Moving Objects Information Management: The Database Challenge

(Vision Paper)

Ouri Wolfson

Department of Computer Science, University of Illinois, Chicago, IL, 60607

and Mobitrac Inc., Chicago, IL 60610

wolfson@cs.uic.edu

Abstract Miniaturization of computing devices, and advances in wireless communication and sensor technology are some of the forces that are propagating computing from the stationary desktop to the mobile outdoors. Some important classes of new applications that will be enabled by this revolutionary development include location-based services, tourist services, mobile electronic commerce, and digital battlefield. Some existing application classes that will benefit from the development include transportation and air traffic control, weather forecasting, emergency response, mobile resource management, and mobile workforce. Location management, i.e. the management of transient location information, is an enabling technology for all these applications. Location management is also a fundamental component of other technologies such as fly-through visualization, context awareness, augmented reality, cellular communication, and dynamic resource discovery.

In this paper we present our view of the important research issues in location management. These include modeling of location information, uncertainty management, spatio-temporal data access languages, indexing and scalability issues, data mining (including traffic and location prediction), location dissemination, privacy and security, location fusion and synchronization.

1. Introduction

In 1996, the Federal Communications Commission (FCC) mandated that all wireless carriers offer a 911 service with the ability to pinpoint the location of callers making emergency requests. This requirement is forcing wireless operators to roll out costly new infrastructure that provides location data about mobile devices. In part to facilitate the rollout of these services, in May 2000, the U.S. government stopped jamming the signals from global positioning system (GPS) satellites for use in civilian applications, dramatically improving the accuracy of GPS-based location data to 5-50 meters.

As prices of basic enabling equipment like smart cell phones, hand helds, wireless modems, and GPS devices and services continue to drop rapidly, International Data Corp (IDC) predicts that the number of wireless subscribers worldwide will soar to 1.1 billion in 2003. Spurred by the combination of expensive new location-based infrastructure and an enormous market of mobile users, companies will roll out new wireless applications to re-coop their technology investments and increase customer loyalty and switching costs. These applications are collectively called location-based services.

Emerging commercial location-based services fall into one of the following two categories. First, Mobile Resource Management (MRM) applications that include

systems for mobile workforce management, automatic vehicle location, fleet management, logistics, transportation management and support (including air traffic control). These systems use location data combined with route schedules to track and manage service personnel or transportation systems. Call centers and dispatch operators can use these applications to notify customers of accurate arrival times, optimize personnel utilization, handle emergency requests, and adjust for external conditions like weather and traffic. Second, Location-aware Content Delivery services that use location data to tailor the information delivered to the mobile user in order to increase relevancy, for example delivering accurate driving directions, instant coupons to customers nearing a store, or nearest resource information like local restaurants, hospitals, ATM machines, or gas stations. Analyses Ltd. estimates that location based services will generate $18.5B in sales by 2006.

In addition to commercial systems, management of moving objects in location based systems arises in the military (see [6, 7]), in the context of the digital battlefield. In a military application one would like to ask queries such as "retrieve the helicopters that are scheduled to enter region R within the next 10 minutes".

Location management, i.e. the management of transient location information, is an enabling technology for all these applications. Location management is also a fundamental component of other technologies such as fly-through visualization (the visualized terrain changes continuously with the location of the user), context awareness (location of the user determines the content, format, or timing of information delivered), augmented reality (location of both the viewer and the viewed object determines the type of information delivered to viewer), and cellular communication.

Location management has been studied extensively in the cellular architecture context. The problem is as follows. In order to complete the connection to a cellular user u, the network has to know the cell id of u. Thus the network maintains a database of location records (key, cell-id), and it needs to support two types of operations: (1) Point query when a cellular user needs to be located in order to complete a call or send a message, e.g., find the current location (cell) of moving object with key 707-476-2276, and (2) Point update when a cellular user moves beyond the boundary of its current cell, e.g., update the current location (cell) of moving object with key 707-476-2276. The question addressed in the literature is how to distribute, replicate, and cache the database of location records, such that the two type of operations are executed as efficiently as possible. Related questions are how frequently to update, and how to search the database. Many papers have addressed this question, and two good surveys of the subject are [4, 14].

However, the location management problem is much broader. The main limitations of the cellular work are that the only relevant operations are point queries and updates that pertain to the current time, and they are only concerned with cell-resolution locations. For the applications we discussed, queries are often set oriented, location of a finer resolution is necessary, queries may pertain to the future or the past, and triggers are often more important than queries. Some examples of queries/triggers are: during the past year, how many times was bus#5 late by more than 10 minutes at some station (past query); send me message when a helicopter is in a given geographic area (trigger); retrieve the trucks that will reach their destination within the next 20 minutes (set oriented future query).

In terms of MRM software development, the current approach is to build a separate, independent location management component for each application.

However, this results in significant complexity and duplication of efforts, in the same sense that that data management functionality was duplicated before the development of Database Management Systems. To continue the analogy, we need to develop location management technology that addresses the common requirements, and serves as a development platform in the same sense that DBMS technology extracted concurrency control, recovery, query language and query processing, and serves as a platform for inventory and personnel application development. And this is the objective of our DOMINO project (see [20]), and our venture-funded startup company called Mobitrac (www.mobitrac.com). In this paper we describe the main research challenges in building a general purpose location management system.

The rest of the paper is organized as follows. In section 2, we discuss modeling and spatio-temporal operators. In section 3, we discuss uncertainty in location management, and in section 4, we discuss location management in a distributed/mobile environment. In section 5, we discuss location and traffic prediction by data mining, and in section 6 we discuss indexing. In section 7 we discuss other issues, particularly privacy and location fusion. In section 8 we conclude the paper.

2. Location Modeling and Linguistic Issues

In this section we first outline a naïve solution and outline its drawbacks (subsection 2.1), then we describe a more sophisticated model that we proposed (subsection 2.2) and explain how it addresses some of the drawbacks. In subsection 2.3 we present a possible set of spatio-temporal operators that capture uncertainty, and can be used for querying location information.

2.1 A Naïve Solution and Its Drawbacks

A fundamental capability of location management is modeling of transient location information, particularly the location of mobile devices such as cell phones, personal digital assistants, laptops, etc. These devices are carried by people, or mounted on moving objects such as vehicles, aircraft, or vessels. The location information is updated by positioning technologies. Examples of such technologies include 1) GPS (that is transmitted from the device to the location server via a wireless network), 2) network based positioning that computes the location by triangulation of transmission towers, 3) fixed sensors in the environment (e.g. at a toll booth) that identify the moving object, and 4) cell-id that identifies the cell in which the moving object is located (a low resolution method).

A straightforward approach that is used by existing industrial applications such as fleet management and Automatic Vehicle Location (AVL), is to model the location as follows. For each moving object, a location-time point of the form (l, t) is generated periodically, indicating that the object is at location l at time t. l may be a coordinate pair (x,y), or a cell-id. The point is stored in a database managed by a Database Management System (DBMS), and SQL is used to retrieve the location information.

This method is called point-location management, and it has several critical drawbacks. First, the method does not enable interpolation or extrapolation. For example, assume that a user needs to know which police officers were within one mile from the location of an emergency that occurred at 3pm. This information can only be retrieved for the moving objects that happened to generate a location update at 3pm. If an object did not generate an update at 3pm, then its whereabouts at that time are unknown. The problem is even more severe for extrapolation, i.e. if a future location is requested; for example, which field service employees will be closest to a

customer location at 5pm? This query cannot be answered by the point-location method, even though the future location of service personnel can be estimated based on current work schedules.

The second problem of the point-location method is that it leads to a critical precision/resource trade-off. An accurate picture of the precise location of moving objects would require frequent location updates that consume precious resources such as bandwidth and processing power.

Finally, a third problem of this method is that it leads to cumbersome and inefficient software development. Specifically, location based services will require the development of a vast array of new software applications. Doing so on top of existing DBMS technology has several drawbacks. First, existing DBMS's are not well equipped to handle continuously changing data, such as the location of moving objects. The reason for this is that in databases, data is assumed to be constant unless it is explicitly modified. For example, if the salary field is 60K, then this salary is assumed to hold (i.e. $60K is returned in response to queries) until explicitly updated. This constant-until-modified assumption does not hold for the location of moving objects which changes continuously. The second drawback is that location based services applications need to manage space and time information, whereas SQL is not designed and optimized for this type of queries and triggers. For example, the query "retrieve the vehicles that are inside region R always between 4pm and 5pm" would be very difficult to express in SQL. Finally, the location of a moving object is inherently imprecise because the database location of the object (i.e. the object-location stored in the database) cannot always be identical to the actual location of the object. This inherent uncertainty has various implications for database modeling, querying, and indexing. For example, there can be two different kinds of answers to queries, i.e. the set of objects that "may" satisfy the query, and the set that "must" satisfy the query. SQL semantics cannot account for this difference.

An interesting observation is that the point location management is used for two different cases. One in which the route of the moving object is known a priori (e.g. trucking fleet management, municipal transit), and the other in which such information is not available. For example, in location-aware advertising consumers usually cannot be assumed to provide their destination, and this is also the case for the enemy in digital battlefield applications. In other words, the information available a priori is not utilized for tracking, and it is not updated as a result of tracking.

2.2 Trajectory location management

In this section we outline our proposed model of a trajectory, explain how to construct it, and explain how it solves the problems associated with point location management. Let us observe that there are alternatives to the approach proposed here (see for example [9, 17]). If possible, our model makes use of a priori or inferred information about the destination of an object. For example, the destination can be inferred based on a motion pattern (e.g. the person travels to the office between 8am and 9am), or access to auxiliary information (e.g. a calendar may indicate a meeting at given time and address).

The method proposed is called trajectory location management. In this method we first obtain or estimate the source and destination of the moving object. For example, the object starts in New York City at 57th street at 8th Ave. at 7am and heads for Chicago at the intersection of Oak and State streets. Then, by using an electronic map geocoded with distance and travel-time information for every road section, a

trajectory is constructed.

Before defining the trajectory, let us define the format of an electronic map. An electronic map is a relation. Each tuple in the relation represents a city block, i.e. the road section in between two intersections, with the following attributes:

- *Polyline*: the block polyline given by a sequence of 2D *x,y* coordinates: $(x_1,y_1),(x_2,y_2),...,(x_n,y_n)$. Usually the block is a straight line segment, i.e. given by two *(x,y)* coordinates.
- *Fid*: The block id number

The following attributes are used for geocoding, i.e. translating between an (x,y) coordinate and an address such as "1030 North State St.":

- *L_f_add*: Left side from street number
- *L_t_add*: Left side to street number
- *R_f_add*: Right side from street number
- *R_t_add*: Right side to street number
- *Name*: street name
- *Type*: ST or AVE
- *Zipl*: Left side Zip code
- *Zipr*: Right side Zip code
- *Speed*: speed limit on this city block
- *One way*: a Boolean One way flag.

The following attributes are used for computing travel-time and travel-distance.

- *Meters*: length of the block in meters
- *Drive Time*: typical drive time from one end of the block to the other, in minutes

Such maps are provided by, among others, Geographic Data Technology Co. (www.geographic.com) An intersection of two streets is the endpoint of the four block-polylines. Thus each map is an undirected graph, with the tuples representing edges of the graph.

The route of a moving object O is specified by giving the starting address or (x,y) coordinate (start_point), the starting time, and the ending address or (x,y) coordinate (end_point). An external routine available in most existing Geographic Information Systems, and which we assume is given a priori, computes the shortest cost (distance or travel-time) path in the map graph. This path denoted P(O) is given as a sequence of blocks (edges), i.e. tuples of the map. Since P(O) is a path in the map graph, the endpoint of one block polyline is the beginning point of the next block polyline. Thus the whole route represented by P(O) is a polyline denoted L(O). For the purpose of processing spatiotemporal range queries, the only relevant attributes of the tuples in P(O) are Polyline and Drive-Time.

Given that the trip has a starting time, for each straight line segment on L(O), we can compute the time at which the object O will arrive to the point at the beginning of the segment (using the Drive-Time attribute). This is the certain-trajectory, or c-

trajectory. Intuitively, the c-trajectory gives the route of a moving object, along with the time at which the object will be at each point on the route. More formally, a c-trajectory is a sequence of straight-line (x_1, y_1, t_1), (x_2, y_2, t_2), ... ,(x_n, y_n, t_n) in 3-dimensional space. The c-trajectory means that when the object starts at a location having coordinates (x_1, y_1) at time t_1, it will move on a straight line at constant speed and will reach location (x_2, y_2) at time t_2, and then it will move on a straight line at constant speed and will reach location (x_3, y_3) at time t_3, etc. The c-trajectory is an approximation of the expected motion of the object in space and time. The reason it is only an approximation is that the object does not move in straight lines at constant speed. However, given enough straight lines, the approximation can be accurate up to an arbitrary precision. The number of line segments on the trajectory has an important implication on the performance and precision of queries and triggers. Specifically, the performance increases and the precision decrease as the number of line segments decreases. We adjust and fine-tune the number of line segments on each trajectory by using a method that has been studied in computer graphics, namely line simplification (see [2, 8]).

The c-trajectory is stored in the server database and in a computer on board the moving object. At any point in time t between t_i and t_{i+1} the server can compute the expected location of the moving object at time t. Observe that this technique solves the first problem associated with point location management. Namely, trajectory location management enables both, location interpolation and extrapolation. The server can compute the expected location of the moving object at any point in time between the start and end times of the trip. For example, if it is known that the object is at location (x_5, y_5) at 5pm and at location (x_6, y_6) at 6pm, and it moves in a straight line at constant speed between the two locations, then the location at 5:16pm can be computed at anytime, i.e. before 5:16 (extrapolation) or after (interpolation).

Finally, the trajectory (or the uncertain trajectory) is obtained by associating an uncertainty threshold u_i with the i'th line segment on the c-trajectory. The line segment together with the uncertainty threshold constitute an "agreement" between the moving object and the server. The agreement specifies the following. The moving object will update the server if and only if it deviates from its expected location according to the trajectory by u_i or more. How does the moving object compute the deviation at any point in time? Its computer receives a GPS update every two seconds, so it knows its actual location at any point in time. It has the trajectory, so by interpolation it can compute its expected location at any point in time. The deviation is simply the distance between the actual and the expected location. More formally, a trajectory is polyline (x_1, y_1, t_1, u_1), (x_2, y_2, t_2, u_2), ..., (x_n, y_n, t_n, u_n) in 4-dimensional space.

At the server, the trajectory is maintained by revising it according to location-updates from the moving object, and according to real-time traffic conditions obtained from traffic web sites. We have developed a traffic incident model, and a method of identifying the trajectories affected by a traffic incident. Observe that determining whether or not a trajectory is affected by a traffic incident is not a simple matter, and it requires prediction capabilities. For example, suppose that according to the current information Joe's van is scheduled to pass through highway section x twenty minutes from now, and suppose that a web site currently reports a traffic jam on highway section x. Will Joe's expected arrival time at his destination be affected by this? Clearly it depends on whether or not the jam will clear by the time Joe arrives at highway section x. We use historical information and a novel traffic model to make this prediction.

Observe that the agreement (namely the trajectory plus the uncertainty threshold) between the moving object and the server solves the second problem of point location management. Namely, the tradeoff between resource/bandwidth consumption and precision has been broken. In trajectory location management the location of a moving object can be computed with a high degree of precision, using a small number of location updates, or no updates at all. In particular, if the moving object is "on schedule", i.e., it does not deviate from its prescribed trajectory by more than the uncertainty threshold, then no resources are consumed for updates.

Finally, let us observe that a trajectory can be constructed based on past motion in which an object used the point location management. Namely, the trajectory can be constructed from a set of 3D points (x_1, y_1, t_1), (x_2, y_2, t_2), ... ,(x_n, y_n, t_n) that were transmitted by a moving object using the point location management method. One can simply connect the points along the shortest path on the map, and then associate an uncertainty u_i with line segment i. The uncertainty u_i can be bounded given the maximum speed of the object and the known times of the two GPS points immediately preceding and succeeding the i'th line segment (see [15]).

2.3 Data Access Operators

Finally, we propose to solve the third problem associated with point location management using a novel set of operators by which the database is accessed. The operators are used to query the database, and also to set triggers (or alerts) that are fired when interesting conditions are satisfied by the database (e.g. an object is expected to be late by more than one hour). The operators are designed to express when/where questions in an uncertain environment. They can be incorporated into the traditional SQL query language that has been widely adopted by commercial database systems. This means that one can ask queries and set triggers that combine the traditional database conditions with the new operators. This means, for example, that a dispatcher can ask a query such as: retrieve the service-personnel who have Qualification="dsl", and will be within 1 mile of 851 S. Morgan St. at 5pm. This also means that the operators can be combined using boolean operators such as and and or. An additional implication is that these operators/queries can be entered by a user on a client computer, and the same set of operators can be invoked from a program. The latter option enables development of complex spatial and temporal applications. Obviously the proposed set of operators is not exhaustive, but each operator in the set can on one hand be implemented efficiently, and on the other hand we believe that it is useful in a large class of applications. Although the operators are given here in textual form, but they can be implemented in point-and-click, drag-and-drop, graphical and visual form.

The new operators are divided into three classes, operators that pertain to a single trajectory, operators that pertain to the relationship of trajectories to fixed-location facilities or regions, and the relationship among multiple trajectories. These loosely correspond to point queries, range queries, and join queries, respectively, in traditional databases (see [10]). Each one of the immediately following subsections discusses one of these classes.

2.3.1 Some operators that analyze a single trajectory:

- *WHEN object o CLOSEST TO address x*. The operator returns a list of times at which the object passes by or stops at address *x*. Observe that there may be a list of times, since the object may visit or pass by the same location more than once. If the object never passes by or visits *x*, then the operator returns the time when the object passes by the closest location to *x* on its trajectory. This

operator is used, for example, when the customer at *x* needs to know when the technician will arrive at her location according to the current schedule.

- *VCR object o*. The operator "replays" the trajectory of object *o*. The replay can be done on a certain time-scale (e.g. a minute per second), and it can fast forward/rewind to a certain point in time.

2.3.2 Some operators for retrieving trajectories that stand in certain relationships to a region or a facility

- Each one of the operators in this class is a condition. The condition is satisfied by the objects that stand in a certain relationship (e.g. within distance *x*) to a fixed facility (i.e. a point on a map) or a region *R*, during *T*. Thus the conditions correspond to a spatio-temporal range query. Why then is there more than a single operator? The answer is threefold. First, since the expected location of an object changes continuously, one may ask for the objects that satisfy the condition *always* within *T*, or the objects that satisfy it *sometime* during *T*; similarly one can ask for the objects that satisfy the condition *somewhere* in R or *everywhere* in R. Second, there is an uncertainty associated with the location, and thus one can ask for the objects that *possibly* satisfy the condition, or the ones that *definitely* do so. Third, it turns out that the order in which the temporal quantifier is combined with the certainty quantifier important.

2.3.3 Some operators for identifying relationships between trajectories

Each one of the relationship-to-facilities operators can be applied as a relationship-between-trajectories operator. These are called join operators. For example:

- *Possibly-Within [distance d | travel-time t], Sometime in the time interval T.* The condition is satisfied by the pairs of trajectories which are within distance d or travel time t from each other, sometime in the time interval T. This operator is used, for example, in an air-traffic-control system that stores the trajectories of planes. We assume that, in contrast to the existing system in which planes fly on "highways in the sky", the new free-flight system has been implemented (see www.faa.gov/freeflight). The air traffic controller needs to know which planes are expected to be within distance *d* from each other, thus representing a safety hazard.

The opposite operator also applies. Specifically:

- *Possibly-Fartherthan [distance d | travel-time t], Sometime in the time interval T.*

3. Uncertainty management

The location of a moving object is inherently imprecise because, due to continuous motion and due to the fact that the location cannot be reliably obtained at any point in time, the database location (i.e. the object-location stored in the database) cannot always be identical to the actual location of the object. Systems that do not manage this uncertainty delegate to the user the responsibility of understanding and taking into consideration its implications. Furthermore, even if the database is precise, applications that access this database may need query-imprecision support. For example, a user may ask if there is a traffic jam on highway I90, where the notion of a traffic jam is imprecise.

The objective of uncertainty management is to assist the user in accounting for uncertainty, and in expressing imprecise queries/triggers. This objective has various implications for database modeling (in our model the uncertainty is part of the trajectory), querying (possibly and definitely operators), indexing, and resource consumption.

Assuming that one can control the amount of uncertainty in the system, how should it be determined? Obviously, lowering the uncertainty would come at a cost. For example, if a moving object transmits its location to a location-database every x minutes or every x miles, then lowering x would decrease the uncertainty in the system, but increase bandwidth consumption and location-update processing cost; and vice versa, increasing x would reduce the uncertainty, but increase resource consumption. Similarly, adjusting the uncertainty thresholds u_j in our trajectory model has the same tradeoffs concerning resource consumption.

Next we outline our cost based approach to quantify this tradeoff as a demonstration of a possible formalization of the problem (see [21]). The information cost of a trip has the following three components: deviation cost, uncertainty cost, and communication cost. Using these costs we define a function that represents the overall information cost of a trip, and define the optimal uncertainty threshold as the value that minimizes this function.

We believe that the method of defining an uncertainty threshold and communicating only values that exceed the threshold is an important paradigm that has applications beyond location management. Indeed the uncertainty threshold paradigm was used in the context of data warehousing (see [19]) and general sensors (see [12]).

4. Location dissemination in a distributed environment

It is often impractical or impossible to store the location database in a centralized location. This is the case, for example, in the cellular database discussed in the introduction, where a centralized architecture would create an intolerable performance problem. So the question is how to allocate, update and query trajectories in a geographically distributed environment. Another complication arises when the location database is not only distributed but also mobile. This is the case, for example, in a Mobile Ad-hoc Network (MANET). This is a system of mobile computers (or nodes) equipped with wireless broadcast transmitters and receivers which are used for communicating within the system. Such networks provide an attractive and inexpensive alternative to the cellular infrastructures when this infrastructure is unavailable (e.g. in remote and disaster areas), or inefficient, or too expensive to use (see [11]). Mobile Ad-hoc Networks are used to communicate among the nodes of a military unit, in rescue and disaster relief operations, in collaborative mobile data exchange (e.g. the set of attendees at a conference), and other "micro-networking" technologies (see [3]). In this case, the natural data allocation is for each moving object to store and maintain its own trajectory, and the problem is processing the queries discussed in section 2 with an acceptable delay, overhead and accuracy.

5. Location and traffic mining/prediction

In many mobile commerce applications the system does not have a priori information about the future motion of a customer. In other words, in contrast to the single enterprise systems we discussed so far, a customer does not provide her location to potential merchants that match her profile. For example, if at 8 A.M. it is

known that at 9 A.M. the customer will be close to a store that has a sale on merchandise that matches her profile, the system could transmit a coupon at 8 A.M.. This would allow the customer to plan a purchase stop. Location prediction is important in other applications such as wireless bandwidth allocation (in a cellular architecture, location prediction enables optimizing allocation of bandwidth to cells).

We have developed methods of motion prediction based on historical trajectories of moving objects. In other words, we are able to predict the location of a moving object at a certain future time. Our prediction methodology is based on the fact that often moving objects have some degree of regularity in their motion. That is, motion has a random part and a regular part, and the regular part has a periodic pattern (hourly, weekly, etc.). A typical example is the home-office-home pattern. If we are able to detect this pattern, then location prediction is relatively easy for rush hour. Therefore we decompose the motion prediction problem into two sub-problems: periodicity detection, and location prediction based on detected periodicity.

Periodicity detection seeks motion patterns. Assuming that we are given time-stamped sets of GPS points, the following are features of the patterns. First, patterns are partially periodic, i.e. sometimes only part of the motion repeats. For example, a person may usually travel from home to work along a fixed route between 7 A.M. and 8 A.M. every workday and back home between 5 P.M. and 6 P.M.. S/he may do other things and go other places during the rest of the day, and this constitutes the random part of the motion.

Second, the patterns are not necessarily repeated perfectly. For example, the home-to-work trajectory on one day may be different than on another day, due to different traffic conditions. Or, the person may decide to stay at home some workdays and thus miss certain periods.

Finally, the motion can have multiple periodic cycles. For example, the person may go fishing every Saturday and every other Sunday. In summary, our goal is to detect motion patterns that can be partially periodic, not perfectly repeated, and have multiple periodic cycles.

Location prediction in other environments and assuming other circumstances is also important. The ultimate challenge is to predict future location of a hostile moving object that is attempting to hide and deceive about its future location and destination. Location prediction is strongly related to another prediction problem, namely traffic prediction. In order to be able to predict the future location of a moving object, particularly the time of arrival at the destination, one has to be able to predict the traffic conditions that will be encountered en route. Work on traffic-conditions mining and prediction has begun (e.g. [22]), but much more remains to be done.

6. Indexing

Most of the database work in the area of Moving Objects was focused on indexing. In particular, the issue addressed is how to index a large number of 3-dimensional trajectories in order to perform efficient querying and updating of trajectories. Each one of the prevalent approaches is classified as primal (where each trajectory segment is represented as a line) or dual (where the trajectory segment is represented as a point). For example, references [16, 18] take the primal approach, and [1, 5, 13] take the dual approach.

Most existing works use synthetic data to verify performance. Much more

research work is necessary in order to verify the performance of existing and new indexing methods in realistic moving objects scenarios; for example, objects moving with realistic speeds and speed variations on road networks given in existing electronic maps. Furthermore, in seems that the dual methods are not suitable to road networks in which moving objects switch from one road to another, particularly if the trajectory is known a priori. More research is required to adapt the dual method for road networks. Another issue is to analyze in-memory indexing and compare them with brute force methods. For many applications trajectories are several tens of miles, and the number of moving objects are in the hundreds or thousands, which implies that the trajectories can be maintained in main memory, but frequent trajectory-updates and queries may still create a severe performance problem. Finally, most existing work propose and evaluate a particular indexing method, but much more work is required in comparing various methods.

7. Other Issues

Privacy is an issue that immediately arises as a major concern when people are presented with location based applications. This makes systems in which clients learn their location without centralized tracking (e.g. [23]) particularly appealing. In our (controversial) opinion, the privacy problem will be solved by a trusted third party approach, similarly to the way the approach solved the privacy problem in financial credit card transactions and telephone communication. In other words, we feel that financial transactions and telephone communication that are at least as privacy-sensitive as tracking, have been satisfactorily solved by trust of credit card and telephone companies.

An important research problem is the fusion and synchronization of location-information obtained from multiple sources (e.g. GPS, dead-reckoning, proximity sensors, scene analysis). Such fusion is important in order to increase the accuracy of location information and its availability in variable terrain conditions. One particular approach to the problem is introduced in [24], but much more work is necessary in this area.

8. Conclusion

We believe that the pervasive, wireless, mobile computing is a revolutionary development, and location based services and mobile resource management are some of the initial manifestations of this revolution. In this paper we focused on location management, which is an enabling technology for a variety of applications and technologies related to this revolution. We discussed the research issues that need to be addressed in order to make location management a plug-in component in these applications and technologies. The first of these issues are location modeling. We discussed the drawbacks of existing approaches, and we proposed the trajectory as a four dimensional piece-wise linear function that captures the essential aspects of the moving object location. These aspects are two-dimensional space, time, and uncertainty. We also proposed a set of operators to access a database of trajectories.

Another research issue is uncertainty and imprecision management and control. We also discussed location management in a distributed and/or mobile environment, and issues of data mining, particularly prediction of location and traffic patterns. Finally, we briefly discussed the existing and necessary work in the areas of indexing and location fusion.

References:

1. A.K. Agarwal, L. Arge, J. Ericskon: "Indexing moving points". *Proc. of ACMPODS 2000 conference*.
2. P.K. Agarwal and K. R. Varadarajan. Efficient Algorithms for Approximating Polygonal Chains. *Discrete Comput. Geom.*, 23:273-291(2000)
3. F. Bennett, D. Clarke, J. Evans, A. Hopper, A. Jones, and D. Leask, Piconet: Embedded Mobile Networking, *IEEE Personal Communications*, 4(5), October 1997.
4. A. Bhattacharya, S. K. Das, Lezi-Update: An Information-Theoretic Approcah to Track Mobile Users in PCS Networks, *Proceedings of the fifth ACM/IEEE International Conference on MobileComputing and Networking (MOBICOM99*), Seattle, WA, August, 1999.
5. H.D. Chon, D. Agrawal, A. El Abbadi: "Storage and Retrieval of moving objects" *in Proc. of Mobile Data Management 2001 conference, Springer Verlag Lecture Notes in Computer Science* No. 1987.
6. S. Chamberlain. Automated information distribution in bandwidth-constrained environments. *MILCON-94 conference*, 1994
7. S. Chamberlain. Model-based battle command: A paradigm whose time has come. *1995 Symposium on C2 Research & Technology, NDU*, June 1995
8. D.H. Douglas and T. K Peucker Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Canad, Cartog*. 10(2):112-122, Dec. 1973
9. Ralf Hartmut Guting, Michael H. Bohlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos. Markus Schneider, and Michalis Vazirgiannis.: A Foundation for Representing and Querying Moving Objects, in *ACM-Transactions on Database Systems Journal* (2000), 25(1), 1-42
10. H. Garcia-Molina, J. D. Ullman, J. Widom. *Database System Implementation*, Prentice Hall, Upper Saddle River, NJ.
11. Z. J. Has, Panel Report on Ad Hoc Networks - MILCOM'97*, Mobile Computing and Communications Review*, Vol. 2, No. 1, January 1998.
12. Y. Huang, R. Sloan, O. Wolfson, "Divergence Caching in Client-Server Architectures*", Proceedings of the third International Conference on Parallel and Distributed Information Systems (PDIS*), Austin, TX, Sept. 1994, pp. 131-139.]).
13. G. Kollios, D. Gunopulos, V.J. Tsotras: "On indexing moving objects". *ACM PODS 1999 conference*. ACM Press.
14. E. Pitoura, G. Samaras: "Locating Objects in Mobile Computing". *IEEE Transactions on Knowledge and Data Engineering*, Vol. 13, No. 4, July/August 2001
15. D. Pfoser, C.S. Jensen: "Capturing the uncertainty of moving objects representations". *Proc. of the 12 Intl. Conf. on Scientific and Statistical Database Management*, 2000. IEEE Computer Society.
16. S. Saltenis, C.S. Jensen, S.T. Leutenegger, M.A. Lopez: "Indexing the Positions of Continuously Moving Objects. *ACM SIGMOD Conference 2000*.
17. A. P. Sistla, O. Wolfson, S Chamberlain, and S. Dao: Modeling and Querying Moving Objects, In *Proc. of the International Conference on Data Engineering* (1997) pp. 422-432.
18. J. Tayeb, O. Ulusoy, and O Wolfson. A Quadtree-based dynamic attribute indexing method. *The computer Journal*, (41(3), 1998
19. C. Olston, J. Widom, Offering a Precision-Performance Tradeoff for Aggregation Queries over Replicated Data, *Twenty-Sixty International Conference on Very Large Data Bases (VLDB 2000),* Cairo, Egypt, September 2000.
20. O. Wolfson, P. Sistla, B. Xu, J. Zhou, S. Chamberlain, N. Rishe,Y. Yesha,

Tracking Moving Objects Using Database Technology in DOMINO*, Springer-Verlag Lecture Notes in Computer Science, number 1649, Proceedings of NGITS'99, The Fourth Workshop on Next Generation Information Technologies and Systems*, Zikhron-Yaakov, Israel, July 1999, pp. 112-119.

21. O. Wolfson, A. P. Sistla, S. Chamberlain, Yelena Yesha. Updating and Querying Databases that Track Mobile Units. *Distributed and Parallel Databases*, 7, 257-287, 1999
22. S. Handley, P. Langley, F. Rauscher, Learning to Predict the Duration of an Automobile Trip. *Proc. of the 4th International Conference on Knowledge Discovery and Data Mining (1998).*
23. N. Priyantha, A. Chakraborty, H. Balakrishnan, The Cricket Location-Support System, *Proc. of the 6th ACM Int. Conf. on Mobile Computing and Networking (MOBICOM),* Boston, MA, Aug. 2000.
24. J. Myllymaki, S. Edlund, Location Aggregation from Multiple Sources, *Proc. of the 3rd Int. Conference on Mobile Data Management (MDM)*, Singapore, Jan. 02.