Classifying spatiotemporal object trajectories using unsupervised learning in the coefficient feature space

Andrew Naftel Shehzad Khalid

School of Informatics
University of Manchester
Manchester M60 1QD, United Kingdom
+44 161 306 5837

a.naftel@manchester.ac.uk s.khalid-2@postgrad.manchester.ac.uk

Abstract

This paper proposes a novel technique for clustering and classification of object trajectory-based video motion clips using spatiotemporal function approximations. Assuming the clusters of trajectory points are distributed normally in the coefficient feature space, we propose a Mahalanobis classifier for the detection of anomalous trajectories. Motion trajectories are considered as time series and modeled using orthogonal basis function representations. We have compared three different function approximations – least squares polynomials, Chebyshev polynomials and Fourier series obtained by Discrete Fourier Transform (DFT). Trajectory clustering is then carried out in the chosen coefficient feature space to discover patterns of similar object motions. The coefficients of the basis functions are used as input feature vectors to a Self-Organising Map which can learn similarities between object trajectories in an unsupervised manner. Encoding trajectories in this way leads to efficiency gains over existing approaches that use discrete point-based flow vectors to represent the whole trajectory. Our proposed techniques are validated on three different datasets - Australian sign language, hand-labelled object trajectories from video surveillance footage and real-time tracking data obtained in the laboratory. Applications to event detection and motion data mining for multimedia video surveillance systems are envisaged.

Keywords

Object trajectory, event mining, motion classification, trajectory clustering, anomaly detection

1. Introduction

The current ubiquity of video surveillance systems has prompted a flurry of research activity aimed at the development of sophisticated content-based video data management techniques. General purpose tools are now urgently required for video event mining including discovery and grouping of similar motion patterns, detection of anomalous behaviour and object motion prediction. These techniques are essential for the development of next generation 'actionable intelligence' surveillance systems.

Much of the earlier research focus has been on high-level object trajectory representation schemes that are able to produce compressed forms of motion data [1, 3, 4, 10, 13, 16, 22, 23, 29, 34, 35]. This work presupposes the existence of some low-level visual tracking scheme for reliably extracting object-based trajectories [17, 36]. The literature on trajectory-based motion understanding and pattern discovery is less mature but advances using Learning Vector Quantization (LVQ) [24], Self-Organising Maps (SOMs) [18, 32], hidden Markov Models (HMMs) [5, 6], and fuzzy neural networks [19] have all been reported. Most of these techniques attempt to learn high-level motion behaviour patterns from sample trajectories using discrete point-based flow vectors as input to a machine learning algorithm. For realistic motion sequences, convergence of these techniques is slow and the learning phase is usually carried out offline due to the high dimensionality of the input data space.

Related work within the data mining community on approximation schemes for indexing time series data is highly relevant to the parameterisation of object trajectories. However, computer vision researchers have been slow to realize the potential of this work. For example, Discrete Fourier Transforms (DFT) [14], Discrete Wavelet Transforms (DWT) [9], Adaptive Piecewise Constant Approximations (APCA) [27], and Chebyshev polynomials [12] have been used to conduct similarity search in time series data.

In this paper, we apply time series modeling of spatiotemporal data to the problem of object trajectory classification and show how to learn motion patterns by projecting the high-dimensional trajectory data into a low-dimensional manifold represented by a suitably chosen coefficient feature space. The coefficients are derived using functional approximation. The vector of basis function coefficients is used as an input feature vector to a neural network learning algorithm – in this instance a SOM - which can learn similarities between object trajectories in an unsupervised manner. It is shown that significant improvements in the accuracy of trajectory classification and recognition result when learning takes place in the coefficient feature space rather than in the original high-dimensional point trajectory space.

The remainder of the paper is organized as follow. We review some relevant background material in section 2. In section 3 we present some function approximation approaches to trajectory representation. The system architecture and trajectory learning algorithm is presented in section 4 within the framework of a self-organising map. In section 5, the trajectory classification and anomaly detection procedure is discussed and experimental results for different examples of object tracking data are reported

in section 6. The paper concludes with a discussion and proposals for further work.

2. Background and related work

Trajectory descriptors as proposed in MPEG-7 [22] are known to be useful candidates for compressed representation of video object motion. Previous work has sought to represent moving object trajectories through a wide variety of direction and topological based schemes, symbolic representations, polynomial models and other function approximations [1, 3, 4, 9, 10, 12, 13, 14, 16, 22, 23, 27, 34, 35]. The importance of selecting the most appropriate trajectory model has received relatively scant attention [29]. In the recent literature, the most promising similarity retrieval approach for motion trajectories is based on symbolic approximation and string matching [11, 15]. However, this approach appears to be less suited to trajectory clustering and classification than other techniques based on one-dimensional time series. Edit distance similarity measures [15] and MINDIST search [11] commonly used in string matching incur quadratic programming costs which make the symbolic approach less attractive when object motion-based video retrieval is not the prime motivation.

It is surprising to find that many of these candidate indexing schemes have not yet been applied to the problem of motion data mining and trajectory classification. Recent work has either used probabilistic models such as HMMs [2, 5, 6] or discrete point-based trajectory flow vectors [18, 19, 24] as a means of learning patterns of motion activity. An agglomerative clustering algorithm based on the Longest Common Subsequence (LCSS) approach for grouping similar motion trajectories has been proposed in [7, 37]. Yacoob [38] and Bashir *et al*. [5, 6] have presented a framework for modeling and recognition of human motion based on a trajectory segmentation scheme. Classification is performed using Gaussian Mixture Model (GMMs) and HMMs with trajectory modeling that relies on a PCA-based representation of segmented object trajectories. In [33], a semantic event detection technique based on discrete HMMs is applied to snooker videos. Various machine learning algorithms used for classifying biological motion trajectories are compared in [20].

The contribution of this paper is to show that a trajectory-encoding scheme using a coefficient feature space can be used to learn motion patterns more efficiently than previous approaches relying on discrete point-based flow vectors. Clustering, classification and the detection of anomalous trajectories can then be carried out in the coefficient feature space with reduced computational burden.

3. Trajectory representation using function approximation

The output of a motion tracking algorithm is usually a set of noisy 2-D tracker points (x_i, y_i) representing the object's motion path over a sequence of n frames, where $i = 0,\ldots, n\text{-}1$. Often the representative point is taken to be the centroid or edge midpoint of the object's minimum bounding rectangle. The motion trajectory can be considered as two separate 1-dimensional time series, $\langle t_i, x_i\rangle$ and $\langle t_i, y_i\rangle$, the horizontal and vertical displacement against time where $t_0 < \ldots < t_{n-1}$. We consider three alternative trajectory models: Least Squares polynomials (LS), Chebyshev polynomials (CS) and DFT-derived Fourier series (FS) approximation.

LS polynomials are suitable for modelling simple motion trails in the spatial domain, e.g. vehicles moving uniformly along highways, or for smoothing x-y projections of more complex spatio-temporal trajectories. Chebyshev approximations are more appropriate for modelling highly complex spatiotemporal trajectories such as pedestrian motion exhibiting stop-start and looping motions, whilst Fourier series approximation are suitable for mixed types of trajectory. Occasionally, it may be possible to approximate the motion trail (spatial trajectory shape only) in the x-y plane. In this case, we would replace t by x or y in one of the following equations depending on the choice of principal axis [16]. This would only be worthwhile if all trajectories could be aligned with the same principal axis. An example would be the modelling of vehicle trajectories in highway traffic surveillance. However, spatial modelling neglects the temporal component inherent in motion trajectories.

In applications to fixed-camera surveillance, it is not necessary to apply shift and scale transformations to the data before model fitting. We wish to preserve shift and scale dependence at the clustering stage. The performance of the three different trajectory representation schemes is compared experimentally in section 6.

3.1 Least squares polynomials

The trajectory projected in the (x, t) space can be modelled by a polynomial $P_{m(x)}(t)$ of degree $m < n$ as

$$x \approx P_{m(x)}(t) = a_{0(x)} + a_{1(x)}t + \ldots + a_{m(x)}t^m \tag{1}$$

The projection of the trajectory in the (y, t) space can be modelled using a similar polynomial expression, $y \approx P_{m(y)}(t)$. The unknown $2(m+1)$ coefficients $a_{i(x)}$, $a_{i(y)}$ $(i = 0,..., m)$ can be determined using a least squares approximation by minimising the function E with respect to a_0, a_1, ... using trajectory points (x_i, t_i) and (y_i, t_i) for $a_{i(x)}$ and $a_{i(y)}$ respectively. The function for $a_{i(x)}$ is given as:

$$E(a_{0(x)}, a_{1(x)}, \ldots, a_{m(x)}) = \sum_{i=0}^{n-1} \{x_i - (a_{0(x)} + a_{1(x)}t_i + \ldots + a_{m(x)}t_i^m)\}^2 \tag{2}$$

The motion trajectories are therefore modelled by a feature vector of LS polynomial coefficients $\mathbf{A} = \{a_{0(x)},..., a_{m(x)}, a_{0(y)},..., a_{m(y)}\}$.

3.2 Chebyshev polynomials

Alternatively, a spatiotemporal trajectory (t_i, x_i) can be modelled by a function $f_{(x)}(t)$ expressed as a weighted sum of Chebyshev polynomials $C_{k(x)}(t)$ up to degree m, defined as

$$x = f_{(x)}(t) \approx \sum_{k=0}^{m} b_{k(x)} C_{k(x)} \tag{3}$$

where $C_{k(x)}(t) = \cos(k\cos^{-1}(t))$ and

$$b_{0(x)} = \frac{1}{m}\sum_{k=1}^{m} f_{(x)}(t_k) \;\; b_{i(x)} = \frac{2}{m}\sum_{k=1}^{m} f_{(x)}(t_k) C_{i(x)}(t_k) \tag{4}$$

for $t \in [-1,1]$ and $i = 1,..., m$. The k roots of $C_{k(x)}(t)$ are given by t_j for $1 \le j \le k$. Similar expressions can be obtained for projection in (y, t) trajectory space. Thus, the motion trajectories are represented by a feature vector of Chebyshev polynomial

coefficients **B** = $\{b_{0(x)},..., b_{m(x)}, b_{0(y)},..., b_{m(y)}\}$. Further implementation details can be found in [12].

3.3 Discrete Fourier transform

Without loss of generality, a spatiotemporal trajectory (t_i, x_i), i = 0,…, n-1 can be considered as a 1-D time series $\{x_i\}$ if $t_i = i$. The n-point DFT of $\{x_i\}$ is defined to be a sequence $\{X_f\}$ of n complex numbers, f = 0,..., n-1 given in eq.(5). A similar expression can be defined for $\{y_i\}$ given in eq.(6).

$$X_f = \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} x_i \exp\left(-j2\pi ft/n\right), \quad f = 0,1,...,n-1 \tag{5}$$

$$Y_f = \frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} y_i \exp\left(-j2\pi ft/n\right), \quad f = 0,1,...,n-1 \tag{6}$$

where $j = \sqrt{-1}$. Typically, the DFT sequence is truncated after m terms, f = 0,.., m–1, where X_0 and Y_0 are real numbers. In this case, the motion trajectory feature vector consists of 2m+2 entries (from real and imaginary parts) for each time series in $\{x_i\}$ and $\{y_i\}$.

3.4 Similarity search metric

The Euclidean distance in the feature vector space is used as the basis for comparing the similarity of two motion trajectories. Each function approximation produces a coefficient feature vector which can be used to index a 2-D trajectory. Given two trajectories Q and S, we can index these by a concatenated feature vector of coefficients q_j and s_j (j = 0, 2m+1) of dimension 2m+2, i.e. $Q = \{q_0,..., q_{2m+1}\}$, $S = \{s_0,..., s_{2m+1}\}$. The overall trajectory feature vector is formed by concatenating the separate x_i, y_i time series coefficient feature vectors. A Euclidean distance (ED) on the coefficient feature space can be expressed as:

$$ED(Q,S) = \sqrt{\sum_{j=0}^{(2m+1)} (q_j - s_j)^2} \tag{7}$$

4. Learning trajectory patterns using Self-Organising Maps

Self-organising maps (SOMs) have been previously used for motion activity classification [18, 32] with trajectories encoded as point-based flow vectors. However, the use of point-based encoded trajectories results in a high dimensional learning space and reduced system efficiency. In this case, unsupervised learning of motion patterns normally takes place offline. To achieve dimensionality reduction, we consider object trajectories as motion time series and index using a low-dimensional coefficient feature space. An overview of the system architecture used for trajectory learning is shown in Fig. 1.

4.1 Network model

The network topology chosen for the SOM is a layer of input neurons connected directly to a single 1-dimensional output layer. Each input neuron is connected to every output neuron with the connection represented by a weight vector. The network topology is shown in Fig. 2. A similar network model was proposed in [18] to learn vehicle trajectories for accident prediction.

In a SOM network, physically adjacent output nodes encode patterns in the trajectory data that are similar and hence, it is known as a topology-preserving map. Consequently, similar object trajectories are mapped to the same output neuron. The number of input neurons is determined by the size of the feature vector which relates to the selected number of coefficients in the model.

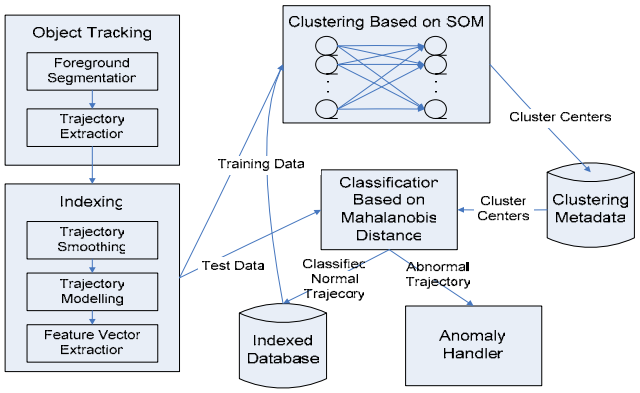


Figure 1. Overview of system architecture for learning object trajectories.

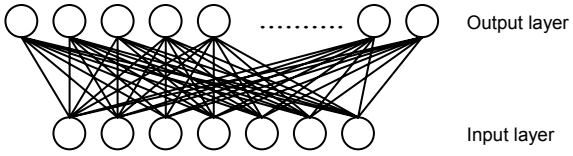


Figure 2. SOM network architecture used for trajectory learning.

4.2 Learning Algorithm

The algorithm used to cluster the trajectories differs slightly from the original SOM proposed by Kohonen [30]. The number of output neurons representing the number of distinct patterns in the data is initially set to a value greater than the desired number of cluster patterns that we wish to produce. After training the network, clusters representing the most similar patterns are merged in an agglomerative manner until the cluster count is reduced to the target number. The final number of trajectory cluster patterns is empirically chosen at present.

Let B be the input feature vector representing the set of trajectory basis function coefficients, and W the weight vector associated to each output neuron. The learning algorithm comprises the following steps:

1. Determine the winning output node k (indexed by c) such that the Euclidean distance between the current input vector B and the weight vector W_k is a minimum amongst all output neurons, given by the condition

$$c = \arg\min_k \left\| B - W_k(t) \right\| \quad \forall k \tag{8}$$

2. Train the network by updating the weights. A subset of the weights constituting a neighbourhood centred around node c are updated using

$$W_k(t+1) = W_k(t) + \alpha(t)\eta(k,c)(B - W_k(t)) \quad (9)$$

where $\eta(k, c) = \exp(-|r_k - r_c|^2 / 2\sigma_t^2)$ is a neighbourhood function that has a value of 1 when $k = c$ and falls off with distance $|r_k - r_c|$ between output nodes k and c, σ_t is a width parameter that is gradually decreased and t is the training cycle index.

3. Decrease the learning rate $\alpha(t)$ linearly over time.

4. After a pre-determined number of training cycles, decrease the neighbourhood size.

5. At the end of the training phase, merge the most similar cluster pairs until the desired number of groupings is achieved. Assuming W_a and W_b are the weight vectors associated with output neurons representing the most similar clusters, and m, n are the number of sample trajectories mapped to these neurons respectively, a new weight value W_{ab} for the merged cluster can be calculated as

$$W_{ab} = \frac{mW_a + nW_b}{m+n} \quad (10)$$

5. Trajectory classification and anomaly detection

The SOM algorithm can be used to learn a set of motion patterns for the trajectory training dataset. The resulting labelled classes can then be used to classify new unseen trajectory data as normal (i.e. belonging to one of the existing labelled classes) or abnormal (sufficiently distant from one of the existing classes). We use a simple k-NN classifier with the optimum value of k chosen by leave-one-out analysis. This involves training the classifier on all the labelled trajectories apart from the single trajectory to be tested. The step is then repeated over all trajectories in the dataset and the mean classification accuracy determined. We select the value of k that achieves the best classification result.

Classification results are presented in the following section using hand-labelled trajectories as ground truth. Visualisation of the clusters in the coefficient feature space shows that it is a reasonable assumption to represent class conditional probability density functions as multivariate normal. Anomalous trajectories can be detected through analysis of the covariance structure of a pattern at each output node. Hotelling's T^2 test is used to determine if the Mahalanobis distance of a sample trajectory to its nearest class centre makes it an outlier and thus abnormal. The test is now described in more detail.

Assume that instance feature vector x belongs to pattern class Γ_i, where $\#\{\Gamma_i\}$ denotes the number of sample vectors x assigned to class Γ_i and $i = 1,\ldots,K$. The class mean is denoted by μ_i and the covariance estimate $\sum_i$ is given by

$$\sum_i = \sum_{x \in \Gamma_i} (x - \mu_i)(x - \mu_i)^T / (\#\{\Gamma_i\} - 1) \quad (11)$$

where μ_i and $\sum_i$ are calculated during training. The T^2 statistic based on the Mahalanobis distance can be calculated as

$$T^2 = \frac{n}{n+1}(x - \mu_i)^T \sum_i^{-1} (x - \mu_i) \quad (12)$$

where $n = \#\{\Gamma_i\}$ and μ_i is the class mean to which the sample vector is closest. A hypothesis test [25] can be conducted to determine whether x is 'too far' from μ_i and hence denoted as anomalous. Given an input feature vector of dimension p in the coefficient space, we have that

$$\alpha = P\left(T^2 > \frac{(n-1)p}{n-p} F_{p,n-p} \right) \quad (13)$$

where $F_{p,n-p}$ is a random variable with an F-distribution and p, n-p degrees of freedom. $F_{p,n-p}(\alpha)$ is the upper $(100\alpha)^{\text{th}}$ percentile of the $F_{p,n-p}$ distribution.

6. Experimental results

We now present some results to demonstrate the effectiveness of the proposed clustering, classification and anomaly detection techniques in the coefficient feature space. The experiments have been performed on three different tracking datasets – hand-labelled object trajectories taken from the CAVIAR dataset [8], real-time object tracking data recorded in our laboratory, and high quality recordings of hand signs taken from the Australian Sign Language (ASL) UCI KDD archive [26]. The latter dataset (although not taken from a surveillance scenario) is included to permit comparison with previously reported motion classification techniques.

6.1 Peformance evaluation of trajectory modelling schemes

The performance of the three different trajectory representation schemes has been compared. The purpose of the experiment was to test the retrieval accuracy for each approximation scheme and to investigate the effect of varying the number of coefficients used for model fitting. We also wished to examine the effect of introducing additive noise and simulating object occlusion on retrieval performance.

Experiments have been performed using the CAVIAR dataset which consists of hand-labelled video sequences of moving and stationary people. This was originally established to provide a testbed for benchmarking vision understanding algorithms. Semantic descriptions of target object behaviours and motion coordinates had been previously generated using a interactive labelling program and the results have been stored in XML files [8]. These files have been parsed to extract ground truth labelled object trajectories.

The dataset, S, contains 222 individual object trajectories extracted from 22 different video sequences as shown in Fig. 3. A corrupted dataset S_C is produced by adding the term η*$U[0,1]$**rangeValues* to each (x, y) coordinate in the original set S, where η is a scaling factor such that $0 \le \eta \le 1$, $U[0,1]$ is uniform random noise on the interval [0,1], and *rangeValues* is the range on x and y coordinates. Each corrupted trajectory in S_C then serves as an example query Q_C and we search for its closest match Q in the original dataset S by searching for $\text{argmin}_{Q \in S}\{ED(Q_C, Q)\}$. A set of rankings $\forall Q_C \in S_C$ is produced. In the absence of noise and when no data points are excluded, the closest match to Q_C should be its corresponding uncorrupted version in S which produces a rank value of 1. For ease of

comparison we record the proportion of times (as a percentage) the query trajectory is ranked correctly as 1 when taken over all S_C. This is repeated for different number of coefficients in LS, Chebyshev and Fourier approximations and for various values of scale factor η. The results are summarised in Fig. 4.

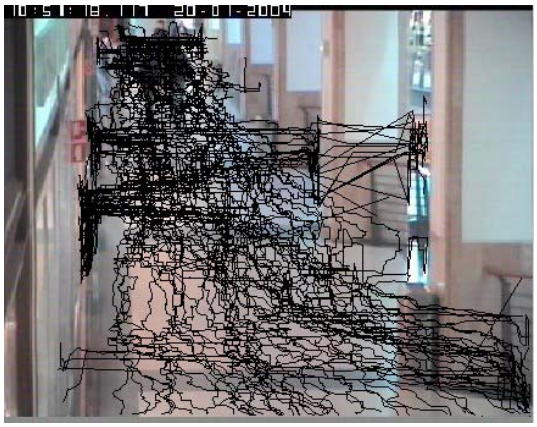


Figure 3. Background scene containing ground truth labelled object trajectories extracted from the CAVIAR dataset [8].

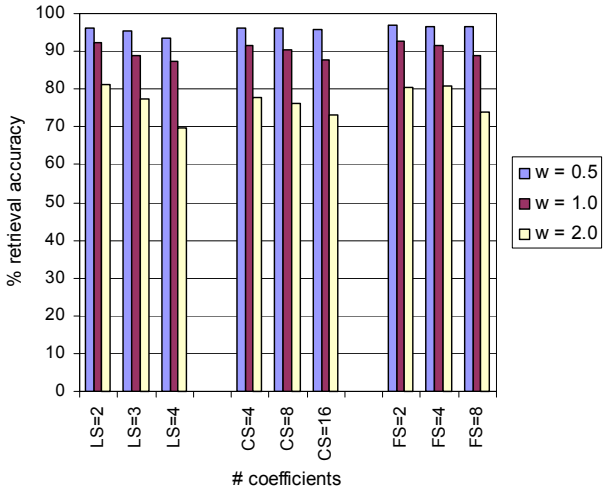


Figure 4. Effect of scaled uniform noise on trajectory retrieval accuracy. *w* is the scaling factor, LS= least squares, CS = Chebyshev polynomials, and FS = Fourier series approximation using DFT.

For small amounts of noise, both the choice of approximation scheme and number of coefficients does not appear to be too critical, although there is a slightly higher fall off in performance for LS as *m* increases. For higher noise levels, it is apparent that Fourier series approximations outperform both LS and Chebyshev polynomials. This may be explained by the fact that Euclidean distance defined over Fourier coefficients is more noise resistant in the frequency domain. In previous work, it has been shown that a LS-RANSAC approach would be beneficial if it is known that the tracking algorithm produces very noisy estimates with a significant number of outliers [29].

The retrieval experiment was then repeated but this time under simulated object occlusion by removing at random a continuous subsequence of points. This experiment determine the effect of having partial or abruptly interrupted trajectories on the choice of representation scheme. The proportion of points removed (*p*) varied between 10, 20 and 30% of the trajectories' length. Each of the results obtained were averaged over 10 random subsequence removals. In this instance there was no added noise. The percentage retrieval accuracy over all query trajectories was determined as before for each choice of approximation scheme and number of coefficients. The results are shown in Fig. 5. In this case LS polynomials perform best in presence of partial or abruptly occluded trajectories. This can be explained by the larger smoothing effects of LS operator compared to Chebyshev or Fourier series approximations.

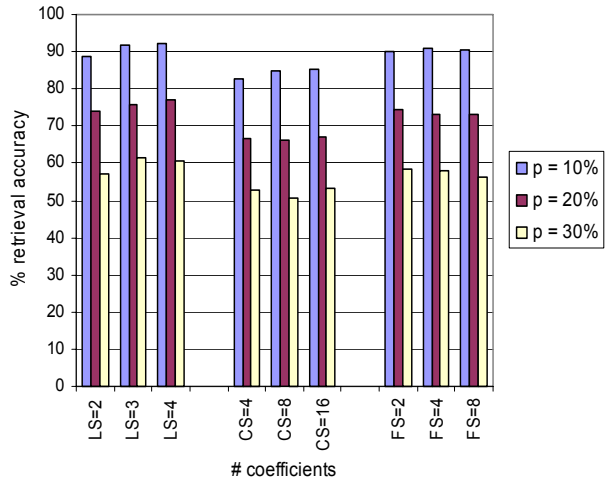


Figure 5. Effect of occluding subsequences on trajectory retrieval accuracy. *p* is the percentage of subsequence length removed from the original trajectory. Results are averaged over 10 random removals. LS= least squares, CS = Chebyshev polynomials, and FS = Fourier series approximation.

6.2 Comparison of trajectory clustering techniques

The *ED*-metric defined over the coefficient feature space is now used to perform the trajectory clustering step. We ran the SOM clustering algorithm on the previous dataset using the 3 different spatiotemporal approximations. From empirical observation, it was noticed that if the number of coefficients is too low (typically $m < 3$), poor clustering results are obtained. As a sanity check, we repeated the clustering using a standard *K*-Means algorithm [21] and the same result was observed. Although satisfactory results are obtained in retrieval experiments with a small number of coefficients, there is insufficient discriminatory power in a very low dimensional coefficient subspace to achieve a meaningful clustering outcome. This was an unexpected result that warrants further investigation.

In practice, we have found no discernible differences in SOM clustering results between spatiotemporal trajectory models generated by Chebyshev polynomials with m = 4, 6 and 8

coefficients and DFT approximation with $m = 4$ and 6 terms. The SOM algorithm always produces visually better cluster separations than K-means. This is to be expected given that SOM is better at preserving the topology of the original trajectory space. We do not attempt to first normalise trajectories to achieve scale or translational invariance since we wish to preserve these differences in the clustering stage. Preserving scale and translation dependence is a desired outcome in fixed camera surveillance applications.

The individual object trajectories have widely varying lengths ranging from 20-1600 points with a mean length of 342. The trajectory points are rectified to the ground plane using the homography mapping data provided [8]. We have indexed the object trajectories using 9 DFT coefficient feature vectors for each spatial coordinate ($m = 4$). We initially train a SOM network with 50 output neurons and then reduce these to 9 using the agglomerative clustering method described in section 4.2. The final choice for the number of clusters in the dataset is empirically determined. However, this process can be automated by applying a threshold on the distance between the closest clusters when merging the most similar clusters together. The merging process is stopped when the distance between the closest clusters lies below a certain threshold value.

In the SOM learning algorithm the neighbourhood size σ_t is decreased linearly after every Q training cycles, where Q is fixed at the start of training. The learning rate $\alpha(t)$ is reduced linearly over time until it reaches a preset minimum value and then remains constant over the fine tuning stage until the maximum number of iterations is achieved. The weight vectors are randomly initialized to lie within the expected range of the input feature vectors. This type of initialization improves the stability of the training network during the learning phase.

Sample trajectories from the test set are then classified using the classification technique described in section 5. The resulting trajectory cluster patterns are shown in Fig. 6. Visual inspection confirms that qualitatively similar motion trajectories have been clustered together quite successfully. Motions across the shopping mall corridor from left-to-right and right-to-left are grouped into separate clusters as expected. Although the proposed time series representation is velocity dependent, spatial similarities in object trajectories can still be identified in the cluster patterns. In this case we have chosen a motion representation that is view dependent and this would necessitate training the system on each camera separately. A method that deals with small PTZ motions of the camera can be developed based on techniques described in [4].

In order to visualise the effects of trajectory clustering in the transformed feature space, we perform Principal Component Analysis (PCA) on the DFT coefficient vectors. The first 3 PCs account for 94% of the total variability. Fig. 7 shows the trajectories plotted in the PCA subspace of DFT coefficients. Each point represents an instance trajectory and these are given different symbols to highlight the separate cluster groups each trajectory is allocated to. These plots show a good degree of cluster separation in the low-dimensional PCA subspace.

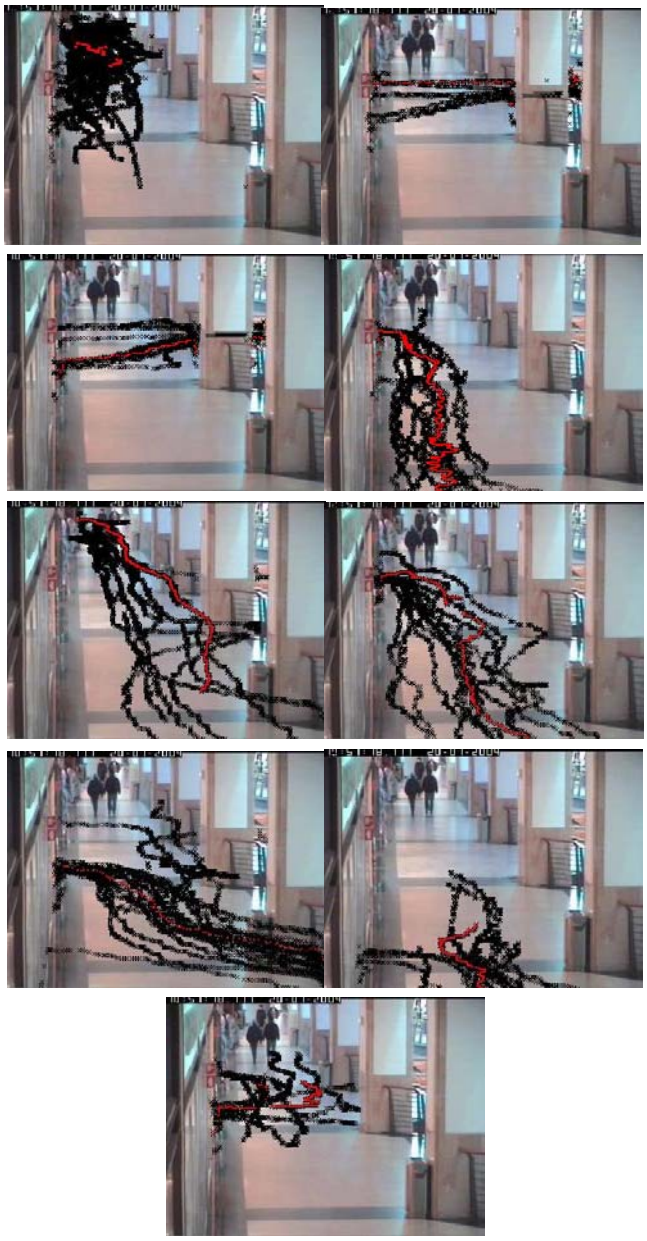


Figure 6. Clustering of motion trajectories in CAVIAR dataset using SOM with DFT-based coefficient input feature vectors. The complete set is shown in Fig. 3.

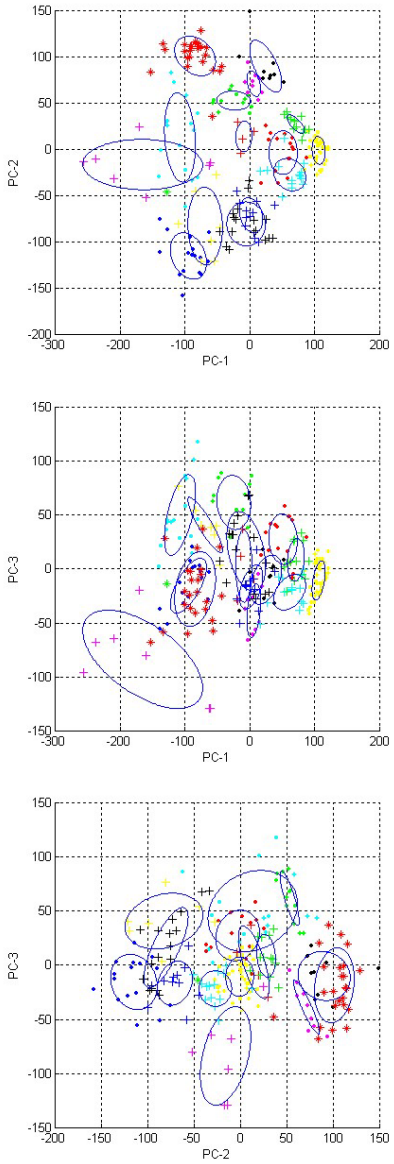


Figure 7. Trajectory clustering in the Fourier coefficient subspace (*m* = 4). For visualisation purposes we have performed PCA on the DFT coefficient feature vector and produced pairwise plots of PC_1 vs PC_2 (upper), PC_1 vs PC_3 (middle), and PC_2 vs PC_3 (lower). Error ellipses for the covariance matrices are shown for each cluster group.

We repeated the clustering experiment on a synchronised set of frontal views taken from the same dataset. The resulting trajectory cluster patterns are shown in Fig. 9 with the complete set of trajectories depicted in Fig. 8. Again visual inspection confirms that qualitatively similar motion trajectories have been clustered together quite successfully. Motion clusters approximately fall into one of the following 6 classes – motion left-to-right and right-to-left across the corridor, motion into the store from left/right direction, and motion out of the store and towards left/right direction.

Figure 8. Background scene of synchronized frontal view of shopping mall corridor with overlaid set of hand-labelled trajectories.

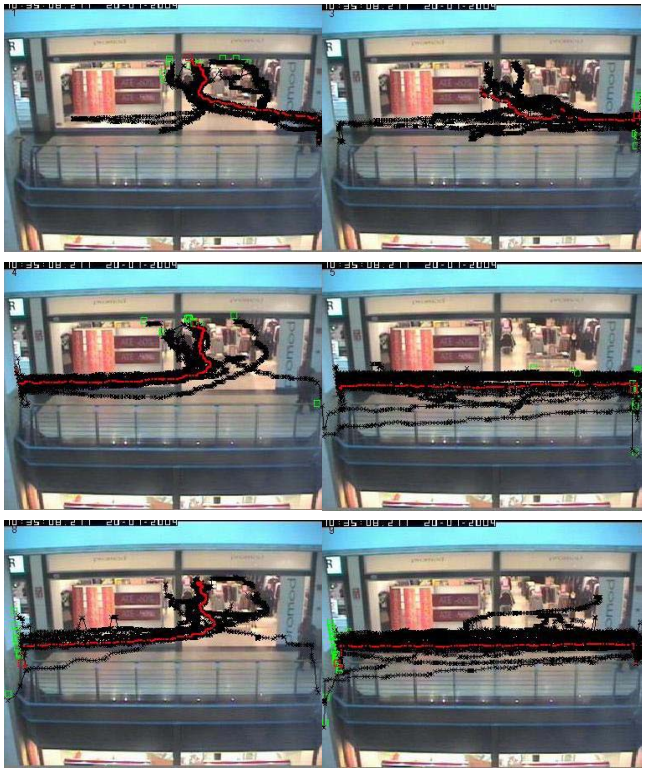


Figure 9. Motion trajectory cluster patterns in synchronised view of Fig. 6 using SOM with DFT coefficient feature vectors.

To investigate the effectiveness of clustering in the coefficient feature space compared to clustering with point-based flow vectors, we performed some additional classification tests. The class labels of the motion patterns shown in Fig. 9 were assigned using the SOM unsupervised learning algorithm described in section 4. This dataset was chosen as it demonstrates good class separation.

For comparison purposes this was repeated using *K*-means clustering [21]. The assigned labels displayed in Fig. 9 were taken to represent ground truth. The dataset, S_T, was then randomly partitioned into equal-sized training and test sets for cross validation purposes. We used a *k-NN* classifier (k = 1) to classify all instance trajectories from the test set and generated the overall classification accuracy. To avoid bias, we repeated the random partitioning 500 times and averaged the classification errors over the test set. The results summarised in Table 1 demonstrate the superiority of learning trajectory patterns in the coefficient feature space. The classification accuracy obtained using coefficient feature space learning is higher than point-based trajectory encoding for both SOM and *K*-means algorithms.

Table 1. Comparison of mean overall classification accuracy for 2 different clustering techniques (SOM and *K*-means) and 2 different trajectory encodings (DFT coefficient feature space and point-based flow vectors). #classes : #trajectories = 6 : 62 (# training set = 63, # test set = 62).

Method type	% Accuracy
SOM: DFT coefficients	93.7
SOM: point flow vectors	81.2
K-Means: DFT coefficients	89.7
K-Means: point flow vectors	83.0

For the next experiment, we compared the performance of all 4 methods in trajectory classification and prediction. From the original set S_T, we define a set of partial trajectories S_P by removing 10% of the data points from the end of each trajectory. This is increased up to 50% in steps of 10. The partial trajectories are then passed to the learning algorithm for classification. The class assigned to the complete trajectory is treated as the ground truth when assessing classification accuracy for the partial trajectory. We compare the point-based flow vector and DFT coefficient feature vector representation for both SOM and *K*-means trajectory learning techniques. The classification is based on the Mahalanobis distance between the input vector x_P representing the partial trajectory and cluster mean μ_i associated with *i*th output neuron/cluster centre. The sample x_P is assigned to class *k* if

$$k = \arg\min_{i \in K} \{(x_P - \mu_i)^T \sum_i^{-1} (x_P - \mu_i)\} \quad (14)$$

where Σ_i is covariance estimate and is calculated using eq.(11). x_P is said to be misclassified if it is not assigned to the same class when trained on S_T. The mean classification errors based on motion prediction for the partial trajectories using each of the four approaches can be seen in Fig. 10. Once again, the classifier derived from an SOM-based learning technique combined with trajectory representation in the DFT-coefficient feature space outperforms *K*-means and point-based flow vector encoding as it achieves lower misclassification errors. Hence, parameterized models prove more effective than point-based flow vectors in the trajectory prediction and classification task.

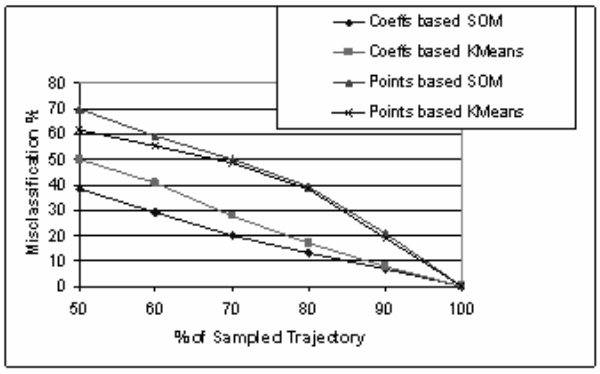


Figure 10. Comparison of mean overall classification accuracy in motion activity prediction using 2 different clustering techniques (SOM and *K*-means) and 2 different trajectory representations (DFT-coefficient feature space and point-based flow vectors). #classes : #trajectories = 6:125.

6.3 Experiments using Australian Sign Language data

Classification experiments have also been performed on the Australian Sign Language (ASL) dataset [26]. The trajectories are derived from the (x, y) coordinates of the signer's hand over a sequence of frames as different word classes are signed. The results presented in this section may be compared with those reported in [5] for HMM-based motion recognition. Other trajectory classification methods have have also been tested on this dataset [2, 4, 6, 11]. Hand sign trajectories for the word class *forget* are shown in Fig. 11.

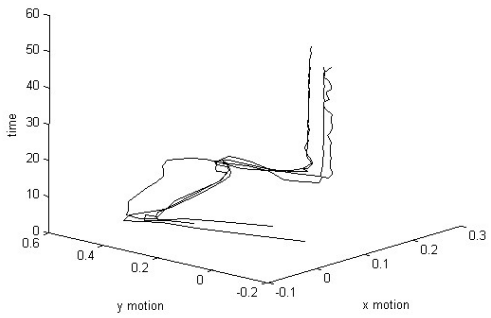


Figure 11. Three examples of hand sign trajectories for the word class *forget* in the Australian Sign Language (ASL) dataset. The vertical axis represents the time.

Each word class has 27 examples of signs with a trajectory point length of 57. A set of 30 word classes has been chosen. We used a supervised form of SOM to determine the classification accuracy for DFT-coefficient feature space trajectory learning. For the required number of word classes, the motion signing data is randomly partioned into equal-sized training and test sets. The codebook vectors are then learnt using the training data. The SOM is initialized with the number of output neurons set equal to the number of word classes present. The weight vectors are

initialized with the mean of the trajectories that belong to each class. Then the training data is presented sequentially to the network and the cluster centers for the specific class are updated. After this step, the test set is passed to the classifier and the class labels obtained are compared with the ground truth. The experiment is repeated with different numbers and combinations of word classes. Each classification experiment is averaged over 100 runs to reduce any bias resulting from favourable word selection. The classification accuracies are reported in Table 2. Since ground truth is available for the ASL dataset, this experiment gives some indication of motion recognition rates achievable using our trajectory learning system. We achieved similar performance to a HMM-based recognition system described by Bashir [5] who reported a classification accuracy of 91.2% over 3 word classes. However, our approach is conceptually simpler and computationally less expensive.

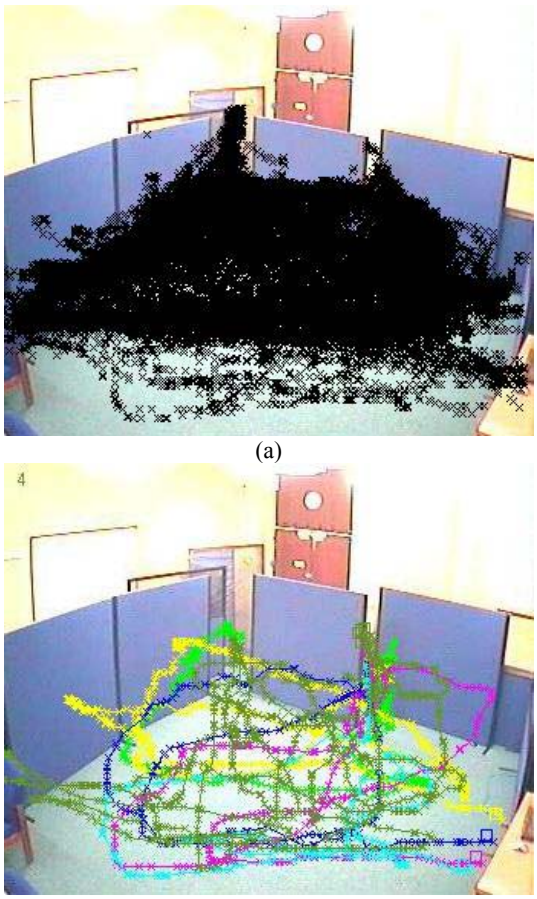
Table 2. Trajectory classification results for the Australian Sign Language (ASL) dataset. Trajectories are modeled in the DFT-coefficient feature space.

# classes : # trajectories	% Accuracy
2 : 54	95.7
3 : 81	91.0
4 : 108	89.9
8 : 216	82.1
16 : 432	76.3
24 : 648	70.1

6.4 Detecting anomalous trajectories

In the final experiment, we test the performance of the anomaly detection component of our vision system for trajectory-based motion understanding. The dataset was obtained from tracking people moving around in our laboratory. The tracking algorithm used to collect the data is described elsewhere [31]. Human movements were planned so that object trajectories could be grouped into 4 distinct classes and hence ground truth was directly available. The motion trajectory dataset superimposed over the background scene is shown in Figure 12(a).

The dataset also included some motion activities that varied deliberately from the planned movement patterns. This motion data was excluded from the training set. We randomly selected half the dataset for unsupervised learning and then presented the whole dataset together with the unusual trajectories as a test set. The clustering results are shown in Fig. 13. Visual inspection confirms that qualitatively similar motion trajectories have been grouped together as expected. This can be seen by observing the points marked by rectangles which indicate the trajectory initial points. Abnormal trajectories which are defined to be sufficiently distant from all the identified classes such that eq.(13) is satisfied for $P < 0.01$ are shown in Figure 12(b). All the trajectories that deliberately diverged from the 4 planned movement patterns were correctly identified as anomalous.



(a)

(b)

Figure 12. (a) Laboratory background scene with object trajectories derived from motion tracking scheme [31]. (b) Trajectories identified as anomalous using Hotelling's test with $P < 0.01$.

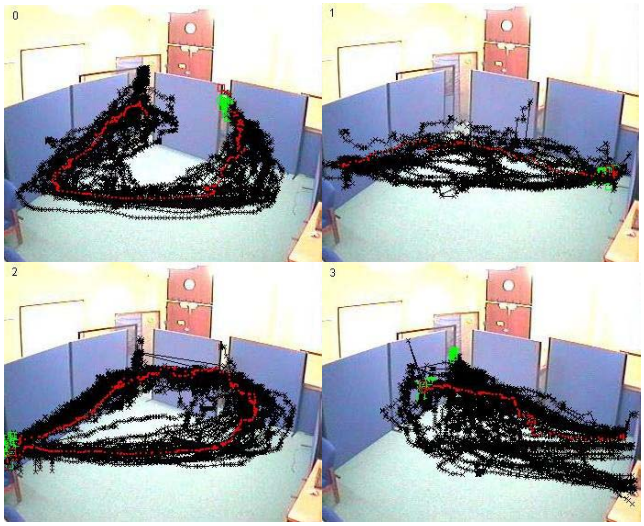


Figure 13. Trajectory cluster patterns obtained using SOM clustering in the DFT coefficient feature space.

7. Discussion and conclusions

This paper presents a neural network learning algorithm for classifying spatiotemporal object trajectories. Global features of motion trajectories are found to be well-represented by DFT-based Fourier series approximations and this is apparent in the cluster visualizations. Using the coefficients of basis functions as input feature vectors to a neural network learning algorithm offers an efficient alternative to the use of discrete point-based flow vectors for trajectory classification and anomaly detection.

A possible drawback of this approach is for representation of highly complex trajectories resulting from partial trajectories stitched together over multiple camera views. These are inherently unsuited to a global function approximation. One possibility is to use a trajectory segmentation scheme or multiscale approach and augment the feature vector with additional entries relating to object shape or colour.

A more comprehensive performance evaluation is now required using realistic crowded video sequences where occlusions and target misdetections will result in highly fragmented, partial and noisy trajectories. The robustness of the classification technique requires thorough investigation under these circumstances. We would also like to compare other dimensionality reduction and machine learning techniques for trajectory classification, e.g. ICA, HMMs and semi-supervised learning.

This paper presents a novel vision system component for trajectory-based event detection. Trajectories are modeled as motion time series using DFT coefficients and activity patterns are then learnt in this reduced feature space using a SOM network. Improvements in recognition accuracy and learning efficiency are achieved when compared to point-based trajectory encoding and other clustering techniques. Assuming trajectories are distributed normally in the transformed coefficient feature space, a Mahalanobis classifier can be used to distinguish between normal and abnormal trajectory patterns. Our techniques have been validated using three different types of video tracking data.

8. References

1. Aghbari, Z., Kaneko, K., Makinouchi, A.: Content-trajectory approach for searching video databases. *IEEE Trans. Multimedia* 5(4), 516-531 (2003)
2. Alon, J., Sclaroff, S., Kollios, G., Pavlovic, V.: Discovering clusters in motion time-series data. In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, (2004)
3. Bashir, F., Khokhar, A., Schonfeld, D.: Segmented trajectory-based indexing and retrieval of video data. In: *Proceedings of. IEEE International Conference on Image Processing*, Spain, pp. 623-626 (2003)
4. Bashir, F., Khokhar, A., Schonfeld, D.: A hybrid system for affine-invariant trajectory retrieval. In: *Proceedings of ACM SIGMM Multimedia Information Retrieval Workshop*, pp. 235-242 (2004)
5. Bashir, F., Khokhar, A., Schonfeld, D.: HMM-based motion recognition system using segmented PCA. In: *Proc*eedings of *IEEE International Conference on Image Processing (ICIP 2005)*, Genoa, Italy (2005)
6. Bashir, F., Ashfaq, A., Khokhar, A., Schonfeld, D.: View-invariant motion trajectory-based activity classification and recognition. *ACM Multimedia Systems*, Accepted to appear (2006)
7. Buzan, D., Sclaroff, S., Kollios, G.: Extraction and clustering of motion trajectories in video. In: *Proc*eedings of *International Conference on Pattern Recognition* (2004)
8. CAVIAR: Context aware vision using image-based active recognition. [Online]. Available: [http://homepages.inf.ed.ac.uk/rbf/CAVIAR]
9. Chan, K., Fu, A.: Efficient time series matching by wavelets. In: *Proceedings of International Conference on Data Engineering*, Sydney, pp. 126-133 (1999)
10. Chang, S.-F., Chen, W., Meng, H.J., Sundaram, H., Zhong, D.: A fully automated content-based video search engine supporting spatiotemporal queries. *IEEE Trans. Circuits Syst. Video Technol,* 8(5), 602-615 (1998)
11. Chen, L., Ozsu, M.T., Oria, V.: Robust and fast similarity search for moving object databases. In *Proceedings of ACM SIGMOD,* Maryland, USA, pp. 491-502 (2005)
12. Cui, Y., Ng, R.: Indexing spatio-temporal trajectories with Chebyshev polynomials. In: *Proceedings of ACM SIGMOD Conference*, pp. 599-610 (2004)
13. Dagtas, S., Ali-Khatib, W., Ghafor, A., Kashyap, R.L.: Models for motion-based video indexing and retrieval. *IEEE Trans. Image Proc.* 9(1), 88-101 (2000)
14. Faloutsas, C. Ranganathan, M., Manolopoulos, Y.: Fast subsequence matching in time-series databases. In: *Proceedings of ACM SIGMOD Conference*, pp. 419-429 (1994)
15. Hsieh, J.W., Yu, S-L., Chen, Y-S.: Trajectory-based video retrieval by string matching. In: *Proceedings of International Conference on Image Processing*, pp. 2243-2246 (2004)
16. Hsu, C.-T., Teng, S.-J.: Motion trajectory based video indexing and retrieval. In: *Proceedings of IEEE International Conference on Image Processing*, pp. 605-608 (2002)
17. Hu, W., Tan, T., Wang, L., Maybank, S.: A survey on visual surveillance of object motion and behaviors. *IEEE Trans. Systems, Man & Cybernetic. Part C*, 34(3), 334-352 (2004)
18. Hu, W., Xiao, X., Xie, D., Tan, T., Maybank, S.: Traffic accident prediction using 3-D model-based vehicle tracking. *IEEE Trans. Vehicular Tech*. 53(3), 677-694 (2004)
19. Hu, W., Xie, D., Tan, T., Maybank, S.: Learning activity patterns using fuzzy self-organizing neural networks. *IEEE Trans. Systems, Man & Cybernetic. Part. B*, 34(3), 1618-1626 (2004)
20. Ivo, F., Sbalzarinii, J.T.: Machine learning for biological classification applications. Technical Report *Center for Turbulence Research, Proceedings of the Summer Program* (2002)

21. Jain, A.K., Dubes, R.C.: *Algorithms for Clustering Data*, Prentice Hall (1998)
22. Jeannin, S., Divakaran, A.: MPEG-7 visual motion descriptors. *IEEE Trans. Circuits Syst. Video Technol.* 11(6), 720-724 (2001)
23. Jin, Y., Mokhtarian, F.: Efficient video retrieval by motion trajectory. In: *Proceedings of British Machine Vision Conference* (2004)
24. Johnson, N., Hogg, D.: Learning the distribution of object trajectories for event recognition. *Image Vis. Comput.*14(8), 609-615 (1996)
25. Johnson, R.A., Wichern, D.W.: *Applied Multivariate Statistical Analysis*, 4th Edition. Prentice-Hall, New Jersey, (1998)
26. KDD archive [Online]. Available: [http://kdd.ics.uci.edu/databases/auslan2/auslan.data.html]
27. Keogh, E., Chakrabarti, K., Pazzani, M., Mehrota, S.: Locally adaptive dimensionality reduction for indexing large time series databases. In: *Proceedings of ACM SIGMOD Conference*, pp. 151-162 (2001)
28. Keogh, E., Kasetty, S.: On the need for time series data mining benchmarks: A survey and empirical demonstration. *Knowledge and Data Discovery*, pp. 102-111 (2002)
29. Khalid, S., Naftel, A.: Evaluation of matching metrics for trajectory-based indexing and retrieval of video clips. In: *Proceedings of 7th IEEE Workshop on Applications of Computer Vision*, Colorado, USA, pp. 242-249 (2005)
30. Kohonen, T.: *Self-Organizing Maps,* 2nd Edition. Springer-Verlag, New York (1997)
31. Naftel, A., Khalid, S.: Video sequence indexing through recovery of object-based motion trajectories. In *Proceedings of Irish Machine Vision and Image Processing Conference* (IMVIP'04), Dublin, Eire. pp 232-239 (2004)
32. Owens, J., Hunter, A.: Application of the self-organising map to trajectory classification. In: *Proceedings of IEEE International Workshop on Visual Surveillance*, pp. 77-83 (2000)
33. Rea, N., Dahyot, R., Kokaram, A.: Semantic event detection in sports through motion understanding. In: *Proceedings of Conference on Image and Video Retrieval*, Dublin, Ireland, (2004)
34. Shim, C., Chang, J.: Content-based retrieval using trajectories of moving objects in video databases. In: *Proceedings of IEEE 7th International Conference on Database Systems for Advanced Applications*, pp. 169-170 (2001)
35. Shim, C., Chang, J.: Trajectory-based video retrieval for multimedia information systems. In: *Proceedings of ADVIS*, pp. 372-382 (2004)
36. Vlachos, M., Kollios, G., Gunopulos, D.: Discovering similar multidimensional trajectories. In: *Proceedings of International Conference on Data Engineering,* pp. 673, (2002)
37. Wang, L., Hu, W., Tan, T.: Recent developments in human motion analysis. *Pattern Recognition* 36(3), 585-601 (2003)
38. Yacoob, Y., Black, M.J.: Parameterized modeling and recognition of activities. *Computer Vision and Image Understanding,* 73(2), 232-247 (1999)