Adaptive Cell-Based Index For Moving Objects In Indoor

SoongSun Shin[1], Gyoungbae Kim[2] and HaeYoung Bae[1]
[1] Dept. Of Computer Science and Information Engineering, Inha University, In Koran
[2] Dept. Of Computer Education, Seowon University, in Korea
[e-mail: hermit7999@gmail.com, hybae@inha.ac.kr, gbkim@seowon.ac.kr]

Received March 1, 2012; revised May 15, 2012; revised June 25, 2012;
accepted June 29, 2012; published July 25, 2012

Abstract

Existing R-tree that is based on a variety of outdoor-based techniques to manage moving objects have been investigated. Due to the different characteristics of the indoor and outdoor, it is difficult to management of moving object using existed methods in indoor setting. We propose a new index structure called ACII(adaptive Cell-based index for Indoor moving objects) for Indoor moving objects. ACII is Cell-based access structure adopting an overlapping technique. The ACII refines cells adaptively to handle indoor regional data, which may change its locations over time. The ACII consumed at most 30% of the space required by R-tree based methods, and achieved higher query performance compared with r-tree based methods.

Keywords: Indoor, moving objects, spatio-temporal, Cell-based access structure

http://dx.doi.org/10.3837/tiis.2012.07.006

1. Introduction

Continuouly, spatio-temporal index will play a crucial role in tracking users efficiently and providing better utilization service in LBS. A lot of the study on data management for moving objects has assumed an outdoor environment in which objects move in Euclidean space (possibly constrained) or some form of spatial network and in which GPS(Global Positioning System) or GPS-like positioning is assumed explicitly or implicitly. That body of research provides part of an enabling foundation for the growing Location-Based Services industry[1][2][3].

However, many people lives in indoor space: homes, office buildings, shopping and leisure facilities, and collective transportation infrastructures. The latter may be large: For example, each day in 2009, London Heathrow Airport, UK had on average 180,000 passengers, and the Tokyo Subway (Tokyo Metro and Toei Subway), Japan delivered a daily average of 8.7 million passenger rides in 2008. Tokyo's Shinjuku Station alone was used by an average of 3.64 million passengers per day in 2007[4]. However, existing techniques for indexing in outdoor space are not easily applicable in indoor spaces, for three reasons[5][6].

First, the location of indoor space is not enough to available the existed geometric model. The GPS is unavailable. Also, the geometric model is used to euclidian distance method for calculation of position. But, this indoor movement different from outdoor Euclidean and spatial-network constrained movement. A corridors, rooms and interlayer need to clarify the distinction in indoor spaces modol. Accordingly, the new models is needed for indoor envrionment in which setting occurs is featured by semantic entities such as doors, rooms, hallways and floor.

Second, the location of moving objects in indoor space is expressed in the same space even if the position is different in same cell space. For instance, when there are moving obecjt MBa and MBb, each of them has (x, y) position. And each moving object is included in Cell space as MBa and MBb $\in$ Cell space. That is to say, although there are two moving objects that have different positions, they corresponds to the same location in the Cell space.

Third, existing positioning system was using GPS location for positioning. GPS-like positioning is typically unavailable in indoor spaces. In order to solve indoor positioning problems of indoor location-based positioning techniques have been investigated. There are a variety of techniques based on indoor positioning techniques like as RFID(Radio Frequency IDentification), UWB(Ultra WideBand), WLAN(Wireless LAN), etc. But WLAN location positioning techniques take advantage the bond in real life [7][8].

To overcome the limitation of indoor access methods, we propose ACII(Adaptive cell-based Index for Indoor Moving Objects). The ACII index consists of two main structures: MC(Memory Cell) and MEMO(Memory). The MC is a cell-based adaptive index structure to handle moving objects searches. The MEMO is a trajectory stored list that stores time-varying positions of moving objects (i.e., trajectories), and is used for retrieving trajectories. This two-body structure uses both management of indoor moving objects and overlapping techniques to produce a time and space efficient index structure, and is particularly useful for processing various spatio-temporal queries. The MC partitions an cell space into a set of fixed-size each cells to form a Rooms, Hallways and Doors that keeps making belong to indoor topologies. Each cell space stores a record of moving objects in its corresponding subspace. The MC don't change its basic cell space structure. Proposed method should manage to minimize the storage requirement, the MC and MEMO an overlapping technique. When the Moving Object moves in the same cell space, the information of MC will not be updated. But, inserted moving objects in other cell, inserted moving object is included with other cell and to write the memo for exchange MC spcace. Thus, new data record are created only when it needs to accommodate changes made to the data set sampled previously. The MEMO stores the entire trajectories of moving objects in a space-efficient way. The space efficiency is important because historical databases like such trajectories tend to grow large rapidly

The rest of the paper is organized as follows. In Section 2, we survey related work, discuss their advantages and analyze their problems. Section 3 presents the design principles of the ACII and Section 4 describes the structure of the MC and MEMO in detail. In Section 5, we provide a detailed description of algorithms for query processing. Section 6 contains an extensive experimental

evaluation. Finally, Section 7 summarizes the conclusion and future work directions for further research.

2. Related Work

2.1 Indoor Model

Symbolic space model is suitable for modeling the indoor spaces. Because, Indoor space is impossible to use the GPS or GPS like as positioning system. Existed geometric model used the distance to calculate the coordinate of location using topology feature by GPS positioning system. Information about locations is presented in different formats. Geometric coordinates, as used by GPS, refer to a point or geometric figure in a multi-dimensional space, typically, a plane or a three-dimensional space. The topological properties of such a space allow the calculation of distances between locations and their inclusion in other locations.

Symbolic coordinates, on the other hand, do not provide any reasoning about their spatial properties(distance and inclusion) without any additional information. Such coordinates are available via cell-IDs in cellular networks, such as GSM(Global System for Mobile communication) or WLAN, as well as via other positioning technologies, such as radio frequency tags(RFIDs) or infrared(IR) beacons. Therefore, symbolic coordinates define positions in the form of abstract symbols, e.g., the sensor identifiers of the active badge system, or room and street names, etc. In contrast to geometric coordinates, the distance between two symbolic coordinates is not implicitly defined. Also, topological relations like spatial containment cannot be determined without further information about the relationship between symbolic coordinates. Symbolic location models provide this additional information on symbolic coordinates.

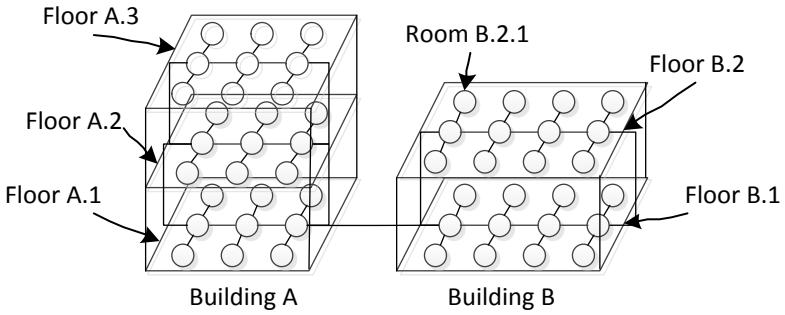


Fig. 1. Symbolic location model

Fig. 1. shows an example of the symbolic model that combined graph-based and set-based. Floor *A.1.2* and *3* shows the floors of *building A*. *Floor A.1* and *A.2* have a connection. The *B.2.1* is a room and their connections. The *building A* and *B* depicts buildings and the paths between them. The latter could be used in a scenario where only coarse-grained location information suffices, and so, it allows generating small models that cover large areas.

In the graph-based part of the combined model, locations are connected by edges if a connection between these locations exists in the real world. For instance, two rooms will be connected in the graph if there is a door between them; two floors will be connected if stairs lead from one floor to the other, etc.

The set-based part of the combined symbolic location model consists of a set of symbolic coordinates. Location are subsets of this set of locations, representing rooms, floors, buildings, etc. this part of the model is used for range queries as described in the section about set-based models.

Therefore, symbolic space models are often preferred over geometric models when modeling indoor space[5][9].

2.2 Indoor Positioning and AIM

Traditional spatio-temporal index methods are designed mainly to support outdoor query processing [10][11][12][13][14]. Updating traditional structures are not easily applicable in indoor spaces[15].

The TPR-tree[13] based on the R*-tree with velocities to index linear functions of time. Specifically, an object is represented by its position as of a reference time and its velocity vector. The sides of the bounding rectangles (BRs) employed are functions of time as well, and the BRs are chosen in order to bound all the included moving objects or BRs at any time in the future. If no updates occur on a TPR-tree, its BRs will expand, and query and update performance will deteriorate. When updates occur, objects are placed in the BRs that then now fit into, and BRs that have grown too much are tightened.

TPR*-tree[14] is a variant of the TPR-tree. The TPR*-tree uses the same data structure as the TPR-tree, but applies different algorithms for maintaining the index. Especially, the aim of the algorithms is to optimize time-range queries rather than time slice queries, as done by the TPR-tree. And when the TPR-tree decides on where to insert an object on a level-by-level basis, the TPR*-tree focuses more on insertions and makes more global decisions. In addition, the TPR*-tree is more aggressive than the TPR-tree when it comes to the tightening of BRs while tightening costs I/O, it may save I/O subsequently.

The claim is that updates are not frequent in traditional application. However, recently moving objects data continuously send location updates to the index structure as they move. For the past decade, several research efforts focus on developing variation of traditional access methods to support continuously moving objects.

Various index-based strategies have been proposed for spatio-temporal index. These works assume a Euclidean space and accurate positions. This paper studies spatio-temporal index in indoor environment. It employs an augmented r-tree on time intervals to retrieve relevant historical indoor.

The SCRM[15] is continuous range monitoring adapts the symbolic indoor space to exploit monitoring of moving objects to handle spatio-temporal joins on historical indoor tracking data. By using SCRM, there are many advantages shown as follows: First, it is possible to share query processing cost among concurrent queries so as to reduce the overall system overhead. Critical devices common to multiple queries can be identified and exploited for that purpose. Second, it is of interest to consider other types of monitoring queries, e.g., range and kNN queries that are attached to moving objects. Third, it is interesting to conduct probabilistic analysis on other kinds of object distributions, e.g., Gaussian distribution. But its contributions are orthogonal to issues such as low-level RFID data management [16][17]. Rather, the paper aims to offer, at a higher level, a uniform foundation for joining indoor symbolic tracking data that may be obtained from a variety of RFID technologies and also from other technologies such as Bluetooth. And the positions of moving objects are often characterized by uncertainty. Between two consecutive GPS reports, an outdoor moving object's position can be captured by an ellipse [18]. So, Based on an analysis on the uncertainty inherent to indoor tracking data, effective join probabilities are formalized and evaluated for object.

The MR-tree, the HR-tree and Overlapping Linear quad-tree are all based on the concept of overlapping trees. The idea was applied to handle the evolving B^+-tree[19]. Later, this technique was extended to R-trees and quadtrees, to index spatio-temporal moving objects [20]. The basic idea is that, given two trees where the second one is an evolution of the first one, the second tree can be represented by registering only the modified branches of the first one. That is, only the modified branches are actually created and the branches that do not change are simply re-used, while each tree keeps a root of its own.

The MV3R-tree proposed a structure that combines the concepts of multiversion B-trees and 3DR-tree. Although the MV3R-tree can deal efficiently with both time-slice and time interval queries by combining the benefits of both structures the MV3R-tree requires more space than the 3DR-tree and still needs high management cost due to the combination of the MVR-tree and the 3DR-tree.

LUGrid[21] is Lazy-Update Grid-based index adapts the grid file with two important features, namely, lazy insertion and lazy deletion. Lazy insertion reduces the updateI/Os by adding an additional in-memory structure layer over the disk index. Therefore, a batch of updates can be flushed to disk at one time, and consequently the cost of multiple updates is amortized. Lazy deletion reduces

update cost by avoiding deleting single obsolete entry out of the index immediately. Instead, this way, the obsolete entries can be referred from the current entries in the memo and are lazily removed when their disk pages are accessed.

The RUM-tree[22] minimizes the frequent update of moving objects. The RUM-tree processes updates in a memo-based approach that avoids disk accesses for purging old entries during an update process. Therefore, the total cost for update processing is reduced. Various mechanisms are used by garbage cleaner to remove obsolete entries including a vacuum cleaner approach that regularly sweeps the space with some frequency and a clean-upon-touch approach that cleans a leaf node whenever it is fetched during an insert or update operation.

Bx-tree[23] is optimized according to the finding B+-tree, which is dynamically selected according to the number of objects in the dataset. It divides two partitions. The maximum update interval in the Bx-tree is the same as the optimizing time interval time unit of the TPR-tree.

The Bdual-tree[24]uses the same B+-tree as the Bx-tree and also has two partitions. The order of the Hilbert-curve is also optimized as that of the Bx-tree, but one degree smaller since the Bdual-tree partitions two more dimensions(the velocity dimension).

Our proposed ACII structure distinguishes itself from all other approaches where it has all the following properties: (1) ACII indexes the current positions of moving objects, no predication scheme is used, (2) ACII is based on the cell space belong to graphic model structure where cells are not equal sized, cells can adapt to data distribution through cell structure that is room, hallways, floor and etc. on, (3) ACII efficiently resolves the update frequently using the cell space in indoor. Thus, no overhead or I/O is reduce due to update.

3. Design of the ACII

In this section, we describe what assumptions we make and present a sample applications scenario. Types of queries the proposed ACII index aims at are also described.

3.1 Design Model

We use the term indoor partition to indicate a room, a hallway, or a staircase. Semantically, every indoor partition is a smallest piece of independent space that is connected to others with one or more doors.

Theodoratos et al. proposed a specification and classification scheme for Indoor spatio-temporal Model, and suggested a list of specifications that can be followed by an Symbolic Model[9]. They addressed issues on position queries, nearest neighbor queries, navigation, and range queries. A few existing proposals such as set-based, hierarchical and graph based models are presented. We adopt the semantic Location model[25] to precisely describe the specifications of the ACII index.

Table 1. Specifications of Semantic Location model and compares with other existing Symbolic models.

	Supported Coordinates		*Supported Queries*		
	Sym	*geom*	*P*	*R*	*N*
Set-based	*Yes*	*No*	*Good*	*Good*	*Basic*
Graph-based	*Yes*	*No*	*Good*	*Basic*	*Good*
Hierarchical	*Yes*	*No*	*Good*	*Good*	*Basic*
Combined symbolic	*Yes*	*No*	*Good*	*Good*	*Good*
Subspace(hybrid model)	*Yes*	*Yes*	*Good*	*Good*	*Good*
Partial suspace(hybrid model)	*Yes*	*Yes*	*Good*	*Good*	*Good*

Table 1 summarizes the specifications of Semantic Location model and compares with other existing Symbolic models. It classifies the location models with respect to the supported coordinate type(sym = symbolic, geom=geometric), the supported queries(P=position, R=range, N=nearest neighbor). The ACII index considers point objects moving in a discrete manner within a Cell space using semantic

model. In this paper, we assume that an object moves from Cell to Cell space. Our approach can be easily expanded into an others positioning system.

3.2 An application scenario

To facilitate a variety of applications, positioning systems are deployed in indoor settings. For instance, Bluetooth, RFID and wireless positioning are deployed in airports to support real-time monitoring of delays as well as off-line flow and space usage analyses. Such deployments generate large collections of tracking, tracking and recording the changing positions of objects are be-coming increasingly feasible. The need for indexing moving objects and processing various types of spatial-temporal queries arises in Indoor LBS(Location Based Service), including large collections of tracking data in the Building, shopping malls, conference facilities, airports, and other transportation infrastructures[26]. As an example consider the User Safety systems, which monitor the location and motion patterns of users in order to provide services such as congestion detection, optimal route guidance and so on. Each user is equipped with a smart phone or embedded device, and automatically and periodically transmits its position to a server at regular intervals using either RSSI(Received signal strength indication) in WLAN. The server stores these current positions of all users which constitute the trajectory of moving object, and processes spatial-temporal queries. Our objective is to minimize the management cost and the storage requirement for such an application. In addition, we aim at efficient processing of the following types of queries: position, range, kNN(nearest neighbor), and trajectory query.

- Position query retrieves all moving objects within a certain region at a specific Cell space.
- Range query returns all objects within a certain region at a specific Cell space.
- Nearest Neighbor query retrieves the n objects closet within a certain position at a specific Cell space.
- Trajectory query retrieves (partial) trajectories of moving objects that satisfy a certain condition specified in a spatio-temporal domain.

4. The structure of the ACII

Formally, in our proposed ACII index structures, moving objects are modeled within Symbolic Cells. A Symbolic Cell of locations $S = (V, E)$, where V denotes all cell locations c_i and $<c_i,c_j> \in E$ *iff* $c_i \backslash c_j$, is a rooted tree. Any vertex V connects with the root. V is a hierarchy leveled, where V *denotes all edge* $<e_i, e_j> \in E$ *iff* $e_i \rightarrow e_j$. The level of each vertex v is defined as:

0, if V is a root cells; or $\min_c$ $c.level+1$, $c \in V$ *and* $<c,v> \in E$

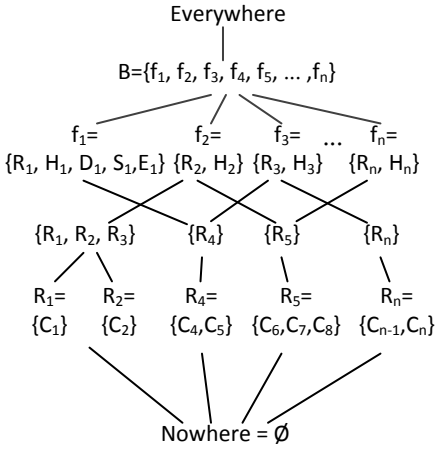


Fig. 2. Extend Symbolic Model

Fig. 2 shows an example of such a building model. The set of location l consists of the building B, the floors f_1, …,f_n and several R includes a room, hallway, door, stair and elevator, etc on. C_1,…C_n explains the Cell of such as location space. It shows relationship of the Symbolic model to the set-based approach. Location can be interpreted as sets of symbolic coordinates.

A moving object M moves in a Fourth-dimensional space $B \times F \times C \times T$, where $B = [b_0, b_{n-1}]$, $F = [f_0, f_{n-1}]$, $C = [c_0, c_{n-1}]$ and $T = [t_0, t_{n-1}]$. The fourth-dimensional space is partitioned into cells such that

$$B = [\, b_0, b_1) \cup \cdots \cup [b_{n-2}, b_{n-1}],$$
$$F = [\, f_0, f_1) \cup \cdots \cup [f_{n-2}, f_{n-1}],$$
$$C = [\, c_0, c_1) \cup \cdots \cup [c_{n-2}, c_{n-1}],$$
$$T = [\, t_0, t_1) \cup \cdots \cup [t_{n-2}, t_{n-1}].$$

Where t_p means timestamp. The lifespan of each cell is (t_p, t_{i+1}). The position of M at a certain Cell space in a time dimension T is a point in a fourth-dimensional space $B \times F \times C \times T$. Thus, the trajectory of M is a Cell space in a fourth-dimensional space $B \times F \times C \times T$. The part of the trajectory is defined as follows:

$$m(u) = \bigcup_{i=0} tr_i,$$

Where tr_i = {<b_i, f_i, c_i, t_i>, <b_{i+1},f_{i+1},c_{i+1},t_{i+1}>} and u is an identifier of group of Cell spaces. An element <b_i, f_i, c_i, t_i> is the ith version of cell position of $M(id)$. Thus, the trajectory of $M(id)$ is a set of groups of cell space $m(u)$, and is modeled as follows:

$$M(id) = \bigcup_{u=0} m(u),$$

4.1 ACII indexing Structure

ACII adopts a cell structure that is similar to the building-based structure. In ACII, however, the directory of cells is maintained in memory instead of being stored on disk. Also, we extend the cell directory to buffer object updates. We refer to the extended in-memory directory as the Memory Cell, and refer to the set of in bucket pages in the Memo. **Fig. 3** illustrates the structure of the ACII index. R-tree is configured using the topology of indoor space. Each bucket of MC consists Cell space. Each MC space to a Memo where it's flushed data is stored. The leafnode of R-tree is one of cell space. For a Cell(room, hallway and indoor space) corresponding MC_n. Each cell includes MBR(Minimum Boundary Rectangle), Size, Distance, etc. The Ob_n that is moving objects are stored within cell space.

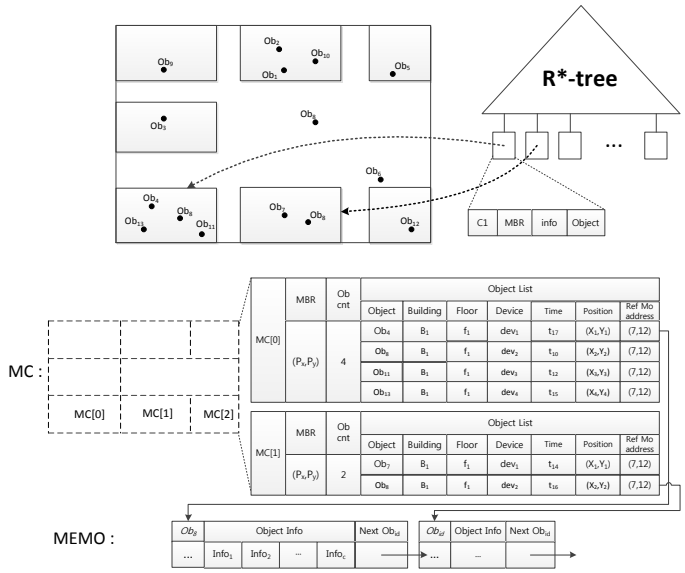


Fig. 3. Structure of the ACII index

Memo to store the trajectories information of moving objects is used, stores the bucket list in memory. Memo is managed in a way that the LRU(Least Recently Used). If the MC stored new data or the moving objects is moved from MC to another MC, the update data is stored in Memo. When the Memo bucket size is full, the trajectories moving objects data in Memo is flushed to disk. Also, the Memo can quickly stores the frequently trajectory moving objects by LRU method.

For instance, MC[0] stored a number of 4 moving objects. The ob_4, ob_8, ob_{11} and ob_{13} are moving objects in same cell. MC save the most recently inserted data. When the moving objects updated in cell space, it is stored in MC. If a moving object includes the MBR of MC, its data inserts in the MC, and the object count of MC increased.

If the same moving object is updated in same cell space, the inserted moving object information in MC is exchanged only time parameter. This technique doesn't need to continuously storage the updated moving objects, when the same moving objects inserted in same cell.

But, where a moving objects is observed in the MC[1], the moving objects is stored and check the MEMO. At this time, the first step of verification is to find the Memo bucket list of same moving objects id. If there didn't includes the same id in Memo, the Memo is created for new updated moving objects. However, if there was an existing, the moving objects are inserted into the last bucket. And existed moving objects information is eliminated in MC without MC[1].

Through, the MC by using the Memo can be reduced within overlap of the trajectory data. When the moving objects is updating in the same cell, the moving objects information in MC need to exchange the time parameter without the others value. It isn't necessary to continuously store the update data in same cell space.

4.2 Processing Update

In this section, we discuss update processing in ACII index. An update sent from a continuously moving object to the ACII contains the object identifier and the object's new location. To illustrate the basic ideas described above, let us discuss the following example of ACII that organized records with a moving object identity. The initial situation of our ACII is given in **Fig 4**. **Fig. 4** depicts an overview of update processing in ACII that distinguish 3 cases.

First, in case of inserted new moving objects, when the same Ob_{id} didn't include in MC, the moving objects is inserted in MC. Then after bucket of Memo of Ob_{id} is retrieved, if the Memo didn't store the bucket of Ob_{id}, the Memo of Ob_{id} is created.

Second step occurred to update the same moving objects in same cell space. In this cases, already had moving objects information in MC. So, in MC don't need to update the moving objects without timestamp and real-position. This method reduce the overlap of update storage by storage only timestamp.

In third case, when the moving objects is inserted with the others cell space, this moving objects is storing the other MC that is related with cell space by cell MBR. Then if the MC has the Memo of Ob_{id}, the bucket of Ob_{id} is writing in end of list bucket of Memo. The pointer of Memo exchanges from existed address to new MC address.

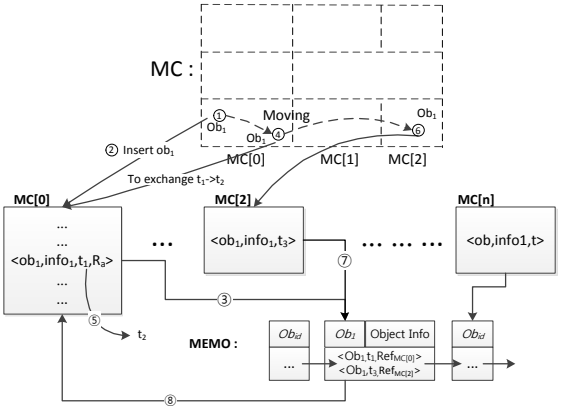


Fig. 4. For instance of Updates in ACII index

For the sake of simplicity of our example, we assume that already 3 data records are included in same building and floor. The ACII consists of MC and MEMO. The parameters of the MC are set up in the following moving object = ob1, information that includes <*Building ID, FloorID, ObjectID, deviceID* > = $info_1$, Time = t_1, Reference address = R_a. The update step will proceed as follows.

① Updated the Ob_1: the Ob1 that is moving object.

② The Ob_1 is inserted in MC[0]: the ob1 is included in MBR of MC[0], already MC is built by r-tree.

③ Memo is created: by matching the Ob_1 for management trajectory data.

④ Updated the Ob_1: To update the moving objects again in same cell space.

⑤ Timestamps: To only exchange the timestamp in MC[0] for update.

⑥ Updated the Ob_1& After inserted data: To update the moving objects in the other cell space and inserted data in MC[2].

⑦ The updated moving objects written in Memo[Ob_1]: the Memo need to update of new moving objects for managing the trajectory data. Because data is exchanged

⑧ To exchange the pointer of address and eliminate the existed Ob_1 information: the pointer of Memo of Ob_1 is changed at new MC[2]. Then existed Ob1 information in MC[0] is eliminated.

The MEMO adopts an overlapping technique[MB-tree, …]. In each cell space, several moving objects are stored at different timestamps and managed using LRU method. However, in order to save Memory or Disk, common data records of MEMO may be combined without data replication, if the contents of two buckets are identical.

Table 2 shows a more detailed example. It assumes that only 4 buckets are allowed in MEMO, and 4 blocks can reside in the MEMO memory. In this example, 6 moving objects are inserted belong to time. The Cache memory does not influence the insert pattern, and these writes evoke 6 disk accesses in the LRU. If we apply MEMO LRU for exactly the same inserted moving objects pattern, we can reduce the disk accesses counts to three.

Table 2. Comparison of LRU and LRU of MEMO

Update a Moving Objects		*LRU*		*MC & MEMO LRU*	
t	*Objects*	*Cache Memory(C1,C2)*	*LRU of 4 bucket Disk Block*	*MC of Cache Memory*	*MEMO(4 bucket) Disk Block*
t_1	C_1∋A,B,C,D C_2∋ ∅	C_1 : D,C,B,A	[D_{t1}],[C_{t1}],[B_{t1}],[A_{t1}]	C_1 : D,C,B,A	[D_{t1}],[C_{t1}],[B_{t1}],[A_{t1}]
t_2	C_1∋E,F C_2∋ ∅	C_1: F,E,D,C,B,A C_2:	[F_{t2}],[E_{t2}],[D_{t1}],[C_{t1}] D←{A, B}	C_1: F,E,D,C,B,A	[F_{t2}],[E_{t2}],[D_{t1}],[C_{t1}] D←{A, B}
t_3	C_1∋A,B,F C_2∋ ∅	C_1: F,B,A,E,D,C C_2:	[F_{t3}],[B_{t3}],[A_{t3}],[E_{t2}] D←{C, D} M←{A, B}	C_1: F,E,D,C,B,A	[F_{t3}],[B_{t3}],[A_{t3}],[E_{t2}] D←{C, D} M←{A, B}
t_4	C_1∋ ∅ C_2∋B	C_1: F,A,E,D,C C_2: B	[B_{t4}],[F_{t3}],[A_{t3}],[E_{t2}]	C_1: F,E,D,C,A C_2: B	[B_{t4}],[F_{t3}],[A_{t3}],[E_{t2}]
t_5	C_1∋ ∅ C_2∋E,B	C_1: C,D,F,A,F C_2: B,E	[B_{t4}],[E_{t5}],[F_{t3}],[A_{t3}] D←{B}	C_1: F,D,C,A C_2: E,B	[B_{t4}],[E_{t5}],[F_{t3}],[A_{t3}]
t_6	C_1∋F C_2∋E,B	C_1: F,A,C,D C_2: B,E	[F_{t6}],[B_{t6}],[E_{t6}],[A_{t3}] D←{E}	C_1: F,D,C,A C_2: E,B	[F_{t6}],[B_{t6}],[E_{t6}],[A_{t3}]
t_7	C_1∋ ∅ C_2∋A,B,C	C_1: F,D C_2: C,B,A,E	[C_{t7}],[B_{t7}],[A_{t7}],[E_{t6}] D←{E} M←{C}	C_1: F,D C_2: A,C,E,B	[C_{t7}],[B_{t7}],[A_{t7}],[E_{t6}] D←{E} M←{C}
t_8	C_1∋ ∅ C_2∋C, A	C_1: F,D C_2: C,A,B,E	[A_{t8}],[C_{t8}],[B_{t7}],[E_{t6}] D←{A}	C_1: F,D C_2: A,C,E,B	[A_{t8}],[C_{t8}],[B_{t7}],[E_{t6}]

All objects in the example insert sequence are update only once. Thus, LRU cache memory acts just like FIFO(First In First Out) queue, and does not influence update performance. Meanwhile, MEMO LRU can reduce the number of stored in Disk by not rebuilding the same cell moving objects in the bucket of MC memory.

5. Query Processing in ACII

In this section, we discuss query processing in ACII index. We discuss the processing steps for standing queries, and provide the algorithm for processing queries. **Fig. 5** depicts an overview of query processing in ACII that has the following fourth stages: according to query that is position query, range query, nearest neighbor and trajectory query.

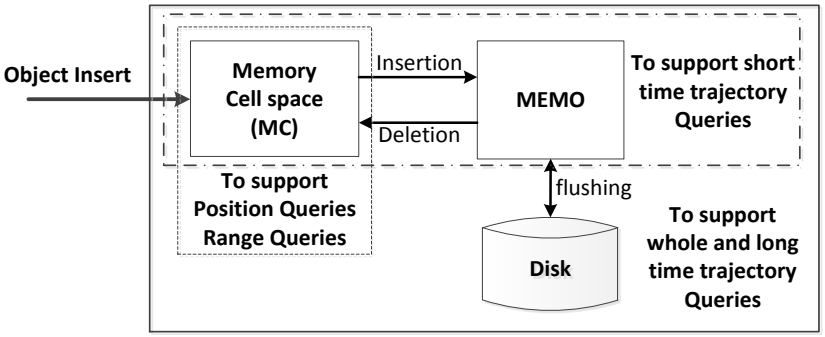


Fig. 5. Overview of query processing in ACII

Stage 1: Current Position Queries
The determination of the positions of moving object and static objects like users, buildings, hallways, etc. in indoor is a different from outdoor positioning system. The tasks described below cannot be carried without the known positions
This shows that a general location model has to support different coordinate reference position that is cell space based on. The queries of current position are processing to retrieve the alive moving objects by based-on in-memory. The alive moving objects means is active or continuously update. Therefore, moving objects that is continuously used, can retrieve in the MC and get the cell information.

Algorithm. 1. Current Position Queries

Input: Object ID = ob_{id}, t_e
Output: QueryAnswer(ret)
Procedure QueryProcessing(*m*)
01: **Begin**
02: **If** (Meta data isn't in oid) **return null**;
03: **while** MC isn't null **do**
04: The MC contains ob_{id} **then** return current cell of position.
05: MC ← MC.next;
06: **end while**
07: Bucket ← if MC doesn't contain ob_{id}, Set the disk reference pointer
08: **while** (Bucket is null)
09: To get the trajectory of ob_{id} in disk
10: Bucket ← Bucket.next
11: **end while**
12: **return** current cell of position or null
13: **End**

Stage 2: Range Queries
A range query returns all objects within a certain cell area. When a rage query processing in indoor space, it is used by feature of cell space(MBR) instead of Euclidian distance calculate in outdoor. The moving objects are returns within all of cell space that includes in range.

Algorithm. 2. Range Queries

Input: Queryarea(*rgn*)
Output: QueryAnswer()
Procedure RangeQueryProcessing(*m*)
01: **Begin**
02: Search for MC whose space coverage overlap with rgn.
03: put pointers to such MC cells into a set Set_m;
04: **For** each MC cell m referred in Set_m **end if**
05: **while** (Set_m isn't null)
06: **if** (u.cell includes rgn), QueryAnwser ← u;
07: Set_m ← Set_m.next
08: **end while**
14: **End**

Stage 3: nearest neighbor Queries
A nearest neighbor query is the search for the n objects closet to a certain cell space. For instance, a user can search for the nearest friend with respect to his current position. Its query returns within cell space. The nearest neighbor relationship on the concept is used to define the optimal distance between two entities that is a cell space.

Algorithm. 3. Nearest neighbor Queries

Input: Queryarea(*rgn*)
Output: QueryAnswer()
Procedure RangeQueryProcessing(*m*)
01: **Begin**
02: Search for MC whose space coverage overlap with rgn.
03: put pointers to such MC cells into a set Set_m;
04: **For** each MC cell m referred in Set_m **end if**
05: **while** (Set_m isn't null)
06: QueryAnwser ← to get the moving objects in cell
07: Set_m ← Set_m.next
08: **end while**
09: **return** the moving object of minimum distance based on cell in QueryAnwer.
10: **End**

Stage 4: Trajectory Queries
A trajectory query is the search for the trace of object. When a trajectory queries processing, the return value is the trace of moving objects that includes the cell identity in indoor space. The step of processing as follows.

The Ob_{id} of moving objects is retrieved in Meta file. Then, the objects are searched in MC and next searching in the Memo. Finial if the trajectories query time didn't gratify the requested time, searching in the disk. Therefore, trajectory queries can improve the retrieval time before to access the disk.

Algorithm. 4: Trajectory Queries

Input: Object ID = oid, t_e
Output: QueryAnswer
Procedure TrajectoryQueryProcessing(*m*)
01: **Begin**
02: **If** (Meta data isn't in oid) **return null**;
03: **while** MC isn't null **do**
04: The MC contains oid **then** QueryAnswer←to get the obinfo break until $t(t_s < t_{now} < t_e)$.
05: MC ← MC.next;
05: **end while**
06: MEMO ← obinfo.ref;
07: **while** (MEMO isn't null)
08: QueryAnswer←to get the trajectory of ob_{id} in memory until $t(t_s < t_{now} < t_e)$.
09: MEMO ← MEMO.next
10: **end while**
11: Bucket ← MEMO.disk
12: **while** (Bucket is null)
13: QueryAnswer←to get the trajectory of ob_{id} in disk until $t(t_s < t_{now} < t_e)$.
14: Bucket ← Bucket.next
15: **end while**
16: **return** QueryAnswer
17: **End**

---

6. Performance Evaluation

In this section, we evaluate the ACII empirically and compare with the previous work through extensive experimentation. The data set we generate moving objects, and the results of experiments are given next.

6.1 Performance Test Environment

A performance test was conducted by respective field values generated by a self-developed data generator program and generated synthetic data sets in the experimental study. All the experiments are conducted on an AMD Phenom™ 2 with Processor 3.20GHz and 512MB virtual memory of RAM. In all experiments, the number of objects varies from 100K. Objects are continuously moving with given velocities. We count the number of object processes in cycles, each cycle takes 20th. It is used to run all experiments, which were implemented in C++. The several spatio-temporal index is implemented based on the Benchmark[24].

The spatio- temporal indoor data generator is following a scenario where users move in building. The building has a 4 floor with 30 rooms. All rooms and staircases are connected by doors to a hallway. Each user is equipped with a Smart phone, and transmits its position to the LDT server[1]. Users are initially located around in the building and are randomly moving in the building.

Table. 2. Experiment Parameters and Range of Data

Experimental Factors	*Range of Data*	*Experimental Factors*	*Range of Data*
Moving Objects	*1,000,000*	*Device*	*1,000,000*
Update Cycle	*1~10sec*	*Time*	*Milli-second*
Floor	*0 ~ 4*	*Velocity*	*1~7km/hour*

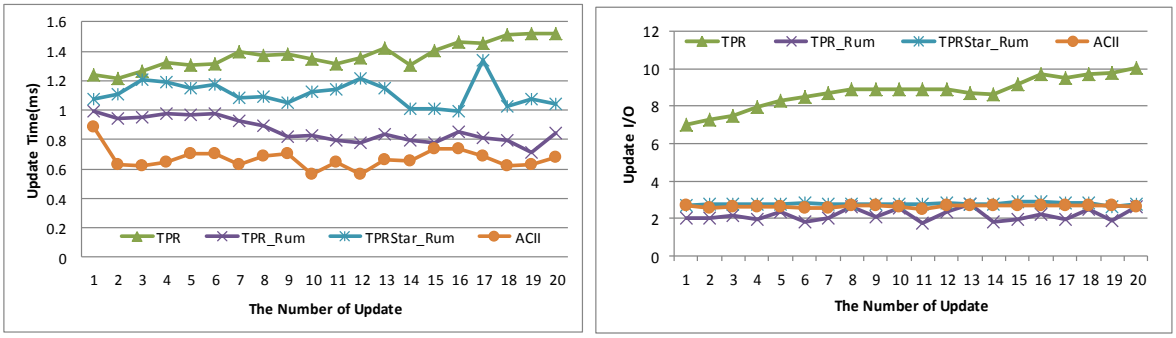
There are important three rules used to generate movements that affect the performance: First, an objects in a room can move to the hallway or move inside the room; second, an objects in staircase can move to the floor or move in the room; an object in the hallway can move in the hallway, move to

one of the Cell space. At each step, an object randomly chooses a room as the destination. All objects move with the constant speed of 4km/hour. The proposed algorithm is named after ACII which is compared with other algorithms. And I have used the RUM-tree, TPR*-tree, TPR-tree, Bx-tree, B+-tree and Bdual-tree algorithms for performance test.

6.2 Performance Test

We study the update performance of ACII and compare it with RUM studies[24]. In this evaluation, we analyzed the reuse rates of the ACII, the TPR-tree[13], TPR-tree using RUM and the TPR*-tree[14] using RUM which adopt an overlapping technique structure for update performance. For ACII, we use different velocity of moving objects. Specifically, the velocity of moving objects is set as 1, 4 and 7 km/hour. The efficiency in updates in ACII with low velocity comes from the proposed technique. When the moving objects velocity becomes slow or the MEMO buffer becomes larger, the update cost becomes lower. Because the MC can decrease update in same cell or more updates are flushed to disk at one time.

Fig. 6. shows the results of the performance that are updated according to movement of moving object. The update time is the update processing time. The number of update is cycle number to finish the update. Each index shows the average update query performance. As expected, both the update and query cost varying exchanged from update the data. This is because more objects need to be retrieved in a given query region for a larger dataset. We also observe several differences in performance among the indexes. **Fig.6 (b)** shows the Update I/O according to update data. From the figure we can see that TPR-tree is higher than others algorithms. It is because that TPR-tree doesn't use the memory.

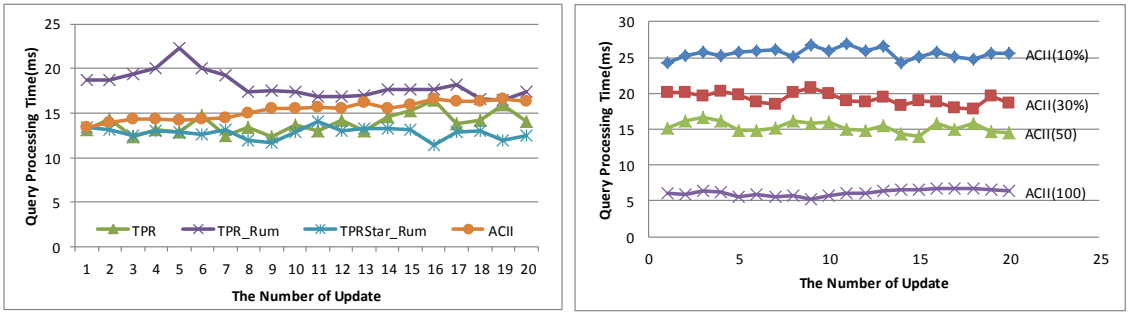


(a) Update time(ms) (b) Update I/O

Fig. 6. Effect of Update performance

In this experiment, we investigate the effect of query size by varying the square window from 1000×1000. As expected, the results in **Fig. 7.(a)**, show that the query cost of each index increases with an increasing query window size. Lager windows contain more objects and therefore lead to more node accesses. The TPR-tree have the lowest I/O cost. The query cost of the TPR_Rum and TPR*_Rum is a little higher than that of the TPR-tree mainly because of the existence of obsolete entries in the RUM-tree.

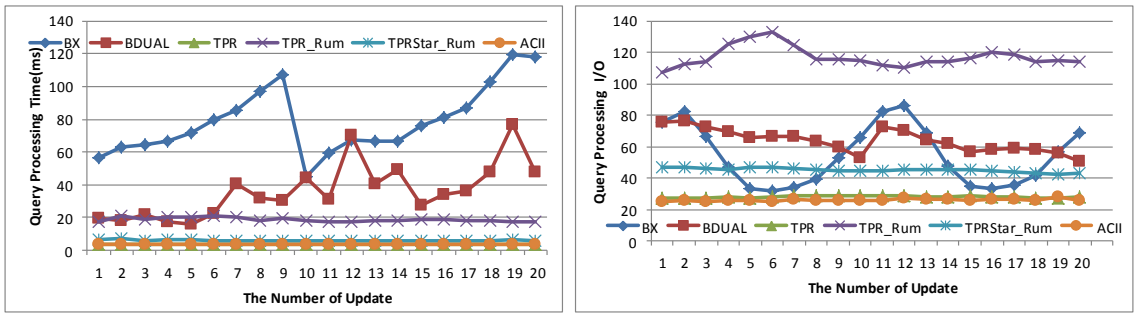
Modification: And ACII is a bit higher than that of the TRP_Rum and TPR*_Rum. However, if the matching rate is more than 50% between the scope of Cell space and range of query, ACII is better than TRP_Rum and TPR*_Rum. Because, the ACII found all objects in the Cell space that is overlapped Cell MBR and query space. **Fig. 7.(b)** show that the ACII index query cost of each percentage. The 10% explains to overlap space only 10%, and the 100% query square is same the cell space MBR.



(a) Effect of Range Query (b) Variation of matching Rate

Fig. 7. Effect of Range Query

Fig. 8 is considering kNN queries, all indexes exhibit slight increase query processing time. A kNN query is processed using incremental range queries. Due to the difficulty of tin estimating the initial search radius, the query range may be extended several times in order to find all the neighbors. **Fig. 8**. shows the same relative performance among the indexes as does **Fig. 7**(range queries). Such repeated range queries result in an increased query processing time while the I/O cost remains unchanged due to the buffer.



(a) kNN time(ms) (b) kNN I/O

Fig. 8. Effect of Number of kNN Query

7. Conclusion

In this paper, we proposed ACII; an adaptive cell-based index for moving objects in indoor. This is the study that adopts a cell-based index structure for the indoor information of moving objects, instead of outdoor index access methods such as R-trees. We have proposed an adaptive cell-based approach in indoor, called ACII, which consists of two main structures: MC and MEMO. The MC utilizes a managing technique to reduce the update frequency and configures cells adaptively belong to building structure. MEMO utilizes an overlapping technique to reduce the storage requirements, and uses the LRU method for efficiently managing of trajectories moving objects in indoor space. We believe that the proposed ACII techniques can be applied to other indoor positioning system.

References

[1] S. S. Shin, S. O, Kim, J. Y, Du, T. S, Kim and S. H. Kim, "The development of an indoor and outdoor navigation system," in *Proc. of 17th ITS World Congress*, Busan, 2010.

[2] B. GreBmann, H. Klimek and V. Turau, "Towards ubiquitous indoor location based services and indoor navigation," *Positioning navigation and communication Workshop on IEEE*, pp.107-112, 2010. Article(Crossref Link)

[3] M. Segura, V. Mut, and C. Sisterna, "Ultra wideband indoor navigation system," *Radar, Sonar & Navigation, IET JOURNALS*, pp.402-411, 2012. Article(Crossref Link)

[4] C. S. Jensen and H. Lu, "The great indoors: A data management frontier," in *Proc. of the Second Workshop on Research Directions in Situational-aware Self-managed Proactive Computing in Wireless Adhoc Networks*, May.2010.
[5] H. Lu, B. Yang and C. S. Jensen, "Spatio-temporal joins on symbolic indoor tracking data," *27th IEEE International Conference on Data Engineering*, pp.816-827, 2011. Article(Crossref Link)
[6] C.S. Jensen, H. Lu, and B. Yang, "Graph model based indoor tracking," *Mobile Data Management: Systems, Services and Middleware*, 2009. Article(Crossref Link)
[7] F. Barcelo-Arroyo, M. Ciurana, I. Watt, F. Evenou, L. D. Nardis and P. Tome, "indoor location for safety applications using wireless networks," in *Proc of the 1st ERCIM Workshop on Mobility*, 2007.
[8] H. M. Khoury and V. R. Kamat, "Evaluation of position tracking technologies for user localization in indoor construction environments," *Automation in Construction 18*, pp.444-457, 2009. Article(Crossref Link)
[9] C. Becker and F. Durr, "On location models for ubiquitous computing," *Personal Ubiquitous Computing,* Jan. 2005. Article(Crossref Link)
[10] W. I. Choi, B. K. Moon and S. H. Lee "Adaptive cell-based index for moving objects," Data & Knowledge Engineering, Vol.48, No.1, pp.75–101, 2004. Article(Crossref Link)
[11] Y. Theodoridis, M. Vazirgiannis and T. K. Sellis, "Spatio-temporal indexing for large multimedia applications," in *Proc. of the 3rd IEEE Conference on Multimedia Computing and Systems*, pp.441–448, 1996. Article(Crossref Link)
[12] M. A. Nascimento and J. R.O. Silva, "Towards historical R-trees," in *Proc. of the 1998 ACM Symposium on Applied Computing*, pp.235–240, Feb.1998. Article(Crossref Link)
[13] S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez, "Indexing the positions of continuously moving objects," In *Proc. of ACM SIGMOD*, pp.331–342, 2000. Article(Crossref Link)
[14] Y. Tao, D. Papadias, and J. Sun, "The TPR*-tree: an optimized spatio-temporal access method for predictive queries," In *Proc. of VLDB*, pp.790–801, 2003. Article(Crossref Link)
[15] B. Yang, H. Lu, and C. S. Jensen, "Scalable continuous range monitoring of moving objects in symbolic indoor space," In *Proc. of the 18th ACM conference on Information and knowledge management*, 2009. Article(Crossref Link)
[16] R. Want. "RFID Explained: A primer on radio frequency identification technologies," *Morgan and Claypool*, 2006.
[17] E. Welbourne, N. Khoussainova, J. Letchner, Y. Li, M. Balazinska, G. Borriello and D. S. Cascadia, "A system for specifying, detecting, and managing RFID events," In *Proc. of MobiSys,* pp.281–294, 2008. Article(Crossref Link)
[18] D. Pfoser and C. S. Jensen, "Capturing the uncertainty of Moving-object representations," in *Proc. of the Advances in Spatial Databases*, pp.111–132. Article(Crossref Link)
[19] Y. Manolopoulos and G. Kapetanakis, "Overlapping b+-trees for temporal data," in *Proc. of the 5th Jerusalem Conference on Information Technology*, pp.491–498, 1990.
[20] T. Tzouramanis, M. Vassilakopoulos and Yannis Manolopoulos, "Overlapping linear quadtrees: a spatio-temporal access method," in *Proc. of the 6th ACM International Workshop on Geographical Information-Systems*, pp. 1–7, Nov. 1998,. Article(Crossref Link)
[21] X. P. Xiong, M. F. Mokbel, and W. G. Aref, "LUGrid: Update-tolerant Grid-based Indexing for Moving Objects." in *Proc. of the 7th International Conference on Mobile Data Management IEEE Computer Society*, 2006. Article(Crossref Link)
[22] X. P. Xiong and W. G. Aref, "R-trees with Updated Memos," in *ICDE*, 2006. Article(Crossref Link)
[23] C. S. Jensen, D. Lin, and B. C. Ooi, "Query and update efficient B + -tree based indexing of moving objects. InProc," *VLDB*, pp.768–779, 2004. Article(Crossref Link)
[24] Y. Tao and X. Xiao, "Primal or dual: which promises faster spatiotemporal search?," *The VLDB Journal*, pp.18. Article(Crossref Link)
[25] H. Hu and D. Lee, "Semantic location modeling for location navigation in mobile environment," in *Proc. of the 2004 IEEE international conference on mobile data management,* 2004. Article(Crossref Link)
[26] S. Chen, S. Christian, Jensen, and L. Dan, "A benchmark for evaluating moving object indexes," in *Proc. of VLDB Endow*, 1574-1585, Aug. 2008. Article(Crossref Link)

Soong Sun Shin Received B.S. degree from Seowen University, South Korea, in 2006, and M.S. degree from Inha University, South Korea, in 2008. He is currently a Ph.D. Candidate with the Department of Computer Science & Information Engineering at Inha University, S. Korea. His research interests include spatial DBMS and Indoor LBS for ubiquitous environments.

Gyoung Bae Kin is an Associate Professor at Seowon University, Korea Republic. He received a B.S. degree from Inha University, Korea Republic, in 1992, an M.S. degree from Inha University, Korea Republic, in 1994. and a Ph. D in Computer Science and Engineering from Inha University, Korea Re public, in 2000. He worked as a senior researcher from 2000 to 2004 and he worked as an assistance professor at Seowon
University from 2004 to 2009. Prof. Kim's areas of interest include moving object
databases, storage systems etc.

Hae Young Bae is a Professor at Inha University, Korea Republic. He received a B.S. degree from Inha University, Korea Republic, in 1974, an M.S. degree from Yonsei University, Korea Republic, in 1978, and a Ph. D in Computer Engineering from the Soongsil University, Korea Republic, in 1989. He worked as the dean of the Graduate School of Information Technology and Tel ecommunication at Inha University from 2004 to 2006, and he worked as the dean of the Graduate School at Inha University from 2006 to 2009. Prof. Bae's areas of interest include spatial databases, multimedia databases etc.