Future Generation Computer Systems ▮ (▮▮▮▮) ▮▮▮–▮▮▮

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

A taxonomy for moving object queries in spatial databases

Sultan Alamri [a,*], David Taniar [a], Maytham Safar [b]

[a] Clayton School of Information Technology, Monash University, Australia

[b] Computer Engineering, Kuwait University, Kuwait

HIGHLIGHTS

- This study explains the variety of queries in moving objects databases.
- The movement patterns and the new raised queries of the moving objects databases.
- This taxonomy shows the targeted queries of the current moving objects data structures.

ARTICLE INFO

Article history:
Received 14 April 2013
Received in revised form
21 October 2013
Accepted 17 February 2014
Available online xxxx

Keywords:
Moving objects
Spatial databases
Movement patterns
Query taxonomy

ABSTRACT

In the past decade, many works have focused on the development of moving object database indexing and querying. Most of those works have concentrated on the common spatial queries which are used with static objects as well. However, moving objects have different features from static objects which may lead to a variety of queries. Therefore, it is important to understand the full spectrum of moving object queries, even before starting to build an index structure for such objects. The aim of this paper is to provide a complete picture of the capabilities of moving object queries. Thus motivated, in this paper we propose a taxonomy of moving object queries, comprising five perspectives: (i) *Location* perspective, (ii) *Motion* perspective, (iii) *Object* perspective, (vi) *Temporal* perspective and (v) *Patterns* perspective. These give an overall view of what moving object queries are about. In this work, each perspective is described and examples are given.

1. Introduction

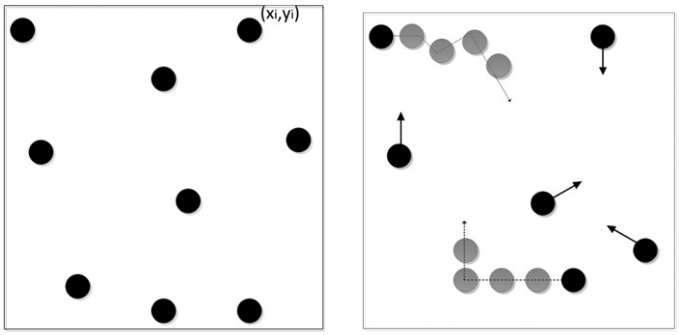
Moving objects are objects (points) that change their locations (geometric attributes) over time [1], which requires a higher update frequency. For instance, taxis traveling around the cities are considered as moving objects because they frequently change their locations over time, which requires a high level of updating in the taxi centers' databases. Other examples are cars, aircraft, ships, mobile phone users, armies, individuals and many more. Moreover, tracking these moving objects is essential for big data applications. Therefore, the indexing of moving objects plays a critical role in query processing, and our aim is to present a complete study regarding the capabilities of querying moving objects.

In addition, the storage and manipulation of moving objects will be based on spatial information representing static geographical objects alongside temporal information. There is an important difference between the indexing and querying of moving objects and that of static objects. With static objects, spatial data structures basically assume that the objects are constant unless conspicuously updated, whereas moving objects require frequent updates of the locations. Fig. 1 illustrates the difference between moving objects and static objects. Fig. 1(a) shows that the locations of static objects are constant unless conspicuously altered (e.g. change address). On the other hand, Fig. 1(b) shows that the locations of moving objects are constantly updated. Moreover, Fig. 1(b) shows the new features of the moving objects such as direction, velocity and movement patterns, which do not exist for static objects.

With the development of tracking and positioning systems, such as GPS and WI-FI, the correct recording of locations has become available which allows the *querying* of the moving objects. A large number of moving object applications have different perspectives regarding the querying of the moving objects [2,3]. Besides the typical types of queries such as point queries, range queries and k-nearest neighbor queries which are widely used in static object applications, querying moving objects has many other dimensions. For example, the moving objects change their locations with time; therefore, tracking moving objects at certain times

* Corresponding author. Tel.: +61 431685904.
E-mail addresses: Sultan.alamri@monash.edu (S. Alamri), david.taniar@monash.edu (D. Taniar), maytham.safar@ku.edu.kw (M. Safar).

http://dx.doi.org/10.1016/j.future.2014.02.007
0167-739X/



(a) Static objects. (b) Moving objects.

Fig. 1. Moving objects and static objects.

(temporal queries) is essential in security applications. Moreover, tracking moving objects that entered or left a certain area (topological queries), is another unique perspective of moving objects querying. These queries illustrate the different vectors of the moving objects which do not exist in static object applications.

In addition, different researches have been developed to analyse objects which change their spatial location over time in order to track and monitor cars, trains, and ships [4–7]. Most of these researches on moving objects have concentrated on the common spatio-temporal queries with a lack of attention being paid to the variety of queries about moving objects. In our work, we concentrated on building a taxonomy that assists to achieve a better understanding of the moving object queries in order to build data structures for moving objects. The main goal is to disclose the variety of possible queries about moving objects in spatio-temporal databases.

In this paper, we present a taxonomy for moving object queries. The queries about moving objects can be performed in different environments (underlying structures) which include the Euclidean space, spatial road network and cellular space. Therefore, we explain the moving objects' environments in Section 2. Then Section 3 explains the moving objects query taxonomy from various perspectives. First is the *Location* perspective, which includes common spatial queries such as K nearest neighbors (KNNs), range queries and others. Second is the *Motion* perspective, which covers direction, velocity, distance and displacement queries. Third is the *Object* perspective, which includes the type status queries and the form status queries. Fourth is the *Temporal* perspective which includes the trajectory, timestamped, inside, disjoint, meet, equal, contain, overlap and period queries. Last we explain the *Patterns* perspective, where the moving objects are using undefined movement or predefined movement patterns which include many patterns such as spatial movement patterns and temporal movement patterns. Each perspective is explained with illustrated examples. Section 4 summarizes our taxonomy with targeted queries in most of the current moving object data structures.

2. Background

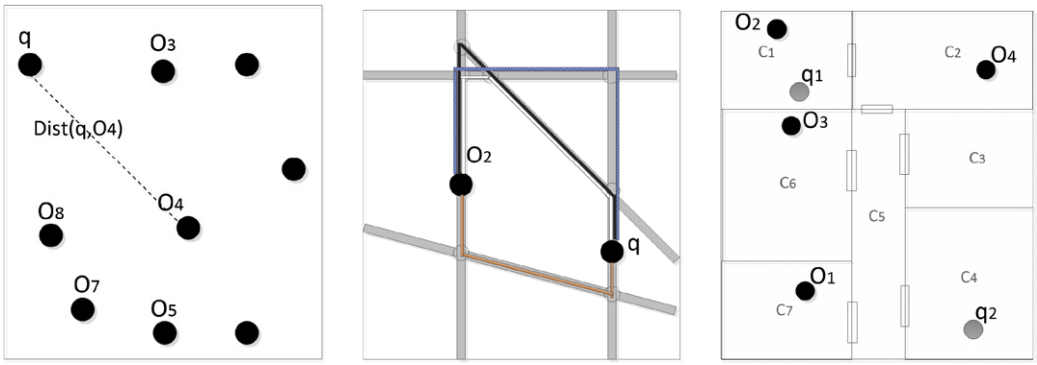
In this section, we explain the environments (underlying structures) of moving objects. The queries of moving objects are performed in different environments of moving objects. Therefore, understanding the underlying structures (the moving object environments) is essential. Moreover, the measurements for moving objects queries are different in each environment; hence, we need to illustrate the measurement differences of each environment in order to obtain a full picture of the query classifications.

The environments of the moving objects include the *Euclidean space, spatial road network* and *cellular space*. The example in the introduction section (see Fig. 1) is based on *Euclidean space*. In Euclidean space, the distance between object O_i and O_j is the straight and direct distance between them [8,9]. In this space, the queries will be considered from the Euclidean space measurement point of view. For example, a range query performed in Euclidean space could return the moving objects in a 2 km radius. In this example, the Euclidean space's measurement will be used to measure the distance from the query point to the points of interest up to the 2 km radius. Fig. 2(a) illustrates an example of a Euclidean space measurement query, where q wants to retrieve the Euclidean distance between it and moving object O_4.

The *spatial road network* is the other space that can be considered for moving objects. In a geospatial database setting, moving objects can be located in the road network where the distance between O_i and O_j is a network distance (or shortest road) between them [10–12]. Note that a spatial road network has *topological restrictions* and the distance between two objects is the actual road distance (avoiding the obstacles). In this space, the queries will be considered in terms of the road network measurement such as a 2NN query performed in a spatial road network to return the nearest two moving objects to q. In this example, the shortest path measurement will be used to retrieve the results. Fig. 2(b), illustrates an example of the spatial road network, where q wants to retrieve the shortest path's between its position and moving object O_2.

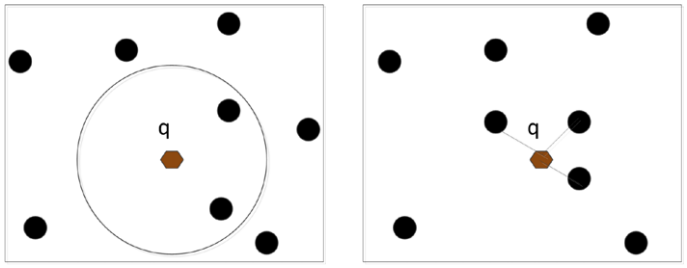
Cellular space is a representation of location by sets of cells that include moving objects. The basic difference between cellular space and Euclidean space or spatial road networks is the dependence on the geometric representation of spatial property [11,10]. A query in Euclidean space or a spatial road network is given with coordinates such as $(x_i; y_i)$ and $(x_j; y_j)$. On the other hand, the queries in cellular space are usually based on cellular notations such as: "What are the moving objects in cell 48?" In this example, the cell number represents a cell identifier which is the main difference from the outdoor coordinates in Euclidean space or spatial road networks [13,14,10,15]. To further explain, in a geometric space, both the located objects and locations will be represented as coordinate n-tuples, represented as points, areas, and volumes. Basically, this type is based on a reference coordinate system (RCS) which is used in many works [16–18,3]. On the other hand, the location of a moving object is indicated by abstract symbols or cells in cellular space. In cellular space, the location descriptions are represented by sets and a located object in that case is considered as a member of these sets. Indoor spaces are usually treated as *cellular space* where the object locations are not given with coordinates. The locations of the moving objects in indoor spaces are usually based on cellular notations. Moreover, many outdoor applications depend on the outdoor cellular space, where the exact locations of the moving objects are not useful. In this space, the queries will be considered from the cellular spaces measurement point of view. The distance is not metric, but is based

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007



(a) Euclidean space. (b) Spatial road network. (c) Cellular space.

Fig. 2. Space structures of moving objects.



(a) Range query. (b) 3NN query.

Fig. 3. Range and KNN queries.

on the number of hops. For example, if the q_1 in Fig. 2(c) wants to retrieve the nearest object to its location, O_2 will be considered the 1NN to q_1 since it is located in the same cell. Another example based on Fig. 2(c) where q_2 wants to return the nearest object to its location, the measurement will be based on the number of the hops between the cells, where O_1 is the nearest to q_2.

3. Moving object queries taxonomy

Our taxonomy of moving object queries covers five perspectives. First, the *Location* perspective includes common spatial queries such as *K* nearest neighbors (KNNs), range queries and others; second is the *Motion* perspective which covers direction, velocity, distance and displacement queries; third is the *Object* perspective which includes the type and form status queries; fourth is the *Temporal* perspective which includes the trajectory, timestamp, inside, disjoint, meet, equal, contain, overlap and period queries; and finally there is the *Patterns* perspective, whereby the moving object patterns can be undefined movement or predefined movements. Each perspective is explained with illustrated examples.

3.1. Location perspective

The location perspective considers the location as the key element in the moving object queries. Therefore, the results will be retrieved based on the location requirements in the queries. For example, range queries use the location as the main element to return the points of interests in a certain location with a certain range distance to the query point. This perspective includes the common location queries which are widely used in spatial databases. Location perspective includes many query types; however, we will confine the illustration to some examples such as *spatial, navigational, topological*, *N-body constraints* and *aggregate* queries.

Spatial queries are common spatial queries which are widely-used in spatial databases such as join, point, *K* Nearest Neighbor (KNN), Reverse Nearest Neighbor (RNN), range, spatio-temporal range and within-distance queries [19–22]. These queries usually depend on the location (the coordinates) of the query (see Fig. 3). Examples of KNN queries and range queries are "return the three nearest taxis to location *B* and all taxis in 2 km radius range". Spatio-temporal range queries consider the moving objects within the spatial range and temporal interval, such as "retrieve moving objects inside 2 km radius within 5 min from now". A within-distance query example is as follows: "return moving objects in the last 1 km distance to *q*".

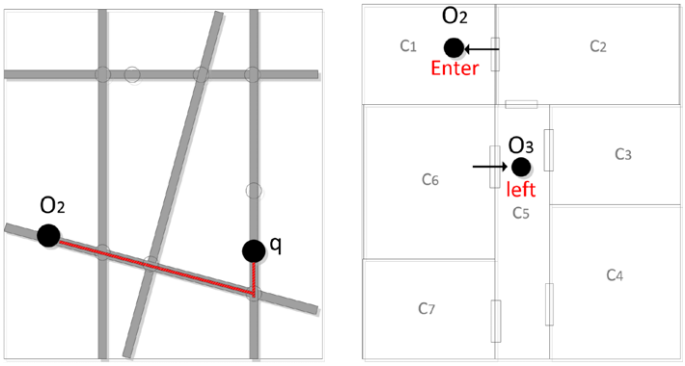
Navigational queries focus on data such as *heading, speed, travel distance, location, path* and many more (see Fig. 4(a)). Note that the overhead of computing these answers will be paid because the information that is requested is not stored directly. Next we present some examples of navigational queries. A traffic management center wants to return the current *location* or the current speed of a particular vehicle, or another example, "return the *distance* between object O_2 and q".

Topological queries concentrate on the data stored partially or fully of a trajectory object [5,4,23]. *Enters, leaves, bypasses* and *crosses* are common operations that are involved in topological queries (see Fig. 4(b)). Next we give some examples of topological based queries: "Which vehicles entered Monash University most recently?" Another example is: "Which objects crosses C_6?" Note that the complexity of processing this type of query includes the consideration of the moving object locations as cellular notations.

N-body constraint queries are the types of queries in which we can specify location constraints such as greater than or less than a specific distance. Therefore, the query returns a set of *N* objects which fulfil the alert location constraint. A query example to satisfy a 2-body constraint (based on Fig. 5) is: "retrieve taxis within a circular range with $r = 3$ from the center station". Fig. 5 shows an example of *N*-body constraint queries.

Aggregate queries or window aggregate queries concentrate on aggregate data over regions that satisfy some spatio-temporal predicates. This type of query is processed somewhat differently

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007



(a) Path query. (b) Topological query.

Fig. 4. Topological and navigational queries.

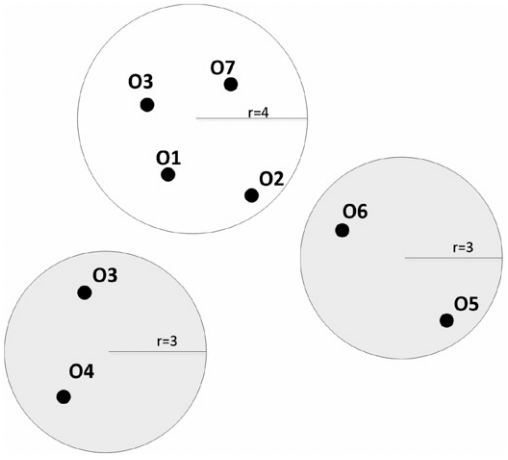


Fig. 5. N-body constraint queries. $\langle O_3, O_4 \rangle$ and the pair $\langle O_5, O_6 \rangle$ satisfy a 2-body constraint with alerting distance $r = 3$.

from a traditional relational database [24,25]. The difference is that in window aggregate queries, detecting the predefined hierarchies from ranges and locations of the spatio-temporal query window is quite difficult, which leads to some difficulties and complexities in using OLAP operations. For example: "return the number of moving objects that have been in *A* area between 2 p.m. and 3 p.m.". Another query example is: "return the maximum number of the visitors to *B* area".

To sum up, the results from this perspective are retrieved based on the location requirements in the queries. We explain several examples such as spatial, navigational, topological, *N*-body constraints and aggregate queries. One of the important complexities in this perspective is maintaining the frequent update of the queries; for example, maintaining the update of moving objects which are involved in a range query, or maintaining the update of moving objects' speeds or travel distances in navigational queries.

3.2. Motion perspective

Motion can be observed by linking a reference to a moving object and measuring its location change relative to another reference frame [26]. The motion perspective is a unique dimension of moving objects which does not exist for static objects. Therefore, motion queries can be raised for moving objects. There are many motion vectors of the moving objects which can be classified as motion vectors related to non geo-referenced moving objects (such as wind) and geo-referenced moving objects (such as vehicles). Since we focus on geo-referenced moving objects which consume geographical space, the motion vectors are classified as follows: *velocity*, *direction*, *distance* and *displacement*.

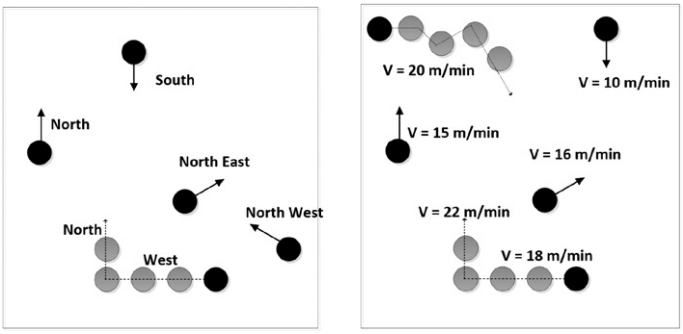
Velocity queries are queries that illustrate the relevance of the different speeds of each moving object. There is no doubt that each moving object will have a different velocity range for its own movement. For example, a transport company has a number of moving points (such as taxis); hence, the velocity range between the moving objects will be different. Moreover, the velocity range of each single object trip will be different as well. Therefore, object velocity queries can be classified as *similar range velocities* or *different range velocities*. *Similar range velocities* indicate that the target's moving objects on the map are moving at a similar speed range, whereas *different range velocities* means that the query is interested in whether objects are moving at a different speed range [3]. Note that speed similarity or difference can be applied to one single moving object, or to multiple moving objects. For example: "What are the objects that are moving within the range 60 kmph to 65 kmph in the northern area?" This illustrates a query that is related to each moving object's velocity.

Direction queries are the queries that depend explicitly on the moving objects' directions and whether or not they have a similar direction. For *similar direction query*, an example is: "return the vehicles in the suburb of Clayton that changed their direction to First St North", While a *different direction query*, "the traffic management system wants to return all moving vehicles in the suburb of Clayton which are moving in opposite directions". Note that the moving object's velocity and direction can make a difference because it involves different data structures and algorithms. The complexity of processing these types of queries includes considering both the velocity and direction in the tree structure of the moving object. Fig. 6 illustrates the direction and velocity of moving objects.

Distance queries are queries that illustrate the relevance of the different distances of each moving object. The distance vector is the path followed by the moving object during its motion. Here we focus on the relevant distances between the moving objects [26]. Note that relevant distances can be applied to one single moving object, or to multiple moving objects; for example: "return the moving objects that moved 2000 m distant". Another example which illustrates a query that is related to the moving object's distance is: "return the location of moving object *A* when its mileage (moved distance) reaches half the moving object *B* mileage".

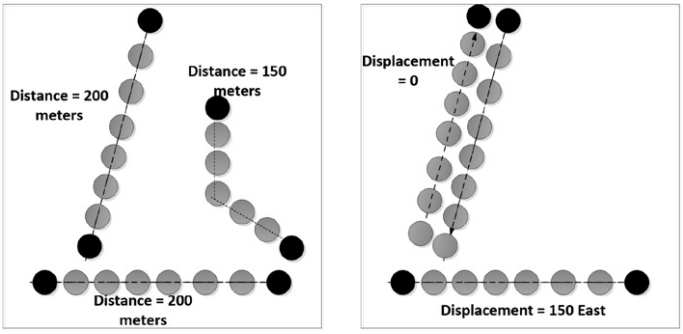
Displacement queries are queries that illustrate the relevance of the different displacement of each moving object. The displacement vector is the shortest distance starting from the initial point to the last destination point in the motion of a moving object [27]. Moreover, displacement has two vectors which are distance and

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007



(a) Moving object's direction. (b) Moving object's velocity.

Fig. 6. Direction and velocity of the moving objects.



(a) Moving objects' distance. (b) Moving objects' displacement.

Fig. 7. Distance and displacement of the moving objects.

direction [28]. Here we focus on the relevance of the between displacement of the moving objects. For example: "return all moving objects that moved around the field with the displacement $= 0$". In this example the results will include all moving objects that moved from an initial point and went back to the same initial point. Another example which illustrates queries that are related to moving object's displacement is: "return the location of moving object *A* when its displacement is equal to its *moved distance*/2". Fig. 7 illustrates the distance and displacement of moving objects.

To sum up, the motion perspective can be observed by linking a motion vector to a movement. Our focus on the geo-referenced moving objects includes velocity, direction, distance and displacement. The issues raised in this perspective include how to associate the motion vectors in the query processing, and how to involve the location perspective with the motion vectors.

3.3. Object perspective

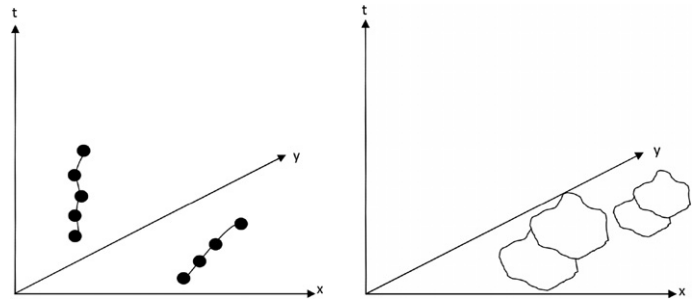
The object perspective considers the object characteristics as the key element in the query. Therefore, the results will be retrieved based on the object characteristics (type or form) that are indicated in the query. There are many types of moving objects; therefore, the object queries make a difference because they involve different data structures and algorithms. Queries about a moving object can involve different types of moving objects, or can specify only one type. Consequently, we need to illustrate the queries that are related to the object perspective of moving objects. Object perspective can be classified as follows: *object-type queries* which include single-type and multi-types and *object-form queries* which include point, line and region objects.

Object-type queries are queries that depend explicitly on the moving object types and whether or not they are of a similar type. There is no doubt that the types of moving objects are different and are determined by the main sponsor of the moving object. For example, a transport company has different types of moving points such as taxis, buses etc. The *type query* will consider the object from a *single-type* or *multi-types* perspective. *Single type queries* means that the outcomes of the query will be only one type of moving object (object type $= 1$), which means the objects are similar in all factors of the queries. For example, *cohort* [29–31], which means a group that has a factor in common that will be statistically relevant. For example, a similar gender (females): "return the locations of all females employees of the company". *Multi-type queries* means that the outcome of the query will be more than one type of moving object (object type > 1). For example: "return the locations of all moving objects in Second St". Here, the objects are *heterogeneous* because they are not the same type (such as same vehicle). Therefore, the outcome of this query might include different types of moving objects such as different makes of car and different mobile users.

Object-form queries are related to the body/form of the moving objects. Defining the object form to a large degree influences the successful design of a spatial data structure and definitely influences the performance of spatial database systems [32]. The majority of geo-referenced moving objects in spatial databases are considered as points such as individuals, mobile users, vehicles, and many more. However, the moving objects can also be considered as *lines* and *regions* [33,34]. A group of moving objects which are moving in a line formatting can be considered as line moving objects. For example, a procession of cars constitutes a moving line on the roads. A query example is: "return the length of the line that is moving towards Second Street". Another body status of moving objects is the moving region. This is clear in the movements of army troops and land extensions. The queries about moving regions can be about locations, size and other characteristics. Fig. 8 shows a representation of moving points and moving regions.

To sum up, the object perspective focuses on the object characteristics as the key element in the queries. This perspective can

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007



(a) Moving points. (b) Moving regions.

Fig. 8. An example of moving points and moving regions.

include object-type queries and object-form queries which include points, lines and regions objects. The issues include the difference that can be made in the data structures and algorithms considering new types of moving objects, and the issues associated with indexing unstructured moving objects.

3.4. Temporal perspective

The temporal perspective includes queries about the moving objects which concern to the temporal aspects and characteristics of the moving objects. Therefore, the temporal aspects will be included in this perspective's queries in different ways as given or requested. The time aspect is a unique vector of moving objects compared with static objects. Moreover, historical (temporal) queries are based on the time of the moving object and show the time difference for objects moving from one place to another [35]. Each database state has a link with a *time-stamp* [36]. In the state, the value of the dynamic object will be taken to be the value of the moving object at the time (t) [32,33,37]. We classify the temporal perspective as follows: *trajectory, timestamped, inside, disjoint, meet, equal, contain, overlap* and *period* queries.

Trajectory means that the sequence time of a moving object with the timestamps of each visit is stored and available for analysis. Trajectories can be used to describe the movement behavior of objects and therefore they can assist in determining groups of objects with the same trajectory. A query example is: "return the trajectory of object O_l". In this example, the location path of O_l is retrieved with each time-stamp of each location.

The database history is a sequence of database states, one for each time, depending on the fixed global clock. Hence, *timestamps* will be strictly applied on the database history. Moreover, the value of an object will be different in two respective database states; therefore, timestamps queries can be classified as *lower time-stamp*, *current time-stamp*, *future time-stamp* and *interval (range) timestamps* [32,4,35,38,39]. *A lower time-stamp* shows the recorded times of each moving object over previously visited location (past). For example, a delivery company wants to know the recorded time of delivery car X at its last location (historical queries). A lower time-stamp is based on the previously recorded information for a moving object in the database. *A current time-stamp* shows the current time of the moving object visited locations [40,41]. For example, "where is Object O_i Now". On the other hand, *the estimation future time-stamp* is based on the estimated time for the moving object to reach its next location (predictive queries) (see Fig. 9) [42]. Here, the query must cooperate with the distance of the next location and the current velocity of the moving object to determine the expected future time, for example, "return the nearest ambulance that will reach Monash University Clayton Campus in the next 2 h". Interval (range) timestamps specify a certain period of time which has a start and an end, for example, "return the ambulances that entered Monash University Clayton Campus between 2 p.m. and 5 p.m.".

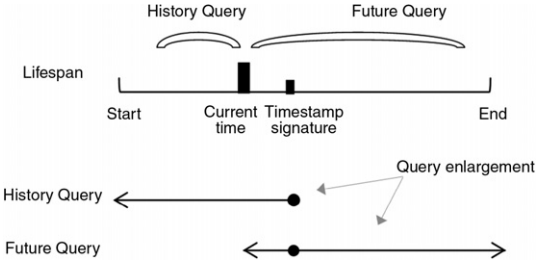


Fig. 9. Timestamped queries (lower, current and future timestamps).

Inside temporal queries indicate moving objects inside a certain time determined by the query. Inside temporal queries are different from the range time-stamp queries because the specified times are not timestamps, it can be a day or a week, for example, "return the moving objects that were active yesterday". In this example, the result should be the moving objects that were moving at any time during the previous day.

Disjoint temporal queries indicate moving objects that have nothing in common from the temporal perspective. In these queries, the times that are determined for each moving object are different. For example, "return two moving objects where one works on night shift and the other on day shift". In this example, the result should be two moving objects that have not worked together on the same shift.

Meet temporal queries are queries about moving objects that touch a common portion of their time boundaries. In other words, these are queries about moving objects that meet in their event times. For example, "return the moving objects that meet in any of their finish or start working times". In this example, the result would be the moving objects that their finishing working time meets with other objects' starting working time (or otherwise) (e.g. return O_1 and O_2, because O_1 working time finishes at 5 p.m. and O_2 working time starts at 5 p.m.).

Equal temporal queries are queries about moving objects that are equal on the temporal side. The results of the equal temporal queries should have exactly equal times. An example of an equal temporal query is: "return the moving objects that arrived at building A yesterday at 2 p.m. and left the building at 3 p.m.". In this example, the result should be only the moving objects that have met all the temporal aspects of the query, which are yesterday, 2 p.m. and 3 p.m.

Contain temporal queries are queries about moving objects that contain similarities in some temporal aspects. Contain temporal queries have some equal times between the moving objects. Example for a contain temporal query is "return the moving objects that arrived of building A at 3 p.m. or 5 p.m. or 6 p.m.". In this example, the result should be the moving objects that have any of the temporal aspects in the query, which are 3 p.m. or 5 p.m. or 6 p.m.

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007

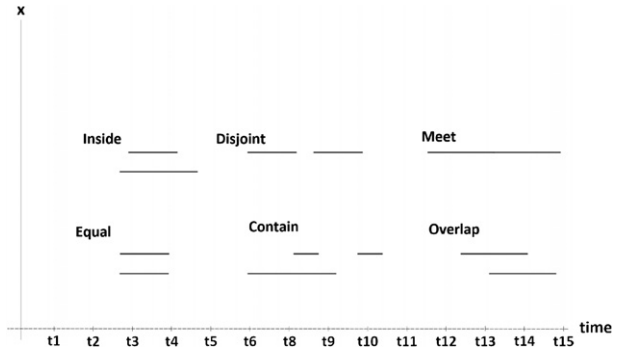


Fig. 10. Temporal operations.

Overlap temporal queries are queries about moving objects that overlap some targeted times. An example of the overlap temporal query is: "return all moving objects where their working duties are overlapped with the day shift and the night shift". The result should be the moving objects that have been in that place at both shift times. Fig. 10 illustrates some of the temporal operations.

Period temporal queries are queries about moving objects that have a specified period of time. An example of the period temporal query is: "return all taxis that were in Monash University for 20 min". The result should be all taxis that were in the specified place (Monash University) for the specified period of time (20 min).

To sum up, the temporal perspective is associated with queries about moving objects which concentrate on the temporal aspects and characteristics of moving objects. Many issues and complexities can be considered in temporal perspective queries such as maintaining the ongoing updating of the time with the location of the moving objects. Moreover, it is important to develop an index structure that facilitates temporal operation queries such as overlap and contains temporal queries.

3.5. Patterns perspective

Here we argue that according to this perspective some queries will be raised only when their movement *patterns* or behaviors are known. Movement patterns are any interesting relationships in a set of moving objects or any recognizable regularity in spatio-temporal data. In this perspective, the query depends entirely on the objects' predefined movement patterns. Patterns perspective queries are classified as: *spatial patterns, spatio-temporal movement* and *temporal patterns*. We explain these pattern categories with some movement pattern examples in terms of a number of characteristics and query examples.

3.5.1. Spatial patterns

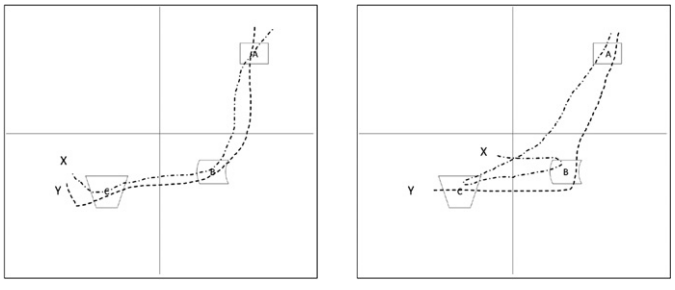
The spatial patterns are the patterns of the movement that include the spatial concentration of the moving objects. In other words, spatial patterns are concerned with the spatial aspects of the moving objects. Therefore, these patterns address as the spatial features of the moving objects and the impact on the existence of unique queries. Spatial patterns have many examples of movement patterns and we will be addressing some of them such as *co-location*, *concentration* and *Levy flights*, with several query examples.

Co-location occurs when the movements of the objects have similar locations in common [2,38]. Co-location patterns have three types. First, *ordered co-location* which exists when the common locations are reached in similar order. Second, *un-ordered co-location* exists when the common locations are reached in different orders. Third, *symmetrical co-location* exists if the common locations are reached in opposite orders [29]. For example, if tourists are visiting four different areas in Melbourne city which are Melbourne Zoo, Melbourne Museum, the Great Ocean Road and the Yarra Valley. If the tourists reach the locations in the same order, this will achieve the ordered co-location pattern, whereas, if the tourists reach the areas in different orders, the un-ordered co-location pattern will be achieved. Moreover, if the areas are reached in reverse order, then we have symmetrical co-location. Fig. 11 illustrates two types of co-location. The query example is: "return the tourist buses that are moving in the same order".

Concentration indicates the spatial concentration of moving objects at a certain instance of time; for example, congestion indicates an area that is affected by a crowd of vehicles in a transportation network. The query example is: "return the ambulances that are located in crowded areas of the city".

The *Levy flights pattern* comes from the name Paul Pierre Levy, the French mathematician. It means a kind of random walk in which the increments are distributed based on a heavy-tailed probability distribution [43,44]. In other words, after many steps, the distance from the original random walk will become a stable distribution [45,44]. The Levy flight concept has been used in a large number of fields such as nature and biology [45,44]. It has also been used in tracking stock market fluctuations, turbulent flow and human travel [43–45].

From our perspective, we are interested in the practical situation of the Levy flights. It is clear from Fig. 12 that there is a range of small random flights, separated by a transformation to some extent jumping to another area. We illustrate this with a simple example of Levy flight random movement; a maintenance company receives a call from Monash University Clayton Campus for a maintenance callout, so the company's maintenance car jumps from the company location to the University. During the operations in the University they received an emergency call from Chadstone Shopping Center for another maintenance callout. We can notice that the movement patterns here are random and without pre-planning, but the movement adopts the Levy flight aspect, where a stable distribution (in the University and Chadstone Shopping Center) has a jumping stage in between. A query example is: "return the total number of jumps which were made by maintenance car *A* today".



(a) Ordered co-location. (b) Un-ordered co-location.

Fig. 11. Co-location patterns.

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007

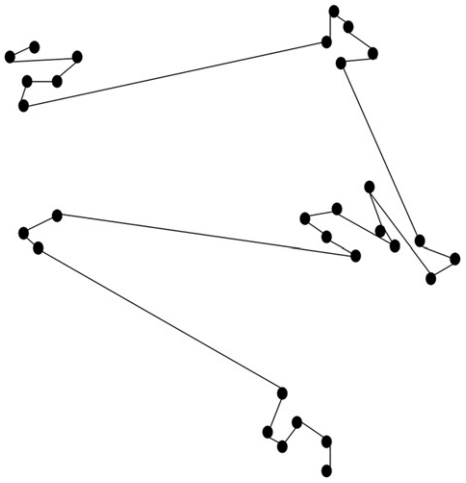


Fig. 12. Illustrates a jumping in Levy flight.

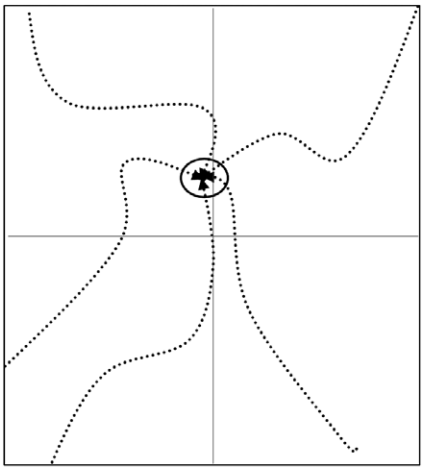


Fig. 13. Meet pattern.

3.5.2. Spatio-temporal patterns

This pattern of movement concerns in the spatio-temporal aspects of the movement. Therefore, these patterns address the spatial and temporal features together of the moving objects and the impact on the existence of unique queries. There is a variety of movement patterns in the spatio-temporal category; however our aim to not limit the movement patterns. Our aim is to illustrate that with knowledge of the movement patterns, some unique queries about the moving objects can be raised. For illustration, we will explain some of the spatio-temporal patterns such as *incidents*, *constancy, sequence, periodicity, meet* and *moving cluster* with some query examples.

First, *incidents* movements among multiple objects can be divided into four pattern types: *concurrence, co-incidence, opposition* and *dispersion* [29]. *Concurrence* refers to an incident that contains a set of entities having similar values of movement attributes for certain duration of time, whereas *co-incidence* is a specific type of incidence that considers the similarity of the moving objects' positions. This type can be *full co-incidence* which means the same locations are reached at the same time, or *lagged co-incidence*, meaning that the same locations are reached after a time delay. *Opposition* indicates a multi-polar arrangement of movement parameter values, such as a sudden group splitting of moving objects to two opposite movement directions. *Dispersion* indicates a group of moving objects that perform a non-uniform motion (opposite of concurrence) [46]. A query example is, "retrieve the moving objects that reach location A at 9:00 p.m". (full co-incidence pattern).

Second, *constancy* is when the motion consists of parameters still on the same values or only slightly changed for a duration of time [29]. For example, a group of cars are moving on a straight road at a specific velocity, and direction and the derived parameters remain the same for a particular duration. The query example is: "track all the vehicles that are heading north and accelerating at an unreasonable speed".

The third pattern is *Sequence*, which is a series of locations that have been visited as an ordered list. This type of pattern indicates a known start and end point in space and time. Sequence patterns are based on the locations and their timestamps. An example of a sequential pattern is a group of tourists visiting a set of places (Zoo, Museum and Gallery) in a particular sequence Zoo $\longrightarrow$ Museum $\longrightarrow$ Gallery within a specified duration of time [47].

Periodicity: this type indicates a cyclical pattern through a period of time (e.g. weekly or daily). This type shows a regular repetition of the movement (spatio-temporal periodicity) for a particular duration. For example, security patrolling cars regularly monitor specific locations through a regular repetition of movement in each period of time. The query example is: "return the locations that security car *x* regularly moves over".

The *Meet pattern* consists of a set of moving objects that meet at a particular point or location. Note that the meet pattern can be a fixed meet or un-fixed meet depending on whether or not the moving objects that stay together for a certain duration are planned for the meeting region [29,48]. For example, a group of friends moving to meet in a certain cafe for a meeting (fixed), or a group of taxis moving towards the airport to drop off passengers (un-fixed). Fig. 13 illustrates the meet pattern. The query example is: "in a marathon running competition, a runner invokes a query to return the number of runners ahead of his/her location".

A *moving cluster* is a set of moving objects that, during movements, stay close to each other while moving in a similar path for a specific duration. Note that it is not necessary that the moving objects participating in the pattern stay the same; for example, a query about moving troops in parallel methods on a military battlefield.

3.5.3. Temporal patterns

The temporal patterns are the patterns of the movement that focus on temporal characteristics of moving objects. In other words, this pattern of movement is interested in the temporal aspects of the movement. There are a variety of movement patterns in the temporal category and for illustration we will explain some of the temporal patterns such as *temporal relations*, and *synchronization in time* with some query examples.

Temporal relations are patterns which are based on the on time axis, for instance, a query about a group of moving objects stopping after a long trip. Fig. 14 explains how the *temporal relations* pattern depends on the time axis.

Synchronization in time is divided into two types. First, *full synchronization* indicates a pattern when changes of movement parameters happen similarly with no time delay (e.g., velocities, direction). *Lagged synchronization* indicates a pattern when changes of movement parameters happen similarly but after a delay of time. For example, "return the police cars that changed their direction on Monash Hwy 5 min from now".

Note that our aim is not to limit the number of movement patterns as there are a large number of other movement patterns. The main idea is to illustrate that with knowledge of movement patterns, new queries about moving objects can be raised. Moreover, these can integrate any type of previous movement patterns and thus produce a set of different queries that depend on the combined movement patterns. Many issues and complexities can be considered using the patterns perspective, for example,

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007

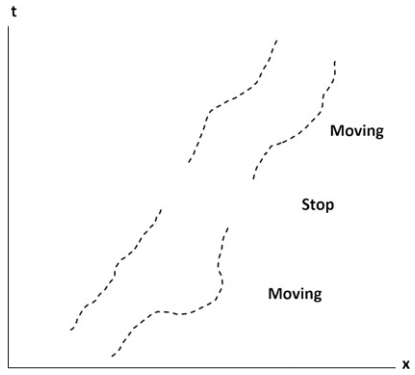


Fig. 14. Temporal relations pattern.

associating certain movement patterns in the index structure. Moreover, predefining movement patterns can influence the updating of the moving objects.

4. Moving object data structures and targeted queries

Many works have been proposed to accommodate the intensive updating which is the main issue when indexing moving objects databases. As an illustrative example, suppose that we are tracking the positions of 2 million mobile phone users in Melbourne. Each user updates his/her position every 10 s, and a single location server keeps track of them. The location server continuously receives the location update stream as a sequence of location update records in a form of (*ObjectID*, p (x, y)), where *ObjectID* is the moving object identifier and p is its location coordinate. The location server needs to efficiently handle 200,000 updates per second. Here, for each update operation, the location server will update the locations of the moving objects which maintain the spatial index. This is necessary in order to answer the variety of spatial and location queries. Therefore, many works have been focusing on minimizing the above-mentioned update cost for each update. Moreover, index structures also aim to obtain a logarithmic search complexity. For example, works such TPR-tree are basically based on R*-tree where the search complexity $O(\log M^n)$ [16,49,50].

In this section, we summarize our taxonomy with targets queries in most of the current moving object data structures. Note that the movement patterns in all of these index structures do not specify moving patterns for the moving objects. Many data structures have focused on the *location perspective* which includes the common queries in the spatial databases. For example, Saltenis et al. introduced the TPR-tree, (Time Parameterized R-tree) which is based on the R*-tree, in order to construct and manage moving objects. The main idea of the TPR-tree is that the index stores the velocities of objects along with their positions in nodes. Therefore, the TPR-tree and its successors [16,17,3,51,18,36] focused on the *spatial queries* such as range queries.

Works such as [11,15,52,13] index moving objects in symbolic indoor spaces, which can then process the *navigational, topological* and *aggregation* queries. For example, indoor-tree [15] indexes record moving objects in indoor cellular space based on the adjacency between the indoor cells. The data structure considers the indoor characteristics such as doors, rooms and hallways and topology predicates (enter, leave, cross, etc.). Therefore, these queries involve location, path (navigational queries), and enter, leave (topological queries). Moreover, the RTR-tree and TP2R-tree [13] are two R-tree-based indexes for trajectories of objects moving in symbolic space. The RTR-tree is based on a 2D R-tree on the Reader-Time space to organize trajectories as line segments. The TP2R-tree is based also on Reader-Time space; however, it transforms a trajectory to a set of points. Both of them consider characteristics such as doors, rooms, hallways, etc. and topology predicates (enter, leave, cross, etc.), which makes them easily support navigational, topological and aggregation queries.

Moving objects have different characteristics from static objects, such as *motion vectors*. A limited number of data structures concentrate on the motion vectors in the construction of the moving objects' data. An example is the DV-TPR*-tree [3] which indexes moving objects based on the *spatial*, *direction* and *velocity* domains. Therefore, the DV-TPR*-tree easily supports *velocity* and *direction queries*. Another example is distance signature [10], for computation of the distance and query processing over long distances. Therefore, [10] supports *motion distance queries*.

In addition, moving objects have many types; therefore, the object queries can make a difference because they involve different data structures and algorithms. The majority of the current data structures focus on *moving points* with a lack of attention to moving lines and moving regions. For example, MV3R-tree, TPR-tree, FNR-tree, RP-tree and many others index the moving points as coordinates (x, y) [16,52,34,53].

Works that concentrate on *temporal* moving objects include [53–55]. The Historical R-tree (HR-tree) is one of the earliest data structures to focus on historical data [55]. The main idea is to use the time-stamp history to construct the R-tree. R-trees can make use of common paths if objects do not change their positions, and new branches are created only for objects that have moved. It is clear that HR-trees are efficient in cases of *time-stamp queries*, as the search degenerates into a static query for which R-trees are very efficient. Another work that focuses on temporal data is the Multi-version 3D R-tree (MV3R-tree) [53] which basically uses multi-version B-trees [21] and combines them with 3D R-trees [53,33].

Works that concentrate on *trajectories* of moving objects are [56,57], which include the trajectory bundle tree (TB-tree), which is based on the R-tree [21,58]. The idea is to index trajectories by allowing a leaf node to contain line segments only from the same trajectory, which assists in retrieving the trajectory of an individual object, but negatively influences the spatio-temporal range queries [56,21]. Another trajectory index is named the STR tree (Spatio-Temporal R-tree) [56,57,8]. STR is based not only on spatial closeness, but also on trajectory preservation. The STR-tree can keep the line segments within the same trajectory [8]; therefore, it can easily support *trajectory queries*. Many other works focus on indexing the trajectories of moving object such as [37,13,52]. Table 1 summarizes the taxonomy queries of moving objects in spatial databases, with some examples of the moving object data structures of the targeted queries.

5. Conclusion

In this paper, we propose a taxonomy for moving object queries to address the variety of queries that can be raised about moving objects of interest. This research focuses on geo-referenced moving objects, which consume geographical space. We started by illustrating the environments of the moving objects which include Euclidean space, road networks and cellular space. The queries taxonomy mainly uses five perspectives to retrieve moving objects which include: First, the *Location* perspective, which includes common spatial queries such as *spatial, topological, navigational, N-body constraints* and *aggregation* queries; second, the *Motion* perspective, which covers the *distance, velocity, direction* and *displacement* queries; third, the *Object* perspective which covers type queries and form status queries (points, lines and regions); fourth, the *Temporal* perspective, in which the query can be a *trajectory, timestamped, inside, disjoint, meet, equal, contain, overlap or period* queries; and last, from a *Patterns* perspective, whereby the moving objects can be predefined movements which include many

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007

Table 1
Taxonomy queries of moving objects.

Perspective	Queries	Examples of data structures	Unique feature
Location perspective	Spatial queries Navigational queries *N*-Body constraint queries Topological queries Aggregation queries	TPR-tree, TPR*-tree, Indoor-tree, RP-tree, Q+Rtree, RTR-tree, TP2R-tree and LUGrid	Location as main element
Motion perspective	Direction queries Velocity queries Displacement queries Distance queries	DV-TPR*-tree and distance signature	Motion vectors
Object perspective	Type queries Moving points Moving lines Moving regions	TI-tree, STR-tree, TPR-tree, Indoor-tree, TPRuv-tree and GG TPR-tree	Similarity of the objects' type or form
Temporal perspective	Trajectory queries Timestamped queries Inside temporal queries Disjoint temporal queries Equal temporal queries Contain temporal queries Overlap temporal queries Period temporal queries	HR-tree, MV3R-tree, TB-tree, TI-tree, SETI, STR-tree, MOT-tree and ITD-tree	Trajectories or timestamps
Patterns perspective	Spatial patterns Spatio-temporal patterns Temporal patterns		Movement patterns

patterns (spatial movement patterns, spatio-temporal movement patterns and temporal movement patterns). Each perspective has been defined in a concise and expressive way according to a number of characteristics and by using examples.

In our future research, we intend to extend our taxonomy model to include the query processing of queries about the object queries. A data structure needs to be built in order to support some of the new moving object queries, especially after finding a lack of support for these queries in the current moving object data structures. For example, we will build a data structure to fit some of the movement pattern queries which have not received attention on the data structure side.

References

[1] Joachim Gudmundsson, Marc J. van Kreveld, Bettina Speckmann, Efficient detection of motion patterns in spatio-temporal data sets, in: GIS, 2004, pp. 250–257.
[2] Talel Abdessalem, José Moreira, Cristina Ribeiro, Movement query operations for spatio-temporal databases, in: BDA, 2001.
[3] Sultan Alamri, David Taniar, Maytham Safar, Indexing moving objects for directions and velocities queries, Inform. Syst. Front. 15 (2) (2013) 235–248.
[4] Ralf Hartmut Güting, Michael H. Böhlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos, Markus Schneider, Michalis Vazirgiannis, A foundation for representing and querying moving objects, ACM Trans. Database Syst. 25 (1) (2000) 1–42.
[5] Maribel Yasmina Santos, José Mendes, Adriano J.C. Moreira, Monica Wachowicz, Towards a spatio-temporal information system for moving objects, in: ICCSA (1), 2011, pp. 1–16.
[6] Agustinus Borgy Waluyo, Bala Srinivasan, David Taniar, A taxonomy of broadcast indexing schemes for multi channel data dissemination in mobile database, in: AINA (1), 2004, pp. 213–218.
[7] Agustinus Borgy Waluyo, Bala Srinivasan, David Taniar, Research in mobile database query optimization and processing, Mobile Inf. Syst. 1 (4) (2005) 225–252.
[8] Dieter Pfoser, Christian S. Jensen, Yannis Theodoridis, Novel approaches to the indexing of moving object trajectories, in: International Conference on Very Large Databases, 2000, pp. 395–406.
[9] Maytham Safar, Dariush Ebrahimi, David Taniar, Voronoi-based reverse nearest neighbor query processing on spatial networks, Multimedia Syst. 15 (5) (2009) 295–308.
[10] Haibo Hu, Dik Lun Lee, Victor C.S. Lee, Distance indexing on road networks, in: Proceedings of the 32nd International Conference on Very Large Data Bases, VLDB'06, 2006, pp. 894–905.
[11] Sultan Alamri, David Taniar, Maytham Safar, Indexing moving objects in indoor cellular space, in: 2012 15th International Conference on, Network-Based Information Systems, NBiS, September 2012, pp. 38–44.
[12] Elias Frentzos, Indexing objects moving on fixed networks, in: SSTD, 2003, pp. 289–305.
[13] Christian Jensen, Hua Lu, Bin Yang, Indexing the trajectories of moving objects in symbolic indoor space, in: Nikos Mamoulis, Thomas Seidl, Torben Pedersen, Kristian Torp, Ira Assent (Eds.), Advances in Spatial and Temporal Databases, in: Lecture Notes in Computer Science, vol. 5644, Springer, Berlin, Heidelberg, 2009, pp. 208–227.
[14] Zhexuan Song, Nick Roussopoulos, Hashing moving objects, in: Proceedings of the Second International Conference on Mobile Data Management, MDM'01, Springer-Verlag, 2001, pp. 161–172.
[15] Sultan Alamri, David Taniar, Maytham Safar, Haidar Al-Khalidi, A connectivity index for moving objects in an indoor cellular space, Pers. Ubiquitous Comput. (2013) 1–15.
[16] Simonas Saltenis, Christian S. Jensen, Scott T. Leutenegger, Mario A. Lopez, Indexing the positions of continuously moving objects, SIGMOD Rec. 29 (2000) 331–342.
[17] Yufei Tao, Dimitris Papadias, Jimeng Sun, The TPR*-tree: an optimized spatio-temporal access method for predictive queries, in: In VLDB, 2003, pp. 790–801.
[18] Wei Liao, Guifen Tang, Ning Jing, Zhinong Zhong, VTPR-tree: an efficient indexing method for moving objects with frequent updates, in: Advances in Conceptual Modeling—Theory and Practice, in: Lecture Notes in Computer Science, vol. 4231, Springer, Berlin, Heidelberg, 2006, pp. 120–129.
[19] Hyung-Ju Cho, Se Jin Kwon, Tae-Sun Chung, A safe exit algorithm for continuous nearest neighbor monitoring in road networks, Mobile Inf. Syst. 9 (1) (2013) 37–53.
[20] A. Prasad Sistla, Ouri Wolfson, Sam Chamberlain, Son Dao, Modeling and querying moving objects, in: ICDE, 1997, pp. 422–432.
[21] Dan Lin, Indexing and querying moving objects databases, Ph.D. thesis, National University of Singapore, Singapore, 2006.
[22] Kefeng Xuan, Geng Zhao, David Taniar, Bala Srinivasan, Continuous range search query processing in mobile navigation, in: Proceedings of the 2008 14th IEEE International Conference on Parallel and Distributed Systems, IEEE Computer Society, Washington, DC, USA, 2008, pp. 361–368.
[23] David Taniar, John Goh, On mining movement pattern from mobile users, Int. J. Distrib. Sensor Netw. 3 (1) (2007) 69–86.
[24] José Moreira, Cristina Ribeiro, Talel Abdessalem, Query operations for moving objects database systems, in: ACM-GIS, 2000, pp. 108–114.
[25] Natalia V. Andrienko, Gennady L. Andrienko, Designing visual analytics methods for massive collections of movement data, Cartographica 42 (2) (2007) 117–138.
[26] Angel D. Sappa, David Gerónimo, Fadi Dornaika, Mohammad Rouhani, Antonio M. López, Moving object detection from mobile platforms using stereo data registration, in: Marek R. Ogiela, Lakhmi C Jain (Eds.), Computational Intelligence Paradigms in Advanced Pattern Classification, in: Studies in Computational Intelligence, vol. 386, Springer, Berlin, Heidelberg, 2012, pp. 25–37.
[27] Xianfeng Fei, Yasunobu Igarashi, Koichi Hashimoto, Parallel region-based level set method with displacement correction for tracking a single moving object, in: Jacques Blanc-Talon, Wilfried Philips, Dan Popescu, Paul Scheunders (Eds.), Advanced Concepts for Intelligent Vision Systems, in: Lecture Notes in Computer Science, vol. 5807, Springer, Berlin, Heidelberg, 2009, pp. 462–473.

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007

[28] Peter Spoer, Displacement estimation for objects on moving background, in: Thomas Huang (Ed.), Image Sequence Processing and Dynamic Scene Analysis, in: NATO ASI Series, vol. 2, Springer, Berlin, Heidelberg, 1983, pp. 424–436.
[29] Yunyao Qu, Changzhou Wang, Like Gao, Xiaoyang Sean Wang, Supporting movement pattern queries in user-specified scales, IEEE Trans. Knowl. Data Eng. 15 (1) (2003) 26–42.
[30] Anat Levin, Lior Wolf, Amnon Shashua, Time-varying shape tensors for scenes with multiply moving points, in: CVPR (1), 2001, pp. 623–630.
[31] Sungnam Lee, Yohan Chon, Hojung Cha, Smartphone-based indoor pedestrian tracking using geo-magnetic observations, Mobile Inf. Syst. 9 (2) (2013) 123–137.
[32] Martin Erwig, Ralf Hartmut Güting, Markus Schneider, Michalis Vazirgiannis, Spatio-temporal data types: an approach to modeling and querying moving objects in databases, GeoInformatica 3 (3) (1999) 269–296.
[33] Victor Teixeira de Almeida, Ralf Hartmut Güting, Indexing the trajectories of moving objects in networks, GeoInformatica 9 (1) (2005) 33–60.
[34] Yuni Xia, Sunil Prabhakar, Q + Rtree: efficient indexing for moving object database, in: DASFAA, 2003, pp. 175–182.
[35] Reasey Praing, Markus Schneider, A universal abstract model for future movements of moving objects, in: AGILE Conf., 2007, pp. 111–120.
[36] Bin Lin, Jianwen Su, On bulk loading TPR-tree, in: 2004 IEEE International Conference on Mobile Data Management, 2004. Proceedings, 2004, pp. 114–124.
[37] Mindaugas Pelanis, Simonas Saltenis, Christian S. Jensen, Indexing the past, present, and anticipated future positions of moving objects, ACM Trans. Database Syst. 31 (1) (2006) 255–298.
[38] Hsiao-Ping Tsai, De-Nian Yang, Ming-Syan Chen, Mining group movement patterns for tracking moving objects efficiently, IEEE Trans. Knowl. Data Eng. 23 (2) (2011) 266–281.
[39] Jinfeng Ni, Chinya V. Ravishankar, PA-tree: a parametric indexing scheme for spatio-temporal trajectories, in: SSTD, 2005, pp. 254–272.
[40] Kyoung-Sook Kim, Si-Wan Kim, Tae-Wan Kim, Ki-Joune Li, Fast indexing and updating method for moving objects on road networks, in: Proceedings of the Fourth International Conference on Web Information Systems Engineering Workshops, WISEW'03, IEEE Computer Society, Washington, DC, USA, 2003, pp. 34–42.
[41] Xiaopeng Xiong, Mohamed F. Mokbel, Walid G. Aref, Lugrid: update-tolerant grid-based indexing for moving objects, in: MDM, 2006, p. 13.
[42] Jens Dittrich, Lukas Blunschi, Marcos Antonio Vaz Salles, Indexing moving objects using short-lived throwaway indexes, in: SSTD, 2009, pp. 189–207.
[43] Donny K. Sutantyo, Serge Kernbach, Valentin A. Nepomnyashchikh, Paul Levi, Multi-robot searching algorithm using Levy flight and artificial potential field, CoRR, abs/1108.5624, 2011.
[44] Injong Rhee, Minsu Shin, Seongik Hong, Kyunghan Lee, Song Chong, On the Levy-walk nature of human mobility, in: INFOCOM, 2008, pp. 924–932.
[45] Seongik Hong, Injong Rhee, Seong Joon Kim, Kyunghan Lee, Song Chong, Routing performance analysis of human-driven delay tolerant networks using the truncated levy walk model, in: MobilityModels, 2008, pp. 25–32.
[46] Marta C. González, Cesar A. Hidalgo, Albert-László Barabási, Understanding individual human mobility patterns, CoRR, abs/0806.1256, 2008.
[47] Dan Lin, Christian S. Jensen, Beng Chin Ooi, Simonas Saltenis, Efficient indexing of the historical, present, and future positions of moving objects, in: Mobile Data Management, 2005, pp. 59–66.
[48] Huiping Cao, Nikos Mamoulis, David W. Cheung, Mining frequent spatio-temporal sequential patterns, in: ICDM, 2005, pp. 82–89.
[49] Yong-Jin Choi, Jun-Ki Min, Chin-Wan Chung, A cost model for spatio-temporal queries using the TPR-tree, J. Syst. Softw. 73 (1) (2004) 101–112.
[50] Dan Lin, Rui Zhang, Aoying Zhou, Indexing fast moving objects for kNN queries based on nearest landmarks, Geoinformatica 10 (4) (2006) 423–445.
[51] Alex Lau, Processing frequent updates with the TPR*-tree using bottom-up updates, Master's Thesis, University of Waterloo, Ontario Anada N2L 3G1, June 2005.
[52] Sultan Alamri, David Taniar, Maytham Safar, Indexing of spatiotemporal objects in indoor environments, in: 2013 IEEE 27th International Conference on Advanced Information Networking and Applications, AINA, vol. 0, IEEE Computer Society, Los Alamitos, CA, USA, 2013, pp. 453–460.
[53] Yufei Tao, Dimitris Papadias, Mv3r-tree: a spatio-temporal access method for timestamp and interval queries, in: Proceedings of the 27th International Conference on Very Large Data Bases, VLDB'01, Morgan Kaufmann Publishers Inc, San Francisco, CA, USA, 2001, pp. 431–440.
[54] Dieter Pfoser, Christian S. Jensen, Yannis Theodoridis, Novel approaches in query processing for moving object trajectories, in: VLDB, 2000, pp. 395–406.
[55] Mario A. Nascimento, Jefferson R.O. Silva, Towards historical R-trees, in: SAC, 1998, pp. 235–240.
[56] Jae-Woo Chang, Jung-Ho Um, Wang-Chien LeeP, A new trajectory indexing scheme for moving objects on road networks, in: David Bell, Jun Hong (Eds.), Flexible and Efficient Information Handling, in: Lecture Notes in Computer Science, vol. 4042, Springer, Berlin, Heidelberg, 2006, pp. 291–294.
[57] V. Prasad Chakka, Adam Everspaugh, Jignesh M. Patel, Indexing large trajectory data sets with SETI, in: CIDR, 2003.
[58] Antonin Guttman, R-trees: a dynamic index structure for spatial searching, in: International Conference on Management of Data, ACM, 1984, pp. 47–57.

Sultan Alamri received his Bachelor degree in Computer Science from King Khalid University, Saudi Arabia in 2007, and his Master degree in Information Technology from School of Engineering & Mathematical Sciences, La Trobe University, Australia in 2010. He is currently a Ph.D. candidate of the Clayton School of Information Technology at Monash University, Australia. His research interests include moving objects databases, query processing and spatial databases. He can be contacted at Sultan.Alamri@monash.edu.

David Taniar holds Bachelor, Master, and Ph.D. degrees all in Computer Science, with a particular specialty in Databases. His current research interests include mobile/spatial databases, parallel/grid databases, and XML databases. He recently released a book: High Performance Parallel Database Processing and Grid Databases (John Wiley & Sons, 2008). His list of publications can be viewed at the DBLP server (http://www.informatik.uni-trier.de/ley/db/indices/a-tree/t/Taniar:David.html). He is a founding editor-in-chief of Mobile Information Systems, IOS Press, The Netherlands. He is currently an Associate Professor at the Faculty of Information Technology, Monash University, Australia. He can be contacted at David.Taniar@infotech.monash.edu.au.

Maytham Safar is currently a Professor at the Computer Engineering Department at Kuwait University. He received his Ph.D. degree in Computer Science from the University of Southern California in 2000. He has over one hundred books, book chapters, and conference/journal articles. Current research interests include social networks, complex networks, location based services, and geographic information systems. He is a Senior Member of IEEE since 2008, and the first Kuwaiti to become an ACM Senior member in 2009. He is also a member of IEEE Standards Association, IEEE Computer Society, IEEE Geoscience & Remote Sensing Society, IADIS, @WAS, SDIWC and INSNA. Established the first complex networks research group (Synergy) at Kuwait University, http://synergy.ku.edu.kw, 2010.

Please cite this article in press as: S. Alamri, et al., A taxonomy for moving object queries in spatial databases, Future Generation Computer Systems (2014), http://dx.doi.org/10.1016/j.future.2014.02.007