Indexing Moving Objects in Indoor Cellular Space

CONFERENCE PAPER · SEPTEMBER 2012

DOI: 10.1109/NBiS.2012.17

Sultan Alamri
Saudi Electricity Company
12 PUBLICATIONS **65** CITATIONS
SEE PROFILE

Maytham Safar
Kuwait University
116 PUBLICATIONS **821** CITATIONS
SEE PROFILE

Indexing Moving Objects in Indoor Cellular Space

Sultan Alamri, David Taniar
Clayton School of Information Technology
Monash University, Melbourne, Australia
Sultan.alamri, David.taniar@Monash.edu

Maytham Safar
Computer Engineering Department
Kuwait University, Kuwait
Maytham.safar@ku.edu.kw

***Abstract*—Researchers have proven that most individuals spend most of their lives indoors. With the current appropriate indoor positioning devices such as Bluetooth and RFID, WIFI, the locations of moving objects will be an important foundation for a variety of applications such as the tracking moving objects, way finding, and security. Many studies have considered the moving object in outdoor environments such as TPR-Tree and its successors. In this paper, we propose a new index tree for moving objects in cellular space. The Index will be based on the connectivity (adjacency) between the indoor environment cells. The development index can support and enable efficient query processing and efficient updates of moving objects in indoor space.**

***Index Terms*—index structure, indoor space, moving objects**

I. Introduction

People spend the majority of their time in indoor environments, working, living, shopping and entertaining [10, 14]. Hence, indoor environments have become increasingly large and complex. For instance, the London Subway has 268 stations with a network of 253 miles being traversed each hour and passengers numbering in excess of 4 millions daily. Consequently, moving object positioning and tracking is an important research field with many applications including tracking moving objects, way finding, and security [18]. Since the global navigation satellite systems such as GPS are not suitable for indoor environments (spaces), and with the new positioning devices technologies for indoor spaces such as RFID, Bluetooth and Wi-Fi [5, 9], large volumes of tracking data become available which provides a variety of services such as indoor navigation, objects tracking, security, and positioning. Thus motivated, this paper provides a new indexing technique for objects moving in cellular indoor space. The development index will be based on a new idea which makes our structure unique; we named it the 'connectivity' (adjacency) between the indoor environment cells.

In the past few years, much research has gone into the development of outdoor applications involving moving objects [10, 14, 17, 28], and concerning the indexing and querying of the trajectories of moving objects and their locations [2, 4, 13, 15, 20]. However, the outcomes of those researches are not suitable for indoor scenarios for the following reasons. First, indoor space is essentially different from outdoor space in many respects. The measurements in outdoor and indoor space are different. In outdoor space, Euclidean space or a spatial network is typically used; whereas, indoor space is related to the notion of cellular space. Moreover, in indoor space, the environment contains different entities such as rooms, doors and hallways that both enable and constrain movement. As a result, the inability or constraint of movement needs to be considered in the data structure for indoor spaces. Second, the positioning technologies differ between outdoor spaces and indoor spaces. In outdoor space, GPS is capable of continuously reporting the velocity and location of a moving object with varying accuracies. On the other hand, a proximity analysis is based on indoor positioning technologies which are not able to report the exact velocities or exact positions [6, 10, 24, 25]. Examples of indoor positioning devices are RFID reader and Bluetooth which are based on the sensing or activation range of a positioning device. In our work, we focus on RFID and Wi-Fi positioning technologies, and we assume a setting where RFID readers and Wi-Fi are deployed at fixed locations such as building entrances, rooms and hallways, where the Wi-Fi/RFID tags are attached to the moving objects at the building.

In this paper, we propose an index structure based on connectivity for moving objects. The data structure in outdoor spaces is based on the space domain, Euclidean or a spatial network. On the other hand, our work focuses on the construction of the moving objects based on the notion of cellular space. The key idea of our moving objects indexing is to take advantage of the entities that enable and constrain the movement in indoor environment (doors and hallways). Therefore, we obtain the optimal representation of the indoor environment that is different from the outdoor environment. Moreover, dealing with the indoor environment in terms of adjacency and connections-based, we have the opportunity to respond to the indoor spatial queries more efficiently.

Our major contributions are:

- We develop a moving objects index structure for indoor space (indoor-tree) which can manage memory wisely via a neighbours distance lookup table.
- The indoor space cannot precisely be transformed to a straight line; therefore, we provide a connection idea for optimal representation of the filling indoor space.
- We present accompanying algorithms for the process of

constructing the data structure for indoor space.

The rest of the paper is organized as follows. Section II briefly reviews several moving objects structures for outdoor spaces. The detailed indexing description is given in Section III, including the tree construction and maintaining algorithms. Section IV provides the discussion, directions of future work and the conclusion.

II. Literature Survey

The majority of the moving objects indices are based on Euclidean distance and the availability of GPS-type positioning which may be explicit or implicit. In Euclidean space, a moving object is a point that moves in a two-dimensional space (x,y planes) and that time is discrete. The position of moving object o is defined as coordinates that are pairs showing the position of an object o in the x,y plane with time. Furthermore, a GPS receiver can continuously report the location of the moving object (x,y coordinates) and its velocity. Therefore, the locations of a GPS-positioning outdoor moving object is usually represented as a three-dimensional space-time space. The works dealing with moving objects' outdoor space data structure can be classified as follows:

Works that concentrate on the trajectories of the moving objects [2, 3, 17]. We start with the Trajectory Bundle tree (TB-tree), which is based on the R-tree [7, 16]. The idea is to index trajectories by allowing a leaf node to contain line segments only from the same trajectory, which assists in retrieving of the trajectory of an individual object; however, it negatively influences the spatio-temporal range queries [3, 14]. Another trajectory Index named STR tree [17] (Spatio-Temporal R-tree) is based not only on spatial closeness, but also on trajectory preservation. STR was introduced to somehow balance spatial locality with trajectory preservation. The main idea of the STR tree is to maintain line segments belonging to the same trajectory. Works that concentrate on the historical moving objects [15, 21, 23]. The Historical R-tree (HR-tree) is one of the earliest data structures which concentrated on historical data [15]. The main idea is to use timestamp history when constructing 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 the case of timestamp queries, as search degenerates into a static query for which R-trees are very efficient; however, there the massive duplication of objects can lead to large space consumption [15, 21]. Another work focused on the historical data is Multi-version 3D R-tree (MV3R-tree) [21] which basically uses Multi-version B-trees [14, 16] and combines it with 3D R-trees [23]. The MV3R-tree includes significant improvements which produce huge space savings without influencing the performance of the timestamp queries compared to the HR-tree.

Works that focused on the future and the current positions [4, 13, 20]. Saltenis et al. introduced the TPR-tree, (Time Parameterized R) 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 velocities of objects along with their positions in nodes. Moreover, the intermediate nodes' entries will store an MBR (minimum bounding rectangle), beside its velocity vector which they call VBR. As an extension of TPR-Tree, Yufei Tao proposed TPR*-tree [22] which develops the insertion/deletion algorithms in order to improve the performance of the TPR-tree. As successors of the TPR-Tree concept, several methods have been proposed to improve different aspects [4, 13, 20].

Throughout the journey of the outdoor data structures, the availability of GPS-type positioning is explicit or implicit. Moreover, the majority of these works based on Euclidean space or a spatial network are typically used. Whereas, indoor space is related to the notion of cellular space. Indoor space is an environment that contains different entities such as rooms, doors and hallways that enable or constrain movement [5, 25]. Therefore, outdoor data structures cannot be applied to indoor space, at least without reasonable enhancement. Indoor space needs to be fixed with indoor positioning devices, such as WIFI, RFID reader and Bluetooth, which are based on sensing or activation. Furthermore, these positioning technologies determine the existence of moving objects at predefined cellular locations which is different in outdoor space. Besides, the limitation of the positioning device sensing ranges might cause a large volatility of the data if using outdoor data structures [10, 14, 19].

To the best of our knowledge, we are the first to build an index structure for the indoor environment that is to be based on adjacency (connections) between floor cells. Indoor environments are not based on Euclidean space; hence, treating the indoor environments based on their connectivity is the optimal way to build a data structure for indoors.

III. Indexing Moving Objects in Cellular Space

In this section, we start by explaining the representation of cellular indoor space. Then we define the uniqueness of our structure which comprises the connections (adjacency) in the indoor space cells with its structure and algorithm. Then, the expansion algorithm, which is the structure that is based on the cells connections, that facilitate the tree construction and the insert and delete algorithms. Then, we describe the construction of the tree and the maintaining algorithms.

The goal of indoor positioning is to capture the locations of moving objects at a certain point of time. We propose an index structure for tracking the moving objects in the floor plane. Our developed index is intended to support and enable efficient query processing and efficient updates of moving objects in indoor space. Moreover, the index can efficiently support different types of queries such as spatial queries

(KNN, indoor range, positioning and future locations queries) and topological queries and aggregation queries.

The moving object m in cellular indoor space is represented as follows:

$$\{m, (c, t)\,|c \in C, t = [Time], f\}$$

Where m (moving object) is located during the current time t at cell c, and f is the floor number. The space will be divided into grids based on its positioning device's coverage (Wi-Fi or RFID) (assume that the coverage will be fixed as grid). The division of the cells will depend on the technical resources of the positioning device's coverage.

For illustration purposes, we present the next example of a floor plane of 7 rooms including stairs $R7$ and corridor $R6$. Each venue will be divided based on the sensors' coverage. For example, the corridor will be divided into 5 cells ($C14, C10, C5, C2$ and $C1$) as shown in Figure 1. The result of the new cells distribution and the connections is as shown in Figure 1.

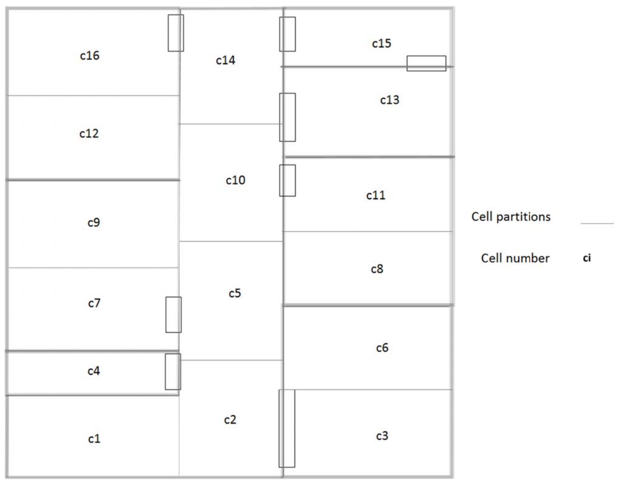


Figure 1. The coverage cells distribution of the floor

The indoor space is an overlapped environment, which has many constraints such as rooms, doors and hallways. Therefore, the indoor space requires more than one positioning device, which is more likely to cause overlapping between the positioning devices' coverage. We eliminate this issue by dealing with any cells' overlap as an individual cell.

Definition 1. Given a set of cells $C = \{C1, C2, ..., Cn\}$, a spatial object x, an overlap between any cells will produce ***new cell*** Ci Cj = $\bar{Cij}$ iff
$x \in$ Ci $(INsignal)$ and x $\in$ Cj $(INsignal)$, for any 1 $\leq$ i, j $\leq$ n.

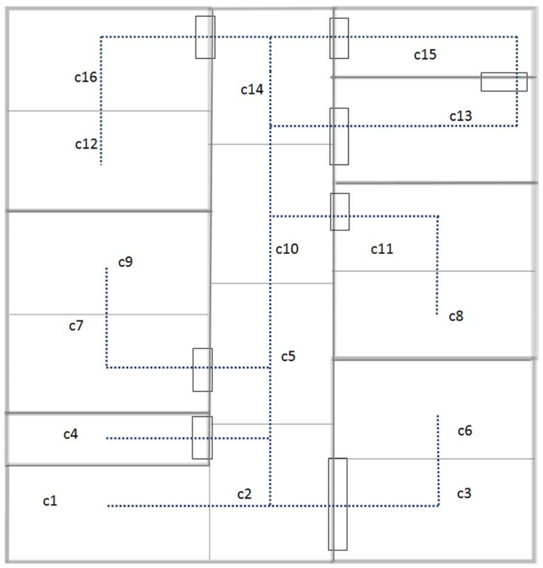


Figure 2. Illustrates the connectivity between the cells

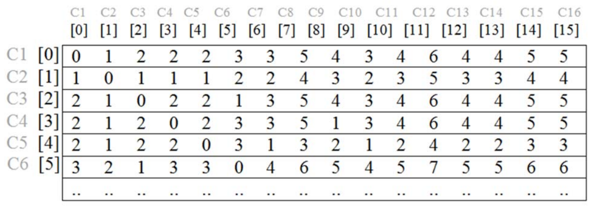
A. Indoor Connections

From the cells distributions, it is clear that for any object located in a particular cell, its next location must be in one of its adjacent cells. For example, for (Figure 2) objects located at $C12$, their next location must be $C16$. Therefore, we can establish connections between the cells which indicate the possible movements between the cells.

Definition 2. Given a set of cells $C = \{C1, C2, ..., Cn\}$, and a spatial object x, Ci is indicated as ***an adjacent cell*** to Cj if x can move between Ci and Cj directly without crossing other cells.

Definition 3. Let Oi be an object located in a particular cell Ci where $C1, C2, ..., Cn$ are the only adjacent cells of Ci. The ***next location*** of Oi *NX(Oi)* must be any of $C1, C2, ..., Cn$.

Next we give an example of the connection on our structure. A floor has been divided into 16 cells as shown in Figure 3. The connectivity between cells is shown by the dashed line between the centres of the cells. Here we can notice that any object located in $C12$, in order to reach $C15$, must go through $C16$ and $C14$ based on Definition 3.



	C1 [0]	C2 [1]	C3 [2]	C4 [3]	C5 [4]	C6 [5]	C7 [6]	C8 [7]	C9 [8]	C10 [9]	C11 [10]	C12 [11]	C13 [12]	C14 [13]	C15 [14]	C16 [15]
C1 [0]	0	1	2	2	2	3	3	5	4	3	4	6	4	4	5	5
C2 [1]	1	0	1	1	1	2	2	4	3	2	3	5	3	3	4	4
C3 [2]	2	1	0	2	2	1	3	5	4	3	4	6	4	4	5	5
C4 [3]	2	1	2	0	2	3	3	5	1	3	4	6	4	4	5	5
C5 [4]	2	1	2	2	0	3	1	3	2	1	2	4	2	2	3	3
C6 [5]	3	2	1	3	3	0	4	6	5	4	5	7	5	5	6	6
	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..	..

Figure 3. The neighbours distance

The connection between the cells will be stored in a neighbours distance lookup table to facilitate the grouping in the data structure. In this table, we have pre-computed neighbours distances between cells; the distance is not metric, but is based on number of hops, where 0 means the cell itself, 1 means the first level neighbour, 2 means the second level neighbour and so on. We assume that the positioning devices' coverage and the neighbours distance lookup is pre-fixed. A neighbours distance lookup table is established to assist in building the expansion algorithm. Moreover, it will be used in cells adjacency comparison to obtain the adjacent cells in case of insertion, deletion or updating (to determine the suitable nodes). Note that the size of the neighbours distance lookup table is quite insignificant (Figure 3). Moreover, we store the cells' connections as a multi-dimensional array list where the search complexity is $O(n)$ [1]. The checking algorithm that will use the table is called the Adjacency Comparison algorithm, which is used to return the nearest connected cell (the cell that has the MIN value in the neighbours distance lookup table).

Algorithm 1 Adjacency Comparison algorithm

```
1: /* check the inserted to $Ci$ with set of $C$. */
2: /* Adjacency Comparison algorithm will return the cell
   that has the MIN value */
3: for $Ci$ adjacent cells do
4:     Check the neighbours distance lookup table
5:     Return MIN(adjacent value)
6: end for
```

B. Expansion Algorithm

Indoor space is a filling space based on connection and adjacency [6, 12, 25]. Since the indoor space is usually based on cellular notation, unlike outdoor space which is based on coordinates (xi, yi) [26, 27], we need to establish a pre-computed expansion algorithm (extracted from the neighbours distance table) to represent overlapping between the cells. Since, the indoor space cannot precisely be transformed to be a straight line, the expansion idea can assist us to represent the filling indoor space cells to assign the higher (with more connection) and the lower cell (with less connection). Our objective is to use the data of the expansion algorithm (stored similarly to the neighbours distance table), in the construction the tree and the maintaining operations such as insertion and deletion. It is clear that each cell must be connected to expand the cell (point). Therefore, we will take advantage of that and record the two expand points with non-leaf nodes as range and the largest expand point will be recorded in the leaf nodes (for comparison and choose nodes goals).

As mentioned, the indoor environment is characterized by entities that enable and constrain movement (doors and hallways). Hence, we take advantage of that in the indoor environment to build data structure based on the adjacency. Traditional data structure such as R-trees and their followers based their comparison on the Euclidean space, so the objects will be grouped together based on MBR least enlargement [7]. Moreover, another traditional data structure such as Hilbert R-trees and their followers based their comparison on the Hilbert value, so the objects will be grouped together based on MBR based on LHV (Largest Hilbert Value) [8, 11]. These techniques are not applicable in indoor environment, because the indoor environment is not based on Euclidean distance. Furthermore, the indoor environment has overlapped areas, which cannot be treated as straight lines as Hilbert R-trees. Therefore, the expansion idea provides an optimal representation of the indoor space, which assist us in comparing between the cells and group the object based on their connections. Therefore, we store two expand points with non-leaf nodes as range and the largest expand point will be recorded in the leaf nodes (for comparison and choose nodes goals).

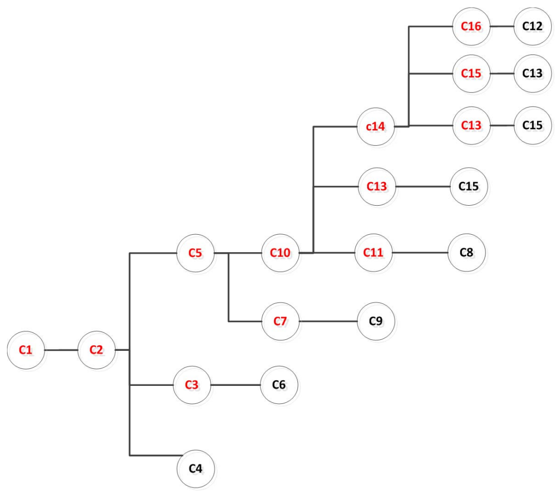


Figure 4. the expansion algorithm steps of the floor example (red illustrates the expand points)

C. Tree Construction

Based on the adjacency mentioned in the previous section, the data structure will group the entries based on their adjacency cells. In this way, we represent the best means of measuring the indoor environment which is based on the adjacency and connections between venue and rooms; conversely, the outdoor environment is based on metric measurement. Therefore, the idea is to start by grouping the objects inside the similar cell. In case of overflowing MBR, splitting will be performed to group it with one of the objects in adjacent cells based on the adjacency comparison algorithm. The idea is to check the RC (Range cells) at the non-leaf node or the LE (the largest expand point) at the leaf node by comparing them with the inserted cells (by the adjacency comparison algorithm) and choosing the cell that has the MIN value (the nearest connected cell). In our indoor tree, the data then is ordered according to the adjacency between the cells. There are two main properties of Indoor-tree:

1) non-leaf nodes contain (RC and $ChildPTR$) where RC is the range of the cells. We store the range of the cells in each non-leaf node; this is based on the expansion algorithm (red in Figure 4). Thus, each non-leaf node will have a range of two expand points as the maximum cell and minimum cells. $ChildPTR$ is the pointer to the child node. Non-leaf node contains at most O_n entries, which is the maximum capacity of the non-leaf nodes.
2) Leaf nodes contain (LE, *obj*, and PTR) where LE is the largest expand point based on the expansion algorithm, where each cell must be connected with expand cell (point); therefore, we record one expand cell only in the leaf node which is the largest connected cell at the node. For example, assume that a leaf node has three cells Ci,Cj and Cx where $Ci > Cj > Cx$ (three cells that contain the objects at that leaf node), we record Ci at the LE, because Ci is the largest expand point at that leaf node. The objects that are contained in the MBR at that time are denoted as *obj*, PTR is the pointer to the data. leaf node contains at most O_n entries, which is the maximum capacity of the leaf. The structure is illustrated in Figure 5.

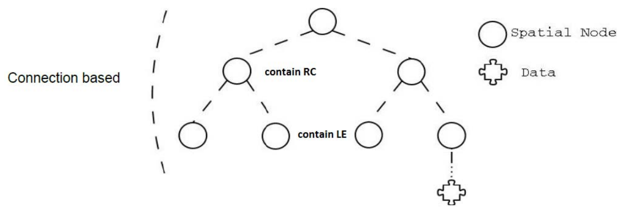


Figure 5. Indoor-tree

Using the sample data in Figure 6, suppose that the four MBRs are clustered into larger MBR, where moving objects = 1, 2, 3, 4,..., 10 and M = 3 and m = 2. Note that the objects' positions are shown only for simplification purposes. Thus, $N1$ RC is ($C14, C16$) (where $C14$ is the highest expand cell contain in $N1$ and $C16$ is the lowest) and $N2$ RC is ($C1, C3$) based on the expansion algorithm. In the leaf node $N1a$ LE is ($C16$), $N1b$ LE is ($C14$), $N2a$ LE is ($C2$), $N2b$ LE is ($C3$). The indoor tree is shown in Figure 7.

IV. Conclusion and Future Work

This paper addresses the challenge of building an index data structure that is appropriate for indoor spaces. The measurement of indoor space is different from that of outdoor space that is based on Euclidean space or a spatial network. Indoor space is related to the notion of cellular space that contains different entities such as rooms, doors and hallways that enable and constrain movement. The proposed index takes advantage of the entities that enable and constrain the movement in the indoor environment (doors and hallways). Our tree structure is based on adjacency and cells connections, which provides us the with opportunity to respond to the spatial queries more efficiently. In addition, our data structure

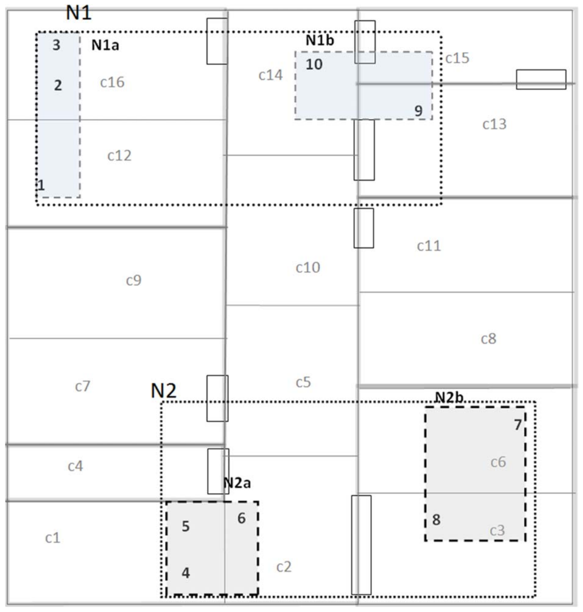


Figure 6. Illustrates the MBRs grouping based on the connection

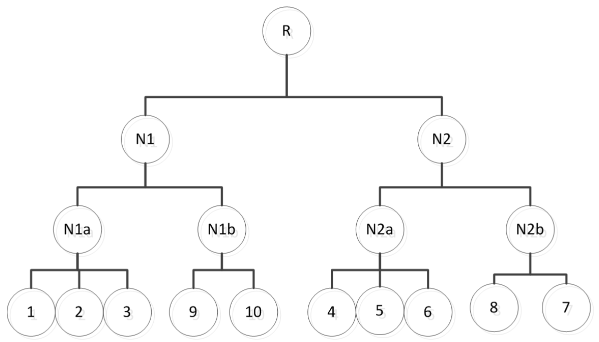


Figure 7. An example of Indoor-tree

is based also on the indoor expansion, which means an illustration of the way that the indoor floor is divided from its main entrance until last cell. This provides us the opportunity to respond to future queries more efficiently. The expansion algorithm gives us the opportunity to use the expand points and record the two expand points with non-leaf nodes as range and the largest expand point will be recorded in the leaf nodes (for comparison and choose nodes goals). To the best of our knowledge, using expand points in leaf nodes comparison is unique in our tree data structure.

This work can be extended in several directions. First, evaluating the indoor tree will be our next step in order to measure the reliability and the robustness of the indoor structure. The construction tree and the query processing performance will be measured in different indoor cases. Furthermore, the density of the moving objects and the complexity of the expansion and the connections will be tested as well. Our second future work direction is the maintaining operations algorithms. The insertion, deletion and updating algorithms will be developed taking into consideration the overflow, underflow and other issues.

References

[1] Arne Andersson, Torben Hagerup, Johan Håstad, and Ola Petersson. The complexity of searching a sorted array of strings. In *Proceedings of the Twenty-sixth Annual ACM Symposium on Theory of Computing*, STOC '94, pages 317–325, New York, NY, USA, 1994.

[2] V. Prasad Chakka, V. Prasad, Chakka Adam, Adam C. Everspaugh, and Jignesh M. Patel. Indexing large trajectory data sets with seti, 2003.

[3] Jae-Woo Chang, Jung-Ho Um, and Wang-Chien LeeP. A new trajectory indexing scheme for moving objects on road networks. In David Bell and Jun Hong, editors, *Flexible and Efficient Information Handling*, volume 4042 of *Lecture Notes in Computer Science*, pages 291–294. Springer Berlin / Heidelberg, 2006.

[4] Yong-Jin Choi, Jun-Ki Min, and Chin-Wan Chung. A cost model for spatio-temporal queries using the TPR-tree. *Journal of Systems and Software*, 73(1):101 – 112, 2004.

[5] F. Forno, G. Malnati, and G. Portelli. Design and implementation of a bluetooth ad hoc network for indoor positioning. *Software, IEE Proceedings -*, 152(5):223 – 228, oct. 2005.

[6] Ralf Hartmut Güting, Michael H. Böhlen, Martin Erwig, Christian S. Jensen, Nikos A. Lorentzos, Markus Schneider, and Michalis Vazirgiannis. A foundation for representing and querying moving objects. *ACM Trans. Database Syst.*, 25(1):1–42, March 2000.

[7] Antonin Guttman. R-trees: A dynamic index structure for spatial searching. In *International Conference on Management of Data*, pages 47–57, 1984.

[8] Haibo Hu, Dik Lun Lee, and Victor C. S. Lee. Distance indexing on road networks. In *Proceedings of the 32nd International conference on Very Large Data Bases*, VLDB '06, pages 894–905. VLDB Endowment, 2006.

[9] James Jayaputera and David Taniar. Data retrieval for location-dependent queries in a multi-cell wireless environment. *Mob. Inf. Syst.*, 1(2):91–108, April 2005.

[10] Christian Jensen, Hua Lu, and Bin Yang. Indexing the trajectories of moving objects in symbolic indoor space. In Nikos Mamoulis, Thomas Seidl, Torben Pedersen, Kristian Torp, and Ira Assent, editors, *Advances in Spatial and Temporal Databases*, volume 5644 of *Lecture Notes in Computer Science*, pages 208–227. Springer Berlin / Heidelberg, 2009.

[11] Ibrahim Kamel and Christos Faloutsos. Hilbert r-tree: An improved r-tree using fractals. In *Proceedings of the 20th International Conference on Very Large Data Bases*, VLDB '94, pages 500–509, San Francisco, CA, USA, 1994.

[12] Hye-Young Kang, Joon-Seok Kim, and Ki-Joune Li. Similarity measures for trajectory of moving objects in cellular space. In *Proceedings of the 2009 ACM Symposium on Applied Computing*, SAC '09, pages 1325–1330, New York, NY, USA, 2009. ACM.

[13] Bin Lin and Jianwen Su. On bulk loading TPR-tree. In *Proceedings of the International Conference on Mobile Data Management*, pages 114 – 124, 2004.

[14] Dan Lin. *Indexing and Querying Moving Objects Databases*. PhD thesis, National University of Singapore, Singapore, 2006.

[15] Mario A. Nascimento and Jefferson R. O. Silva. Towards historical r-trees. In *Proceedings of the 1998 ACM Symposium on Applied Computing*, SAC '98, pages 235–240, New York, NY, USA, 1998. ACM.

[16] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, Sung Nok Chiu, and D. G. Kendall. *Spatial Tessellations:Concepts and Applications of Voronoi Diagrams*, pages 585–655. John Wiley Sons, Inc., 2008.

[17] Dieter Pfoser, Christian S. Jensen, and Yannis Theodoridis. Novel approaches to the indexing of moving object trajectories. In *International Conference on Very Large Databases*, pages 395–406, 2000.

[18] Toms Ruiz-Lpez, Jos Garrido, Kawtar Benghazi, and Lawrence Chung. A survey on indoor positioning systems: Foreseeing a quality design. In Andre de Leon F. de Carvalho, Sara Rodrguez-Gonzlez, Juan De Paz Santana, and Juan Rodrguez, editors, *Distributed Computing and Artificial Intelligence*, volume 79 of *Advances in Intelligent and Soft Computing*, pages 373–380. Springer Berlin / Heidelberg, 2010.

[19] Maytham Safar, Dariush Ibrahimi, and David Taniar. Voronoi-based reverse nearest neighbor query processing on spatial networks. *Multimedia Systems*, 15:295–308, 2009.

[20] Simonas Saltenis, Christian S. Jensen, Scott T. Leutenegger, and Mario A. Lopez. Indexing the positions of continuously moving objects. *SIGMOD Rec.*, 29:331–342, May 2000.

[21] Yufei Tao and 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, pages 431–440, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[22] Yufei Tao, Dimitris Papadias, and Jimeng Sun. The TPR*-tree: An optimized spatio-temporal access method for predictive queries. In *In VLDB*, pages 790–801, 2003.

[23] Yannis Theodoridis, Michael Vazirgiannis . Timos Sellis, Michael Vazirgiannis, and Timos Sellis. Spatio-temporal indexing for large multimedia applications. In *ICMCS*, pages 441–448, 1996.

[24] Agustinus Borgy Waluyo, Bala Srinivasan, and David Taniar. Research in mobile database query optimization and processing. *Mob. Inf. Syst.*, 1(4):225–252, December 2005.

[25] O. Wolfson, B. Xu, S. Chamberlain, and L. Jiang. Moving objects databases: issues and solutions. In

Scientific and Statistical Database Management, 1998. Proceedings. Tenth International Conference on, pages 111 –122, jul 1998.

[26] Kefeng Xuan, Geng Zhao, David Taniar, Wenny Rahayu, Maytham Safar, and Bala Srinivasan. Voronoi-based range and continuous range query processing in mobile databases. *J. Comput. Syst. Sci.*, 77(4):637–651, July 2011.

[27] Kefeng Xuan, Geng Zhao, David Taniar, and Bala Srinivasan. Continuous range search query processing in mobile navigation. In *Proceedings of the 2008 14th IEEE International Conference on Parallel and Distributed Systems*, pages 361–368, Washington, DC, USA, 2008. IEEE Computer Society.

[28] Geng Zhao, Kefeng Xuan, W. Rahayu, D. Taniar, M. Safar, M.L. Gavrilova, and B. Srinivasan. Voronoi-based continuous k nearest neighbor search in mobile navigation. *IEEE Transactions on Industrial Electronics*, 58(6):2247 –2257, june 2011.