Indexing the Trajectories of Moving Objects

Dieter Pfoser
Computer Technology Institute
11851 Athens, Hellas
pfoser@cti.gr

Abstract

Spatiotemporal applications offer a treasure trove of new types of data and queries. In this work, the focus is on a spatiotemporal sub-domain, namely the trajectories of moving point objects. We examine the issues posed by this type of data with respect to indexing and point out existing approaches and research directions. An important aspect of movement is the scenario in which it occurs. Three different scenarios, namely unconstrained movement, constrained movement, and movement in networks are used to categorize various indexing approaches. Each of these scenarios gives us different means to either simplify indexing or to improve the overall query processing performance.

1 Introduction

Several application areas contribute to the growing number of different types of spatiotemporal data. For example, we are currently experiencing rapid technological developments that promise widespread use of on-line mobile personal information appliances [12]. Mobility is a concern to many applications and services. One aspect of mobility is movement and thus the change of location. Applications in this context include emerging ones such as location-based services, but also "classical" ones such as fleet management and the optimal spatiotemporal distribution of vehicles [1].

Applications such as these warrant the study of indexing mobile objects. In particular, our interest is in *recording the movements of mobile objects, i.e., their trajectories, and indexing those for post-processing (e.g., data mining) purposes*. Thus, we will not concern ourselves with the indexing of the current positions and the predicted movements of objects such as treated in, e.g., [14, 15]. The size and shape of an object is often fixed and of little importance-only its position matters. Thus, the problem becomes one of recording the position of a moving object across time. The movement of an object may then be represented by a trajectory in the three dimensional space composed of two spatial dimensions and one time dimension [9].

Typically, access methods are developed having only the data and the queries in mind. However, for trajectory data we can also consider the constraints that the objects in their movement are subjected to. Specifically, we may distinguish among three movement scenarios, namely unconstrained movement (e.g., vessels at sea), constrained movement (e.g., pedestrians), and movement in transportation networks (e.g., trains and, typically, cars). As we will see in this work, the latter two scenarios allow us to optimize query processing techniques or to use simpler access methods to index the data.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

The outline of this paper is as follows. Section 2 defines the basic concepts and indentifies the challenges in indexing trajectories. Sections 3, 4, and 5 point out indexing approaches grouped according to the three movement scenarios. Finally, Section 6 concludes and points to research directions.

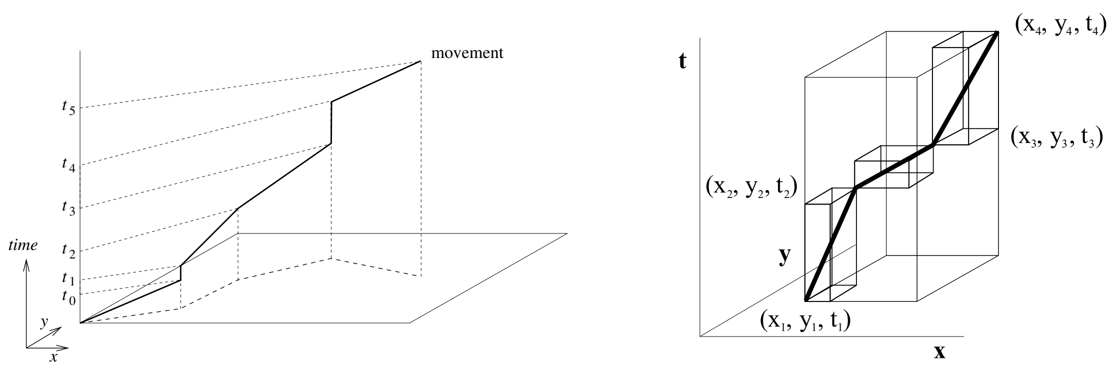
2 On Trajectories and Queries

The central question in this work is how we can ease the task of indexing point-object movements. This section gives a brief description of the data, the related queries, and the respective indexing challenges.

2.1 Trajectories

A trajectory is the data we obtain by recording the position of a moving point object across time. Consider the following application context. Optimizing transportation, especially in highly populated and thus congested areas, is a very challenging task that may be supported by an information system. A core application in this context is fleet management [1]. Vehicles equipped with GPS receivers transmit their positions to a central computer using either radio communication links or mobile phones. At the central site, the data is processed and utilized.

To record the movement of an object, we would have to know its position at all times, i.e., on a continuous basis. However, GPS and telecommunications technologies only allow us to sample an object's position, i.e., to obtain the position at discrete instances of time, such as every few seconds. A first approach to represent the movements of objects would be to simply store the position samples. This would mean that we could not answer queries about the objects' movements at times in-between those of the sampled positions. Rather, to obtain the entire movement, we have to interpolate. The simplest approach is to use linear interpolation, as opposed to other methods such as polynomial splines. The sampled positions then become the endpoints of line segments of polylines, and the movement of an object is represented by an entire polyline in 3D space. The solid line in Figure 1(a) represents the movement of an object. Space and time coordinates are combined to form a single coordinate system. The dashed line shows the projection of the movement into the 2D (spatial) plane [8, 9].



(a) (b)

Figure 1: Trajectories: (a) 2+1 Dimensional Space, (b) Approximating Trajectories Using MBBs

In classical spatial databases only position information is available. In our case, however, we have also derived information, e.g., speed, acceleration, traveled distance, etc. Information is derived from the combination of spatial and temporal data. Further, we do not just index collections of line segments—these are parts of larger, semantically meaningful objects, namely trajectories. These semantic properties of the data are reflected in the types of queries of interest for the data.

Query Type		Operation	Signature
Coordinate-Based Queries		overlap, inside, etc.	range
Trajectory -Based Queries	Topological Queries	enter, leave, cross, bypass	range $\times$ {segments} $\rightarrow$ {segments}
	Navigational Queries	traveled distance, covered area (top or average), speed, heading, parked	{segments} $\rightarrow$ int {segments} $\rightarrow$ real {segments} $\rightarrow$ bool

Table 1: Types of Spatiotemporal Queries

2.2 Queries

The queries of interest for trajectories comprise adapted spatial queries, e.g., range queries of the form “find all objects within a given area at some time during a given time interval” and queries with no spatial counterparts. Here, the so-called trajectory-based queries are classified in *topological* queries, which involve the entire movement of an object (enter, leave, cross, and bypass), and *navigational* queries, which involve derived information, such as speed and heading. Table 1 summarizes the query types. The “Operation” column lists the operations used for several query types, and the “Signature” column presents the types involved. For example, a coordinate-based query uses the inside operation to determine the segments within the specified range. The notation {segments} simply refers to a set, it does not capture whether this set constitutes one or more trajectories. It is important to be able to extract information related to (partial) trajectories, e.g., “What were the trajectories of objects after they left Tucson between 7 a.m. and 8 a.m. today, in the next hour?” This type of query is referred to as *combined* query, since we first have to select the respective trajectories (“...left Tucson between 7 a.m. and 8 a.m. today...”) and subsequently the respective portions (“...in the next hour?”). More details on trajectory-related queries can be found in [8, 11].

2.3 Indexing Fundamentals

Trajectories are three-dimensional and can be indexed using spatial access methods. However, there are difficulties. Trajectories are decomposed into their constituent line segments, which are then indexed. The use of the R-tree [2] is illustrated in Figure 1(b). The R-tree approximates the data objects by Minimum Bounding Boxes (MBBs), here three-dimensional intervals. Approximating segments using MBBs proves to be inefficient. Figure 1(b) shows that we introduce large amounts of “dead space,” meaning that the MBBs cover large parts of space, whereas the actual space occupied by the trajectory is small. This leads to high overlap and consequently to a small discrimination capability of the index structure.

Other trajectory indexing problems include trajectory preservation and skewed data growth. As for the first problem, spatial indices tend to group segments into nodes according to spatial proximity. However, in the case of trajectories, it is beneficial to some queries if the index preserves trajectories, i.e., to group segments according to their trajectory and only then according to proximity (cf. [11]). The second problem refers to the fact that trajectory data grows mostly in the temporal dimension. The spatial dimensions are fixed, e.g., the city limits. Exploiting this property of the data can further increase query performance.

2.4 Movement Scenarios

Depending on the particular objects and applications under consideration, the movements of objects may be subject to constraints. Specifically, we may distinguish among three movement scenarios, namely unconstrained movement (e.g., vessels at sea), constrained movement (e.g., pedestrians), and movement in transportation networks (e.g., trains and, typically, cars). Unconstrained movement is the scenario mostly asserted in work on spatiotemporal access methods. However, it hardly represents reality. Constrained movement and movement in networks represent similar movement scenarios. The former assumes that there exist spatial objects that constrain the movement, e.g., when considering the movement of cars, houses, lakes, parks, etc. Movement in networks is then merely an abstraction of constrained movement. Here, one is only interested in positions of objects with respect to the network and not with respect to a two-dimensional reference system. For example, we may expect that many applications will be interested only in the positions of cars with respect to the road network, rather than in their absolute coordinates. The movement effectively occurs in a different space than for the first two scenarios.

Section 2.3 illustrated the indexing challenges related to trajectories. In exploiting auxiliary information such as infrastructure and networks, we can improve query performance and simplify indexing. The following three sections explore indexing and query processing approaches in relation to the three movement scenarios.

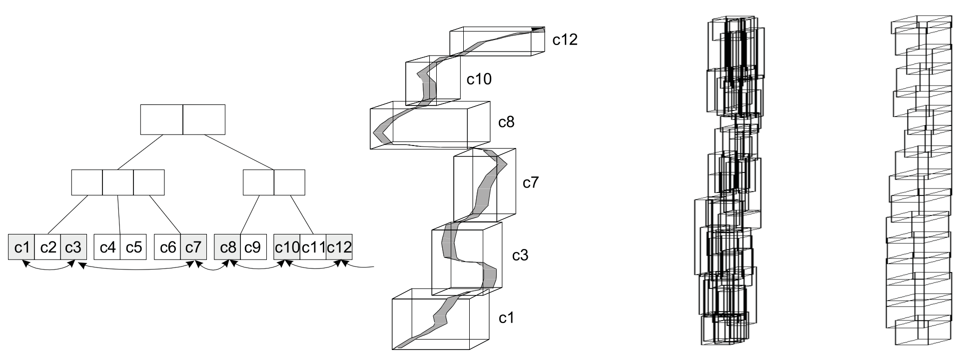
3 On Indexing Unconstrained Movement

Unconstrained movement is conceptually the simplest case for trajectory indexing. In the following, we describe several access methods that are tailored to the requirements of the data and the queries.

3.1 The TB-tree

The TB-tree [11] is an access method that considers the particularities of the data, namely trajectory preservation and temporal growth, and aims at efficiently processing related types of queries.

An underlying assumption for the R-tree is that all inserted geometries are independent. In our context this translates to all line segments being independent. However, line segments are part of trajectories and the R-tree only implicitly maintains this knowledge. With the TB-tree, we aim for an access method that strictly preserves trajectories. As a consequence, each leaf node in the index should only contain segments belonging to one trajectory. Thus, the index becomes actually a *trajectory bundle*. This approach is only feasible by making some concessions to the most important R-tree property, namely node overlap, or spatial discrimination. As a drawback, spatially close line segments from different trajectories will be stored in different nodes. This, in turn, increases the node overlap, decreases the space discrimination, and, thus, increases the classical range query cost. Thus, the TB-tree trades space discrimination for trajectory preservation.



(a) (b) (c)

Figure 2: TB-Tree: (a) Structure, Non-Leaf Level MBB Structure (B) R-Tree and (c) TB-Tree

Employing the above constraints, the TB-tree structure consists of a set of leaf nodes, each containing a partial trajectory, organized in a tree hierarchy. For query processing, it is necessary to be able to retrieve segments based on their trajectory identifier. Thus, leaf nodes containing segments of the same trajectory are linked together by a doubly-linked list data structure. This preserves trajectory evolution and overall improves especially the performance of trajectory-based and combined queries. Figure 2(a) shows a part of a TB-tree structure and a trajectory illustrating the overall data structure. For clarity, the trajectory is drawn as a band rather then a line. The trajectory symbolized by the grey band is fragmented across six nodes, c1, c3, etc. In the TB-tree these leaf nodes are connected through the doubly-linked list.

Figures 2(b) and (c) give an impression of the quality of the non-leaf level arrangement of bounding boxes in the R- and the TB-tree, respectively. The R-tree grouping is dominated by purely spatial characteristics such as proximity, ignoring the temporal growth of the data. The TB-tree structure is dominated by trajectory preservation. Since the data is inserted with a growing time horizon, the MBBs exhibit a temporal clustering.

The TB-tree structure is not the only approach to the indexing of trajectories. The following section gives a brief overview of what other techniques exist.

3.2 Other Approaches

Nasciemento et al. [7] adopt the trajectory model as described in Section 2.1 and investigate the suitability of multidimensional access methods for trajectory data. They compare the performance of various R-tree versions for different range-query loads.

Hadjileftheriou et al. [6] define an approach to reduce the dead space introduced by MBB approximations of trajectories (cf. Section 2.3) by introducing "artificial object updates." They effectively manipulate the partitioning of a trajectory into segments. A partially persistent tree structure [4, 16] is used to index the data. This approach generalizes previous work [3] in which it was assumed that the objects move with a linear function of time, whereas in [6] more complex functions are permitted.

Porkaew et al. [13, 5] examine the indexing of trajectories in native space (cf. Section 2.1) vs. parametric space. In parametric space, segments of trajectories are represented in terms of a location and a motion vector. In their experimental evaluation, the authors use an R-tree structure as an index for both representations.

Other works further propose access methods that go beyond trajectories towards the indexing of moving spatial shapes in general. These approaches however do not always consider continuous but only discrete changes, e.g., Tzouramanis et al. [17] employ overlapping linear quad-trees to index discretely changing spatial objects.

Tao and Papadias [16] propose a method called MV3R-tree to index the past locations of moving shapes. Their method combines multi-version B-trees and 3D R-trees to process timestamp and interval queries.

4 On Constrained Movement

We now assert that the movement of objects is constrained by elements termed *infrastructure*.

In terms of data, infrastructure represents "black-out" areas for movement and, thus, there are no objects and trajectories where there are infrastructure elements. However, in the index, approximations of the data are used, which introduce dead space. Consequently, the areas covered by infrastructure are not empty. This will incur unnecessary search in the index as well as produce a certain number of falsely reported answers, which must subsequently be eliminated. Both lead to extra I/O operations.

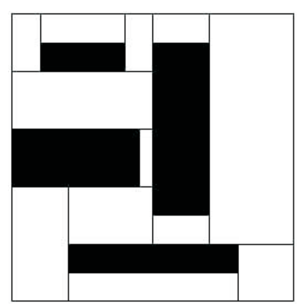


Figure 3: A Query Window Segmentation Example

To eliminate this extra I/O, we can use infrastructure in a pre-processing step, the idea being to not look for objects where there cannot be any? The strategy we choose is to query the infrastructure to save on querying the trajectory data. Overall, this will turn out to be favorable, since the number of infrastructure elements can be assumed to be very small compared to the trajectory data. Further, the trajectory data is growing with time, whereas the infrastructure data remains more or less constant.

The general principle is to decompose a given query window based on the infrastructure contained in it. The intuition is to segment the parts of the query window not occupied by infrastructure into well-shaped rectangles, i.e., as square as possible. In Figure 3, few but large infrastructure elements are shown as black rectangles, the possible outcome of such a segmentation process is shown as white rectangles.

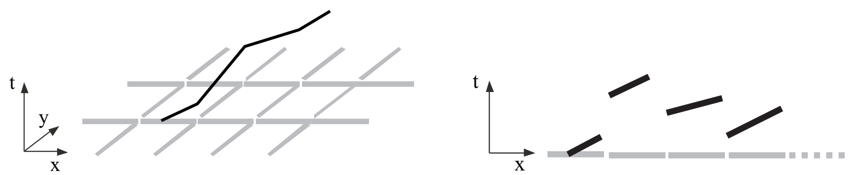
The query windows resulting from this segmentation (instead of the large query window ranging over infrastructure) are subsequently used to query the trajectory data index. In [10] the conditions under which this approach is favorable are established.

5 On Movement in Networks

In many applications, movement occurs in a network. When dealing with network-constrained movement, one is not interested in spatial extents, e.g., the thickness of the road, or the absolute position of the object in terms of its (x, y)-coordinates, but rather in relative positions with respect to the network, e.g., kilometer 21 of Highway 101.

The space defined by a network is quite different from the Euclidean space that the network is embedded into. Intuitively, the dimensionality of the networked-constrained space is lower than that of the space it is embedded in. In the literature, the term 1.5 dimensional has been used.

Modeling movement with respect to a network simplifies the trajectory data obtained. The two spatial dimensions are essentially reduced to one. Figure 4 illustrates this principle by showing the same trajectory in a three-dimensional and a two-dimensional space. A two-dimensional network is reduced to a set of one-dimensional segments, and the trajectory data is mapped accordingly.



(a) (b)

Figure 4: Network Movement: Trajectories in (a) 3D and (b) 2D Space

Lowering the dimensionality of the data reduces the indexing challenge considerably. Off-the-shelf database management systems typically do not offer three-dimensional indexes and thus do not contend with trajectories. Although it is desirable to design new access methods for new types of data, it may not be attractive in the short or medium term. For example, it took a dozen years before the R-tree found its way into some commercial database products. Depending on the type of data, it can be beneficial to use existing access methods by transforming the data. Considering movement in a network is such a transformation. We can use a simple access method such as a two-dimensional R-tree and this, in turn, allows for an easy integration of the new type of data into commercial database management systems.

6 Conclusions and Future Work

Spatiotemporal data is emerging from a broad range of applications. In this work, we present selected methods for the indexing of trajectories, a type of data that stems from recording the movement of point objects. The existing approaches are grouped according to *three movement scenarios*, constrained movement, unconstrained movement, and movement in networks. Each of these scenarios can aid the processing of spatiotemporal queries in different ways. Unconstrained movement is the typical showcase for the definition of new access methods. Constrained movement allows us to reduce the extent of query windows. Movement in networks reduces the dimensionality of the data and thus of the index.

Directions for future work can either be to define more efficient and/or more specialized access methods, or to satisfy existing needs arising from real applications [1]. This can be achieved by trying to handle spatiotemporal data with our current knowledge in connection with the means available from off-the-shelf database management systems [18].

Acknowledgements

The author would like to thank his co-authors on this subject, Christian S. Jensen and Yannis Theodoridis. Part of this research was supported by the CHOROCHRONOS project, funded by the European Commission DG XII, contract no. ERBFMRX-CT96-0056, and the Nykredit Corporation.

References

[1] V. Delis, D. Pfoser, Y. Theodoridis, and N. Tryfona. Movement mining in fleet management applications. Project Proposal, General Secretariat of Science and Technology, Athens, Hellas, 2001.

[2] A. Guttman. R-trees: a dynamic index structure for spatial searching. In *Proc. ACM SIGMOD*, pp. 47–57, 1984.

[3] G. Kollios, D. Gunopulos, V. Tsotras, A. Delis, and M. Hadjieleftheriou. Indexing animated objects using spatio-temporal access methods. *IEEE TKDE*, 13(5):758–777, 2001.

[4] A. Kumar, V. Tsotras, and C. Faloutsos. Designing access methods for bitemporal databases. *IEEE TKDE*, 10(1):1–20, 1998.

[5] I. Lazaridis, K. Porkaew, and S. Mehrotra. Dynamic queries over mobile objects. In *Proc. EDBT*, pp. 269–286, 2002.

[6] Hadjieleftheriou M, G. Kollios, V. J. Tsotras, and D. Gunopulos. Efficient indexing of spatiotemporal objects. In *Proc. EDBT*, pp. 251–268, 2002.

[7] M. Nascimento, J. R. O. Silva, and Y. Theodoridis. Evaluation of access structures for discretely moving points. In *Proc. of the International Workshop on Spatio-Temporal Database Management*, pp. 171–188, 1999.

[8] D. Pfoser. *Issues in the Management of Moving Point Objects*. Ph.D. thesis, Department of Computer Science, Aalborg University, Denmark, 2000.

[9] D. Pfoser and C. S. Jensen. Capturing the uncertainty of moving-object representations. In *Proc. of the 6th International Symposium on Spatial Databases*, pp. 111–132, 1999.

[10] D. Pfoser and C. S. Jensen. Querying the trajectories of on-line mobile objects. In *Proc. of the 2nd ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pp. 66–73, 2001.

[11] D. Pfoser, C. S. Jensen, and Y. Theodoridis. Novel approaches to the indexing of moving object trajectories. In *Proc. VLDB*, pp. 395–406, 2000.

[12] E. Pitoura. DB-Globe: an IST-FET research project on global computing. European Commission DG XII, contract no. IST-2001-32645, http://softsys.cs.uoi.gr/dbglobe/, 2002.

[13] K. Porkaew, I. Lazaridis, and S. Mehrotra. Querying mobile objects in spatio-temporal databases. In *Proc. of the 7th International Symposium on Advances in Spatial and Temporal Databases*, pp. 55–78, 2001.

[14] S. Saltenis and C. S. Jensen. Indexing of moving objects for location-based services. In *Proc. ICDE*, pp. 463–472, 2002.

[15] S. Saltenis, C. S. Jensen, S. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. In *Proc. ACM SIGMOD*, pp. 331–342, 2000.

[16] Y. Tao and D. Papadias. Mv3R-tree: a spatiotemporal access method for timestamp and interval queries. In *Proc. VLDB*, pp. 431–440, 2001.

[17] T. Tzouramanis, M. Vassilakopoulos, and Y. Manolopoulos. Overlapping linear quadtrees and spatio-temporal query processing. *The Computer Journal*, 43(4):325–343, 2000.

[18] M. Vazirgiannis and O. Wolfson. A spatiotemporal model and language for moving objects on road networks. In *Proc. of the 7th International Symposium on Advances in Spatial and Temporal Databases*, pp. 20–35, 2001.