Aggregation and Comparison of Trajectories

Nirvana Meratnia
Department of Computer Science
University of Twente
P.O. Box 217, 7500 AE Enschede
The Netherlands
meratnia@cs.utwente.nl

Rolf A. de By
Intl. Inst. for Geo-information Science
& Earth Observation (ITC)
P.O. Box 6, 7500 AA Enschede
The Netherlands
deby@itc.nl

Categories and Subject Descriptors

H.2.8 [**Database applications**]: Spatial database and GIS; H.4 [**Information Systems**]: Information Storage and Retrieval; H.4 [**Information Systems**]: Database Management

General Terms

Algorithmics, Design, Theory

Keywords

moving object, trajectory, aggregation, pattern of movement, spatial similarity, spatiotemporal queries

ABSTRACT

Dealing with moving objects necessitates having available complete geographical traces for determining exact or possible locations that objects have had, have or will have. This is where trajectory determination plays an important role, and on which classification, aggregation and comparison methods must be built. The purpose of aggregation is to identify similar trajectories and to represent them by a single trajectory.

Although much work has been done in similarity measurements for time series data, they mainly deal with one dimensional time series. On the other hand, they are good for short time series and in absence of noise, which is definitely not the case for moving objects. This paper describes different approaches to aggregate similar trajectories.

1. INTRODUCTION

Moving object representation and computing has received a fair share of attention over recent years in the database community [27, 14, 29, 22, 1, 6, 26]. This is understandable as positioning technology is rapidly making its way into the consumer market, not only through the already ubiquitous cell phone but soon also through small, on-board positioning devices in many means of transport and types of portable equipment. It is thus to be expected that all these devices will start to generate an unprecedented data stream of time-stamped positions. Hence, the need for database support.

Databases have not accommodated such data in the past, as their design paradigm always was one of 'snapshot representation'. Their present support for *spatial* time series is at best rudimentary. *General* time series support is an active field of research, however [3, 7, 8, 11, 13].

The central theme of this paper concerns the process of turning raw data streams into object trajectories, and turning a collection of the latter into an *aggregate result*. Thus, we study the problem of aggregating moving object trajectories.

There are many applications of trajectory aggregation, all having to do with *traffic analysis*. Truck fleet behaviour analysis, commuter traffic in a city, passenger traffic in an airport, shopping behaviour in a mall, or even inside a single supermarket. The behaviour of all these 'moving objects' is already or will soon be traceable. The benefits of traffic analysis are obvious: truck theft protection, improving commuter traffic throughput or passenger throughput, and building shopping profiles for marketing purposes.

Aggregation of data means the determination of equivalence classes of data. Compared to the typical problem domain of aggregating data from an ordinary database, the aggregation of trajectories is harder, due to the inherent imprecision in spatiotemporal data. This has urged us to develop a notion of *trajectory similarity*: when are two trajectories sufficiently similar (for the analytic purposes at hand) to be classified in the same way? Trajectories may be similar in total or in part; they may approximately coincide in start and end points, yet be largely dissimilar internally.

There are a number of fundamental problems with defining and determining moving object trajectory similarity: differences in sequence length and sampling rates, as well as inherent spatial uncertainty.

The simplest approach to define similarity between two sequences is viewing them as vector and using Euclidean distance as similarity measure [2, 7, 9, 12, 13, 15, 16, 24, 25]. Such techniques cannot be easily applied to sequences having different length or sampling rate. Also, they are not effective in the presence of noise in the data, as is common in spatial time series.

To remedy some of these problems, [17, 20, 30] used a time warping technique to measure similarity, concentrating on the whole sequence. This technique suffers from performance degradation for long time sequences [21], as it has to

GIS'02 November 8–9, 2002, McLean, Virginia, USA

scan all the data sequences sequentially to perform a similarity search [18].

A similar technique is to find the longest common subsequence (LCSS) of two sequences and then define the distance using the length of this subsequence [3, 8]. The LCSS shows how well the two sequences can match one another if we are allowed to stretch without re-sequencing values. Due to using dynamic programming to find the longest common sequence, this technique is very expensive in terms of time and memory.

An interesting approach to represent a time series using the direction of the sequence at regular time intervals is presented in [23]. Ge and Smyth [11] present an interesting alternative approach for sequence similarity that is based on probabilistic matching. This addresses the issue of data uncertainty, not covered in any of the above.

An important consideration is that all the above work deals mainly with one-dimensional time series, while our trajectories are spatial time series. The most related papers to our work therefore are Bozkaya [4] and Vlachos [28]. The first discusses similarity measures for sequences of multidimensional points using an approach equivalent to LCSS. Their emphasis, however, is on feature vectors extracted from images.

A recent work that proposes a method to cluster trajectory data is due to Gaffney and Smyth [10]. Their method is a model-based approach that seems to suffer from a scalability problem. Also, it assumes a model of movement, which is not easy to determine in real data sets. In fact, it is what we would hope to find out from an aggregate query.

The aggregation approach that we propose in this paper does not necessarily require equal sampling rates or same speed for objects, and addresses spatial uncertainty. It is based on a parametric representation of trajectories deduced from raw data, and a subsequent mapping to a raster domain for aggregate purposes. This makes the approach rather insensitive to some of the problems mentioned above: differences in sequence length and sampling rate are largely hidden by the parametric representation, and spatial uncertainty is handled through appropriate choice of raster cell size.

Computationally, our algorithm can perform the aggregation faster as it does not compare pairs of trajectories, but compares trajectories, one at a time, with the reference raster. Finally, retrieving information and analysing grid structures are more efficient and faster. Well-known digital image processing tools can be used for such purposes.

2. AGGREGATION METHODS

The raw data of mobile object movements consists of position time series with different, and sometimes variable sampling rates. Such series cannot be directly compared, and require some form of data standardization. We defined various techniques in [19], and used one of them, a spline method, for the work reported in this paper.

This spline method provides a symbolic representation of a trajectory, which allows us to derive from it a re-discretized position time series, for each moving object, but now with synchronized and constant sampling rate. The choice of sampling rate is an important parameter of the method as it determines the precision of the data, as well as data and time complexity of the algorithms involved.

The remainder of this section reviews different aggregation methods, as well as different situations in which they are best applied. We first discuss naïve aggregation using thresholds, and argue why it is not very suitable for our problem. Then, we propose two other methods: *spatial unit aggregation*, and *spatiotemporal unit aggregation*.

2.1 Aggregation based on a distance threshold

A naïve technique of defining trajectory similarity would be to use shortest distances between re-discretized positions a and b of two trajectories A and B being compared. In the case of spatial dimension we would discard epoch information, in the case of spatiotemporal dimension, we would not discard epoch information. Positions a and b would be considered similar if they are within a pre-defined threshold distance. A similarity measure for A and B could subsequently be defined on the number of similar positions they have in common, or on the maximum length of a subseries of similar positions they have in common.

We have various arguments for not proceeding along these lines. The most important objection is that a trajectory similarity notion like discussed above is a non-transitive characteristic. This is illustrated in Figure 1. Such non-transitivity precludes the derivation of equivalence classes that we are after in aggregate queries. Essentially, we cannot define or derive a prototype representative of an equivalence class of trajectories based on distance thresholds.

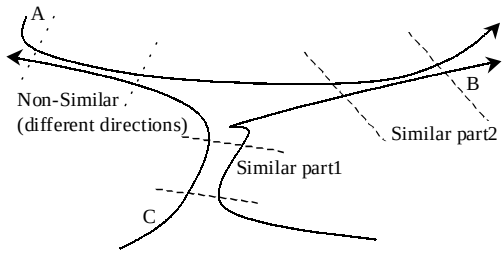


Figure 1: Non-transitivity of naïve trajectory similarity notion

Another strong argument is the time complexity involved in computations of distance threshold similarity. The notion demands pairwise comparisons of trajectories, thus is $O(N^2)$ where N is the number of objects, and each comparison of two trajectories — barring the possibly unlikely case of availability of spatial indices on object positions — would be $O(n^2)$, with n being the number of object positions within a trajectory.

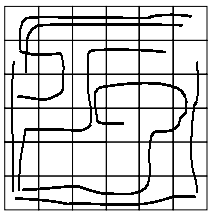
To overcome these problems, we propose two raster-based methods, in which a distance threshold notion is implicit in a fixation of the raster cell size. The fundamental idea in these methods is to determine similar overall trajectories on the basis of underlying raster cells.

2.2 Spatial unit aggregation

In this alternative method, the study area of interest is divided into homogeneous spatial units. Such a unit is defined as the 'minimum spatial extent (cell size) of interest in aggregating the trajectories'. Its size is *a priori* determined.

In principle, each cell is assigned the number of trajectory visits, i.e., the number of times that any object passes through it. If trajectories are widely scattered (resulting in many non-zero cell values), a raster is used as storage means.

Otherwise, non-zero elements together with their cell index can be stored in the form of $\langle \text{cell\_id}, \text{number\_of\_hits} \rangle$ sequences. Figure 2 shows a raster correspondence to a defined area of interest.



2	2	2	2	2	2
3	1	1	1	1	1
2	1	2	1	1	2
2	1	2	1	1	2
2	0	0	0	1	1
4	2	2	2	2	2

Figure 2: Aggregation based on spatial unit

The choice of cell size (γ [m]) is an important issue that has a direct effect on accuracy of the obtained results. This choice is affected by the maximum object speed $v_{\max}$ [ms^{-1}] and the chosen re-sampling rate (ρ [s^{-1}]). Rate and cell size must be picked so that a trajectory produces at least one hit in each cell that it visits. As a rule of thumb — to assure each relevant raster cell is 'hit' by a re-sampling point — parameters γ and ρ must be chosen such that

$$\frac{v_{\max}}{\rho} \ll \frac{\gamma}{\sqrt{2}}.$$

One should also remark that

- A too large cell size may result in information loss. It may be too large an aggregation unit to be useful.
- Some choices of re-sampling rate and cell size may amount to multiple hits per cell for the same trajectory. This may be due to low speed of the object or to a long path through the area that the cell represents. Though multiple hits are not entirely meaningless, it will be difficult to discriminate between these two cases, and we will generally assume, and algorithmically ensure, that multiple hits will count for just one hit.
- A too small cell size will produce large amounts of data, but also risks missing hitting cells when the resampling rate isn't high enough. In addition, the aggregation effect may become useless.

A	10	0	B
C	0	10	D

Figure 3: The number of trajectory hits may ra from 10 to 20.

Multiple cell sizes can be accommodated by applying image pyramid technique. In our case, this application, l ever, is non-trivial. This is because the relation between number of trajectory hits per cell and that of trajectory in its eight neighbor cells is not clear: they may relat the same or to different trajectory visits). An examp shown in Figure 3. There is no clear evidence indica the relation between hits in cells D and A: they can be same, partially the same or completely different trajec visits. Therefore, in such a small area, the overall number of hits may range from 10 to 20.

The above exemplifies the counting problem one encounters when attempting to assess traffic parameters for an arbitrary area on the basis of number of trajectory hits per cell.

Recall that raw positional time series for an object are turned into a symbolic stepwise spline representation, which in turn is transformed into a re-sampled (ρ) and synchronized positional time series. A straightforward linear scan of this series allows to assign the raster $Num\_hit(i,j)$ the number of visits ('hits') that this object has made to each cell (i,j). A visit is defined as an uninterrupted stay in (the areas of) the cell. An object can visit the same cell multiple times.

To remedy the above counting problem, knowing the object traffic between neighboring cells is a must. During the same linear scans as for Num_hit, we therefore also build up the rasters *Left*, *Right*, *Up* and *Down*, indicating, respectively, the number of subsequent object visits to a left, right, up and down neighbor cell. They are illustrated in Figure 4. These direction rasters, at essentially no complexity cost, provides piecewise, summary trajectory information. (Actually, two more rasters are built: *Start* and *End*, administering the number of trajectory starts and ends per raster cell.)

An important class of aggregate traffic queries identifies a contiguous area A within the study area, or a collection of mutually exclusive contiguous areas A_i, and requests to determine the number of object movements in A. The answer to this query can be defined as:

$$\begin{aligned} \mathit{Traffic}(A) = & \textstyle\sum_{i,j \in A} \mathit{Num\_hits}(i,j) \\ & - \textstyle\sum_{i,j \in A; i-1,j \in A} \mathit{Left}(i,j) - \sum_{i,j \in A; i+1,j \in A} \mathit{Right}(i,j) \\ & - \textstyle\sum_{i,j \in A; i,j+1 \in A} \mathit{Up}(i,j) - \sum_{i,j \in A; i,j-1 \in A} \mathit{Down}(i,j). \end{aligned}$$

To test our method, a simulation was carried out, which concerned about fifty objects travelling from different parts of a city towards the city center, located at the center of Figure 5(a). Cell values in Figure 5(a) are scaled from white (for zero hits) to black (for maximum number of hits). In Figure 5(b), the accompanying *Down* raster is illustrated.

One of the main advantages of our approach is that each raster can be populated for each object trajectory independently. The raster cells, so to say, provide an 'equivalence class mold'. The following reasons show the superiority of our raster-based approach to vector-based approach in aggregation procedure:

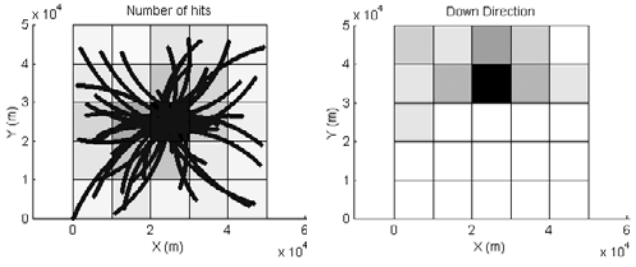


Figure 5: (a) Number of hits matrix. (b) Down direction matrix

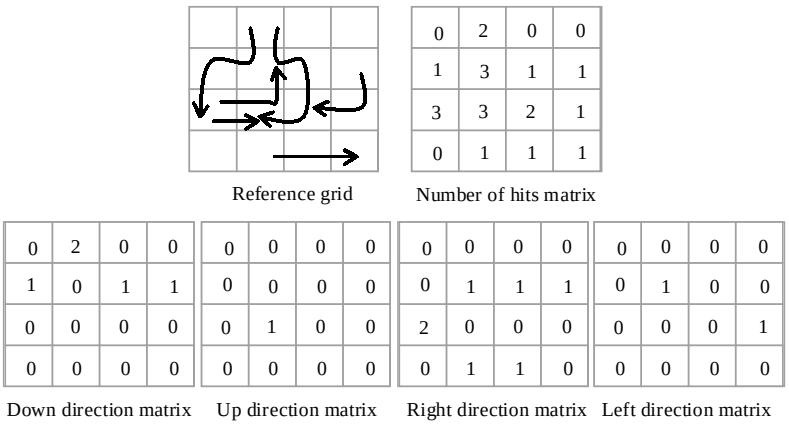


Figure 4: An example explaining aggregation based on number of hits and direction matrices

- Fewer computations: In the raster approach the number of computations is reduced to $O(nN)$, with flexibility of experimentation through the parameters γ and ρ.

- Independency to individual trajectories: In the raster approach, individual trajectories are overlaid on a reference grid and the number of hits per cell is computed easily (as overlay is performed easily in raster). One of the main advantages of using the raster approach comes from the fact that its reference frame (grid) is independent of individual trajectories and therefore, helps avoiding trajectory pair comparisons. In other words, comparisons are made between a trajectory and the reference grid rather than between pairs of trajectories.

- Easier generalization of information and immediate comprehension between reference grid and other spatial entities.

  Generalization of the information is easier in the raster approach, as there is an immediate relation between reference grid and other spatial entities. This allows using raster overlay methods to obtain information regarding any geo-referenced area of interest.

  The traffic formula above furthermore provides a basis for a pyramid approach to the data.

2.3 Spatiotemporal unit aggregation

Support for more data types helps to increase the domain of queries that can be answered, therefore in our next method, spatial data (as in the previous approach) are used together with time.

Different than in the previous method, the number of trajectories hits per cell specifically in a certain time interval is computed. The unit τ we use is a spatiotemporal unit, defined as the "minimum spatial and temporal extent of interest in aggregating the trajectories." It is user-defined, known *a priori* and may change depending on the application.

Having time included in the data schema, results in using a space-time cube or sequences of pairs ⟨voxel_id, number_of_hits⟩ instead. The latter option would be used in case of sparse rasters. Figure 6 shows an example of using such a technique.

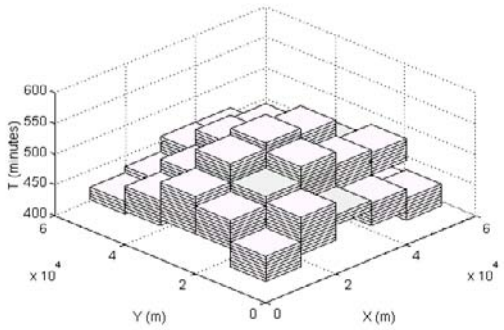


Figure 6: Aggregation based on spatiotemporal unit

Regarding spatiotemporal aggregation unit size, the same considerations as in the previous method apply. There is one additional rule of thumb, relating to the time unit τ. It should be chosen such that $1 \ll \rho\tau$, as $\rho\tau$ is a measure for hit number expectation per cell.

The method not only answers queries discussed before but also provides more information such as time. For instance, now, not only the most crowded area can be located but also the time instant at which this happened.

The last two methods are based on storing the number of hits per (spatiotemporal) cell. The traffic query alluded to earlier, would now also take a time interval T as input parameter, indicating the period during which object traffic is aggregated.

Observe that our generalization defines $Num\_hits(i, j, t)$ as the number of trajectory visits to the cell (i, j) during epoch t. Likewise, the rasters $Left(i, j, t)$ et cetera get their natural definitional extension. Although it is derivable information, we will find use for one additional raster $Stay(i, j, t)$, representing the number of objects present in cell (i, j) at epoch t and not leaving it during that epoch. Rasters *Start* and *End* are special cases of it.

The query $Traffic(A, T)$ now finds an answer in the formula:

$$Traffic(A,T) = \sum_{i,j \in A; t \in T} Num\_hits(i,j,t) - \sum_{i,j \in A; i-1,j \in A; t \in T} Left(i,j,t) - \sum_{i,j \in A; i+1,j \in A; t \in T} Right(i,j,t) - \sum_{i,j \in A; i,j+1 \in A; t \in T} Up(i,j,t) - \sum_{i,j \in A; i,j-1 \in A; t \in T} Down(i,j,t) - \sum_{i,j \in A; t \in T; t+1 \in T} Stay(i,j,t).$$

3. QUERIES

Moving object aggregate queries study behavior of groups of objects, not that of individual objects. Our first method allows various 'where'-type of aggregate queries, whereas the second method addresses 'where-and-when'-type queries. The latter has 'when' as an important special case, by setting area parameter A to be the total study area. We can say that all address issues of object traffic analysis.

We discussed these queries as being parameterized by the area A, or by a collection of mutually exclusive areas A_i. An appropriate choice of others parameters allows tuning the query formulation to the analysis needs. Different traffic arteries or other traffic hotspots can be identified, and for different time intervals.

The use of rasters like *Right* allows us to identify the direction of traffic in the spatial case. After all, by algebraic summation of the *Up* and *Down* matrices, we can obtain the trend in the up-down direction. Similarly, by algebraic summation of the *Right* and *Left* matrices, we can obtain the trend in the right-left direction. The angle of the trend is then determined. In the spatiotemporal case even the direction and throughput of traffic can be assessed. At advanced resolutions, in addition aggregate acceleration figures may be computable.

We have yet to experiment more fully with real traffic data to determine the sensitivity of our methods to choices of parameter values.

4. CONCLUSIONS

In this paper, we discussed three approaches towards trajectory aggregation and comparison as well as their strength in answering some typical queries. Each method has its strengths in answering certain types of query. We argued that (vector) trajectory aggregation benefits from taking a raster analysis approach, even though this at first may seem like an unnatural break of paradigm.

The raster-based aggregation methods show superiority to vector-based approaches due to the smaller computational overhead, the easier generalization of information, and the independency of individual objects.

The proposed methods have been tested only in a prototype environment, not in a full-fledged GIS environment and the question of how to implement these ideas there is what we currently work on. More experience is needed to validate the methods.

Otherwise, our interest is in more advanced raster analysis, general trend analysis and pyramid representation of the rasters in a traffic analysis context.

5. ACKNOWLEDGEMENTS

We are grateful to the anonymous reviewers of a draft of this paper for their insightful remarks and useful comments.

6. REFERENCES

[1] P. K. Agarwal, L. Arge, and J. Erickson. Indexing moving points. In *Symposium on Principles of Database Systems*, pp. 175–186, 2000.

[2] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In D. B. Lomet, editor, *Proc. 4th Intl. Conf. on Foundations of Data Organization and Algorithms*, volume 730 of *Lecture Notes in Computer Science*, pp. 69–86, Chicago, Il, USA, 13–15 Oct. 1993. Springer Verlag.

[3] B. Bollobás, G. Das, D. Gunopulos, and H. Mannila. Time-series similarity problems and well-separated geometric sets. In *Proc. 13th Annual Symp. on Computational Geometry*, pp. 454–456, Nice, France, 4–6 Jun. 1997. ACM Press.

[4] T. Bozkaya, N. Yazdani, and Z. M. Özsoyoglu. Matching and indexing sequences of different lengths. In F. Golshani and K. Makki, eds., *Proc. 6th Intl. Conf. on Information and Knowledge Management*, pp. 128–135, Las Vegas, Nv, USA, 10–14 Nov. 1997. ACM Press.

[5] J. Charlton. Understanding second moment of area. Lecture note for the Department of Building Engineering, UMIST University, 1997.

[6] Y.-J. Choi and C.-W. Chung. Selectivity estimation for spatiotemporal queries to moving objects. In *Proc. ACM SIGMOD Intl. Conf. on Management of Data*, pp. 440–451. ACM Press, 2002.

[7] K. K. W. Chu and M. H. Wong. Fast time-series searching with scaling and shifting. In V. Vianu and C. Papadimitriou, eds., *Proc. 18th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems*, pp. 237–248, Philadelphia, Pa, USA, 31 May – 2 Jun. 1999. ACM Press.

[8] G. Das, D. Gunopulos, and H. Mannila. Finding similar time series. In H. J. Komorowski and J. M. Zytkow, eds., *Proc. First European Symp. on Principles of Data Mining and Knowledge Discovery*, volume 1263 of *Lecture Notes in Computer Science*, pp. 88–100, Trondheim, Norway, 24–27 Jun. 1997. Springer Verlag.

[9] C. Faloutsos and K.-I. Lin. FastMap: A fast algorithm for indexing, data-mining and visualization of traditional and multimedia datasets. In M. Carey and D. Schneider, eds., *Proc. 1995 ACM SIGMOD Intl. Conf. on Management of Data*, pp. 163–174, San Jose, Ca, USA, 23–25 May 1995. ACM Press.

[10] S. Gaffney and P. Smyth. Trajectory clustering with mixtures of regression models. In U. Fayyad, S. Chaudhuri, and D. Madigan, eds., *Proc. 5th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pp. 63–72, San Diego, Ca, USA, 15–18 Aug. 1999. ACM Press.

[11] X. Ge and P. Smyth. Deformable Markov model templates for time-series pattern matching. In R. Ng, editor, *Proc. 6th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pp. 81–90. ACM Press, 2000.

[12] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In M. P. Atkinson, M. E. Orlowska, P. Valduriez, S. B. Zdonik, and M. L. Brodie, eds., *Proc. 25th Intl. Conf. on Very*

Large Data Bases, pp. 518–529, Edinburgh, Scotland, UK, 7–10 Sep. 1999. Morgan Kaufmann.

[13] D. Q. Goldin and P. C. Kanellakis. On similarity queries for time-series data: Constraint specification and implementation. In U. Montanari and F. Rossi, eds., *Proc. First Intl. Conf. on Principles and Practice of Constraint Programming*, volume 976 of *Lecture Notes in Computer Science*, pp. 137–153, Cassis, France, 19–22 Sep. 1995. Springer Verlag.

[14] R. H. Güting, M. H. Böhlen, M. Erwig, C. S. Jensen, M. Schneider, N. A. Lorentzos, and M. Vazirgiannis. A foundation for representing and querying moving objects. *ACM Trans. on Database Systems*, 25(1):1–42, 2000.

[15] T. Kahveci and A. K. Singh. Variable length queries for time series data. In *Proc. 17*th *Intl. Conf. on Data Engineering*, pp. 273–282. IEEE Computer Society, 2001.

[16] E. J. Keogh, K. Chakrabarti, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In W. G. Aref, editor, *Electronic Proc. 2001 ACM SIGMOD Intl. Conf. on Management of Data*, pp. 151–162, Santa Barbara, Ca, USA, 21–24 May 2001.

[17] E. J. Keogh and M. J. Pazzani. Scaling up dynamic time warping for data mining applications. In R. Ng, editor, *Proc. 6th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, pp. 258–289. ACM Press, 2000.

[18] S.-W. Kim, S. Park, and W. W. Chu. An index-based approach for similarity search supporting time warping in large sequence databases. In *Proc. 17*th *Intl. Conf. on Data Engineering*, pp. 607–614. IEEE Computer Society, 2001.

[19] N. Meratnia. Implementation of data structures and operations for moving objects modeling. Master's thesis, ITC, Enschede, The Netherlands, July 2000.

[20] S. Park, W. W. Chu, J. Yoon, and C. Hsu. Efficient searches for similar subsequences of different lengths in sequence databases. In *Proc. 16th Intl. Conf. on Data Engineering*, pp. 23–32, San Diego, Ca, USA, 28 Feb.–3 Mar. 2000. IEEE Computer Society.

[21] S. Park, D. Lee, and W. W. Chu. Fast retrieval of similar subsequences in long sequence databases. Technical Report UCLA-CS-TR990028, University of California, Los Angeles, Ca, USA, 1999.

[22] D. Pfoser, C. S. Jensen, and Y. Theodoridis. Novel approaches in query processing for moving object trajectories. In *The VLDB Journal*, pp. 395–406, 2000.

[23] Y. Qu, C. Wang, and X. S. Wang. Supporting fast search in time series for movement patterns in multiple scales. In G. Gardarin, J. C. French, N. Pissinou, K. Makki, and L. Bouganim, eds., *Proc. 1998 ACM CIKM Intl. Conf. on Information and Knowledge Management*, pp. 251–258, Bethesda, Md, USA, 3–7 Nov. 1998. ACM Press.

[24] D. Rafiei and A. Mendelzon. Similarity-based queries for time series data. In J. M. Peckman, S. Ram, and M. Franklin, eds., *Proc. 1997 ACM SIGMOD Intl. Conf. on Management of Data*, pp. 13–25, Tucson, Az, USA, 1997. ACM Press.

[25] D. Rafiei and A. Mendelzon. Querying time series data based on similarity. *IEEE Trans. on Knowledge and Data Engineering*, 12(5):675–693, Jan./Feb. 2000.

[26] S. Šaltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. In *Proc. 2000 ACM SIGMOD Intl. Conf. on Management of Data*, pp. 331–342. ACM Press, 2000.

[27] A. P. Sistla, O. Wolfson, S. Chamberlain, and S. Dao. Modeling and querying moving objects. In A. Gray and P.-Å. Larson, eds., *Proc. 13th Intl. Conf. on Data Engineering*, pp. 422–432. IEEE Computer Society, 7–11 Apr. 1997.

[28] M. Vlachos, G. Kollios, and D. Gunopulos. Discovering similar multidimensional trajectories. In *Proc. 18th Intl. Conf. on Data Engineering*, San Diego, Ca, USA, 26 Feb.–1 Mar. 2002. IEEE Computer Society.

[29] O. Wolfson, P. Sistla, B. Xu, J. Zhou, and S. Chamberlain. DOMINO: Databases fOr MovINg Objects tracking. In *Proc. 1999 ACM SIGMOD Intl. Conf. on Management of Data*, pp. 547–549, 1999.

[30] B.-K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary L_p norms. In A. E. Abbadi, M. L. Brodie, S. Chakravarthy, U. Dayal, N. Kamel, G. Schlageter, and K.-Y. Whang, eds., *Proc. 26th Intl. Conf. on Very Large Data Bases*, pp. 385–394, Cairo, Egypt, 10–14 Sep. 2000. Morgan Kaufmann.