Modeling and Querying Moving Object Trajectories Using Dimension Reduction

*Time and space are modes by which we think and not conditions in which we live*

Albert Einstein (1879-1955)

2.1 Introduction

The advent of modern monitoring applications such as location based services, presents several new challenges when dealing with continuously evolving spatio-temporal information. Location Based Services (LBS) is one major area in spatio-temporal data management, which involves the ability to find the geographical location of mobile devices and provide services based on this location information [18Prasad]. Location Based Services can offer tremendous benefits in the security informatics area. For example, tracking the location of a person who needs urgent help, or of a criminal who is wanted or giving proper guidance for a traveler whose vehicle is entering into critical area such as insurgency, terrorist attack and communal violence. In the case of emergency calls (eg terrorist attack, robbery and murder) it is obvious that if the call responders have the information concerning the location of the people making the call, then the rescue time can be get reduced.

The spatio-temporal data analysis frame work and related computational methods discussed in [19Zeng2009] are relevant to many security-related and tourism applications in the context of transportation systems. For instance, recent crime analysis in security informatics has discovered that criminals are increasingly utilizing various types of vehicles to assist their criminal activities. Mining and analyzing spatial, temporal or spatio-temporal interaction patterns from transportation data can provide many valuable insights to facilitate crime fighting and counter terrorism efforts. For example, spatio-temporal analysis can help to identify emerging hotspots of crime activities in a sensitive area, which in turn provides useful information to help border control agencies to efficiently allocate...
patrol resources. At the same time by identifying such hotspot area, if the system could give appropriate alarm to vehicles moving in that area it would be an important security measure for people moving in that vehicle.

One of the popular services under heavy demand is the location-based service (LBS) that exploits the spatial information of moving objects per temporal changes. In order to support LBS well, in this chapter, the researcher investigates how spatio-temporal information of moving objects can be efficiently stored by reducing the data into one dimension and also how this dimension reduction is advantageous in identifying sensitive areas on road networks. The proposed system also explains how to provide security alarm to objects currently in an emergency area using, the trigger concept available in relational database systems.

It is assumed that the database stores the complete history of moving objects through time and must answer queries about any time in the history of objects. Here each database record has the format \((oid, location, time)\), where \(oid\) identifies an object, \(location\) is the spatial coordinates\((x, y)\) represented as binary string (to be explained in section 2.3), and \(time\) indicates the time in which the object remained at position \((x, y)\). A typical domain where such a model fits is mobile device tracking, e.g., of GPS, PDA, or wireless phone devices.

The research proposal made in this chapter also contains a trigger which will fire on updating the moving object database by an object traveling on road network when it enters/crosses a pre-defined sensitive area. The trigger will provide alarming message to the object traveling on the vehicle as well as it will provide information to highway police (as discussed in section 2.5). There are many products for vehicle tracking system which uses GPS receivers to identify user’s location and managing data in multi-dimensional way using GIS packages leading to complexities in data analysis. We propose a system which maintains a remote moving object database where information is stored with reduced dimension in a relational database. As the
method follows dimension reduction, the data management becomes easier and the system will provide fast response.

This chapter is organized as follows. Section 2.2 contains the related work on storing location information on road network using coordinates and dimension reduction. Section 2.3 describes the necessary method available to express location information using hierarchical administrative district as of binary string. The necessary relational database modeling and query processing is discussed in section 2.4. A description of the performance of the methods, comparison with similar methods is being done in section 2.5. In section 2.6 an illustration of the proposed security alarming scheme is described with necessary logical system diagram and needed algorithms.

2.2 Related Work

There are many index structures such as 3DR-tree [33Theodoridis1996], HR-tree [34Nascimento 1998], STR-tree [35Pfoser1999], TB-tree [36Pfoser2000], or MV3R-tree [37Tao 2007] which represent location information of moving objects as (x,y) coordinates in two-dimensional space, and index them using R-tree [38Guttman1984] structures with at least three dimensions where time is the third dimension. The size of these indices rapidly increases as time passes, resulting in non-negligible search cost. In addition, since these schemes use geometric coordinates for capturing location information, they may include locations into which moving objects cannot move as movement possible only along a road. This leads to unnecessary wastage in storage that make complexities in indexing and related operations. For instance, consider the four locations, (0, 0), (0, 1), (1, 0), and (1, 1), where only (0, 0) and (1, 1) are valid locations if we assume that road is being represented as straight line. Then, aforementioned schemes require 2 bits to represent the four locations when only 1 bit is sufficient to represent the two valid locations, (0, 0) and (1, 1). In general, dead spaces like (0, 1) and (1, 0), waste a lot of storage spaces. Therefore, to remedy these shortcomings, recent developments
try to exploit location information along with the road network in real world, or to reduce dimensions of indices by separately storing the dimensions of moving objects.

The moving point objects in many cases do not move freely in the 2D plane but rather within spatially embedded networks such as roads. We can then represent movements relative to the network rather than to the 2D space. So instead of describing position of an object by geographic coordinates we can describe it as being at kilometer 220.30 with respect to an origin and towards a particular direction on a particular highway \cite{Jensen2003, Hage2003}. Research using road network \cite{Gupta2004} limits the movable boundary of objects to roads so that it represents location information represented as a pair of (the identifier of the nearest road, the distance from the nearest road). In this scheme where nodes on the road network are systematically converted into binary strings and thus various queries can be efficiently handled by utilizing simple operations on the binary strings (e.g., Hamming distance). In this scheme, however, the length of binary string is proportional to the number of nodes in the road network, and the relative locations of roads and moving objects are not easily obtainable from the binary strings of road network nodes. An index structure \cite{Papadias2003} suitable for such road coordinates, proposes an algorithm that uses the network distance, instead of the Euclidean distance, to measure the distance between two coordinates. Some of the representative works in this area\cite{Almeida2004, Almeida2005, Chakka2003, Frentzos2003} aim at reducing the dimensions of indices to improve query processing performance. However, they still have to use every information of moving objects, which are at least three-dimensional, for query processing.

The method proposed in \cite{Pfoser2003, Pfoser2005} transform two-dimensional locations into one dimensional representation which uses Hilbert Curve\cite{Faloutsos1989} for the transformation process. Since this method does not consider hierarchical administrative district during the transformation, it does not fit well the real world situations. The binary string representation proposed
[32SangYoonLee2007] however, is based on the information of hierarchical administrative district and thus can be easily converted into address formats that are easier for human users to interpret. The proposal made in this chapter is an extension of this method that could be very useful in security informatics which can easily identify sensitive areas and also for providing trigger based security alarm to objects moving in that area. Since each district or road can easily identified by a unique pattern in the new encoding scheme and since the two dimensional spatio-temporal data has been converted into one-dimension, query processing will be easier and the PL/SQL and trigger features can be effectively used to implement the scheme.

2.3 Binary Encoding Method to store Roads and Relative Locations

In conventional approaches, the location information of moving objects were expressed as a geometric coordinate (x,y) in two-dimensional space. However, the proposal to express location information [32SangYoonLee2007] using both hierarchical administrative district and road network in one-dimensional space that fits real world better. For instance, if a moving object is in a building at a coordinate of latitude = 125.58 and longitude = -37.34, then it can be expressed as a set of fields according to an administrative district such as city, road-name, road-block (e.g., Seoul, Main road, 165th block). Furthermore, by converting the fields into a binary string that has efficient way to process queries, we focus the following advantages.

(i) Storage cost can be reduced as the proposed scheme requires to store one-dimensional data against multi-dimensional. The complexity in managing large scale multi-dimensional spatio-temporal data for indexing and query processing can be simplified.

(ii) In real world, moving objects can only follow along the “roads”. However, if one expresses location information as geometric coordinates, then one may include spaces where moving objects can
never move into, so-called dead space, incurring storage waste (Refer section 2.2).

(iii) Since the location information is specified in binary code, entire district or road-block can be easily addressed based on respective number of binary bits.

The proposed binary encoding method of moving object consists of the following procedures:

(i) Transform the real-life address of each location (static in nature like location of a supermarket) into a binary string

(ii) Represent the present location of the Moving object into a binary string

(iii) Convert the given location represented in binary string, back into real-life address.

2.3.1 Transform the real-life address of location into a binary string

As a location name consists of the administrative address which may consists of fields like district, road and location on the road we need to represent each of this component into binary string. Since this is static information we can encode and store it in the database before modeling the moving objects database which is dynamic. So we have to consider this encoding as a three step process.

(i) Mapping of district into binary string

(ii) Mapping of road into binary string

(iii) Mapping of relative location on the road into binary string

(iv) Combine the above three binary string into a single binary string which represents the actual location.

Here the last step is trivial which is just concatenating the binary strings obtained in first 3 steps into a single binary string to represent the location. The first 3 steps are discussed bellow.

(i). Mapping administrative district into binary string

We first discuss how to encode districts within states of a country. For easy illustration, let us consider an imaginary country with 4 states (A, B, C, D) as a
whole and 8 districts in each state centered by their capital cities. The simplest encoding method is to use their lexicographical orders. That is, by using 2 bits for state name and 3 bits for city name, one can encode a district as a 5 bit string whose first two bits represent the lexicographical order of its state name and the remaining three bits represent the lexicographical order of its city name.

As an illustration, one can express the location ‘‘state ‘A’ city ‘a’ ’’ as ‘‘00 000’’, the location ‘‘state ‘A’ city ‘b’’’ as ‘‘00001’’ and ‘‘state ‘B’ city ‘a’ ’’ as ‘‘01 000’’. Although this encoding scheme is simple to implement, it does not provide the relative position of cities as data is stored in lexicographical order. For example, let us consider two moving objects, one located at the city ‘‘00 000’’ and the other at the city ‘‘00 001’’. Comparing these two binary strings, one can deduce that two objects be in the same state but in different cities. These two binary strings, however, do not provide any clue as to the relative positions of the two objects.

A solution to the above problem is to use a mapping technique based on space-filling curves such as Z-ordering [^1]. A space-filling curve is a one-dimensional curve which visits every point within a multi-dimensional space exactly once. In mathematical analysis and computer science, Z-order, Morton order, or Morton code is a space-filling curve which maps multidimensional data to one dimension while preserving locality of the data points. It was introduced in by G.M Morton [^2]. The term "Z-order" refers to the order of objects along the Z-axis. In coordinate geometry, X typically refers to the horizontal axis (left to right), Y to the vertical axis (up and down), and Z refers to the axis perpendicular to the other two (forward or backward). One can think of the windows in a GUI as a series of planes parallel to the surface of the monitor. The windows are therefore stacked along the Z-axis, and the Z-order information thus specifies the front-to-back ordering of the windows on the screen. An analogy would be some sheets of paper scattered on top of a table, each sheet being a window, the table your computer screen, and the top sheet having the highest Z value. The z-value of a point in multi-dimension is simply calculated by

[^1]: Orenstein 1984
[^2]: Morton 1996
interleaving the binary representation of its coordinate values. Once the data are
sorted into this ordering, any one-dimensional data structure can be used such as
binary search trees, B-trees etc for assigning unique codes to each space.

Algorithm 2.1 describes how an administrative district can be represented and
stored as a set of binary string. The method is a recursive procedure which will
successively divide the entire region into sub-regions and finally map each district
into a two-dimensional space and then assign a binary string to each district. To
provide the relative position of districts the mapping is based on space-filling
curves such as Z-ordering.

Algorithm 2.1. Mapping administrative district into binary string

1. Compute the centroid of each district.
2. Divide the region into two sub-regions, south and north, so that the numbers of
centroids in both south and north are similar.
3. If region south has more than one centroid, divide it into two sub-regions, South-
east and south-west, so that the numbers of centroids in both south-east and
South-west are similar.
4. Do the same for region north symmetrically.
5. For each sub-region obtained from steps 3 and 4, if it contains more than one
centroid, repeat steps 2 to 4.
6. Considering the division process undergone, map each city onto a two-
dimensional space.
7. Using a Z-ordering, assign a binary string to each district.

An illustrative example of the algorithm is given in Figure 2.1 shows the division
process of a state into 8 districts.

The proposed encoding based on Z-ordering produces more informative binary
strings. Let us consider two moving objects, one located at the district 00 000 and
other located at 00 001. In addition to the fact that two objects are in the same state
but in different districts, we can infer that (1) since the first two bits for districts are
all 00, the districts are located at southwest area of the state, and (2) since the last
bit is different, the district where the first object is located is south of the district
where the second object is located.
Figure 2.1 Mapping Administrative District into Binary String

(a): Each district is labeled by its centroid (a, b,...h). (b): The hierarchical division process is numbered as 1, 2, 3 and 4 are shown. (c): shows the mapping of the regions into two-dimensional space. (d): Binary string conversion of each district where the relative ordering is based on Z-ordering.

(ii) Mapping of Road into Binary String

Algorithm 2.1 can be modified to map road within each district into binary string by considering region for district and road as line. The changes needed to be made to Algorithm 2.1 are as follows: (1) every instance of word ‘‘region’’ is to be replaced with word ‘‘district’’ and (2) every instance of word ‘‘district’’ is to be replaced with word ‘‘road’’. The modified procedure is shown in Algorithm 2.2 where each road is being assigned with a unique binary string.

Algorithm 2.2 Mapping Road into binary string

1. Compute the centroid of each road.
2. Divide the district into two sub-district, south and north, so that the number of centroids in both south and north are similar.
3. If district south has more than one centroid, divide it into two sub-district, south-east and south-west, so that the number of centroids in both south-east and south-west are similar.
4. Do the same for district north symmetrically.
5. For each sub-district obtained from steps 3 and 4, if it contains more than one centroid, repeat steps 2 to 4.
6. Considering the division process undergone, map each road onto a two-dimensional space.
7. Using Z-ordering, assign a binary string to each road.
However, unlike in Algorithm 2.1, administrative district information that is represented as a “region”, road information is represented as a “line” in Algorithm 2. Therefore, when one uses the center points of roads to represent the location of moving objects, it can be erroneous for curvy roads whose center points may correspond to multiple points.

To overcome this limitation, one needs to partition road as follows:

i. Partition road into sub roads using crossroads. For instance, the roads of Figure 2.2(a) can be partitioned into ten sub-roads using four crossroads (circles in the figure).

ii. Partition curvy or crooked roads into a combination of straight roads such that each straight road is stretched furthest. For instance, Figure 2.2 (b), represent the partition points upon the crossroad and curvy road, respectively.

iii. Finally, if roads are across multiple administrative regions, partition them at the boundary line of regions (refer to Figure 2.2 (c)).

![Figure 2.2. Decomposition of roads into a set of manageable units.](image)

(a) (b) (c)

Figure 2.2. Decomposition of roads into a set of manageable units. (a) Divide the roads at their intersections, (b) express the curvy roads as a list of straight roads, (c) divide the roads at the boundary line.

When a line is too curvy so that the centroid finding will be difficult then it can be partitioned and may be treated as different lines with separate code strings. Once all roads are partitioned into sub-roads, their mappings to binary strings are performed using Algorithm 2 and then combining this with binary strings mapped from
administrative districts. For example the binary string 00000010 which represents a road in district A (code of A is 00000) where code for the road is 010.

(iii) Mapping of relative location on the road into binary string

Now let us consider how to encode the location on road. We first partition a road into $2^n-1$ units of the same size, and then represent each boundary as an $n$-bit binary string as shown in Figure 2.3. Then, we choose the boundary nearest from an object and use its binary string as the location of the object on the road. For example let us take $n=2$, then we divide the road into $2^2-1=3$ units, where boundary of each partition is encoded as 00,01,10,11(00 is the starting boundary).

![Figure 2.3](image)

**Figure 2.3** A road which is partitioned into $2^n-1$ units of the same size

As explained in the last step, the code for location in a road will be concatenated with the binary code for the road and the binary code of the district.

### 2.3.2 Geo-Challenges during the mapping of location information

The major source of error that degrades the accuracy of our proposal is that the location of moving objects on roads may not be accurately mapped to points on the roads. This is mainly caused by the following reasons:

i. Normally GPS will allow error margin of ±10m for an object. Sometimes Government agencies purposefully add an error level to the GPS data, to counter terrorism[143Raju2005].

ii. When moving objects are not located in the middle of roads, the proposal may incur error margin up to the width of the road. This is because roads are only represented as straight lines in this scheme. For instance, Figure 2.4(a) illustrates the locations of moving objects collected by GPS. If one simply relies on the collected location information, the locations of the moving
objects will be represented as in Figure 2.4(b). However, by considering error margins, Figure 2.4(b) must be corrected to Figure 2.4(c).

![Figure 2.4](image)

**Figure 2.4 GPS errors and their corrections.** (a) Locations of moving objects collected by GPS, (b) mapping onto road networks without error correction, (c) mapping onto road networks with error correction.

iii. Another source of error is the granularity of partition. The same issue is important in the traditional two-dimensional approach of (x,y), but becomes more important in this proposed scheme. For instance, consider the partition of a road of 320 m long into 16 sub-roads (20 m each). Although two objects are located in the same sub-road, it is possible that they can be as far as 20 m apart each other – non-negligible distance in some LBS applications. One can increase the granularity of partition further to avoid such a problem at the cost of longer binary strings and thus increased storage. The algorithm **XY2BS** discussed in section 2.3.3.1 will map an (x,y) co-ordinate properly into an available location on the road.

iv. The last source of error in measuring accurate location is when objects intersecting among various regions. The binary string scheme uses the center points of objects in representing their locations. Therefore, even if an object is intersected by many regions, only one region to which the center point of the object belongs represents the object, causing errors in the
search. For instance, consider the city hall intersected by many regions. Now, we want to find out which rooms Sam has walked around in the city hall. Even though Sam has walked into several rooms of the city hall, since the address of the city hall is only represented as its center point, one cannot track down which rooms he has visited. One way to remedy this limitation is to use a range of binary strings such as “000000–000011” for objects intersected by many regions (especially static objects like buildings).

The proposed encoding scheme has the following characteristics:

a. One can find out the lowest common administrative district by extracting the longest common prefix of a given set of binary strings.

b. A district containing a set of lower districts can be represented by the range of binary strings; for example, county “A” in Figure 2.1 is represented by the range [00000, 00111].

2.3.3 System Implementation

The logical system of our proposed scheme is shown in Figure 2.5. It consists of two sub-systems one for population and other for query processing. The population sub-system is responsible for collecting the information of moving objects and storing it into databases, and the query processing sub-system is in charge of answering queries for moving objects. To support the proposed binary encoding scheme, in addition, the LBS system needs three conversion modules, XY2BS, AD2BS and BS2AD as shown in Figure 2.5.

2.3.3.1 Module XY2BS

In this proposed method, we map administrative district information and relative locations of objects on the roads into binary strings. However, not all locations of objects are correctly mapped onto roads. Therefore, we need to fix the mapping of geometric coordinates into the points on the roads. Module XY2BS converts a two-dimensional coordinate for the location of a moving object into a binary string. To
expedite the conversion process, XY2BS maintains an R-tree \cite{Guttman84} built from the roads in administrative districts. For a given road R, let bitstring(R) and rectangle(R) denote the binary string of R and the rectangle for the two end points of R, respectively. For each road R in district, the R-tree stores rectangle(R) and bitstring(R) in one of its leaf nodes.

Algorithm 2.3 will fix the position of a moving object represented by geometric coordinates as usually supplied by the GPS, onto roads. The algorithm uses an R-tree for roads to quickly convert a two-dimensional point into equivalent binary string.

**Algorithm 2.3.** Utilizing an R-tree to quickly convert a two-dimensional point, (x,y), into the equivalent binary String

1. Generate the rectangle uMBR by expanding x to its left and right by uR, and expanding y up and down by uR. uMBR is then expressed as \([x - uR, x + uR], [y - uR, y + uR]\). Here, uR is a system parameter used for determining the nearness of roads from a two-dimensional point.
2. Search the R-tree for the roads whose MBRs overlap uMBR.
3. From the roads obtained in Step 2, select the road \( R \) whose Euclidean distance to \((x,y)\) is the smallest.
4. Project \((x,y)\) onto the road \( R \).
5. Using the relative position of \((x,y)\) on the road \( R \), calculate the binary string for \((x,y)\).
6. Concatenate binary string of road \( R \) and the binary string for \((x,y)\) to obtain the final binary string representation for point \((x,y)\).

The example of Figure 2.6 further illustrates Algorithm 2.3. Figure 2.6(a) depicts the uMBR (Memory Bounded Rectangle) for a given two-dimensional point, \((x,y)\), and shows the three roads, \( R1(00000) \), \( R2(00001) \), and \( R3(00010) \), obtained from the R-tree. Since the Euclidean distance from \( R1 \) to \((x,y)\) is the smallest, \((x,y)\) is projected onto \( R1 \), which is more closed to the partition with boundary string ‘10’ as shown in Figure 2.6(b). Lastly, both binary string of \( R1 \) and that of the relative position of \((x,y)\) on \( R1 \) are concatenated to obtain the final binary string representation(0000010) of the point \((x,y)\).

**Figure 2.6 An example which illustrates how Algorithm 2.3 works** (a) Generate the uMBR by expanding \( x \) and \( y \) by \( uR \), (b) select the closest road and project \((x,y)\) onto it.

### 2.3.3.2 Modules AD2BS and BS2AD

It is preferable to ask queries by users using real-life address such as ‘‘Seoul, Main road, 100’’ than using coordinates such as ‘‘longitude = -65, latitude = 45’’. Similarly it would be fine if the system could give the answer to query in such real-
life address. Therefore, in our prototype, we assume that both users’ queries and query results are in the real address format. Module AD2BS converts these real-life addresses into equivalent binary string representations, and module BS2AD converts binary strings back to their equivalent real-life addresses[Fig 2.5]. For a rapid conversion to binary string and reverse, AD2BS and BS2AD maintains a table containing administrative address and corresponding binary string representation which is being indexed for fast searching.

2.4 Proposed ORDBMS Based Moving Object Database Model with Trajectory representation.

In this section we introduce the model of the trajectory, its construction, related database scheme and the issues related to the modifications of the MOD. Several research prototypes have been implemented with each one dedicated to various categories of problems of interest to MOD[144,Yasser2008, 145Meng2003, 146Pelekis2005, 148Brakatsoulas2004]. However on the commercial side, there have been very few database vendors that have enabled MOD like capabilities in their products. A particularly appealing commercial Object Relational Database Management System (ORDBMS) is oracle 9i [147Pelekis2005, 149Feuerstein1997] which on top the reliability of a stable and mature technology also offers:

i. PL/SQL as an environment for implementing User-Defined Functions (UDF) and specifying User-Defined Types (UDT)

ii. Triggers as declarative means to specify a reactive behavior in response to modifications to the MOD.

Before we go into the modeling of the proposed MOD database schema, the following section will introduce the concept of Object-Relational Database Management Systems (ORDBMS).
2.4.1 Object-Relational Database Management Systems (ORDBMS)

In traditional Databases, the common data types available are integer, float, character and date which are generally used for common database applications. The types of operations performed on these data types are simple arithmetic, logical and specific functions based on these set of operators. These limited set of data types and operations makes the modeling of real-world spatial applications extremely difficult. Hence, the recent advances in commercial database systems have focused on efficiently storing and managing complex information like spatial data.

Extensible database management systems \cite{pelekis2005} aim to make data management easier and more natural to users or applications such as location-based services, urban planning, utilities, transportation, remote sensing and web-content management. A simple example of spatial (or location) data is a street address. Geo-coding (a process of converting an address to a longitude/latitude coordinate pair) results in a location which can then be used to determine the spatial relationships among street addresses or between a street address and some linear spatial feature, such as a road, or an area feature, such as a census block. Roads, census blocks, county and state boundaries are more common examples of spatial data and are usually depicted in a map. A Geographic Information System (GIS)\cite{burrough1995} is often used to store, retrieve, and render this earth-relative spatial data. In such rendered map, this spatial data depicts the locations of the objects on a two dimensional piece of paper or a video monitor.

The database systems that combine the best of both the relational and object oriented databases are called the Object Relational Database Management Systems (ORDBMS). Object-relational database systems \cite{stonebraker1996} facilitate the definition, storage, retrieval, and manipulation of user-defined data types in the database through the use of user-defined functions and index methods. Thus an ORDBMS can now handle spatial information represented using a spatial object data type, and accessed or manipulated using spatial index methods and functions.
Because spatial is now just another attribute represented in the database, users can use it as another qualifier or criteria when searching or browsing the database. There are several benefits in managing the spatial and attribute data in a single database. Key benefits of this approach to spatial data management include:

- Better data management for spatial data. Users gain access to full function of spatial information systems based on industry standards with an open interface to their data (e.g., SQL).

- Spatial data is now stored in enterprise-wide database, thereby facilitating spatially enabling many more applications.

- Reduced complexity of systems management by eliminating the hybrid or file based architectures of traditional GIS-based data management schemes.

- Proprietary data structures are avoided by using an open SQL platform, thereby allowing for the seamless integration of e-business and location based services. This facilitates the task of delivering applications that meet the increasingly demanding analysis and reporting needs of a growing information and knowledge management community.

### 2.4.2 Requirements of a Spatial Database System in ORDBMS Model

Any database system that attempts to deal with enterprise GIS and location based services has to provide the following features:

i) A set of spatial data types to represent the primitive spatial data types (point, line, area), complex spatial data types and operations on these data types like intersection and distance.

ii) The spatial types and operations on them should be part of the standard query language that is used to access and manipulate non-spatial data in the system. For example, in case of relational database systems SQL should be extended to support spatial types and operations.
iii) The systems should also provide performance enhancements such as indexes to process spatial queries (range and join queries), parallel loads and queries, which are available for non-spatial data.

2.4.3 Spatial Indexing and Query Processing

Indexes help speed up the execution of SQL statements in the database by providing a faster access path to data. Spatial indexes [152 Shashi2003] are also the primary means of reducing disk I/O when manipulating the data. Databases provide standard indexing mechanisms that work with scalar data; these indexes are not suitable for spatial data. The main purpose of a spatial index is to facilitate spatial selection. That is, in response to a query, the spatial index will only search through a subset of objects embedded in the space to retrieve the answer set.

Spatial queries are often processed using filter and refine techniques. In the filter step, an approximate representation of a spatial object is used to determine a set of candidate objects that are likely to satisfy the given spatial query. There are several advantages to apply this filter and refine strategy. First, spatial objects tend to be very large and hence consume considerable amounts of main memory. An approximate representation of a spatial object takes considerably less space and time to load into memory. Second, computations on spatial objects tend to be very complex and computationally expensive. The more complex the objects, the more processing is required to compute spatial relationships. Computations using approximate objects tend to be very fast and require far less computational cycles.

2.4.4 Spatio-Temporal Indexing

Spatio-temporal databases store both spatial as well as temporal information related to moving objects. Index structures are defined over both space and time separately but there is a requirement for a single index structure which can store both the spatial and the temporal aspect. Spatio-temporal indexing can be defined for data
changing discretely over time and also for data which changes continuously. Some of the researches done in this area\cite{Sajimon2008} is described below.

Existing indices such as 3DR-tree \cite{Theodoridis1996}, HR-tree \cite{Nascimento1998}, STR-tree \cite{Pfoser1999}, TB-tree \cite{Pfoser2000}, or MV3R-tree \cite{Tao2001} represent location information of moving objects as \((x,y)\) coordinates in two-dimensional space, and index them using R-tree \cite{Guttman1984} structures with at least three dimensions. It is known that 3DR-tree and HR-tree are effective for spatial queries while STR-tree, TB-tree, and MV3R-tree are good for trajectory queries. However, the size of these indices rapidly increases as time passes, resulting in non-negligible search cost. Currently separate index structures are maintained for retrieving spatial and temporal information about a moving object. A spatial index gives the spatial characteristics of object and temporal index returns temporal characteristics. For managing spatial data, R-Trees and its variants seem to be the most efficient ones. Temporal information for these objects can be stored in a similar way in one dimensional data structure. For maintaining uniformity a single multi-dimensional data structure is selected having spatial (2DR-Tree) and temporal (1DR-Tree) index. The drawback behind this scheme is that it demands access to both the indexes and then computation of the intersection set between the two answer sets. Better solution out is merging of the two indexes (spatial and temporal) as adopted by 3DR-Tree. The proposed spatio-temporal indexing scheme consists of only one index: a three-dimensional index for complete spatio-temporal information of the objects. Being R-Tree the indexing mechanism, 3DR-Tree\cite{Yannis1996} is proposed to store spatial as well as temporal information of the objects. Advantages of this scheme is that since it is based on a unified framework, a spatio-temporal query requires to search information in a single index structure eliminating the need for performing spatial joins after querying spatial and temporal index separately in simple spatial and temporal indexing scheme.
2.4.5 System Architecture

One of the important characteristics of this problem domain (Moving Object Database Modeling) is that the majority of the queries of interest to the users are continuous \cite{sistila1997}, i.e., they are relevant to future time intervals and their answers may have to be frequently re-evaluated due to the dynamics of the entities involved. As in this proposed model a sequence of (location, time) updates are periodically transmitted to the MOD by the moving objects \cite{mohamed2004}, the continuous queries have to be re-evaluated rather frequently; as a consequence, their answer updates will have to be sent to the users, incurring a huge processing/communication overhead. So one of the objectives of the modeling is to reduce the frequency of re-evaluation of the query, upon an update operation.

There are different models for representing the motion of the objects in a given MOD\cite{hui2006}. These are (i) sequence of (location, time) frequent updates \cite{mohamed2004}, (ii) sequence of (location, time, velocity) updates\cite{sistila1997} and (iii) trajectory model \cite{wolfson2002}. We follow a combined approach which basically uses the concept of trajectory model which go for frequent updates of (location, time) by a GPS enabled Mobile unit or PDA. The advantages of taking the trajectory model in this context is that it can represent the past, present and future path of the movement of an object.

A trajectory, which is a sequence of two dimensional points in this proposal (1D geolocation, time) like (loc$_{1}$, t$_{1}$), (loc$_{2}$, t$_{2}$) . . . (loc$_{n}$, t$_{n}$) represents vertices of a polyline\cite{bart2006}. It is also assumed that between two vertices, an object is assumed to move along a straight line with constant speed. Using electronic maps augmented with the information about the distribution of the traffic patterns along the road segments \cite{hui2006}, a trajectory between a given origin and destination can be constructed using a time-dependent extension of the Dijkstra’s algorithm \cite{dijkstra1959, wolfson2002}. An advantage of this model is that the MOD has full knowledge of the more distant future. Since in our proposed binary encoding method, each road can be identified by a unique binary string, the future
moving path of an object can be projected by connecting the location codes in sequence, which are available in the database. The distance between nodes (static object like petrol pumps, motels, supermarket) can be taken from the database where the optimum distance were calculated using Dijkstra's algorithm. A location based query (where query objects are static) will be checking current location and time of the moving object with the nodes of the above projected trajectory. An update operation will call for re-evaluation of the query only when there is a change in velocity of the moving object.

In the proposed scheme[101 Sajimon2011], Moving Object Database (MOD) is an Object Relational Database in which the scheme is shown in Figure 2.7.

![Figure 2.7 Logical System Architecture of Query Procedure on Moving Object Database](image)

The scheme includes the following tables:

a) **Moving Object Table - MOT (Object-id, trajectory, ppq, pqa)**: Contains the details of the moving object trajectories, where trajectory is represented as a User
Defined Type (UDT) in an ORDBMS defined as a raw type LIST of points, having location(binary code) and time attributes. Attribute ppq represents pending posed queries and pqa represents part of query answer.

b) **Pending Query Table** – PQT (User-id, Query-id, Expires): Keeps track of the query issued by a particular user with time of expiry of the query.

c) **Query Answer Table** – QAT(Query-id, Object-id): Contains the objects which are part of the answer for a given query. This table will be automatically updated when a query is being processed.

d) **Static Objects Table** – SOT (Object–id, Type, Name, Location bit string): To keep details of static objects on roads. Type denotes the various types of static object like super market, petrol pumps, Bank ATM etc.

e) **Disturbed District Table** (DDT) described and used in section 2.5

f) **Disturbed Road Table** (DRT) described and used in section 2.5

### 2.4.6 PL/SQL Procedures for Query Processing

The procedures which enable the processing the different queries are briefly explained below.

i. **Transmit_Answer(Q,U)**: Used to transmit the answer of the Query Q to the user U who passed it.

ii. **Receive(U,Q)**: Process involved upon receiving Query Q from user U(uid) assign a unique qid to Q and inserts the tuple (uid,qid,t) into the table PQT. If uid is a mobile user it increment 1 to ppq in MOT for this uid. This module will then call Evaluate(qid) followed by **Transmit_Answer(qid,uid)**.

iii. **Evaluate(Q)**: Evaluates a query qid and obtain the answer set containing a set of uid’s. If this is the first Evaluation call for qid then each (qid,uid) tuple will be inserted in table QAT and the pqa attribute in MOT will be incremented by 1. If there is a (qid, uid) tuple in QAT in which uid is not in the answer set of the evaluation, that tuple will be removed and the pqa attribute of this uid in MOT will be decremented by 1. If the tuple (qid,uid)
is not in QAT, it will be inserted and the attribute pqa of this uid in MOT will be incremented by 1.

iv. **Evaluate_All()**: Scans the table PQT and for every query qid whose *Expires* attribute is not less than current time, invokes the Evaluate(qid).

v. **Evaluate_All_Issued(uid)**: Scans the PQT table and for every tuple for which PQT.User-id=uid and PQT.Expires> Current-time, invokes Evaluate(PQT.Query-id). If its execution caused modifications to the QAT and there is an existing instance of the procedure TransmitAnswer(qid,uid) for which qid = PQT.Query-id, the new answer set is transmitted to the user uid.

vi. **Re_Evaluate_Ans(OID)**: Scans the relation PQT and for every tuple for which PQT.Object-id=OID, it invokes the procedure Evaluate(PQT.Query-id).

vii. **Remove(QID)**: Removes the tuple for which PQT.Query-id=QID; decrement the MOT.ppq counter for the respective MOT.oid = PQT.User-id. If a tuple for this User-id is still in MOT(indication of some tuple for this user in QAT table) then remove all tuples from QAT for which QAT.query-id=QID, and decrement the ppq attribute for the corresponding user-id from MOT.

viii. **Flush(PQT)**: Periodically checks(frequency will set by the user) the PQT table. For every tuple for which the value of the *Expires* attribute is less than current_time, it invokes the Remove (PQT.Query-id) procedure.

### 2.4.7 Spatio-Temporal Triggers

In this section the researcher is describing the syntax and semantics of various database triggers which uses the above set of procedures as well as the spatio-temporal trigger “security-alarm”. The security-alarm trigger, by taking advantage over the binary encoding scheme will automatically provide messages to vehicles entering into disturbed area either to change their route or to reduce the travel speed.
Database Triggers

i. Update trigger on MOT

The update triggers will be executed only when the current location is more than two blocks over the previous location on the same road, to avoid frequent re-evaluation of the queries. This trigger is designed to capture both cases: the updated object has posed a query to the MOD; the updated object is part of an answer for a query posed by another object as explained below.

Update Trigger1

\[
\text{CREATE TRIGGER MOD_UPDATES\_1 AFTER UPDATE OF \text{trajectory} ON MOT} \\
\text{REFERENCING OLD AS OldTuple} \\
\text{WHERE MOT.ppq>0 AND abs(OldTuple.trajectory.location - trajectory.location)>2 Evaluate\_All\_Issued(MOT.oid);} \\
\]

Update Trigger2

\[
\text{CREATE TRIGGER MOD_UPDATES\_2 AFTER UPDATE OF \text{trajectory ON MOT REFERENCING OLD AS OldTuple} } \\
\text{WHERE MOT.pqa>0 AND abs(OldTuple.trajectory.location - trajectory.location)>2 Re\_Evaluate\_Answer(MOT.oid);} \\
\]

ii. Insert Trigger on MOT

Insertion of a new tuple in MOT as a new moving object, could possibly affect the answer set to every pending query in the MOD defined as follows.

\[
\text{CREATE TRIGGER MOD_INSERTIONS AFTER INSERT ON MOT Evaluate\_All();} \\
\]

iii. Delete Trigger on MOT

Upon deletion of a moving object from MOT (it will no longer become a traffic participant), any query posed by the affected object itself is no longer of interest.
CREATE TRIGGER MOD_DELETIONS AFTER DELETE ON MO
REFERENCING OLD AS OldTuple WHERE OldTuple.ppq>0
REMOVE(X) WHERE (X IN SELECT Query_id FROM Issued
WHERE Query_id = OldTuple.ID)

Trajectory Query Processing
The purpose of a Trajectory query \[^20Pfoser2000\] is to traverse and display a path of a moving object in a particular period of time. For example the query

“Where the object O5 moving around during the time interval [1 pm, 4 pm]?”.

If the system represents the locations of moving objects as conventional two-dimensional geometric points, the answer to such a query consists of a set of line segments extracted from (x,y) rectangle, which can be meaningfully displayed only on electronic maps. However the answers from the proposed binary coding system can be easily converted to real-life addresses and thus can be delivered to users in text or voice format (in addition to being useful on electronic maps as well). To process trajectory queries, the system first searches the database using the object and the trajectory attribute list the locations that satisfy the time interval. The outcome will be a set of locations represented in binary strings. The system then calls module BS2AD to convert the binary strings in the result into the corresponding administrative district addresses and finally sends out the result in text or voice format to users. While showing the result to users, the system may represent a set of adjacent rows as a single row by extracting their common prefixes.

2.5 Proposed Trigger Based Security Alarming Scheme
The environment in which the alarming system in security informatics applications \[^106Sajimon2008\] will function as shown in Figure 2.8. The two additional tables used in the system along with other objects as per Figure 2.7 are described below.
i) **Disturbed Districts Table – DDT (District binary code, ts, te, type, status):**
Contains the details of district which are sensitive due to a terrorist attack, communal violence, natural calamity, emergency declared by local administration etc. *ts* and *te* are the starting time and ending time of the disturbance occurred or occurring. *Type* shows the type of disturbance like emergency called by police, blockage due to traffic congestion, violence attack etc. *Status* field shows the current status of the disturbance. This field can be a set or reset status according to the condition of disturbance.

ii) **Disturbed Roads Table – DRT (Road binary code, ts, te, type, status):**
Similar to DDT which contains details of disturbed roads.

The MOD will be informed about the sensitive area by highway police through authorized messages or messaging system. The police people will inform the sensitive area in real-life address and algorithm 23 discussed in section 2.3.3 can be used to convert that into binary string notation and will set active status in MOD. Alarming will work for objects entering into or already within the sensitive area, and will be triggered in the next immediate update so that they can be vigilant or change their routes or stop the movement at all [Sajimon2009]. Advantage of this scheme is that a complete district, complete city or the whole single road can be set or reset to sensitive state easily as in the binary encoding scheme discussed earlier has a fixed substring to represent a district or city or road or part of a road respectively. When an object crosses a sensitive area or being in such area, an update of current location in MOD will automatically triggered by sending message to the object with sensitive location information in real-life address. A similar message will pass to highway police with information of object-id and the location it has reached in the last update.

The scheme consists of the following algorithms
**Algorithm 2.4:** Marking sensitive area on MOD

1. Read the address of the area in real-life notation and duration from security people.
2. Check the authority of the message.
3. Search on the B-tree for district/road names to find out the equivalent binary string.
4. Write the obtained binary string into district-sensitive table or road-sensitive table and set the status active.

**Algorithm 2.5:** MOD updating by a moving object at fixed intervals.

1. Get the location(x,y) of the moving object through the mobile set having GPS facility and time of updating.
2. Convert the location(x,y) into binary string using algorithm 2.3 discussed in section 2.3.3.
3. Update the trajectory field of the moving object data table (MOT) with a list record consisting of (location(as binary string), time).
4. During updation if the location is in any of the sensitive area, a trigger “security alarm” will be fired as detailed in the following algorithm.

**Algorithm 2.6:** Trigger “security-alarm”

Create or alter trigger **security-alarm** before update of objects for each row

Begin
1. If district binary substring (or road binary substring) in the location code is in District-sensitive table (or road-sensitive table)
2. If status column in district sensitive table (or sensitive-road table) is “yes”
3. If time to update match with the disturbance time from sensitive-district table (Or sensitive-road table)
   a) Search the B-tree for address of the disturbed area where binary code is the key
   b) Send a security-alarm to object through object-id with message containing Real-life address of location, city, road, part of road affected
   c) Send a security-warning to highway police through Police-id containing the object-id, road/district entered, time of entry

End

In the case of unregistered vehicles like a terrorist vehicle the tracking and messaging to police people will be possible by processing the MOD data along with the object identification data like number plate, obtained from surveillance camera installed at such sensitive locations. This will be addressed in the future work.
Alternatively the message system can be used for set of vehicles controlled for a specified purpose. Typically when company vehicles leave the domain of operation, a message can be sent to concerned authorities. This method is highly useful in oil companies where some specific area is administrated as restricted area.
2.6 Experimental Evaluation

2.6.1 Performance of the binary conversion procedure considering overhead
Since the function XY2BS, which converts the two-dimensional co-ordinate into single dimensional binary string is present in all the modules discussed above, the overhead due to this conversion process has to be estimated to see the efficiency of the methods proposed. Experimental validation proved that the overhead in converting (x,y) into binary string will be negligible for a typical LBS environment and tolerable even for an environment with a huge number of locations. Figure 2.9 shows that the total time spent for the conversion increases almost linearly with the number of two-dimensional locations.

![Conversion Time Graph](image)

Figure 2.9. Elapsed time to convert two-dimensional locations into binary string

2.6.2 Query Processing Performance
The performance of query processing[101Sajimon2011] has been evaluated with INFATI [17Jensen] data set of car movements which has been explained in section 1.10.

(i) Performance of DDL Queries
Figure 2.10, 2.11 and 2.12 shows the comparison of query processing time (msec) for insert, update and delete operations for our 2D Indexing in ORDBMS and that for 3D Indexing in spatial Database system.
Figure 2.10  Query Performance Comparison of INSERT operation

Figure 2.11  Query Performance Comparison of UPDATE operation
The above figures show that the performance of INSERT operation is slower than both UPDATE and DELETE operation in both the cases. This support the common evaluation strategy for the performance management of relational database management system\([129]\text{Afolabi2008}\). For INSERT operation the performances in both the methods are almost same. This may be due to the reason that all the pending queries have to be re-evaluated in an instant of insertion. But for UPDATE and DELETE operations our proposed model shows better performance over conventional 3D indexing. Especially UPDATE operations achieved a clear performance advantage due to the avoidance of the re-evaluation of unaffected pending continuous queries.

(ii) Performance of Trajectory Query Processing

In order to evaluate the performance of trajectory query processing \([106]\text{Sajimon2008}\), we apply the query over set of moving objects in INFATI dataset stored in (1) 3DR-tree as a triplet of (x-coordinate, y-coordinate, timestamp) and (2)
in 2DR-tree as a pair of (binary string representation of location, timestamp). Moving object identification numbers were used as a key and thus stored in leaf nodes of both 3DR-tree and 2DR-tree. The performance in both the model is compared using the evaluation metrics such as index size and query execution time.

**Measurements based on the metric ‘index size’**

The size of the 3DR-tree index and 2DR-tree index for different number of trajectories (in our design which is equal to the number of moving object as trajectory is a list type attribute in MOT table) is shown in Fig 2.13 and index size savings of 2DR is shown in Fig.2.14.

Thus on an average 2DR tree which uses binary string representation uses only 70% of storage space consumed by 3DR tree(which uses two dimensional (x,y) for location data), and thus there is 30% average savings on the storage of index size.

![Figure 2.13 Comparison of Index Size](#)
Measurements based on the metric ‘query execution time’

We shall consider the execution of the trajectory query

“Where the object O5 moving around during the time interval [1 pm, 4 pm]?”

To execute such a query, the system has to traverse through the index using the time constraint. The execution time for 2DR tree and 3DR tree for different numbers of trajectories is shown in Table 2.1 and Figure 2.15.

The performance evaluation shows that the execution time for 2DR tree is significantly better when the number of trajectories is greater than 1500.

<table>
<thead>
<tr>
<th>No of Trajectories</th>
<th>Execution Time(msec)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3DR-tree</td>
</tr>
<tr>
<td>500</td>
<td>450</td>
</tr>
<tr>
<td>1000</td>
<td>830</td>
</tr>
<tr>
<td>1500</td>
<td>1300</td>
</tr>
<tr>
<td>2000</td>
<td>1780</td>
</tr>
<tr>
<td>2500</td>
<td>2180</td>
</tr>
</tbody>
</table>

*Table 2.1 Trajectory Query Execution Time*
The summary of the performance study are as follows. Mapping three dimensional trajectory data to lower dimensions using binary encoding scheme to store and manipulate in an ORDBMS proved beneficial in terms of query processing and update operations. The extra cost arise due to the mapping into binary code has also proved scalable in terms of the number of trajectories concerned. The proposed scheme is relatively simple in its implementation as it uses simple relational schema structure instead of complex storage structures as used in a pure GIS package\cite{Pfoser2003}. Another advantage is the convenience in using the powerful Query Processing and Trigger based procedure building capabilities available in a conventional RDBMS package. In application domain which involves tracking of moving objects on restricted paths like highways a carefully extended implementation model of this proposal could be a better alternative in terms of cost and operational effectiveness.

2.7 Conclusion

Dimension reduction is one of the challenging problems in multidimensional data management which has many applications in area that handle large volumes of data. The proposed moving object database model defined in constrained networks uses one of the dimension reduction methods over large scale spatio-temporal data with
the concept of binary encoding techniques. This spatio-temporal database could be managed using off-the-shelf database methods for indexing and query processing of trajectory data. More specifically, this work proposes an approach that maps two-dimensional network to one-dimensional intervals. Thus, the dimensionality of trajectories can be reduced from three to two (binary coded location data and time). The performance study presents results for varying sets of queries and trajectory data. The topic also discusses a classical application area of the proposed concept in the area of security informatics in the form of trigger based security alarming scheme. The coming chapters discuss how this dimension reduction techniques of trajectory data has application in finding spatio-temporal similarity measure and trajectory clustering.