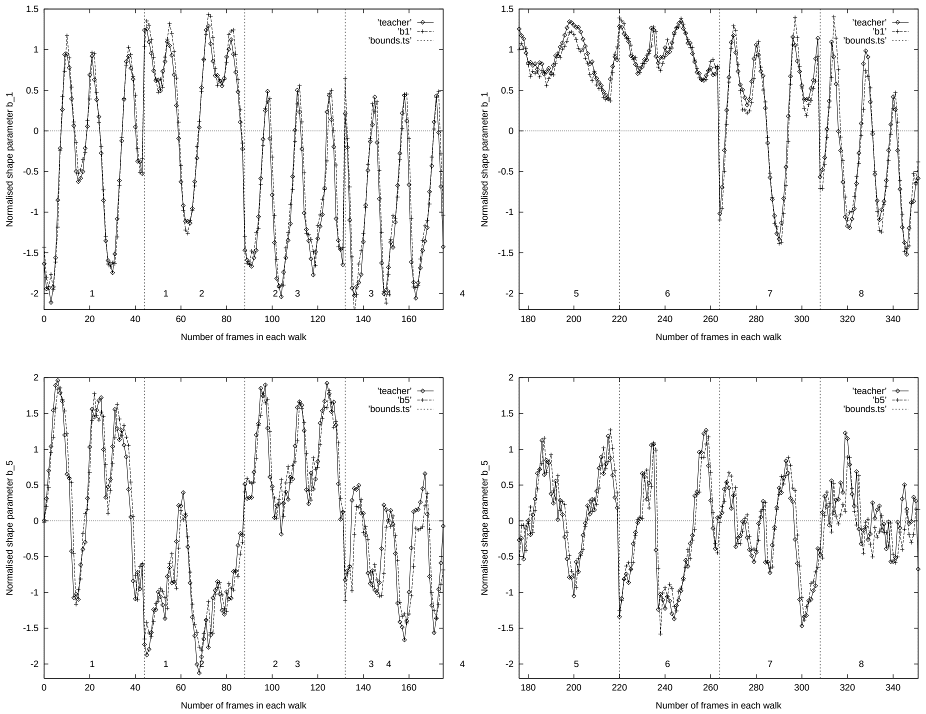

Figure 9: The target '◇' and one-step ahead prediction '+' value, of the shape parameter $b_1$ (top) and $b_5$ (bottom), in each of the eight separate walks between the dotted vertical lines. The directions followed by the pedestrian are from left to right 1.↗, 2.↖, 3.→, 4.←, 5.↑, 6.↓, 7.↘, 8.↙. Each walk is shown in 44 frames and the test set contains 352 patterns.

[7] Refenes, A.N., M. Azemar-Barac, et al., "Currency exchange rate prediction and neural network design strategies," Neural Comp. and App., **1**, pp 46-58, 1993.

[8] Cootes, T.J., C.J. Taylor, D.H. Cooper, and J. Graham, "Training models of shape from sets of examples," Proc. of British Machine Vision Conference, Sept. 1992.

[9] Elman, J.L., "Finding structure in time," Cognitive Science, **14**, pp 179-221.

[10] Rumelhart, D.E., G.E. Hinton, and R.J. Williams, "Learning internal representations by error back-propagation," In McClelland, J, Rumelhart, D.E. (eds.) PDP, **1**, pp 318-360, MIT Press, 1986.

[11] MacKay, D.J.C., "Bayesian non-linear modelling for the energy prediction competition," Tech. Report, Cavendish Laboratory, Cambridge University, 1993.

[12] Jordan, M.I. and R.A. Jacobs, "Hierarchical mixtures of experts and the EM algorithm," Neural Computation, **6**(2), pp 181-214, 1994.

BMVC 1996 doi:10.5244/C.10.65

## 5 Discussions

The task of tracking walking pedestrians in an outdoor scene can be regarded as one of modelling the spatial-temporal behaviours of a moving deformable object. Once an appropriate representation form, a *state vector*, for the object concerned is acquired from an image sequence, we can employ some nonlinear dynamic models to learn these state vectors, trying to find out the underlying characteristics or at least the empirical regularities behind these apparent phenomena. The successfully established system would be ready for forecasting the future behaviours of the state vector.

We have introduced a neural network based motion tracking system to perform the above task. A compact state vector consisting of the first 5 significant shape parameters and two directional displacements has been chosen to represent the most likely changes in the shape contour and the moving directions. The prediction system then comprises a set of seven related but separately trained neural network models, each is responsible for modelling one variable of the compact state vector. The model architecture has been designed to capitalise on crucial spatial-temporal variations in the raw data. The system has been trained using real-world, noisy data sequence of limited types of walks and number of frames. Special attention has been paid to solving the overfitting problem by exploiting the cross-validation technique. An initial evaluation of the system has been made and results presented. The one-step ahead prediction performance on an independent test sequence has been very encouraging.

Further studies include the use of other generalisation enhancement techniques and theoretically well-established frameworks such as Bayesian learning framework [11], hybrid mixture experts network [12]. We are particularly interested in the automatic determination of relevant inputs for a model, the investigation of multi-step ahead prediction of the behaviours of the pedestrian, the integration of the current system into the video rate motion tracking demonstrator.

*Acknowledgments* : The authors thank Adam Baumberg for valuable discussions.

## References

[1] Baumberg A., *Learning Deformable Models for Tracking Human Motion*, PhD Thesis, School of Computer Studies, University of Leeds, October 1995.

[2] Evans, R., “Kalman filtering of pose estimates in applications of the RAPID video rate tracker,” Proc. of British Machine Vision Conference, Sept. Leeds, 1992.

[3] Marslin, R, G.D. Sullivan, and K.D. Baker, “Kalman filters in constrained model based tracking,” Proc. of BMVC’92, Leeds, Sept. 1992.

[4] Blake, A., R. Curwen, and A. Zisserman, “A framework for spatio-temporal control in the tracking of visual contours,” Intl. J. Computer Vision, 1993.

[5] Weigend, A.S., D.E. Rumelhart, and B.A. Huberman, “Back-propagation, weight-elimination and time series prediction,” Proc. of 1990 Connectionist Models Summer School, Morgan Kaufmann.

[6] Chakraborty, K., K. Mehrotra, et al., “Forecasting the behavior of multivariate time series using neural networks,” Neural Networks, **5**, pp 961-970, 1992.

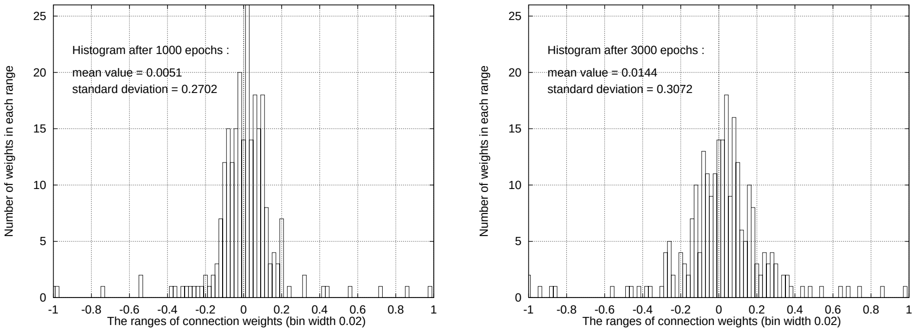

Figure 8: The histograms of the connection weights obtained in modelling $b_1$ after $1,000$ training epochs (left) when the model achieved the best generalisation performance and after $3,000$ training epochs (right) when the model was overfitted.

best prediction performance on the test set, though in this area the performance of the model on training set is much worse than that on both training and test set. This example, however, vindicates the effectiveness of cross-validation technique.

In addition, in the case of modelling $b_1$, Figure 8 gives the distributions of the model's connection weights obtained at two distinctive points : at $1,000$ training epochs or the chosen optimum stopping point and at $3,000$ training epochs when the learning error approached its minimum value about $0.116$. In the later case, the apparent increase in standard deviation of the distribution means that, with more training epochs, more quantisation levels are needed to encode the connection weights of the model, thus increasing the complexity of the model in the sense of large description length [5].

Training process : Take the process of modelling $b_1$ for example, the model was started with a random weight set with uniformly distributed values between $[-0.1, 0.1]$. The *nominal* learning rate used for weight set update was $r = 0.04$, but if the fan-in of a node is $n$, the actual learning rate for a weight leading to this node would be $r' = r/n$. This was very effective in avoiding the instability problem that is often seen in training a model with large fan-in, this also helped to smooth out the fluctuations in validation error for the ease of identifying the optimal stopping point. No momentum term has been involved in the iteration formula. The connection weights were updated following every 12 input patterns, but a training pattern only appeared once in one epoch.

System performance : In the same way as modelling $b_1$, we have obtained 7 models each being responsible for tracking one particular variable of the compact state vector. Five runs were conducted each time with a different initial weight set. The model with the smallest validation error was chosen as the final solution. Due to the limited space, only the prediction results for shape parameters $b_1$ and $b_5$, are given in Figure 9 where the true values of the test set are also depicted for comparison. It can be observed that our models have followed neatly the peaks and troughs of the test data across the eight typical walks, capturing the underlying characteristics of these motion trajectories.

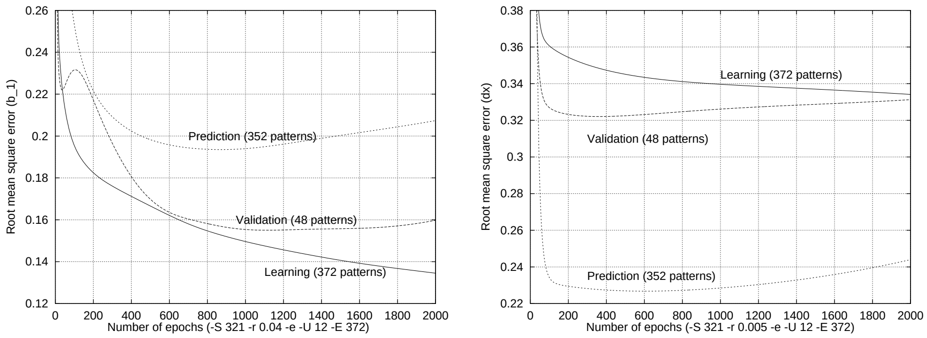

Figure 7: The rms errors versus number of epochs for the training data (solid lines) and validation data (dashed lines). The validation errors suggested that the learning process stop at around $1,000$ epochs in modelling $b_1$ (left) and at around 400 epochs in modelling $dx$ (right). The decisions were corroborated by the respective prediction performance on an independent test set, shown in dotted lines.

The cost function used was the summed square errors between the true value $x_{t+1}$ of a variable at time $t+1$ and the predicted output $\hat{x}_{t+1}$ of its corresponding model, running over the entire training set $S$,

$$E = \frac{1}{2} \sum_{t \in S} (x_{t+1} - \hat{x}_{t+1})^2 \tag{2}$$

## 4.1 Cross-validation

Having made clear the problems we had, cross-validation technique was chosen to prevent the training process from overfitting the network. The two elements of the technique are to acquire an independent validation set and to make decision on early stopping when it is needed.

The procedure : For each of the eight walk sequences, we randomly chose (without replacement) about $1/8$th of its patterns to form a validation set of 48 patterns, the remaining 372 patterns constituted the final training set. In training each model, the validation error was monitored, it decreased initially, during which period the model learned more and more the structural information of the training data, it then started to rise when the model attempted to fit the sampling noises, signifying that the training process should stop.

This phenomenon can be observed from Figure 7 which gives the performance curves in learning shape parameter $b_1$ and $x$-directional displacement $dx$. In the case of modelling $b_1$, the error on training data continued the trend of decrease with prolonged learning process, the longer the better, while the error on validation set took a slow U turn roughly in the region between $1,000$ and $1,200$ epochs, with a minimum rms error about $0.155$, which conformed to the region where the minimum prediction error was obtained. The same story was true for modelling $dx$, the validation error had its minimum value about $0.322$ in the region around 400 epochs where the model gave the

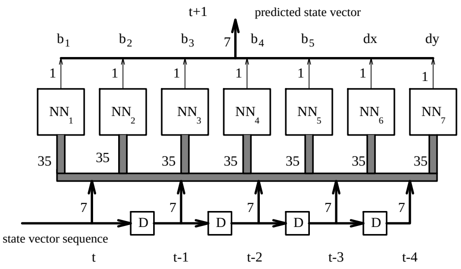

Figure 5: The system for modelling and predicting compact state vector sequences that describe the spatial-temporal variations of a pedestrian walking along eight different directions relative to the camera.

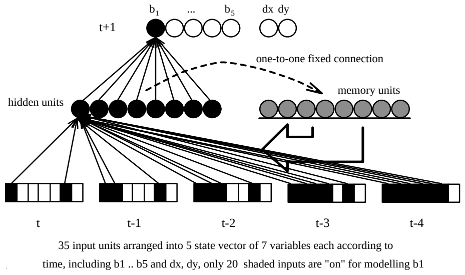
b1
...
b5
dx dy
t+1
one-to-one fixed connection
hidden units
memory units
t
t-1
t-2
t-3
t-4
35 input units arranged into 5 state vector of 7 variables each according to
time, including b1 .. b5 and dx, dy, only 20 shaded inputs are "on" for modelling b1

Figure 6: The general neural network architecture adopted for modelling individual variables of the compact state vector, displayed is for shape parameter $b_1$. The net has one hidden layer with 8 units, each is associated with a *memory unit* via a one-to-one fixed connection.

inputs, which can be the result of non-uniform sampling, and the memory units which act as an exponential filter bank to always keep a *trace* of the activations emerged in the hidden units, or

$$\overline{\mathbf{H}}_t = a\overline{\mathbf{H}}_{t-1} + (1-a)\mathbf{H}_t \tag{1}$$

where $\overline{\mathbf{H}}_t$ is the average hidden layer activation vector, exponentially weighted over the past; $\mathbf{H}_t$ is the current activation vector; $a$ is a constant close to 1. This type of model is often referred to as simple recurrent network (SRN) [9]. The only difference here is the realisation of the memory unit, instead of using Eq: 1 the SRN simply keeps a copy of the hidden units activation vector, or $\overline{\mathbf{H}}_t = \mathbf{H}_t$.

The total number of connection weights for modelling the changes in shape parameter $b_1$ is 241. Note that the actual connections between the input and hidden layer may differ from variable to variable based on our observations on the significance that each input is likely to have on the predicted variable. In Figure 6 only the 20 shaded input units are connected to the hidden layer for modelling $b_1$.[1]

# 4 The experimental studies

Given the system introduced above, we can now proceed to train these networks using the on-line back-propagation learning algorithm [10]. Our primary concern was the study of ways to improve the models generalisation performance, as, for this practical task, the major issue is network overfitting caused by the real-world, noisy, and especially limited number of training patterns, recalling that only one walk sequence is available for each of the eight moving directions and the maximum size of the training set is of the same order as the number of connection weights, e.g. 420 versus 241 in the case of $b_1$.

[1] It would be ideal to let the model decide its significant inputs automatically, like the automatic relevance determination (ARD) theory [11].

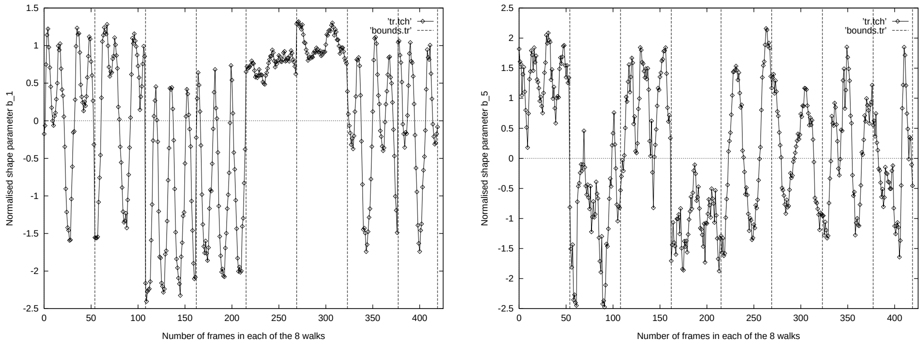

Figure 4: The training data for two components of the compact state vector: the shape parameter $b_1$ (left) and $b_5$ (right). The dotted vertical lines mark the beginning and end of each of the eight separate walks in line with the moving directions of the pedestrian, from left to right, 1.↗, 2.↖, 3.→, 4.←, 5.↑, 6.↓, 7.↘, 8.↙.

Firstly, a *compact state vector* consisting of the first five shape parameters $(b_1, \cdots, b_5)$ and the two directional displacements $(dx, dy)$ is chosen to describe the appearance of a pedestrian at an instant. The five shape parameters represent the major variations that a shape contour is likely to have over the period of time within the scene. Figure 4 shows the data for $b_1$ and $b_5$ in the training set. In the plot of $b_1$, the empirical periodicities look quite clear in most of the eight walks except for walk 5.↑ and 6.↓, whereas, the plot of $b_5$ shows a different picture in these two cases. Note that it is these regular characteristics of the data we need to model to track the motion of the object. Note also that though higher indexed shape parameters beyond $b_5$ can be added up to the compact state vector, the higher the index is, the less regular or more noisy the shape parameter will be, contributing but trivial changes in shape. The two directional displacements $(dx, dy)$ specify the object's moving direction, thus also defining the likely changes in shape contour.

Secondly, each final training pattern for the system is formed by concatenating the compact state vectors over a period of five frames. This is to properly account for the temporal variations of the motion trajectories as well as smooth out its short-term fluctuations. Also, the system can now have chance to explore the nonlinear interactions between shape parameters over the time. In the end, we are left with a total of 420 training patterns and 352 test patterns.

## 3.2 The prediction system

We now present the motion tracking system which is to be trained using the data described above. The system diagram is given in Figure 5 where the inputs are the current and time-delayed compact state vectors and the output is the predicted state vector. There are seven neural network models, each is responsible for modelling one variable of the compact state vector $(b_1, \cdots, b_5, dx, dy)$ and is trained in a separate session.

The architecture for each individual model is given in Figure 6. It has a single linear output unit and up to 8 hidden units each having a hyperbolic tangent transfer function. The dynamics of the model lies in the cooperative effects between the time-delayed

### 2.2 The data sets

The two sets of image sequences were processed according to previous discussions. The results are, for a pedestrian present, a sequence of *state vectors* each taking the form of 15 shape parameters, $(b_1, b_2, \cdots, b_{15})$, plus two measurements $(x, y)$ giving the position of the shape contour in image coordinates. In the discussions followed we choose to use the directional displacements $(dx, dy)$ of the shape within two adjacent frames instead of $(x, y)$.

The *training set* contains eight separate walks, the record length is 58 frames for walk '$\leftarrow$', 48 frames for walk '$\swarrow$', and 59 frames for the rest. This gives us initially a total of 460 state vectors for training purpose. The *test set* also contains eight separate walks, each lasting for 49 frames. There are therefore a total of 392 state vectors in the test set. Note that the test image sequences were recorded some months later and under different weather and lighting conditions. The camera's viewpoint was also subject to slight changes relative to that in the training case. For each walk, the start and finish positions as well as the actual trajectory followed are all different from the cases in the training set.

## 3 Neural networks for motion analysis

The use of state vector sequence to describe the walking trajectory of a pedestrian makes Kalman filters an immediate choice for modelling the human motion. In fact, detailed studies have been carried out and a Kalman filter based video-rate tracking system [1] has achieved some impressive results, though the system adopted several simplified conditions and linearity assumptions, such as the requirement of *constant motion*, *uniform sampling rate*, and that each shape parameter is treated (filtered) *independently* of the others by using a one-dimensional Kalman filter, thus ignoring the nonlinear interactions between the components of the state vector. These conditions, though quite reasonable in many circumstances, may be inappropriate in other situations, resulting in a tracking system susceptible to noise.

### 3.1 The formation of training data

In view of the potential problems above, alternative solutions are sought to use neural networks to learn the motions of walking pedestrians. The effectiveness of a neural network based system depends on the design of appropriate models as well as the formation of good training examples in the sense that they preserve as much useful information as possible of the state vector sequences. The main advantages of such a system are to accommodate non-uniform motion, variable sampling rate and that the trained neural network models can be used as *predictors* to forecast the future moves of individual pedestrians, i.e. the change in shape contour and moving direction. This prediction can be extended to multi-step ahead such that the occlusion of objects (the missing of the whole or part of the shape contour for a number of frames) during tracking process can be recovered, alleviating a major problem in motion tracking.

Based on experience with the current motion tracking system and initial experiments conducted on the data sets, we have come up with the following decision on the formation of training data :

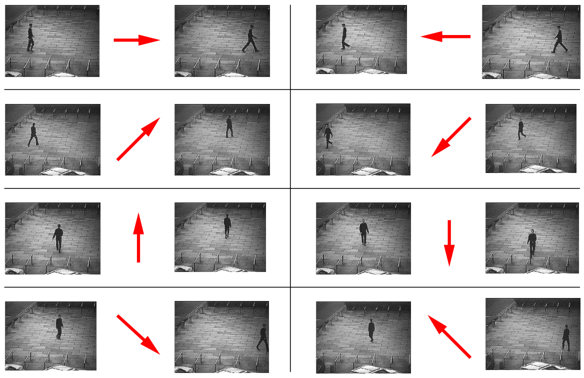

Figure 1: The first and last frame of training image sequence for each of the eight typical walks. The walk ↑, for example, shows the pedestrian walking away from the camera.

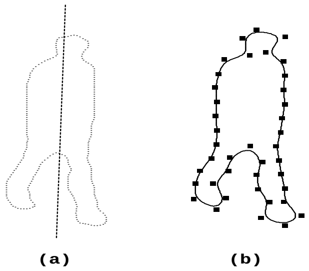

Figure 2: The boundary points and the principal axis of a walking pedestrian silhouette (a). The approximating cubic B-spline with 40 control points (b).

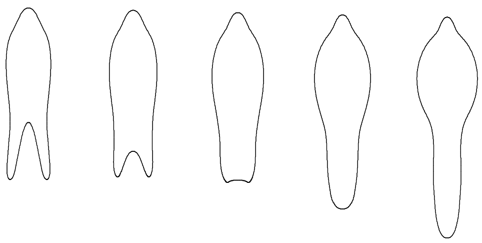

Figure 3: The effect of varying the first mode of variation, or the eigenvector with the largest eigenvalue, by $\pm 1.5$ standard deviations from the *mean shape vector*, representing the shape contour in the middle.

(over 300) for a shape are then reordered against a chosen reference point and approximated by a cubic B-spline of 40 control points. Figure 2 shows the extracted shape contour.

Eigenshape analysis : The control points, or $\mathbf{x}' = (x_i, y_i)$, $i = 1, \cdots, 40$, of cubic B-splines from several hundred frames of the eight walk sequences are aligned. From these aligned shapes a linear point distribution model (LPDM) [8] is then obtained. The model includes an 80-dimensional *mean shape vector* $\overline{\mathbf{x}}$, an 80 × 80 *covariance matrix* and a subset of $m$ *eigenvectors* $\mathbf{P} = (\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_m)$ corresponding to the $m$ most significant modes of variations.

Shape parameters : Given the LPDM above, an arbitrary shape vector $\mathbf{x}'$ can now be represented by $\mathbf{x}' = \overline{\mathbf{x}} + \mathbf{P}\mathbf{b}$, where $\mathbf{b} = \mathbf{P}^T(\mathbf{x}' - \overline{\mathbf{x}})$. $\mathbf{b} = (b_1, \cdots, b_m)$ are henceforth called shape parameters which are the data we are to work on in the remainder of this paper. The left and right two shapes of Figure 3 show the changes occurred from the mean shape by varying shape parameter $b_1$ to a certain degree.

models to automatically learn the typical nonlinear spatial-temporal behaviours that a pedestrian has demonstrated while walking along different directions. This learning process is not subject to any prior assumption or restriction on the circumstances when the motion occurs, e.g. the type of motion, the sampling rate, provided that the data sequences available for learning preserve significant information related to this motion.

The task we faced can be stated as follows: Given a *state vector* that describes the shape contour, scaling factor and position of a pedestrian in image coordinates, and the knowledge of its status for the current and last few time steps, a nonlinear dynamic system based on neural networks is required to learn the spatial and temporal variations underlying the motion that has given rise to these state vector sequences. After the system is established, it can be used to predict the future moves of the pedestrian one-step and/or multi-step ahead, thus following the pedestrian across the scene. In fact, the task is a multivariate time series prediction or *regression* problem with the number of predicted variables (the components of the state vector) being equal to that of predictors, and moreover, the state vector could have more than a dozen components, this distinguishes the current task from other well-studied time series prediction problems, e.g. [5] [6] [7]. Another apparent difficulty regarding this task is the shortage of data, as the training set only has some eight different walk sequences each providing just over 50 training patterns.

The major issues addressed in this paper include the design of motion tracking system and selection of appropriate neural networks adequate for the task, the formation of training patterns to preserve essential information of original data sequences facilitating the learning process of a model, and the exploration of techniques for the enhancement of the generalisation performance of the system. In the next section we discuss the significance of the data sequences, starting with a brief introduction to the modelling of deformable objects. In Section 3, the idea of using neural networks to learn the spatial-temporal behaviours of the motion is explored, the prediction system and component model architecture are proposed. In Section 4 experiments are conducted to evaluate the system's performance using cross-validation techniques. The paper concludes in Section 5 with some discussions of relevant issues.

## 2 The task domain

A typical outdoor scene shown in Figure 1 was studied. This shows a pedestrian walking across the scene in his normal manner along eight different directions as indicated. Two sets of image sequences have been recorded, respectively, for the purpose of training the proposed system and validating it. We now outline the process of generating state vectors from these image sequences.

### 2.1 Modelling of deformable objects

Shape extraction : In this task, the class of objects of interest are the 2D silhouettes of walking pedestrians. The moving object is firstly segmented to obtain a binary image. Morphological filters are applied to remove some fragmentation or to fill gaps. Connected regions satisfying certain constraints are then segmented from the binary image, which are traced clockwise to produce a chain of boundary points. The boundary points

# Neural Networks in Human Motion Tracking – An Experimental Study

Li-Qun Xu
School of Informatics
University of Abertay Dundee
Dundee DD1 1HG, UK
mctlqx@tay.ac.uk

David C Hogg
School of Computer Studies
University of Leeds
Leeds LS2 9JT, UK
dch@scs.leeds.ac.uk

### Abstract

A new method is proposed of tracking the motions of walking pedestrians from a video image sequence. The motions of a pedestrian are firstly summarised by sequences of *state vectors* and each state vector defines a 2D shape contour as well as the position of the pedestrian in image coordinates. Next, the task of tracking the motions is addressed in the context of multivariate time series prediction on the data sequences, and neural networks are designed to model the spatial-temporal variations underlying the motion trajectories. The neural network based tracking system, after being properly configured and trained, is capable of tracking eight typical motions of a walking pedestrian despite the state vector sequences being highly noisy and each having a very short record length. In fact, the system has learned to synthesise sequences (and their representative motions) it has never seen before.

## 1 Introduction

The accurate tracking of human movement in an outdoor scene has been a very challenging issue in machine vision. This problem represents an example of the analysis of the motion of *non-rigid* or *deformable* objects, which basically involves two tasks: The first is to identify and summarise the deformable object of interest, dealing mainly with the extraction and characterisation of static spatial variations of the object from original images. There exist a range of models that could be used for describing this object. The second task is to follow the object being focused through the image sequence, given a spatial-temporal model that has either been constructed based on some prior knowledge of the underlying motion or been learned automatically from the empirical regularities in observations of its many movements. In this paper we concern ourselves with the second task, assuming that the first task has been properly completed, see e.g.[1], which makes available a sequence of *state vectors* for us to carry on tracking.

In recent years there has been substantial research towards the tracking of motions of various kinds for a variety of objects or visual curves, either rigid or non-rigid [2] [3] [4], though the efforts are largely geared towards using traditional techniques, typically (linear) Kalman filters and their various modified versions with the help of some dedicated assumptions and constraints. We are interested, however, in employing neural network