Feature Extraction from Sensor Data Streams for Real-Time Human Behaviour Recognition

Julia Hunter and Martin Colley

Department of Computer Science, University of Essex, Wivenhoe Park,
Colchester CO4 3SQ, U.K.
{jhunte,martin}@essex.ac.uk

Abstract. In this paper we illustrate the potential of motion behaviour analysis in assessing the wellbeing of unsupervised, vulnerable individuals. By learning the routine motion behaviour of the subject (i.e. places visited, routes taken between places) we show it is possible to detect unusual behaviours while they are happening. This requires the processing of continuous sensor data streams, and real-time recognition of the subject's behaviour. To address privacy concerns, analysis will be performed locally to the subject on a small computing device. Current data mining techniques were not developed for restricted computing environments, nor for the demands of real-time behaviour recognition. In this paper we present a novel, online technique for discretizing a sensor data stream that supports both unsupervised learning of human motion behaviours and real-time recognition. We performed experiments using GPS data and compared the results of Dynamic Time Warping.

Keywords: sensor data stream discretization; unsupervised learning; real-time behaviour recognition.

1 Introduction

There are many factors that can limit a person's ability to live a fully independent life, whether as a result of their age or due to physical or mental impairments. Yet often these people desire greater independence than they can safely be granted, such as a young child demanding greater freedom or an elderly person wishing to remain in their own home. The research that we describe here was conceived of with the needs of such people in mind. We are interested in the potential of human motion behaviour analysis in assessing the wellbeing of vulnerable individuals. We learn the routine motion behaviour of the subject (i.e. the places visited, and the routes taken between places) and then show it is possible to detect unusual behaviours while they are happening. An example of unusual behaviour could be as simple as taking a wrong turning and becoming lost. For a young child or a memory-impaired adult this could be a frightening and potentially dangerous situation; the quicker the responsible care-giver can be alerted, the better the outcome will likely be. This type of human motion behaviour analysis has not thus far received much attention within the research

J.N. Kok et al. (Eds.): PKDD 2007, LNAI 4702, pp. 115–126, 2007.

community. However, other kinds of human behaviour have been studied with the same goals of supporting vulnerable people in living independent lives. In particular, a large body of work investigates Activities of Daily Living (ADL) monitoring [1], where the execution of everyday tasks (such as food preparation, having a bath) is recorded and analyzed. It is widely accepted that analysis of the tasks performed can give an indication of a person's wellbeing, and that changes in ADL performance can indicate a change in the subject's condition. Of course, once the individual leaves their home the monitoring stops. We examine the potential of human motion behaviour analysis for extending this kind of monitoring beyond the home.

Section 2 considers some implementation practicalities. Section 3 reviews some data mining techniques that can be applied to the problem of learning and recognizing motion behaviour. Section 4 presents our solution to the same problem. Section 5 discusses the experiments performed to investigate the utility of our method. Section 6 draws conclusions and presents our ideas for further work.

2 Design and Definitions

Let us consider the scenario of a child who is allowed some travel independence yet whose parents would like to be kept aware of changes in their behaviour. If the child starts to take a forbidden shortcut to school the parents would like to find out, as they would if the child decided to go to the park instead of school. The nature of the application is that sensitive data is being collected and the monitored individual may feel that their privacy is being intruded upon. There is also concern that such data collection could be exploited in various ways in the case of a security breach. In order to address some of these concerns we decided that data collection and processing should both happen locally on a small, portable computing device. This means that the time and space complexity of the algorithms used must be kept to a minimum. Additionally, some potential users are computer illiterate; therefore behaviour learning should not require user interaction i.e. the learning algorithms should be unsupervised. Data is obtained from a GPS sensor, so we must work with the inherent coverage and accuracy constraints of the GPS system. These are all points that the design of the system must take into account.

Now let us consider some design issues that will guide implementation. We have decided that an individual's motion behaviour model consists of all the places visited and all the routes travelled; these are connected as a network. The resolution of the model is restricted by the accuracy and availability of GPS data; for example, we will distinguish between buildings but not between rooms in a building. A tested approach to the modelling of places is as a [latitude, longitude] point plus a radius [2] and we follow this example. The resolution of route modelling should support the differentiation between multiple routes connecting the same pair of places.

We define the recognition of an individual's behaviour to be the results of a comparison of what he is currently doing (whether in a place or on a journey) with his model. Real-time recognition means that this process is carried out straight away, rather than waiting for a convenient moment (such as the end of a journey).

3 Related Work

A large body of data mining literature is dedicated to the analysis of time series, and in particular *continuous* time series, where points in the series are real-numbered rather than discrete values (examples of discrete data are the binary values 0/1 or letters of the alphabet). Much work is devoted to the conversion of continuous data to a discrete representation so that discrete analysis can be carried out. This transformation is referred to as "symbolization" or "discretization". The efficiency of numerical computations is greatly increased when the data set is transformed from continuous to discrete [3]; this can lead to faster execution, which is important to real-time monitoring and control operations, and is often less sensitive to measurement noise [3]. It also allows the wealth of discrete data analysis techniques from fields such as bioinformatics and text processing to be applied [4]. Most of the time series data that is the object of data mining research consists of long-term observations of phenomena of a periodic nature, whether this is the performance of financial markets or a patient's heartbeat trace. Many techniques that are suited to this kind of data do not fit our needs. Rather, as explained by [5], the data we measure is the result of a person's intentions; conscious human choice affects every value collected. There are nonetheless some common transformations that we can apply:

Piecewise Approximation (Segmentation)
The time series is decomposed into n homogeneous pieces, segments, such that the data in each segment can be described accurately by a simple model (i.e. Piecewise Constant Approximation, Piecewise Linear Approximation) [6]. This is also referred to as *segmentation*. The distance measure used to compare two segmentations then uses the geometry of the representation [7]. Batch algorithms require n as an input parameter. Online versions, e.g. [13], also exist; here the input parameter is the maximum allowable error per segment. Unfortunately, where the data does not correspond well with the chosen model, over-fitting occurs [14].

Symbolization
SAX [4] takes as input the number of symbols and the number of sub-sequences to decompose each time series into. The initial reduction in dimensionality is obtained by finding the mean of each subsequence. The entire data set is then analysed in order to define the range of values mapping to each symbol, so that symbols are equi-probable overall. Finally the discrete representation is obtained by mapping subsequence means to symbols. Distance between two time series is obtained using a measure derived from Euclidean distance.

Feature Extraction
In order to extract features, some degree of prior knowledge about the data set is required. Domain experts may be used. In [8] motion is represented as a series of symbols, e.g. left turn, straight line. [9] quantify the maxima and minima in a data set and then use these points as a compressed representation.

No Transformation
It is also possible to compare time series using the original, raw data without this initial transformation. Two very common (and long-standing) approaches are [10]:

Euclidean Distance (ED) and Dynamic Time Warping (DTW). ED can only be calculated for equal-length time series, as every point in the candidate time series needs to be compared to its equivalent in the query time series. Thus it may be necessary to carry out interpolation first. ED has time complexity O(n), with additional overhead attributable to interpolation. DTW [11] is able to compare time series of different lengths via a non-linear mapping, also utilizing a Euclidean distance. It is less sensitive to skewing on the time axis than ED and has time complexity $O(n^2)$. Both DTW and ED could have very large n since no dimensionality reduction is carried out.

These data representations and distance measures could be used with the GPS data to perform clustering of the journeys recorded and thus learn models for the routes followed. It is relatively straightforward to perform agglomerative hierarchical clustering [12] once every possible pair of journeys has been compared; any of the techniques described above could be applied to this learning task. However, we wish to use the same techniques to then perform real-time recognition; this requires that data transformation and comparison should be performed online, i.e. while the journey is taking place. The data mining techniques discussed here emphasise accuracy (and in some cases scalability). They are often implemented using batch algorithms that were not intended to give real-time responses of the type we seek. In the context of our application, this would mean waiting for a particular journey to be completed before searching for it in the model. Considering the example of a confused person becoming lost and walking for an hour in the wrong direction, this delay is not acceptable. Real-time recognition demands immediate responses, even if this means basing the response on incomplete information.

In summary, many data mining algorithms implement batch processing, an approach that is unsuited to real-time recognition. Additionally, algorithms have been designed for accurate processing of large data sets; these are not adapted to the processing and storage limitations of small computing devices.

The remainder of this paper presents our solution to the online processing and analysis of sensor data so that it can be used for unsupervised learning and real-time recognition on a device with limited computing resources.

4 Online Data Stream Symbolization (ODSS)

Information relating to the subject's position is obtained from a GPS receiver. The GPS receiver can generate *track angle* data as well as [latitude, longitude] coordinates. The track angle is the angle of travel in degrees with respect to true north, generated by the GPS receiver from the change in [latitude, longitude]. We decided to use track angle as the input to our behaviour model instead of [latitude, longitude] because it efficiently encodes changes in two-dimensional position using one-dimensional data. Fig. 1 and fig. 2 show [latitude, longitude] and track angle plots for a single journey. Inspection of track angle data generated by a person moving about shows frequent areas of steady (or near-steady) state corresponding to periods of straight-line travel. The associated value is the real-numbered track angle. These steady state sections vary in length and frequencies according to the route travelled, but are always present to some degree.

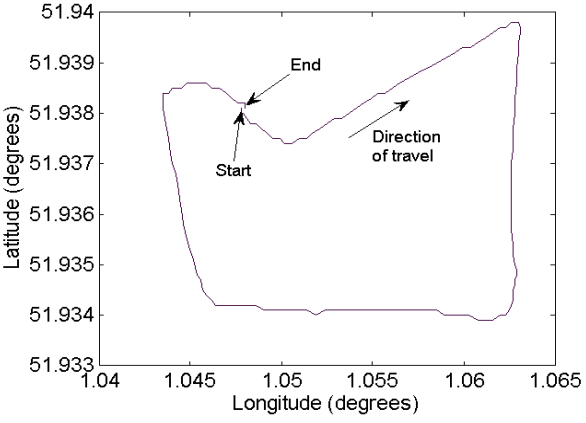

Fig. 1. Plot of latitude against longitude for a circular route travelled clockwise

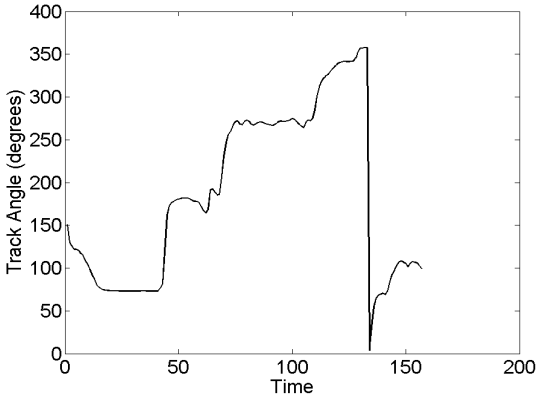

Fig. 2. Plot of track angle against time step for the journey in fig. 1

This characteristic is interesting because it has the potential to uniquely characterize a journey. It can be treated as a representation of the continuous time series to which discrete analysis techniques can be applied. We refer to this representation as a set of symbols, where the symbols are defined in an online fashion as the analysis proceeds. Each symbol consists of a real-numbered value and a tolerance (this is one input to the feature extraction process) describing the degree of variation tolerated with the region of steady state. The second input parameter is the minimum duration of a state for it to be extracted as a symbol. This is the essence of our Online Data Stream Symbolization (ODSS) approach. The results of applying ODSS to the journey in fig. 1 are shown in fig. 3. The input parameters used here are: tolerance of +/- 4 degrees and minimum duration of 12s; these values were selected empirically and had previously been found to work well on a range of data. The overall set of symbols is not restricted, and as more journeys are processed the total number of symbols used by all the journeys could become quite large. In this sense it

is different from other symbol definition processes such as that used by [4], which fix the number and definition of the symbols in advance of performing the transformation. The ordering of the symbols is preserved, as this reflects the information in the time dimension of the original data. The symbols are considered to belong to an *ordered* set.

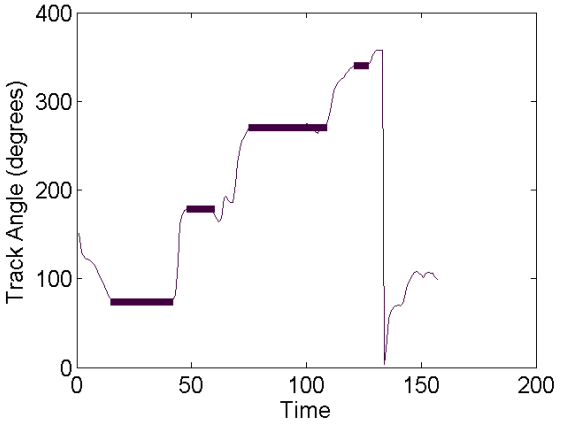

Fig. 3. ODSS results (tolerance = +/- 4 degrees, minimum duration = 12s) superimposed on the track angle data seen in fig. 2. The symbol set can be expressed as the ordered set {73, 178, 270, 340}.

The ODSS approach is similar to Piecewise Constant Approximation (PCA) in that the real-numbered input data maps to a few real-numbered symbols, where these output values are unknown a priori. The difference lies in the fact that PCA fits a constant model to the whole data set, even to those sections that are not constant. Fig. 4 shows a 6-segmentation PCA that has been applied to the track angle data in fig. 2; ODSS and PCA agree about the central 4 segments, while the poorer-fitting 2 outer segments are not identified by ODSS.

The ODSS approach also relates to feature extraction such as [9], in that the feature sought (constant value) is established with prior knowledge of the data set, but can take any value.

Having obtained an ODSS representation for two separate journeys (p and q), a means of comparing the two journeys is required. This in turn depends on being able to compare a pair of symbols. We take one symbol from each journey and compare them using the tolerance, t. The symbols are considered to be a match if equation 1 holds true.

$$(p_i - q_j) \leq 2*t \tag{1}$$

A discrete similarity measure can then be used to compare a pair of journeys. We chose to use the Jaccard similarity coefficient (J). J takes a value in the range [0,1], where high J indicates a high degree of similarity, and vice versa. Equation 2 defines J for two sets A and B. We use equation 1 to work out which symbols are common to both sets.

$$J(A,B) = |A \cap B| / |A \cup B| \tag{2}$$

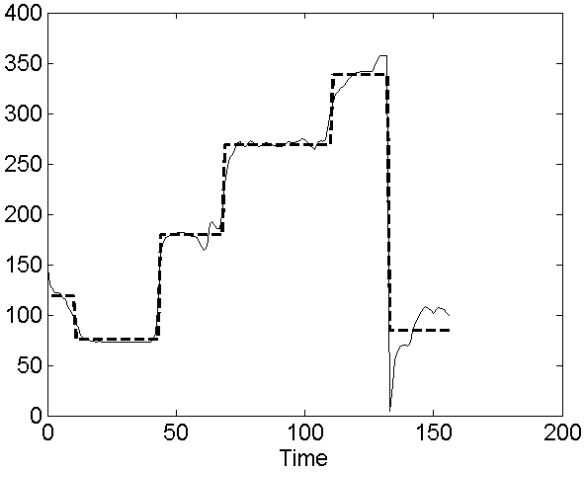

Fig. 4. The 6-segmentation produced by PCA, superimposed onto the track angle data of fig. 2

Our symbol sets are ordered sets, so the set notation in equation 2 is considered here to apply to ordered sets. This important distinction is illustrated in table 1 using the example sets: A={1,3,5,7}, B={7,1,5} (N.B. for the purposes of this example only, symbol equality corresponds to standard mathematical equality).

Given a similarity measure it is then possible to proceed to clustering. The similarity of all possible pairs of journeys is calculated, and used as the input to an agglomerative hierarchical clustering algorithm. The output is used to identify groups of similar journeys.

Table 1. Example showing how ordering affects J, where A={1,3,5,7}, B={7,1,5}

Ordering	$A \cap B$	$A \cup B$	$J = \lvert A \cap B \rvert / \lvert A \cup B \rvert$
No	{1,5,7}	{1,3,5,7}	3/4 = 0.75
Yes	{1,5}	{7,1,3,5,7}	2/5 = 0.4

5 Experiments

5.1 Experimental Scenario

We envisaged a scenario that would provide a framework for the experiments. A young child walks alone through the park (fig. 5) to school in the morning and is accompanied home by a group of friends in the afternoon. In the afternoon the child is allowed to take the direct route home through the park (C – A) that is obscured by trees. In the morning the parents prefer the child to take the longer route to school (A – B – C) that avoids the treed area by a large margin. In this scenario the parents would like to know about changes in their child's behaviour, but would like to be

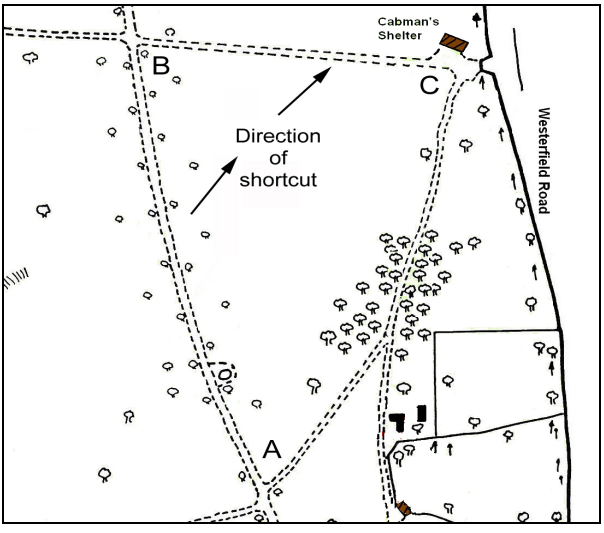

Fig. 5. Map of park where experimental data collected

flexible in setting boundaries for the child. For example, they don't mind if the child cuts the corner at B a bit to shorten their journey but they would like to know if the child is starting to deviate a significant amount from the prescribed route.

5.2 Data Collection and Preparation

Data was collected while walking around the park shown in fig. 5. The data was collected over two visits, with several journeys being walked on each occasion and stationary pauses used to demarcate the journeys. The routes travelled are defined in table 2.

Table 2. Definition of routes travelled

Name of route	Description
Permitted (P)	A – B – C
Return (R)	C – A
Forbidden (F)	A – C
Shortcut (S)	A – (cut the corner at B) – C. The degree to which the corner is cut varies from a few meters to a route running almost parallel with Forbidden

Data was downloaded from the PDA to the PC for processing. Journeys were isolated by identifying velocity = 0 (corresponding to pauses in data collection). All the data was used and there was no attempt to exclude any noisier data from the analysis.

5.3 Experimental Method and Results

Aims. Obtain clusters identifying the main routes travelled. First do this using ODSS, then perform a comparative analysis based on DTW. Investigate a working value for the dissimilarity measure in each case.

Clustering using ODSS. ODSS was applied to each journey (31 in total) in turn. A similarity matrix was then obtained by calculating J for every pair of journeys. This matrix was transformed into a dissimilarity matrix by subtracting all elements from one and then used as the input to the hierarchical agglomerative clustering *linkage* routine available in Matlab. The linkage method used was *average*, meaning that the inter-cluster distance between two clusters x and y is calculated to be the average distance between all pairs of objects in cluster x and cluster y. Results are presented graphically as a dendrogram in fig. 6.

ODSS Results. Dimensionality reduction is significant: for example, a journey consisting of 200 data points might be reduced to 5 symbols using this method. Each tick on the x-axis of the dendrogram represents a single journey; the tick label consists of a letter (e.g. 's' for shortcut) and a number indicating the journey instance (e.g. 's5' is the 5th shortcut). Please refer to table 2 for the meanings of other letters. There is no hard threshold that can divide similar from dissimilar journeys; the dendrogram can be used to ascertain a working value for the application in question. We begin by placing a threshold at 0.5 and consider that all cluster joins below this line are valid. This gives us: a cluster of 'r' journeys; a cluster of 'f' journeys (incorporating the most extreme shortcut that strongly resembles the forbidden route); a cluster of 'p' journeys (incorporating the least extreme shortcut that strongly resembles the permitted route); and several small clusters containing the remaining 's' journeys. Journeys r2 and f2 both join their respective clusters, but above the chosen threshold. Inspection of plots of the raw data shows that these two journeys are noisier than the remainder of the dataset, which explains the greater apparent dissimilarity.

Clustering using DTW. Pairs of continuous time series are used as the input to the DTW algorithm, which uses Euclidean distance to calculate the dissimilarity between the two series. In order to generate results for comparison with ODSS, DTW was used to calculate the distance between all pairs of journeys. This resulted in a dissimilarity matrix that could be clustered in the same way as the one produced by ODSS. The dendrogram is shown in fig. 7. The dissimilarity values have been rescaled so that the values vary between 0 and 1 for this set of results; this is to facilitate comparison with the ODSS results.

DTW Results. Again we select a threshold of 0.5 and then explore the results in the same way as the ODSS clustering: the 'r' journeys form a tight cluster; the 'f' journeys do the same, and pull in the most similar 's' journey too (s10). The next identifiable cluster contains two 'p' journeys plus a fairly similar 's' journey. The remainder of the clusters are more difficult to explain since the 's' and 'p' journeys show strong heterogeneous links; whereas we know that some of the 's' journeys are very different from the 'p' journeys, and very different from each other too.

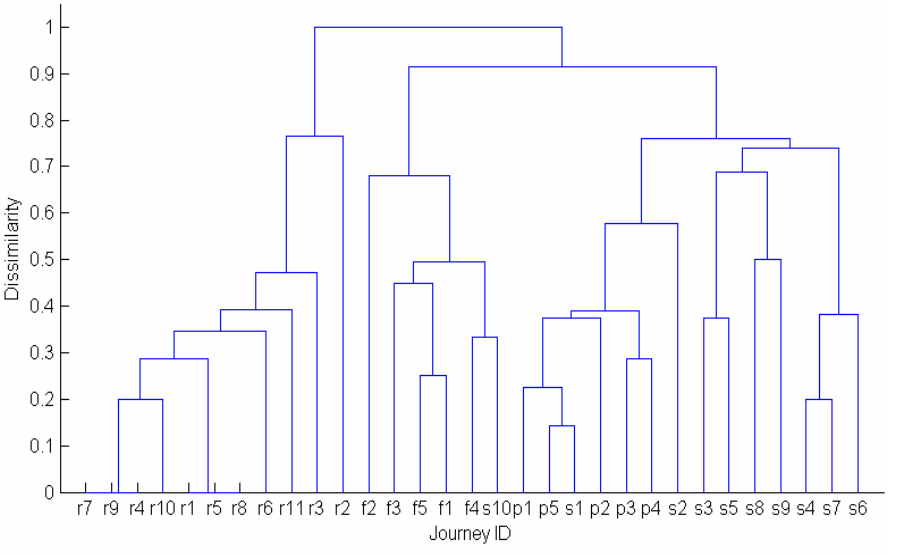

Fig. 6. Results of clustering using ODSS

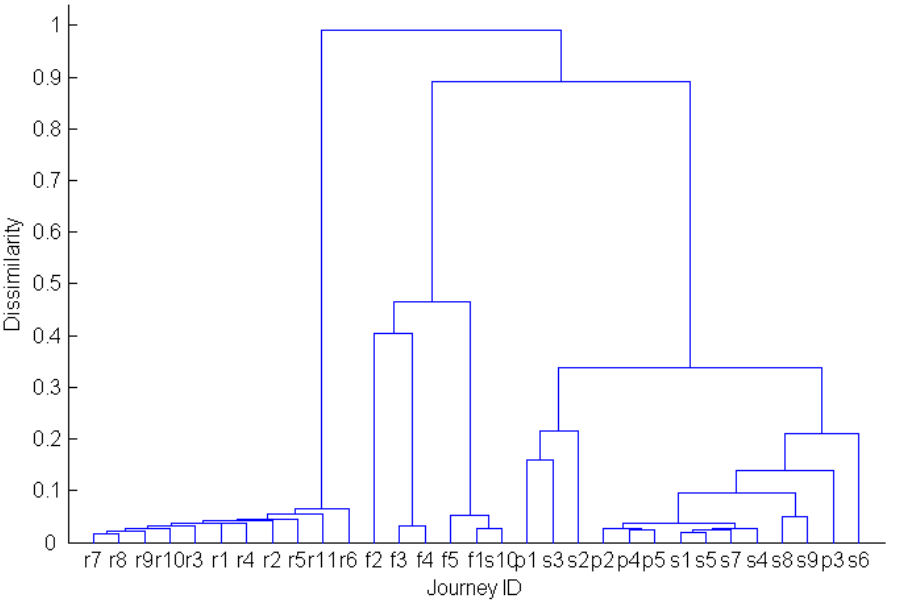

Fig. 7. Results of clustering using DTW

6 Conclusions and Further Work

The results show that ODSS is able to cluster a set of unlabelled routes; it does this at least as well as the equivalent clustering based on DTW for the real GPS data set used. Continuous data is transformed to a discrete representation, with dimensionality reduction of the order of 40:1 for this data set. Transformation can be performed online and requires no batch processing, nor prior definition of symbol sets. A well-established similarity measure, the Jaccard coefficient, can be applied to this discrete representation; this in turn allows hierarchical clustering to be performed. ODSS succeeds in identifying quite small changes in path followed, despite its low-resolution representation. When analysing the behaviour of vulnerable individuals, the ability to detect such small changes could be important.

ODSS forms the basis of our real-time recognition system, which compares partial (ongoing) behaviours with the model of previous behaviours, again using the Jaccard coefficient. A sliding window approach is used to adapt the methods described in this paper to a real-time context. This allows behaviour assessments to be carried out repeatedly, as symbols are extracted from the data stream. The significance of this is the ability to rapidly detect unusual behaviour without waiting, for example, for a journey to end. Initial real-time recognition results are encouraging and are the main focus of ongoing experiments.

References

1. Patterson, D., Fox, D., Kautz, H., Philipose, M.: Expressive, Tractable and Scalable Techniques for Modeling Activities of Daily Living UbiHealth 2003, Seattle, WA (2003)
2. Ashbrook, D., Starner, T.: Learning Significant Locations and Predicting User Movement with GPS. In: ISWC 2002, IEEE, Los Alamitos (2002)
3. Daw, C.S., Finney, C.E.A., Tracy, E.R: A Review of Symbolic Analysis of Experimental Data. Review of Scientific Instruments 74, 916–930 (2003)
4. Lin, J., Keogh, E., Lonardi, S., Chiu, B.A: Symbolic Representation of Time Series, with Implications for Streaming Algorithms. In: Proc. 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, pp. 2–11. ACM Press, New York (2003)
5. Vasquez, D., Large, F., Fraichard, T., Laugier, C.: Intentional Motion, Online Learning and Prediction. In: Proc. Int. Conf. on Field and Service Robotics, Port Douglas (AU) (2005)
6. Bingham, E., Gionis, A., Haiminen, N., Hiisila, H., Mannila, H., Terzi, E.: Segmentation and dimensionality reduction. In: SIAM Data Mining Conference (SDM) (2006)
7. Keogh, E., Pazzani, M.: An enhanced representation of time series which allows fast and accurate classification and relevance feedback. In: proceedings of the 4th Int'l Conference on Knowledge Discovery and Data Mining. New York, NY, pp. 239–241 (1998)
8. Xiaolei Li, X., Han, J., Kim, S.: Motion-Alert: Automatic Anomaly Detection in Massive Moving Objects. In: Proc. 2006 IEEE Int. Conf. on Intelligence and Security Informatics (ISI'06), San Diego, CA (2006)
9. Pratt, K., Fink, E.: Search for Patterns in Compressed Time Series. International Journal of Image and Graphics 2(1), 89–106 (2002)
10. Keogh, E., Kasetty, S.: Data Mining and Knowledge Discovery. 7(4), 349–371 (2003)

11. Keogh, E., Ratanamahatana, A.: Everything you know about Dynamic Time Warping is Wrong. In: 3rd Workshop on Mining Temporal and Sequential Data, in conjunction with 10th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining (KDD-2004), Seattle, WA (2004)
12. Sneath, P.H.A, Sokal, R.R: Numerical taxonomy; the principles and practice of numerical classification. W. H. Freeman, San Francisco (1973)
13. Palpanas, T., Vlachos, M., Keogh, E., Gunopulos, D., Truppel, W.: Online Amnesic Approximation of Streaming Time Series. In: ICDE. Boston, MA, USA (2004)
14. Lemire, D.: A Better Alternative to Piecewise Linear Segmentation. SIAM Data Mining 2007 (2007)