

University of Leeds
SCHOOL OF COMPUTER STUDIES
RESEARCH REPORT SERIES
Report 93.36

Learning Flexible Models from Image Sequences

by

A M Baumberg & D C Hogg
Division of Artificial Intelligence

October 1993

Abstract

The “Point Distribution Model”, derived by analysing the modes of variation of a set of training examples, can be a useful tool in machine vision. One of the drawbacks of this approach to date is that the training data is acquired with human intervention where fixed points must be selected by eye from example images. This is a laborious process and may lead to a non-representative set of training examples being used. A method is described for generating a similar flexible shape model automatically from real image data. A cubic B-spline is used as the shape vector for training the model. Large training sets are used to generate a robust model of the human profile. The resulting modes of variation show the potential of the model for labelling and tracking of pedestrians in real-world scenes. Furthermore, an extended model is described which incorporates the direction of motion of the human, allowing the extrapolation of direction from shape.

1 Introduction

We wish to generate a 2D flexible model of the shape of a non-rigid object such as a walking pedestrian. Flexible shape models have been shown to have applications in tracking and image interpretation (e.g. Kass *et al.* [1], Yuille *et al.* [2]). These tasks are made easier by restricting the solution space of allowed shape deformations. For instance, we can assume affine invariance to accommodate rigid motion of flat objects under weak perspective (Blake *et al.* [3]). Another approach to restricting the solution space is to incorporate some knowledge of the object in question.

Cootes *et al.* [4] describe a point distribution model in which a set of labelled points is hand generated from a set of training images of a particular object in a variety of positions. The shapes are aligned and the deviations from the mean are analysed using principal component analysis. The most significant modes of variation give a compact representation of the generic object shape (based on the training set) whilst the other modes are ignored as they contribute little to the overall shape.

A natural extension of this avenue of research is to generate the model automatically. The problem is to extract a reasonably consistent shape vector from real images containing examples of the object. By processing large amounts of data we can reduce the effects of noise, occlusion and mis-segmentation and thus use a relatively simple segmentation scheme. The processing of image data needs to be sufficiently fast to extract large training sets in a reasonable time. In fact the system has been implemented to run in near real time (processing over 4 image frames per second).

Live video input is used to eliminate the problem of playback quality.

The control points of a B-spline are used as a shape vector, as a spline is convenient for data approximation and fast to render. Moreover, B-splines have successfully been used for tracking image contours (e.g. Blake *et al.* [3]).

In the area of understanding images of humans, work has been done by Hogg [5] and Rohr [6] where an explicit 3D model of the human body is used. Pentland and Horowitz [7] describe a physically based, 3D, hand generated model with spring-like constraints between body parts. These approaches utilise complex, hand crafted 3D models. Other approaches assume the joints of the human body are marked (e.g. Chen and Lee [8]). Murphy *et al.* [9] describe a task-based approach involving eigenimage decomposition of long image sequences of a walking human taken from a fixed viewpoint.

The model described here is essentially 2D but is trained on a selection of arbitrary views. The variation in shape due to different viewpoints is treated as flexibility in 2D shape allowing the model to be used for tracking over the range of viewpoints for which it was trained.

2 Outline of the method

Our system takes live video images from a static camera, processes them and extracts shape-vectors corresponding to the moving objects in the scene. In our experiments the majority of the moving objects are walking pedestrians. The data is then analysed off-line to generate the model. A diagram outlining this system is shown in figure 1.

There are four main stages:-

- *Image preprocessing* to obtain a binary background-foreground image (section 3).
- *Outline extraction* to obtain an ordered set of boundary points for each foreground (moving) shape (sections 4 & 5).
- *Shape vector calculation* to obtain an item of training data (section 6).
- *Off-line analysis* to build the shape model (section 7).

3 Preprocessing stage

The system segments moving objects from a sequence of images by simple differencing from a background image and thresholding. The background image is continually

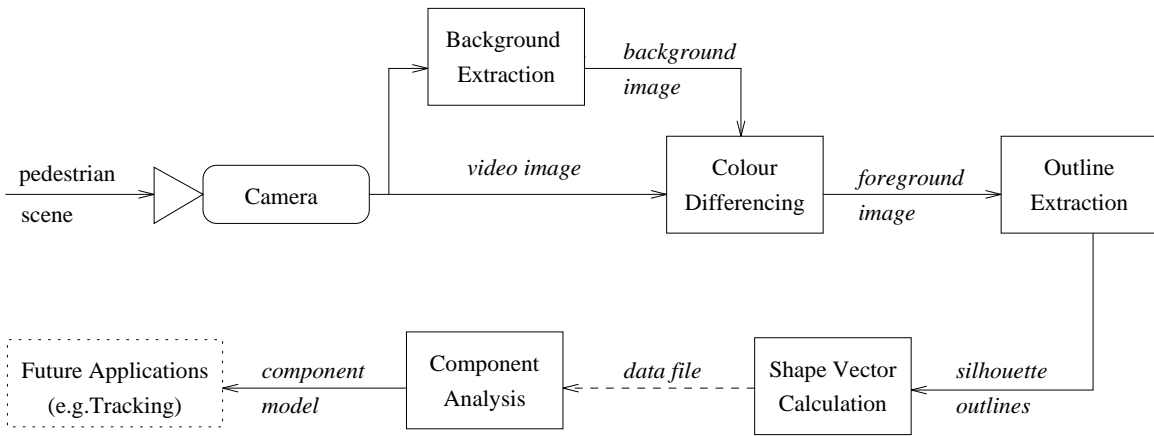


Figure 1: overview of the system

updated (median filtering over time) to account for changing lighting conditions. We use an approximation to the median filter as described by Hyde and Worrall [10].

The background image is subtracted from each input image. The resulting image is blurred (to reduce noise) and then thresholded to produce a binary (background-foreground) image. A conservative threshold is used (i.e. assigning more pixels as foreground) to ensure the foreground objects are well defined connected regions although this increases the effects of noise. These regions correspond to moving objects in the scene.

In order to reduce noise further, colour images are used and the differenced colour images are combined, blurred and thresholded to produce a more robust segmentation.

The results of this preprocessing (illustrated in figure 2) are adequate for our purposes.

4 Extracting the silhouettes

We are interested in the shapes of the foreground (moving) objects in the image. If the object is too small, the detail in the shape is lost. Hence all the connected regions consisting of fewer than a specified number of pixels are removed. Note that this also removes any “spikes” from the image.

The remaining regions correspond to the large moving objects in the scene. (In our experiments, the objects are usually people.) It is assumed that a reasonable proportion of these objects are not occluded and do not overlap. The regions are traced to produce a chain of boundary points. The list of points is smoothed (using Gaussian smoothing) and used as the basis for the calculation of the shape vector.

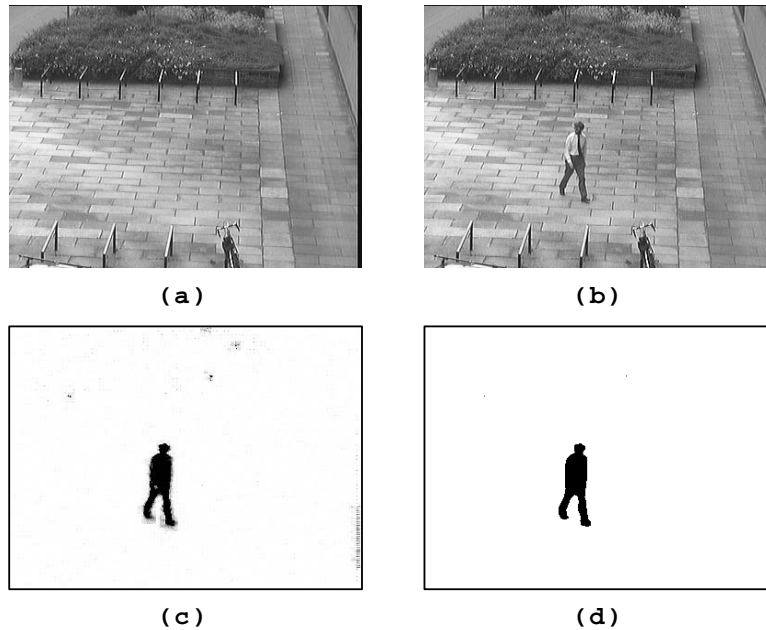


Figure 2: Image Preprocessing: (a) background image, (b) video input image, (c) differenced image, (d) blurred and thresholded image.

5 Finding a point of reference on the boundary

In order to generate a spline, we need to choose an initial starting point on the closed boundary (which will have the associated parameter value $u = 0$). A consistent method is required which is not highly susceptible to noise.

The method used is to find the principal axis (i.e. the axis through the centroid of the boundary points which minimises the sum of the perpendicular distances to that axis). We then select the lower (in terms of image coordinates) of the two points where the axis crosses the boundary. It is assumed that this point will be fixed for humans in the scene. This is reasonable for the scenes we are dealing with where the people always appear in an upright position. A more general method may select the intersection point that is nearest to the centroid (or some other suitable choice).

The boundary points are now reordered so that the first point is the reference point. Each shape is reflected about its principal axis to double the volume of training data (as has been done by Hill, Thornham and Taylor [11]).

6 Calculating a shape vector

The control points of a length-wise uniformly spaced B-spline are used as a shape vector. Previous steps extract from each moving shape a smoothed, ordered set of n

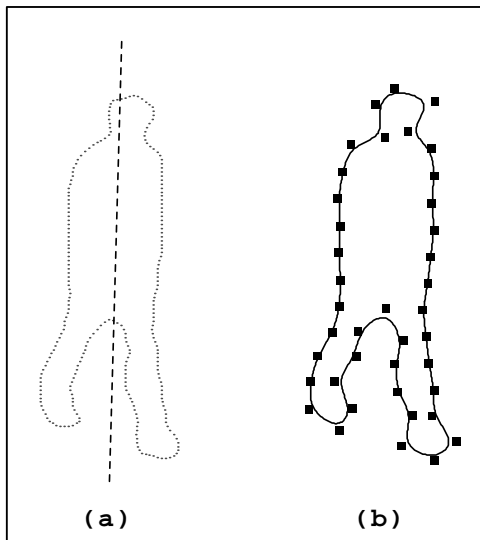


Figure 3: Extracting a spline: (a) data points with principal axis, (b) resulting spline.

boundary points $\mathbf{W}_i = (X_i, Y_i)$, with $0 \leq i < n$ which are to be approximated with a (closed) spline $\mathbf{P}(u) = (P_x(u), P_y(u))$ with N control points $\mathbf{Q}_k = (R_k, S_k)$. We define $\mathbf{P}(u)$ as follows:

$$\mathbf{P}(u) = \sum_{k=0}^{N-1} \mathbf{Q}_k B_k(u)$$

where the B_k are modified B-spline basis functions¹ and $\mathbf{P}(0) = \mathbf{P}(N)$. The required approximating spline minimises the error function

$$\text{Error} = \sum_{i=0}^{n-1} (P_x(u_i) - X_i)^2 + (P_y(u_i) - Y_i)^2$$

where u_k is some parameter value associated with the k th data point.

Using standard methods (see for example Bartels, Beatty and Barsky [12]) the following N equations are obtained:

$$\sum_{k=0}^{N-1} M_{i,k} R_k = \sum_{j=0}^{n-1} B_i(u_j) X_j \quad (1)$$

where $0 \leq i < N$ and $M_{r,s} = \sum_{k=0}^{n-1} B_r(u_k) B_s(u_k)$ and an analogous set of equations are obtained for S_k . For a reasonably close approximation of the boundary, the

¹As the curves are closed we choose the basis functions such that $u = 0$ is equivalent to $u = N$ as follows:

$$B_k(u) = \begin{cases} \mathcal{B}(u - k) & (u - k) \geq 0 \\ \mathcal{B}(u + N - k) & (u - k) < 0 \end{cases}$$

where $\mathcal{B}(u)$ is the standard B-spline basis function which is non-zero on the interval $0 < u < 4$.

parameter values can be set as follows:

$$u_k = \begin{cases} 0 & \text{for } k = 0 \\ \lambda \sum_{i=1}^k |\mathbf{W}_i - \mathbf{W}_{(i-1)}| & \text{for } k > 0 \end{cases} \quad (2)$$

where $\mathbf{W}_n \equiv \mathbf{W}_0$ and λ is chosen such that $u_n = N$.

To calculate the spline control points, \mathbf{Q}_k , the matrix $M_{i,j}$ must be inverted for each shape. However in order to avoid this computationally expensive step, we calculate $n' = wN$ new data points (where w is a whole number), which correspond to the *fixed* uniformly spaced parameter values:

$$u_k \equiv k \frac{N}{n'}$$

For details, see the appendix.

Using these new data points and their associated parameter values, we find that $M_{i,j}$ is fixed and need only be inverted once. This efficiently produces a uniform B-spline with the control points placed at approximately uniformly spaced intervals along the spline (hence “length-wise uniformly spaced”). Moreover the method is fast and robust.

The control points of the spline make up the shape vector \mathbf{x} where

$$\mathbf{x} = (R_0, S_0, R_1, S_1, \dots, R_N, S_N)$$

7 Analysing the shape vectors

7.1 Component analysis

The spline control points are treated in exactly the same way as the points in the “Point distribution model” of Cootes [4]. The shape vectors are aligned to the mean, $\bar{\mathbf{x}}$, and the differences from the mean are analysed using principal component analysis. Hence for each aligned shape the vector \mathbf{dx} is calculated as follows:

$$\mathbf{dx} = \mathbf{x} - \bar{\mathbf{x}}$$

The $2N \times 2N$ covariance matrix C is then calculated using

$$C_{i,j} = E(\mathbf{dx}_i \mathbf{dx}_j)$$

where $E(\dots)$ is the expectation or mean value. The eigenvectors of the covariance matrix correspond to modes of variation of the training data. Moreover, the eigenvector corresponding to the largest eigenvalue describes the most significant mode of variation.

The resulting model consists of the mean shape, $\bar{\mathbf{x}}$, and a subset of t eigenvectors $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_t$ (of unit length) corresponding to the t most significant modes of variation in the training data.

7.2 Experimental Results

In our experiments we used over 700 training shapes extracted from 15 mins of live video of a quiet pedestrian scene containing some moving vehicles. We used 40 control points (and hence 80 parameters) and found that the first 18 modes accounted for 90% of the variance of the training data. The first $t = 18$ eigenvectors can thus be used as an orthonormal basis for the model subspace of allowable shapes. Some of the significant modes of variation of the shape vector are shown in figures 4, 5 and 6.

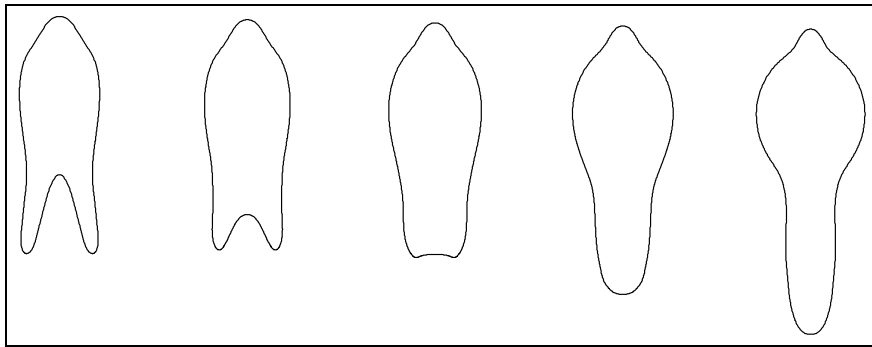


Figure 4: The effect of varying the component of the first mode by ± 1.5 standard deviations

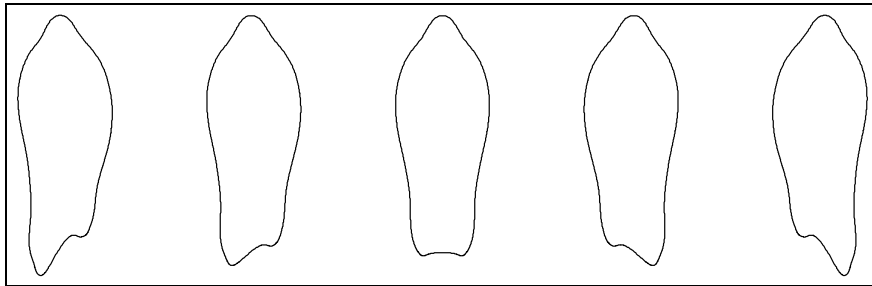


Figure 5: The effect of varying the component of the second mode by ± 1.5 standard deviations

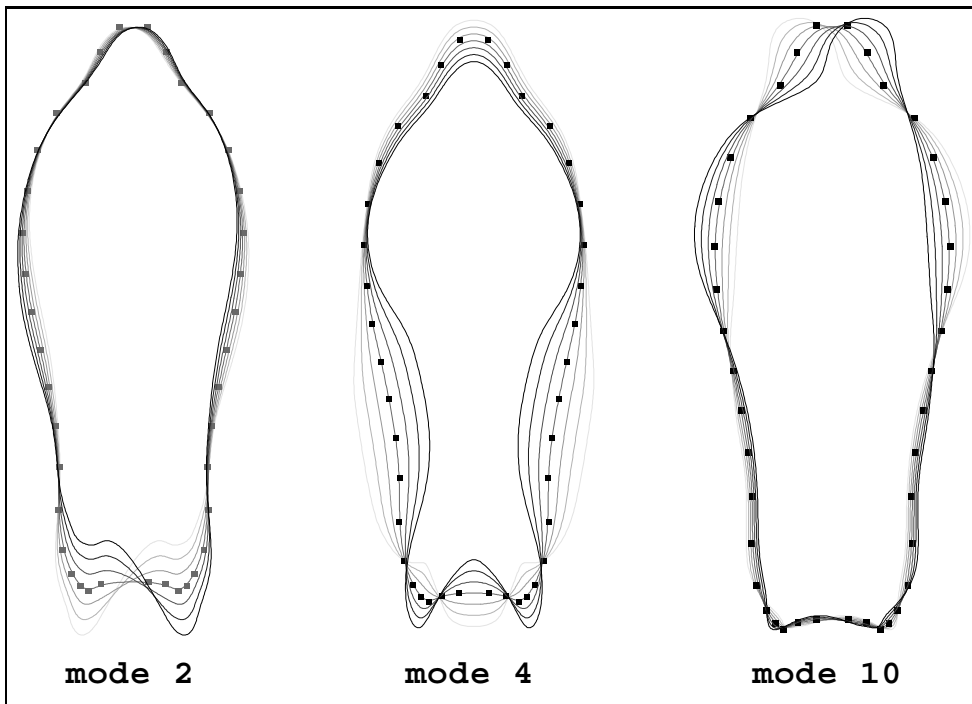


Figure 6: diagrams illustrating some of the modes of variation

8 Projecting into the model subspace

By projecting a noisy shape (obtained as before) into the model subspace (taking the first t modes), some of the effects of noise can be removed. Given a shape vector \mathbf{x} we project to a set of t weights μ_i using

$$\mu_i = (\mathbf{p}_i) \cdot (\mathbf{x} - \bar{\mathbf{x}}) \quad (3)$$

The weights can then be projected back using

$$\mathbf{x} = \bar{\mathbf{x}} + \sum_{i=0}^t \mu_i \mathbf{p}_i \quad (4)$$

to obtain the corresponding “component-filtered” spline. This process calculates the best (in the least squares sense) shape in the model space that matches the input. The results of this process are shown on some real data in figure 7.

9 Adding direction to the model

9.1 Creating an extended model

As the objects are moving, it may be useful to introduce their direction of motion into the model. In order to obtain some training data we must obtain a direction for

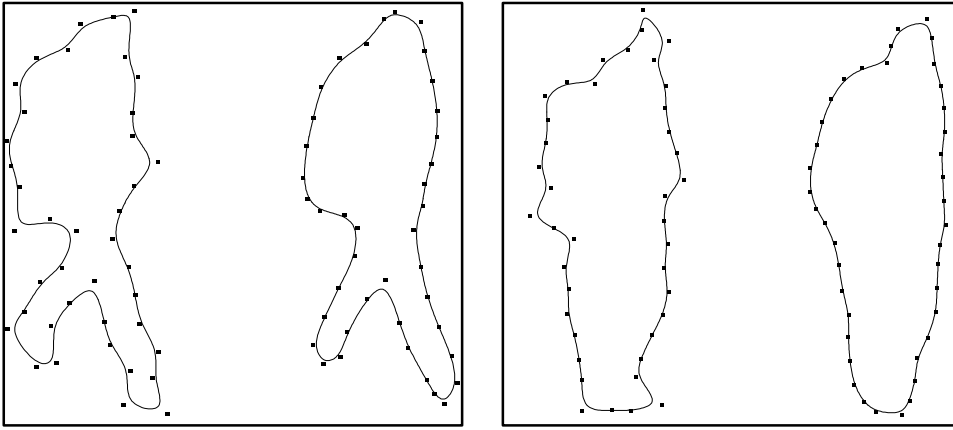


Figure 7: Projecting into the model space: In each case, the component filtered spline is shown to the right of the initial noisy input spline.

each region in the training images. This is done by crudely tracking the regions from one image to another (off-line) by considering similar sized regions in close proximity to be the same object. A simple similarity measure based on that used by Li-Qun *et al.* [13] is used.

For each spline \mathbf{Q}_i , we associate a direction-vector $\mathbf{d} = (d_x, d_y)$ which is set to the displacement of the object from one frame to the next. The direction is normalised as follows:

$$(d'_x, d'_y) = \begin{cases} \frac{1}{d} (d_x, d_y) & d > 0 \\ (0, 0) & \text{otherwise} \end{cases}$$

where $d = |\mathbf{d}|$ and an extended shape vector \mathbf{x}' calculated using:

$$\mathbf{x}' = \{R_0, S_0, R_1, S_1, \dots, R_N, S_N, d'_x, d'_y\} \quad (5)$$

The shape vectors are analysed as before except the mean direction-vector is set to $(0, 0)$. Since the direction-vector is normalised we have introduced an extra redundant dimension to the shape space. Effectively we are adding an extra point to each shape to indicate the direction of travel. This treatment was found to give good results.

N.B. The direction we are dealing with is currently defined in terms of image coordinates.

9.2 Extrapolating direction from shape

Using this extended model we can infer the 2D direction of motion of a person merely from their shape. The object is initially assumed to be stationary. An extended shape vector is calculated using equation (5) and projected into the extended model

subspace as before, using equation (3). Projecting back using equation (4) we obtain a filtered extended shape vector which gives the direction of motion. In effect we are fitting (in the least squares sense) the best moving person to the stationary shape. Some results of this process are shown for real images (figure 8).

The resulting direction is only meaningful if the input shape is from a reasonable segmentation – large errors in the shape give rise to significant errors in the direction.

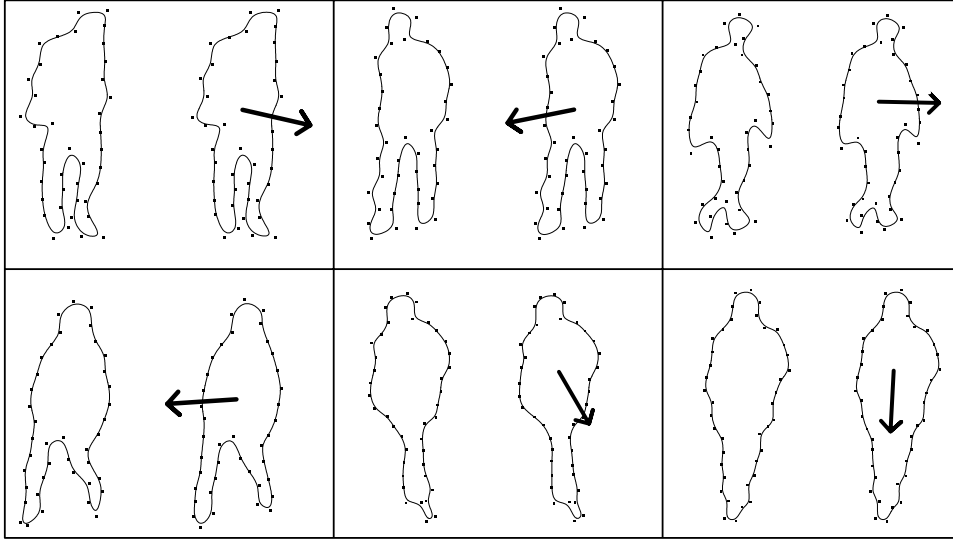


Figure 8: Extrapolating direction: In each case, the extrapolated moving shape is shown to the right of the input shape.

10 Future Research / Work in Progress

Currently work is being done to utilise the model to track one or more pedestrians in real time (using a “smart snake” as described by Cootes and Taylor [14]). The system can then be improved by using the tracker output to update and refine the model on-line. We also intend to incorporate estimates of the ground plane to allow us to calculate the direction of motion of the pedestrian relative to the camera (and hence the relative orientation of the pedestrian).

A large volume of noisy data is used to generate the model. It may be possible to improve the model by rejecting outliers although this will inevitably be computationally expensive. A new approach to this problem is outlined by Shapiro and Brady [15].

The model we have generated attaches a unique label to the shape of a human profile. This may provide a good basis for human posture detection and in the long term detection of motions such as running and walking.

Further work may investigate possible mappings from the spline model space to a high-level 3D model space such as the WALKER model of Hogg [5], where a human is modelled as a set of cylinders connected at articulated joints. The spline model may then be used for fast segmentation (and real time tracking) and to give a global estimate of object pose in the high-level model space (for further refinement or visualisation).

11 Conclusion

We have shown that by using a simple segmentation scheme to produce a large volume of noisy data we can build up a useful model of the human profile. This model can then be used to achieve a more accurate segmentation and hence a refined model.

By restricting the input domain to reasonable quality images from a fixed colour video source, a model has been generated which can then be applied to a less restricted problem domain.

We have demonstrated an efficient method for extracting a shape vector based on a cubic B-spline, from real image sequences. The system can process large amounts of data in near real time to generate a compact data set. Statistical component analysis of the spline data gives a simple but effective model.

Our model is “data-centred” in the sense that it is constructed from real image data. An advantage of this approach is that we can easily fit the model to new input. The model is still not a high level description of a human (e.g. there is no notion of limbs). However for many purposes this description may be unnecessary.

We have also shown how other variables such as direction can be incorporated into a point distribution model. This allows us to extrapolate direction of motion from shape (or alternatively if we know the direction we could constrain the shape appropriately). The results show the effectiveness of using principal component analysis on large noisy training sets.

Appendix

Selecting data points for spline approximation

We have a set of n boundary points $\mathbf{W}_i = (X_i, Y_i)$ to approximate with an N control point closed spline $\mathbf{P}(u)$. We wish to find n' new data points which correspond to fixed parameter values.

Conventionally the parameter values associated with data points \mathbf{W}_k are based on the Euclidean distances between points (see equation 2).

This leads to a set of values u_k corresponding to the data values X_k . The discrete mapping u_k to X_k can then be extended to a continuous mapping u to $X(u)$ by linear interpolation. Hence given $u_k \leq u \leq u_{k+1}$ we can calculate $X(u)$ using

$$X(u) = \left(\frac{u - u_k}{u_{k+1} - u_k} \right) X_{k+1} + \left(\frac{u_{k+1} - u}{u_{k+1} - u_k} \right) X_k$$

Similarly we can interpolate to find $Y(u)$. Hence given a chosen parametric value u we can find a corresponding new data point. Regularly spaced parametric values (between 0 and N) are chosen to find n' new data points. These new data points can now be efficiently approximated with a uniform cubic B-spline.

References

- [1] Kass M., Witkin A., and Terzopoulos D. Snakes: Active contour models. In *First International Conference on Computer Vision*, pages 259–268, 1987.
- [2] Yuille A.L., Cohen D.S., and Hallinan P. Feature extraction from faces using deformable templates. *Computer Vision and Pattern Recognition*, pages 104–109, 1989.
- [3] Blake A., Curwen R., and Zisserman A. A framework for spatio-temporal control in the tracking of visual contours. *International Journal of computer Vision*, 1993.
- [4] Cootes T.J., Taylor C.J., Cooper D.H., and Graham J. Training models of shape from sets of examples. In *British Machine Vision Conference*, pages 9–18, September 1992.
- [5] Hogg D. Model-based vision: A program to see a walking person. *Image and Vision Computing*, 1(1):5–20, 1983.
- [6] Rohr K. Incremental recognition of pedestrians from image sequences. *Computer Vision and Pattern Recognition*, pages 8–13, 1993.
- [7] Pentland A. and Horowitz B. Recovery of non-rigid motion and structure. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 13(7):730–742, July 1991.

- [8] Chen Z. and Lee H.J. Knowledge-guided visual perception of 3d human gait from a single image sequence. *IEEE Trans. on Systems, Man and Cyb.*, 22(2):336–342, 1992.
- [9] Murphy N., Byrne N., and O’Leary K. Long sequence analysis of human motion using eigenvector decomposition. In *Proc. SPIE*, September 1993.
- [10] Worrall A. and Hyde J. A fast algorithm for background generation. VIEWS Working Paper RU-03-WP-T.1.1.1.1-1.
- [11] Hill A., Thornham A., and Taylor C.J. Model-based interpretation of 3d medical images. In *British Machine Vision Conference*, volume 2, pages 339–349, 1993.
- [12] Bartels R., Beatty J., and Barsky B. *An Introduction to Splines for use in Computer Graphics and Geometric Modeling*. Morgan Kaufmann, 1987.
- [13] Li-Qun X., Young D., and Hogg D. Building a model of a road junction using moving vehicle information. In *British Machine Vision Conference*, pages 443–452, September 1992.
- [14] Cootes T.F. and Taylor C.J. Active shape models - ‘smart snakes’. In *British Machine Vision Conference*, pages 276–285, September 1992.
- [15] Shapiro L. and Brady M. Rejecting outliers and estimating errors in an orthogonal regression framework. Ouel, Robotics Research Group, University of Oxford, February 1993.