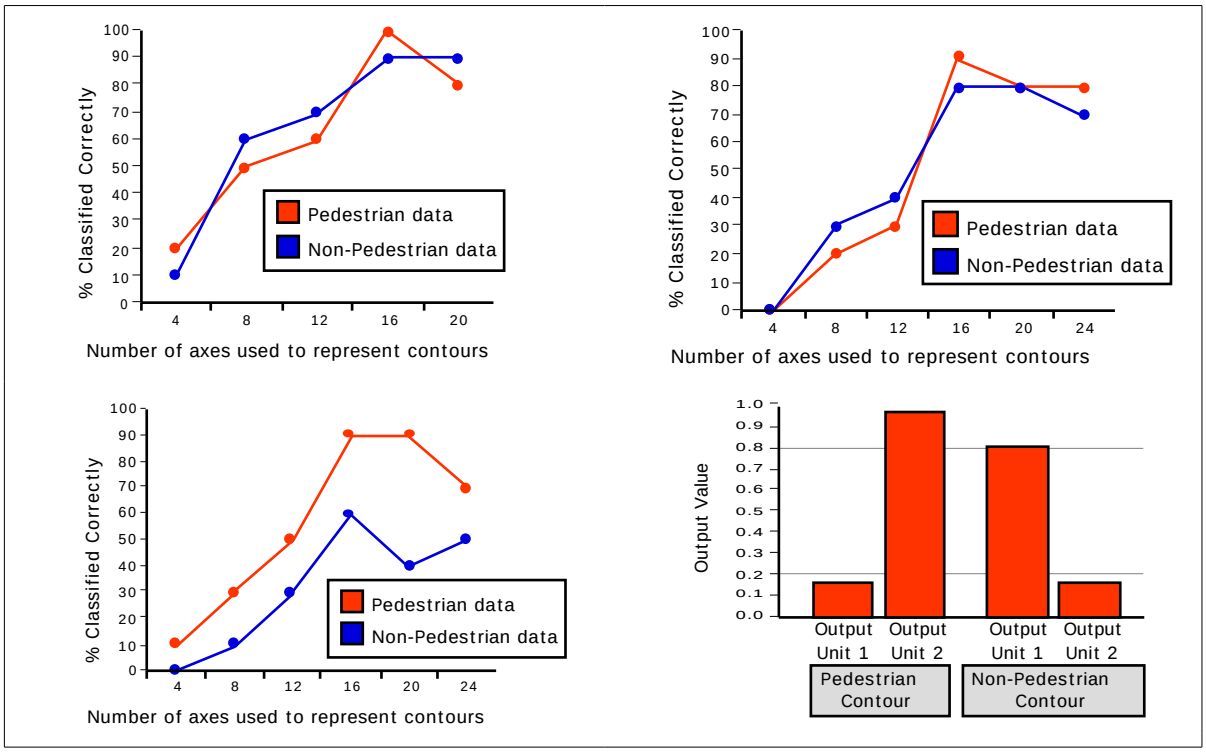

Figure 4: Test set classification results, averaged over 10 runs. The top row uses indoor objects as non-pedestrian data, whereas the bottom row uses outdoor objects. The top left graph shows results with a single output network, all the others are double output networks.

4.4 Analysis and Discussion

The 16-axis representation was adopted as a standard, as it had outperformed the other configurations of the representation, and was judged to be most able to encapsulate the pedestrian qualities of a given contour. The neural network with 16 input units was therefore used henceforth. It was decided to determine the confidence value from the two output units' values as this could be used as a threshold for identifying how 'pedestrian' an object is. To test the chosen network's confidence in its categorisations, it was necessary to look at the average difference between the values output by both of its output units during the experiments described in section 4.3 to see how 'sure' it was that a pedestrian vector was pedestrian and that a non-pedestrian vector was not.

Figure 4 (lower right diagram) shows the average results for the 10 pedestrian and 10 outdoor non-pedestrian vectors. The network's output units are split into a 'non-pedestrian' neuron (unit 1) and a 'pedestrian' neuron (unit 2) which should ideally fire mutually exclusively of one another, as something cannot be both pedestrian

and non-pedestrian. However, as was found in section 4.3, the network is not 100% accurate, resulting in some uncertainties about the answer it gives. Nevertheless a clear divide can be seen between the values output when classifying a pedestrian vector correctly versus classifying a non-pedestrian vector correctly. From the graph, the average confidence value (the difference between the two output units' values) when classifying a pedestrian vector correctly is 0.81, whereas the average confidence value for non-pedestrian vectors is 0.65.

When comparing the generalization results of the networks, it can be seen that pedestrian classification is significantly better than non-pedestrian, when the training/test sets use outdoor, non-pedestrian data objects. The network with two output units produces values that confirm it is more accurate at classifying pedestrian than non-pedestrian vectors. In particular the average value for the output units when given a non-pedestrian vector (0.82 and 0.17) only just fall within the 'classified correctly' zones of 0-0.2 and 0.8-1.0, marked in Figure 4 (lower right diagram).

The choosing of counter examples is clearly a deciding factor to the accuracy of the system; the networks were less accurate at categorising pedestrian vs. outdoor non-pedestrian objects than they were with indoor non-pedestrian objects.

5 Conclusion

A method is presented for categorising objects which are being detected in an image. Active Contour models using the Fast Snake algorithm have been used to detect and track pedestrians around the visual field, although any Active Contour model could in theory be used for this stage of the technique.

The pedestrian contours output from the Active Contour system have then been re-represented as axis crossover vectors to train neural networks. Again this part of the technique is flexible; any number of axes could be used to represent the contours, although this project settled on using 16 axes for pedestrian data. Whilst this project used regularly spaced axes for simplicity, the axes need not be at regular intervals; three axes at 37°, 92° and 246° could be used to represent contours, if this was important to the shapes being represented. The representation is independent of the number of control points on the contour, and therefore is independent of the complexity of the contour's shape. The representation can become scale invariant by normalising the vector. Where the vector length equals the number of axes being projected and not the number of segments or control points on the contour, the vector has the same length even with more complex contours, thus the quantity / frequency of control points on a contour can be experimented with while still using the same contour representation. Finally, the representation is location invariant; two identical shapes in different parts of the image will result in the same vector.

Feedforward neural networks, trained using back-propagation, have been developed which are able to discriminate between pedestrian and non-pedestrian shapes. Moreover they show surprisingly strong generalisation behaviour, to such an extent that over 90% of the unseen data was correctly classified. This suggests that the snake shape recognition system in collaboration with the axis crossover representation provides a powerful method for encoding pedestrian outlines.

By having two output units, the network's confidence value can be used to indicate how 'pedestrian' a contour is. This confidence value could have practical applications, for example it could be used to indicate when an object being tracked is becoming occluded.

References

[1] Kass M., Witkin A. & Terzopoulos D. (1988) *Snakes: active contour models.* In International Journal of Computer Vision (1988), pp.321-331.

[2] Sonka M., Hlavac V. & Boyle R. (1994) Image Processing, Analysis and Machine Vision. Chapman & Hall.

[3] Marr D. (1982) Vision. W.H. Freeman & Company.

[4] Baumberg A.M. & Hogg D.C. (1994). *An Efficient Method for Contour Tracking using Active Shape Models.* University of Leeds School of Computer Studies Research Report 94.11.

[5] Williams D.J. & Shah M. (1992) *A fast algorithm for active contours and curvature estimation.* In CVGIP - Image Understanding 55, pp.14-26.

[6] Blake A. & Isard M. (1998). Active Contours. Springer-Verlag.

[7] Cootes T.F. & Taylor C.J. (1992) *Active shape models - 'smart snakes'.* In British Machine Vision Conference Sept 1992, pp.276-285.

[8] Blake A. & Yuille A. (Eds) (1992). Active Vision. MIT Press.

[9] Tabb K. & George S. (1998). *Snakes and their influence on visual processing.* University of Hertfordshire Department of Computer Science Technical Report No 309 Feb 1998.