Choosing R-tree or Quadtree Spatial Data Indexing in One Oracle Spatial Database System to Make Faster Showing Geographical Map in Mobile Geographical Information System Technology

Maruto Masserie Sardadi, Mohd Shafry bin Mohd Rahim, Zahabidin Jupri, and Daut bin Daman

***Abstract*—**The latest Geographic Information System (GIS) technology makes it possible to administer the spatial components of daily "business object," in the corporate database, and apply suitable geographic analysis efficiently in a desktop-focused application. We can use wireless internet technology for transfer process in spatial data from server to client or vice versa. However, the problem in wireless Internet is system bottlenecks that can make the process of transferring data not efficient. The reason is large amount of spatial data. Optimization in the process of transferring and retrieving data, however, is an essential issue that must be considered. Appropriate decision to choose between R-tree and Quadtree spatial data indexing method can optimize the process. With the rapid proliferation of these databases in the past decade, extensive research has been conducted on the design of efficient data structures to enable fast spatial searching. Commercial database vendors like Oracle have also started implementing these spatial indexing to cater to the large and diverse GIS. This paper focuses on the decisions to choose R-tree and quadtree spatial indexing using Oracle spatial database in mobile GIS application. From our research condition, the result of using Quadtree and R-tree spatial data indexing method in one single spatial database can save the time until 42.5%.

***Keywords*—**Indexing, Mobile GIS, MapViewer, Oracle Spatial Database.

I. Introduction

MOBILE GIS is the extension of a geographic information system (GIS) which has been developed then can be run in the field. Previously GIS can only run in the office with a desktop GIS. With mobile GIS, the user can retrieve, transfer, update, manipulate, analyze and display geographic information everywhere and every time. The standard technologies that integrate in Mobile GIS application are wireless network for transfer and access data, mobile devices for run GIS application in everywhere, and GPS (Global Positioning System) to detect the location.

Authors are with Faculty of Computer Science and Information System, University Technology of Malaysia, 81310 Skudai, Malaysia (e-mails: muserie@gmail.com, shafry@utm.my, zaha@utm.my, daut@utm.my).

Over the past years, Mobile GIS applications and technological trends have a rapid development. The success and emergence of the WWW (World Wide Web) and internet also support this rapid advancement. The geo-information technology had emergence from mainframe computers to stand alone desktop computer GIS, local networking GIS, web GIS and now Mobile GIS where geographical map and the information are run and displayed on small mobile devices like Mobile Phones and Personal Digital Assistants (PDA) [1].

The internet as a telecommunication network over the world can be used for transferring geographic data. The integration of the internet and GIS technologies as we call internet GIS can support the demand for geographic data access and transfer [2]. Additionally, internet can be used for communication data between client and server. With Mobile GIS, field data collection can be settled and send the data to the server for further processing. Furthermore, client can access the information needed for enhance the collection of the geographical data [3].

Rapid improvement in the Mobile GIS technology can solve mobile handheld devices problem like their small bandwidth, limitation of applications capability, color resolution, and small screen display [4]. The recent developments of internet and Mobile GIS technology make it possible the process of spatial data transferring, collection, processing and dissemination with large amount of geographic data [5].

Previously, data collection and editing process in the field take a lot of time and susceptible result some errors. GIS user must visit to the field for take the geographical data in the structure of paper maps. Then they carry out field edit using draft and notes on paper maps and structure. After they got the geographical data, the processes of field edits in the office must be interpreted and manually entered into spatial database. Consequently, the geographical data rarely accurate and up-to-date as it could have been.

With the rapid Mobile GIS developments now a day, GIS can be taken into the field using some devices like compact digital maps, laptop (mobile computer), PDA (Personal Digital Assistant), etc. This new GIS technology enables company or GIS user to get real time geographic information, update data much faster to their database and applications,

efficiently analysis, display geographic data, and making decisions in the field.

The major concern of most wireless technology is the overloading servers because of the system bottleneck phenomenon. Mobile GIS applications use a wireless technology for process transferring and retrieval data, and then it needs concentrating to explore and solve that problem. The system bottleneck occurs in four main areas: database, network, application server and web server. The major problem for system bottleneck phenomenon originates from the database [6]. In mobile computing, considerable research has been focused on the infrastructure and architecture design to try and solve system bottlenecks, such as the development of third generation mobile systems or to resolve cache management of the database. Thus optimizing the existing system such as fine tuning databases is an important step to enhance the overall performance.

To solve the problem, making some optimization in spatial database system is one of the approaches. One of the ways to optimize process of transferring and retrieving data in spatial database system over mobile GIS network is spatial data indexing approach. The best spatial data indexing method are R-tree and Quadtree. Each of those methods have different advantages and disadvantages base on the requirement of the applications and type of data; if Quadtree and R-tree will be used together for index the data in one single spatial database system then the database can be optimized with the appropriate spatial data indexing method and can be contributed to improve spatial data transferring speed.

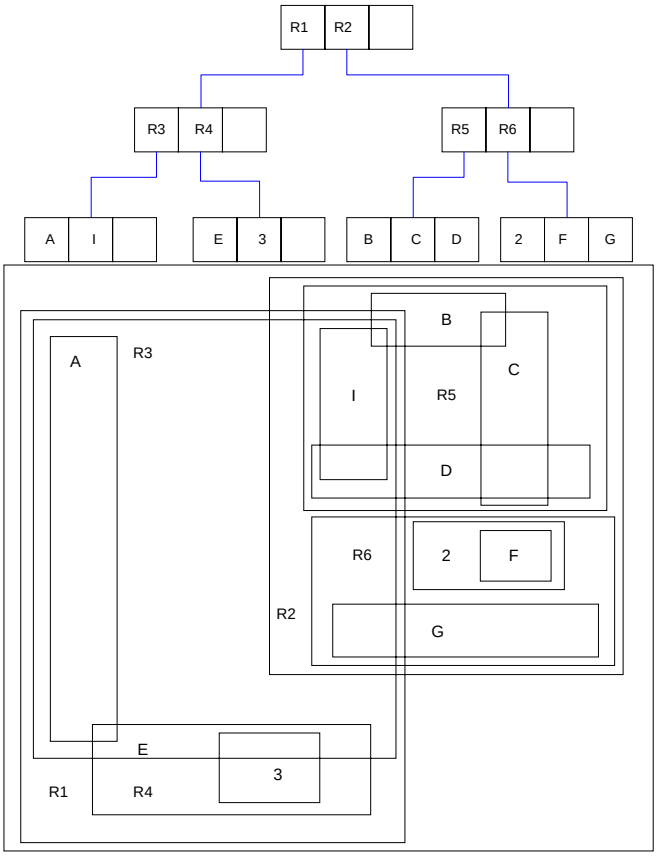

Fig. 1 R-tree Algorithm Schema [7]

II. R-tree Spatial Data Indexing Overview

R-trees are tree data structures that derived from B-trees. It is used for representing multidimensional point data, but is used for spatial process methods for indexing multi-dimensional information. Each node in the tree communicates to the smallest d-dimensional rectangle that surrounds its child nodes. The leaf nodes include cursor to the real geometric entity in the database, as an alternative of child. The entity is represented by the smallest associated rectangle in which they are enclosed [7].

There are many possible approaches to divide a node. The first goal is to allocate the records among the nodes so that probability that the nodes will be visited to subsequent searches will be reduced. The total areas of the wrapping rectangles for the nodes should be minimized to achieve this goal. Another goal is to decrease the probability that both nodes are observe in successive searches. The area common to both nodes should be minimized to achieve this goal. This following figure is to describe R-tree collection of rectangles and the spatial extension of the enveloping rectangles.

Many researchers explored about R-tree spatial indexing, because R-tree is one of the best spatial indexing and recommended in Oracle spatial database. Here, we discuss some of the researchers that already used the advantages of R-tree for the spatial indexing.

A new spatial cluster grouping algorithm and R-tree insertion algorithm has been proposed. It introduces the k-means clustering method and employs the 3D overlap volume, 3D coverage volume and the minimum bounding box shape value of nodes as the integrative grouping criteria [8]. A scalable technique called Seeded Clustering that allows us to maintain R-tree indices by bulk insertion while keeping pace with high data arrival rates has also been proposed [9]. A bottom-up update strategy for R-trees that generalizes existing update techniques and aims to improve update performance has also been proposed [10].

A novel bulk insertion technique for R-Trees using Oracle 10g that is fast and does not compromise on the quality of the resulting already presented [11]. A generalization for the family of R-trees, called the Multi-scale R-tree, that allows efficient retrieval of geometric objects at different levels of detail has also been proposed [12].

The real-time mobile GIS based on the HBR-tree to manage mass of location data efficiently have been explored [13]. A Technique combining existing Q + R-tree and QuadTree in terms of range query execution time by a high order of magnitude has also been proposed [14]. An efficient protocol for the kNN search on a broadcast R-tree, which is a popular

multi-dimensional index tree, in a wireless broadcast environment in terms of latency and tuning time as well as memory usage has also been proposed [15].

III. Quadtree Spatial Data Indexing Overview

A quadtree is a tree data structure in which each internal node has up to four children. Quadtrees are most often used to partition a two dimensional space by recursively subdividing it into four quadrants or regions. The regions may be square or rectangular, or may have arbitrary shapes. A similar partitioning is also known as a Q-tree. Quadtrees has two features which are: (1) they decompose space into adaptable cells and (2) each cell (or bucket) has a maximum capacity. When maximum capacity is reached, the bucket splits, and the tree directory follows the spatial decomposition of the Quadtree [16].

To cover the geometry, Quadtree spatial data indexing uses fixed-size tiles controlled by tile resolution. Each tile has a fixed-size and shape, because the coordinate space has been decomposed a specific number of times. Then the tessellation will be terminated. However all of this process could be happened if the resolution is the sole controlling factor [17].

As we can see in Fig. 2, the tiles can be sorted linearly and systematically at the exact level. Then it will visit the tiles which determined by a space-filling curve. The unique numeric identifiers are also assigned to the tiles.

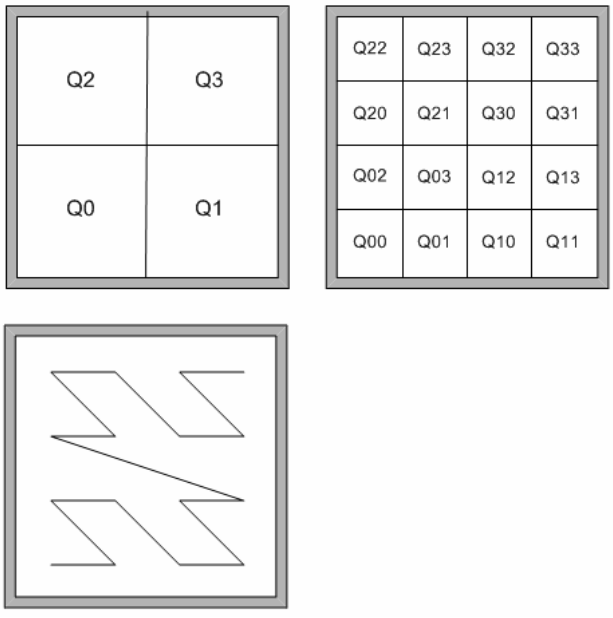

Fig. 2 Quadtree Algorithm Schema

Many researchers had explored about quadtree spatial indexing, even though quadtree is not recommended to use for particular spatial indexing compare to R-tree spatial data indexing. However, there are many advantages using quadtree spatial indexing in the special situation. An algorithm based on applying eigenspace methods has been presented to a quadtree of related set images to solve the pose estimation problem in the presence of occlusion and/or background clutter. The inability to easily locate the desired object and apply the appropriate normalizations, are efficiently overcome by the recursive quadtree procedure [18]. Fluid flow solver based on unstructured quadtree / octree has been explored also. This indexing type Eulerian grids is coupled with a spectral Finite Element (p-FEM) structural mechanics solver based on a Lagrangian description can predict bidirectional fluid-structure interaction (FSI) [19]. The new structure of Multiversion Linear Quadtree (MVLQ) has been present based on spatio-temporal access method. In this indexing structure can be used as an index mechanism for storing and accessing evolving raster images [20].

A novel fractal image coding based on Quadtree partition of the adaptive threshold value has been proposed to show better performance including the improved quality of the decoded image, shorter compression time and higher compression ratio with another spatial indexing structure [21]. An alternative non-pointer quadtree node codification to manage geographical spatial data also has been presented. In this structure, new codification is based on a variable sequence of z-ordered base four digits [22]. A new quadtree-based decomposition of a polygon possibly with holes also has been presented. An approximate this indexing structure is good for ray shooting in the average case as defined by Aronov and Fortune [23].

MPI collective algorithm selection process and explore the applicability of the quadtree encoding method has been proposed. This method constructs quadtrees with different properties from the measured algorithm performance data and analyzes the quality and performance of decision functions generated from these trees. Selecting the close-to-optimal collective algorithm based on the parameters of the collective call at run time is an important step in achieving good performance of MPI applications [24]. The matrix quadtree also has been presented based on the weighted in the form of modification. The technique of distance transform is extended to the weighted regions and applied to the robot path planning [25]. A new data dissemination protocol that exploits "Quadtree-based network space partitioning" to provide more efficient routing among multiple mobile stimuli and sink nodes also has been proposed. Because of a common hierarchy of cluster-head nodes is constructed where the data delivery to mobile sinks is independent of the current position of mobile stimuli [26]. A 3D mesh is generated by using the original quadtree triangulation also algorithm has been proposed with a nontrivial algorithm for quadtree triangulation. The aim of the study was to increase the accuracy of a terrain triangulation while maintaining or reducing the number of triangles [27].

A new data dissemination protocol that exploits "Quadtree-based network space partitioning" to provide more efficient routing among multiple mobile stimuli and sink nodes has been proposed. Simulation results show that in that work significantly reduces average energy consumption while maintaining comparably higher data delivery ratio [28]. A novel computational paradigm for detection of blood vessels in fundus images based on RGB components and quadtree decomposition also has been proposed. The proposed algorithm employs median filtering, quadtree decomposition,

post filtration of detected edges, and morphological reconstruction on retinal images [29]. A distributed quadtree index that adapts the MX-CIF quadtree was described that enables more powerful accesses to data in P2P networks. This index has been implemented for various prototype P2P applications [30].

The coordinate space (for the layer where all geometric objects are located) is issued to a procedure called tessellation in the linear quadtree indexing scheme of oracle spatial 10g, which delineates elite and extensive cover tiles for all of pile up geometry. Tessellation is done by crumbling the coordinate space in a standard hierarchical mode. The range of coordinates, the coordinate space, is viewed as a rectangle. At the first level of decomposition, the rectangle is divided into halves along each coordinate dimension, producing four tiles. Each tile that cooperates with the geometry being tessellated is further decomposed into four tiles. This procedure continues until some termination criteria, such as size of the tiles or the maximum number of tiles to cover the geometry is met. The results of the tessellation process on geometry are stored in a table, referred to as the SDOINDEX table [17].

Spatial quadtree indexing utilizes fixed-size tiles to cover geometry. Fixed-size tiles are managed by tile resolution. If the resolution is the single controlling reason, then tessellation terminates when the coordinate space has been crumbled an exact number of times. Consequently, each tile is of a fixed amount and form. Fixed-size tile resolution is managed by a user-selectable keyword named sdo_level. Smaller fixed-size tiles give better geometry approximations.

IV. Consideration of Choosing R-tree or Quadtree Spatial Data Indexing in One Single Spatial Database System

R-tree and Quadtree indexes that use extensible framework are the best spatial data indexing method from existing spatial indexing method for indexing low-dimensional spatial data. Oracle spatial users can choose between one of two spatial indexes for indexing low-dimensional spatial data: a linear Quadtree [31] or an R-tree [32, 33, 34, 35, 36, 37]. For processing the queries, R-tree approach may be more efficient due to better maintenance of spatial immediacy but may be slow in updates or index creation and implements its own concurrency protocols on top of table-level concurrency mechanisms. Since R-tree is built logically as a tree and physically using tables inside the database and the search involves recursive SQL for traversing the tree from root to relevant leaves. The linear Quadtree give results in simpler index creation, faster updates and inheriting of builtin B-tree concurrency control protocols. Since these indexes computes tile approximations for geometries and uses existing Btree indexes for performing spatial search and other DML operations.

Oracle spatial provides spatial indexing to speed up access and operation spatial data. That spatial index can be an R-tree or Quadtree index, or both. Every type of indexes can give good quality outcome in different situations. R-tree and Quadtree index also can be used together at the same time in one geometry column. To choose the appropriate indexing, analysis of the data and operations indexing must be made in advanced. The default of spatial index in oracle is an R-tree which the dimension of the data is two. If the spatial index has been built in two dimensions of data or more, only one built-up operator (SDO_FILTER, the primary filter or index only in the query) can be used in this situation. This operator considers in all of dimension. Here we explore some of comparisons between R-tree and Quadtree spatial data indexing.

The method to find the interiors of convex polygons is different and can't be applied to concave geometries [39]. Different technique should be used in the concave geometries to get the interior estimations for concave query windows. One of the methods is to break up recursively the MBR (Minimum Bounding Rectangle) of the concave geometry into quadrants four times. This idea also can be applied to carry out a level four tilling via the MBR as the tilling domain. Then the geometries are recognized along with the tiles that wrap the geometry. Aspirant MBRs are also tiled and the tiles for the aspirant MBR are seek out along with the core tiles for the concave query window. Current outcome [39] demonstrates that this two-sided method for applying the intermediary filter in R-trees attains extensive enhancements in query performance. The core area detained using this method could be larger than that for some Quadtree indexes assembled using a tiling level of 14 since the tiling domain in that case is the entire data space while the tiling domain for the R-tree is simply the MBR of the concave geometry. Since this situation happened, then the approximation of geometries cannot be fine-tuned.

Quadtree indexing in Oracle Spatial database use two strategies for index the geographical data. The first one is user choose and specified the tilling level of Quadtree indexing method. The second one is approximate every geometry data thus the covering tiles can be minimize. At the appropriate stage the tile-code associating each tile using z-ordering. Tessellation process has a responsibility to divide the tiles for geometry into internal and external part based on whether or not they are fully interior to the geometry. The spatial_index table receives all the wrapping tiles for geometry data beside geometry rowid. The geometry data as a result of this method may have numerous rows in the spatial_index table where every dissimilar tile-code, rowed, and internal or external status of the geometry stored in a row. Consequently geometry approximation can be fine-tuned by setting the tilling level and number of tiles. The last stage is speeding up queries for construct B-tree index on the tile_code, rowed, and status.

R-tree has easier method in creating the index and tuning rather than Quadtree spatial index creation. Because in Oracle, R-tree will create the index by default without identify any parameter. Moreover, setting the appropriate tuning parameter values of Quadtree can affect performance significantly. In order to optimize performance, Quadtrees necessitate to be fine-tuned by choosing a fitting tiling level. The storage and the index creation costs rise as the tiling level increases. Quadtree require tilling level more than R-tree, as a result the storage that is needed to create the Quadtree index is bigger than using R-tree index.

To recognize the nearest neighbors for a geometry, SDO_NN (nearest neighbor) queries utilize the spatial index. The parameter for tuning the parameter and may affect query's performance utilize SDO_BATCH_SIZE. The SDO_BATCH_SIZE only can be used if SDO_NN utilize an R-tree index to execute the procedure. Additionally this keyword can't be used when Quadtree index combined. The value for this keyword will be affecting the performance of nearest neighbor queries.

In Oracle Spatial, R-trees preserve their rational tree formation and are executed as a table where each node of the R-tree communicates to a row in the table and a child pointer in the R-tree communicates to the row id of child row in the same table. The root rowed (pointer) of the R-tree is stored in the metadata for the index and allows routing from the root of the R-tree to the leaf nodes. Leaf nodes of the R-tree accumulate one MBR for each geometry data together with the geometry row id. Queries and updates attain the root of the R-tree and navigate down the tree to the leaves.

More details on this implementation can be obtained from [38]. Note that each time an R-tree node is called; the matching row from the spatial_index is chosen using a SQL declaration internally. This means query and update processing in R-trees occupies processing of more recursive SQL declarations than in the case of Quadtrees. As a result, some DML processes particularly update are likely to be more costly. And the approximation of geometries cannot be fine-tuned.

In the case of Quadtrees, internal tiles for data geometries are recognized at index creation time. Every geometry tiles are labeled with a "status" representing whether they are internal tiles or external tiles. The status is set to boundary when any element of the boundary of geometry touches a tile. Otherwise, if the tile is totally inside the geometry, the status is set to "interior". These status tags along with the tile-codes are evaluated with those of the query tiles at query time to execute the transitional filtering. Note that such labeling of the tiles that cover query and data geometries inflicts very little transparency (one byte per row) on storage space, and does not change query or update performance.

A spatial R-tree index can index spatial data until four dimensions unlike the Quadtree that only can index only two dimensions data. Each geometry data approximated with R-tree index by a particular rectangle that minimally attaches the geometry using MBR. R-tree index consists of a hierarchical index on the MBRs of the geometries for a sheet of geometries in the layer. In the linear quadtree indexing design, the coordinate space is focused to a procedure called tessellation, which describes restrictedly and extensively cover tiles for every stored geometry data. The coordinate space in a standard hierarchical approach is tessellated by decomposition. At the first level of disintegration, the rectangle is divided into halves along each coordinate dimension creating four tiles. Each tile that interrelates with the geometry being tessellated is further decomposed into four tiles. This procedure keeps on until some termination condition, such as size of the tiles or the highest quantity of tiles to cover the geometry, is met.

Linear referencing is an ordinary and suitable means to correlate attributes or events to locations or portions of a linear element. It has been extensively used in transportation applications (such as for highways, railroads, and transit routes) and efficacies applications (such as for gas and oil pipelines). The major benefit of linear referencing is its potentiality of locating attributes and events along a linear element with only one constraint (usually known as quantify) instead of two (such as latitude/longitude or x/y in Cartesian space). Sections of a linear feature can be referenced and generated dynamically by signifying the start and end locations along the element without explicitly storing them. If LRS (Linear Referencing System) data is indexed using a spatial quadtree index, only the first two dimensions are indexed; the measure dimension and its values are not indexed.

SDO_WITHIN_DISTANCE utilizes the spatial index to recognize the set of spatial objects that are within some particular distance of a given object (such as an area of interest or point of interest). Distance between two extensive objects (nonpoint objects such as lines and polygons) is described as the minimum distance between these two objects. If this operator is used with geodetic data, the data must be indexed with an R-tree spatial index. If this operator is used with geodetic data and if the R-tree spatial index is created with 'geodetic=false' recognized, we cannot use the part parameter.

Geodetic data consists of geometries that have geodetic SDO_SRID values, reflecting the actuality that they are stand on a geodetic coordinate system (such as using longitude and latitude) as contrast to a flat or estimated plane coordinate scheme. Thus indexing geodetic data will improve significantly spatial features. However, quadtree indexes cannot be geodetic indexes. If we generate a non-geodetic index on geodetic data, we cannot apply the part parameter with the SDO_WITHIN_DISTANCE operator.

Distance, region, and angular units are completely supported location data to be stored in the spatial database using the whole-earth geodetic form which guarantees capacities across the earth's exterior will be very precise. Location data can be stored in the database using the whole-earth geodetic model An R-tree index is required for a whole-Earth index. In another hand a quadtree index cannot be used for a whole-Earth index.

To summarize all of those comparisons between R-tree and Quadtree spatial data indexing, below is the table for consideration to choose R-tree or Quadtree spatial data indexing in one spatial database system.

TABLE I
CHOOSING R-TREE OR QUADTREE INDEXING [17]

No	R-Tree Indexing	Quadtree Indexing
1	The approximation of geometries cannot be fine-tuned. (Spatial uses the minimum bounding rectangles.)	The approximation of geometries can be fine-tuned by setting the tiling level and number of tiles.
2	Index creation and tuning are easier.	Tuning is more complex, and setting the appropriate tuning parameter values can affect performance significantly.
3	Less storage is required.	More storage is required.
4	If your application workload includes nearest-neighbor queries (SDO_NN operator), R-tree indexes are faster, and you can use the sdo_batch_size keyword.	If your application workload includes nearest-neighbor queries (SDO_NN operator), quadtree indexes are slower, and you cannot use the sdo_batch_size keyword.
5	Heavy update activity to the spatial column may decrease the R-tree index performance until the index is rebuilt.	Heavy update activity does not affect the performance of a quadtree index.
6	You can index up to four dimensions.	You can index only two dimensions. If LRS (Linear Referencing System) data is indexed using a spatial quadtree index, only the first two dimensions are indexed; the measure dimension and its values are not indexed.
7	An R-tree index is recommended for indexing geodetic data if SDO_WITHIN_DISTANCE queries will be used on it.	A quadtree index is not recommended for Indexing geodetic data if SDO_WITHIN_DISTANCE queries will be used on it.
8	An R-tree index is required for a whole-Earth index.	A quadtree index cannot be used for a whole-Earth index.

V. MAPVIEWER AND APPLICATION SERVER OVERVIEW WITH THE IMPLEMENTATION

This research uses Oracle Application Server 10g as the application server and MapViewer as web base application for our implementation. Oracle Application Server 10g that in mobile computing stands for grid consist of a combined, set-stand software stage. In the main platform, Oracle Application Server consist of and OC4J (OracleAS Containers for J2EE) and Oracle HTTP Server (built by Apache). This application server deploys by J2EE programmable language. This application server turn into the starter platform planed for grid computing with full maintain for Service-Oriented Architecture (SOA). It also has built in a middle tier server infrastructure. Additionally, it has a feature that can help construct our application to become more effective and productive in on-line work environment [41]. All of those functionalities of Oracle Application Server make GIS application server to manage and organize Mobile GIS application that run in web base.

There are some important tasks that we need to set and configure after install Oracle Application Server and MapViewer web application. Firstly we need to run Oracle Application Server instances in the script. Secondly configure Oracle Application Server clusters using Oracle Process Manager and Notification Server (OPMN) commands. Thirdly enable and configure Secure Socket Layer (SSL), if we plan to use Oracle Drive with Oracle Content DB and we are not using Oracle Internet Directory as a user repository.

In the implementation process using Oracle Application Server MapViewer we used J2EE application that already deployed to a J2EE container. In the rendering engine (Java library) named SDOVIS we render cartographic USA Base Map. We used XML and AJAX-based JavaScript to store data that will be retrieving for USA Base Map MapViewer web application without directly connect to the spatial database. A graphical Map builder also used to create map symbols, define spatial data rendering rules, and create and edit USA Base Map objects.

Creating one of the geographical maps should be started from creating the XML transform. Here is one of the example USA Base Map XML transform code to create XML from MAP_ADMIN_AREA2_US table.

XML Transform Code:

```
<?xml version="1.0" standalone="yes"?>
  <info_request datasource="adci" format="strict">
        SELECT * FROM map_admin_area2_us
  </info_request>
```

Result:

```
<ROWSET>
–
<ROW num="1">
          <POLYGON_ID>1717058563</POLYGON_ID>
   <AREA_ID>21013672</AREA_ID>
          <POLYGON_NAME>VIRGINIA</POLYGON_NA
          ME>
   <NAME_LANGCODE>ENG</NAME_LANGC ODE>
          <FEATURE_TYPE>STATE</FEATURE_TYPE >
   <GEOMETRY>null</GEOMETRY>
   <PARTITION_ID>1</PARTITION_ID>
</ROW>
</ROWSET>
```

Here is one of the source codes to display USA Base Map in MapViewer web base application with the main position in Washington.

```
…
          var mapview;

    function showMap()
    {
```

```
                var baseURL  =
                  "http://"+document.location.host
                  +"/mapviewer";
        var mapCenterLon = -77.0425;
        var mapCenterLat =  38.87285;
        var mapZoom      =  4;
        var mpoint =
   MVSdoGeometry.createPoint
   (mapCenterLon,mapCenterLat,          8307);
        mapview = new MVMapView
   (document.getElementById           ("map"), baseURL);
        mapview.addBaseMapLayer (new
   MVBaseMap                ("adci.us_base_map"));
        mapview.setCenter(mpoint);
        mapview.setZoomLevel(mapZoom);
      mapview.addNavigationPanel("EAST");

      mapview.display();
    }
…
```

This is the Map result from that source code.

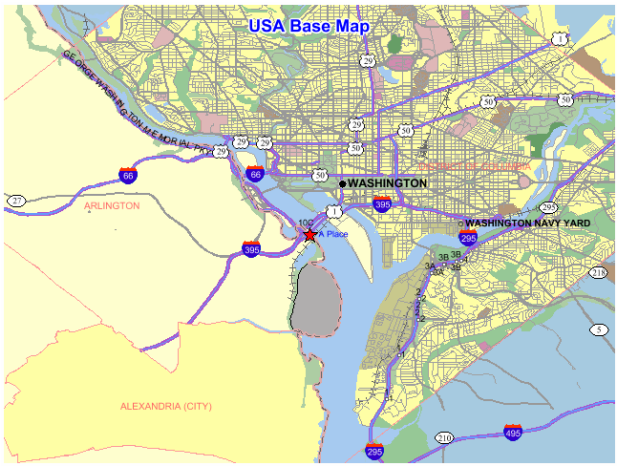

Fig. 3 USA Base Map Basic Flow of Action with Oracle Application Server MapViewer

Here is the XML code to produce that one of US BASE MAP.

```
<?xml version="1.0" standalone="yes"?>
<map_request
        title="USA Base Map"
        basemap="us_base_map"
        datasource = "adci"
        width="640"
        height="480"
        bgcolor="#a6cae0"
        antialiase="false"
        format="PNG_STREAM">
  <center size="0.15">
    <geoFeature render_style="m.star"
                radius="1600,4800"
                 label="A Place"
              text_style="t.Street Name" >
      <geometricProperty typeName="center">
        <Point>
          <coordinates>-77.0425, 38.87285</coordinates>
        </Point>
      </geometricProperty>
    </geoFeature>
  </center>
</map_request>
```

VI. Performance Evaluation

The model used for testing and evaluation in this research has been simplified with the following assumptions [6]:

1. Time is one-dimensional and linearly ordered;
2. Oracle 10g spatial databases are used to develop the GIS model;
3. Connection to the server and data retrieval via an active TCP/IP connection using the Hyper Text Transfer Protocol (HTTP) utilized as data transfer protocol; the Mobile user provides an initial position.

Simplified mobile GIS equipment has demonstrated how process retrieval and transferring data works. A mobile user access data's from server that uses Oracle 10g Spatial as a database to show geographical map. We characterize the performance of this function in the following way. The average response time from the mobile client is measured as the time spent (in seconds) from the moment the query is issued to the moment the results of the query are generated. The hardware settings are summarized in Table II.

TABLE II
Setting of the Experiments

Web Server & GIS Server:	CPU	Intel Pentium Core 2 Duo
	RAM	2 GHz
	Web Server	IIS
	GIS Server	Oracle Application Server & MapViewer
	Database	Oracle 10g Spatial
Mobile Device Equipment:	Model	Tablet PC
	OS	Windows XP Tablet PC
	Web Browser	Internet Explorer

The result for R-tree spatial index is depicted in Fig. 4, which is plotted as a two-dimensional graph to illustrate the results of average response time in Mobile GIS application. The line with blue color corresponds to the time measurement from the data that use an R-tree Spatial Indexing, while the line with red color represents the time measurement from the data that doesn't use Spatial Indexing. We observe that the time saved increases as the database size expands. For the number of records, over 30,000, the time required in the data that use an R-tree Spatial Indexing is almost two times more than the data that doesn't use Spatial Indexing. It is noted, however, that this is related to the actual time of the query, since the database volume is varied as time changes. The response time is slightly different to the result shown in Fig. 4. Nevertheless, we can easily discover that the time response on

the data that use an R-tree Spatial Indexing is faster than the data that doesn't use Spatial Indexing.

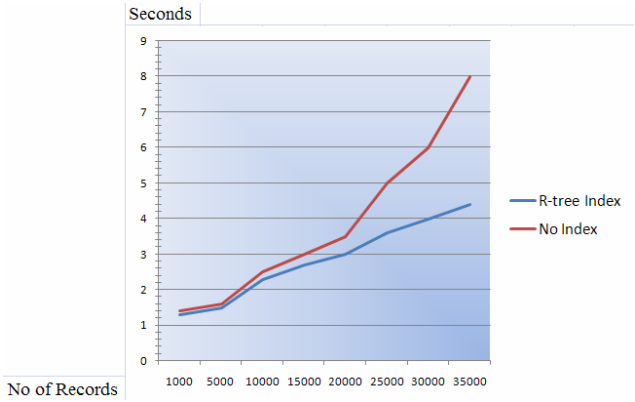

Fig. 4 R-tree Average Response Time

The result for Quadtree spatial index is depicted in Fig. 6, which is plotted the results of average response time in Mobile GIS application. The line with blue color corresponds to the time measurement from the data that use a quadtree Spatial Indexing, while the line with red color represents the time measurement from the data that doesn't use Spatial Indexing. We observe that the time saved increases as the database size expands. For the number of records, over 30,000, the time required in the data that use a quadtree Spatial Indexing is almost two times more than the data that doesn't use Spatial Indexing. It is noted, however, that this is related to the actual time of the query, since the database volume is varied as time changes. The response time is slightly different to the result shown in figure (Fig. 5). Nevertheless, we can easily discover that the time response on the data that use a quadtree Spatial Indexing is faster than the data that doesn't use Spatial Indexing.

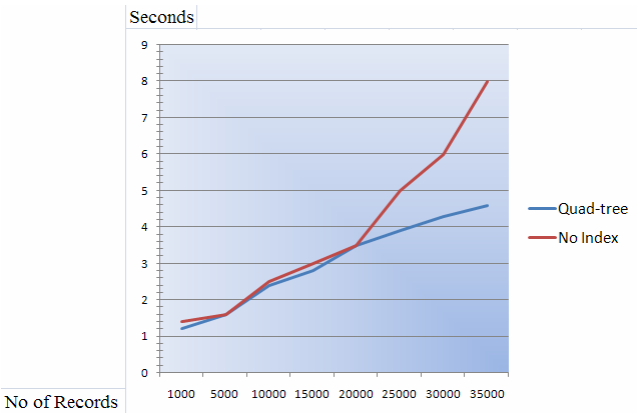

Fig. 5 Quadtree Average Response Time

Below is the table (Table III) to make it clear to see the result.

TABLE III
R-TREE AND QUADTREE EXPERIMENTS RESULT

Number of Records	No Index (Seconds) - Quadtree	Quadtree Index (Seconds)	No Index (Seconds) – R-tree	R-tree Index (Seconds)
1000	1.4	1.2	1.4	1.3
5000	1.6	1.6	1.6	1.5
10000	2.5	2.4	2.5	2.3
15000	3	2.8	3	2.7
20000	3.5	3.5	3.5	3
25000	5	3.9	5	3.6
30000	6	4.3	6	4
35000	8	4.6	8	4.4

VII. CONCLUSION

Database tuning is an important step to solve system bottleneck problem. The use of indexes is an efficient ways of retrieving data from spatial database. R-tree and Quadtree spatial index can be use as spatial database tuning methods to enhance Mobile GIS performance. MapViewer also one of web base applications that suitable, fast, and has a good feature for mobile GIS applications to show geographical map. Oracle Application Server and Oracle Spatial 10g have function for web application server and recommended place to store and retrieve spatial data. The goals are to fine-tune the original database for mobile users with a limited communication bandwidth, to improve the response time, and show geographical map with a good feature. Further research will develop and test this combination of R-tree and Quadtree spatial data indexing in one spatial database system from those of the considerations.

ACKNOWLEDGMENT

The authors would like to thank Ministry of Science, Technology and Innovation Malaysia (MOSTI) and Universiti Teknologi Malaysia (UTM) for their financial support under e-science fund vot no. 79233.

REFERENCES

[1] Rajinder, S. N. (2004). *Cartographic visualisation for mobile application*. Master's thesis, ITC/IIRS.
[2] Peng, Z. R. & Tsou, M. H. (2003). *Internet GIS: Distributed Geographic Information Services for the Internet and Wireless Network*. John Wiley and Sons Inc.
[3] Mensah, E. (2007). *Designing a Prototype Mobile GIS to Support Cadastral Data Collection in Ghana, 44.*
[4] Vckovski, A. (1999). *Interoperability and spacial information theory.* Interoperating Geographic Information Systems.
[5] Kraak, M. J. (2002). *Current trends in visualisation of geographic data with special reference to cartography*. Invited paper: In Proceedings of the XXIIth INCA Congress 2002, Indian National Cartographic Association: Convergence of Imagery Information and Maps, volume 22, 319–324.
[6] Chen, K., & Shi, W. (2002), *A Study of Dynamic Database in Mobile GIS*.
[7] Guttman, A. (1984). *R-Trees: A Dynamic Index Structure for Spatial Searching*. Proc. 1984 ACM SIGMOD International Conference on Management of Data, pp. 47-57. ISBN 0-89791-128-8.
[8] Zhu, Q., Gong, J., Zhang, J. (2007). *An efficient 3D R-tree spatial index method for virtual geographic environments*. ISPRS Journal of

Photogrammetry and Remote Sensing, Volume 62, Issue 3, August 2007, pp 217-224.
[9] Lee, T., Moon, B., Lee, S. (2006). *Bulk insertion for R-trees by seeded clustering*. Data & Knowledge Engineering, Volume 59, Issue 1, October 2006, pp 86-106.
[10] Lee, M. L., Hsu, W., Jensen, C. S., Cui, B., Teo, K. L. (2003). *Supporting Frequent Updates in R-Trees: A Bottom-Up Approach*. Proceedings 2003 VLDB Conference, 2003, pp 608-619.
[11] An, N., Kothuri, R. K. V., Ravada, S. (2003). *Improving Performance with Bulk-Inserts in Oracle R-Trees*. Proceedings 2003 VLDB Conference, 2003, pp 948-951.
[12] Chan, E. P. F., & Chow, K. K. W. (2002). *On multi-scale display of geometric objects*. Data & Knowledge Engineering, Volume 40, Issue 1, January 2002, pp 91-119.
[13] Yun, J. K., Kim, D. O., Hong, D. S., Kim, M. H., Han, K.J. (2006). *A real-time mobile GIS based on the HBR-tree next term for location based services*. Computers & Industrial Engineering, Volume 51, Issue 1, September 2006, pp 58-71.
[14] Francis, D. H., Madria, S., Sabharwal, C. (2008). *A scalable constraint-based Q-hash indexing for moving objects*. Information Sciences, Volume 178, Issue 6, 15 March 2008, pp 1442-1460.
[15] Liu, C. M., & Fu, S. Y. (2008). *Effective protocols for kNN search on broadcast multi-dimensional index trees*. Information Systems, Volume 33, Issue 1, March 2008, pp 18-35.
[16] Finkel, R., & Bentley, J. L. (1974). *Quad Trees: A Data Structure for Retrieval on Composite Keys*. Acta Informatica 4 (1): 1-9.
[17] Oracle Spatial 10g White Paper (2006). Oracle Spatial Quadtree Indexing, 10g Release 1 (10.1).
[18] Chang, C. Y., Maciejewski, A. A., Balakrishnan, V., Roberts, R. G., Saitwal, K. (2006). *Quadtree-based eigen decomposition for pose estimation in the presence of occlusion and background clutter*.
[19] Geller, S., Talke, J., Krafczyk, M. (2007). *Lattice-Boltzmann Method on Quadtree-Type Grids for Fluid Structure Interaction*. Fluid-Structure Interaction, 270-293.
[20] Tzouramanis, T., Vassilakopoulos, M., Manolopoulos, Y. (2000). *Multiversion Linear Quadtree for Spatio-Temporal Data*. Current Issues in Databases and Information Systems, 279-292.
[21] Zhang, L. & Xi, L. F. (2007). *A Novel Fractal Image Coding Based on Quadtree Partition of the Adaptive Threshold Value*. Theoretical Advances and Applications of Fuzzy Logic and Soft Computing, 504-512.
[22] Varas, J. (2007). *Mobile Robot Path Planning Among Weighted Regions Using Quadtree Representations*. Computer Aided Systems Theory - EUROCASTâ€™99, 239-249.
[23] Cheng, S. W. & Lee, K. H. (2008). *Quadtree Decomposition, Steiner Triangulation, and Ray Shooting*. Algorithms and Computation, 368-377.
[24] Grbovic, J. P., Fagg, G. E., Angskun, T., Bosilca, G., Dongarra, J. J. (2006). *MPI Collective Algorithm Selection and Quadtree Encoding*. Recent Advances in Parallel Virtual Machine and Message Passing Interface, 40-48.
[25] Varas, J. (2007). *Mobile Robot Path Planning Among Weighted Regions Using Quadtree Representations*. Computer Aided Systems Theory - EUROCASTâ€™99, 239-249.
[26] Mir, Z. H. & Ko, Y. B. (2006). *A Quadtree-Based Data Dissemination Protocol for Wireless Sensor Networks with Mobile Sinks*. Personal Wireless Communications, 447-458.
[27] Samet, R. & Ozsavas, E. (2007). *Optimization of Quadtree Triangulation for Terrain Models*. Advanced Concepts for Intelligent Vision Systems, 48-59.
[28] Mir, Z. H. & Ko, Y. B. (2007). *A quadtree-based hierarchical data dissemination for mobile sensor networks*. Telecommunication Systems, 117-128.
[29] Reza, A. W., Eswaran, C., Hati, S. (2007). *Diabetic Retinopathy: A Quadtree Based Blood Vessel Detection Algorithm Using RGB Components in Fundus Images*. Journal of Medical Systems, 147-155.
[30] Tanin, E., Harwood, A., Samet, H. (2006). *Using a distributed quadtree index in peer-to-peer networks*. The VLDB Journal, The International Journal on Very Large Data Bases, 165-178.
[31] H. Samet (1989). *The design and analysis of spatial data structures*. Addison-Wesley Publishing Co..
[32] A. Guttman (1984). *R-trees: A dynamic index structure for spatial searching*. Proc. A CM SIGMOD Int. Conf. on Management of Data, pages 47-57.
[33] T. Sellis, N. Roussopoulos, and C. Faloutsos (1988). *The r+-tree: A dynamic index for multi-dimensional objects*. Procdf the Int. Conf. on Very Large Data Bases, 13:507-518.
[34] N. Beckmann, H. Kriegel, R. Schneider, B. Seeger (1990). *The R* tree: An efficient and robust access method for points and rectangles*. In Proc. ACM SIGMOD Int. Conf. on Management of Data, pages 322-331.
[35] S. Berchtold, D. A. Keim, and H. P. Kreigel (1996). *The X-tree: An index structure for high dimensional data*. Procff the Int. Conf. on Very Large Data Bases.
[36] S. T. Leutenegger, M. A. Lopez, J. M. Edgington (1997). *STR: A simple and efficient algorithm for R-tree packing*. In Proe. Int. Conf. on Data Engineering.
[37] V. Gaede & O. Gunther (1998). *Multidimensional access methods*. ACM Computing Surveys, 30(2).
[38] K. V. Ravi Kanth, Siva Ravada, J. Sharma, J. Banerjee (1999). *Indexing medium-dimensionality data in oracle*. In Proc. ACM SIGMOD Int. Conf. on Management of Data.
[39] K. V. Ravi Kanth & Siva Ravada (2001). Efficient processing of large spatial queries using interior approximations. In Symposium on Spatial and Temporal Databases (SSTD).
[40] Oracle Application Server 10g White Paper (2007). Oracle Application Server 10g.

Maruto Masserie Sardadi obtained his B.Sc. degree in Computer Science from the University of Indonesia (UI), Indonesia in 2004. All of his working experienced is in IT area, some in oil industry. He is currently a Computer Science M.Sc. research student in University Technology of Malaysia (UTM). His research interests include database, GIS, and petroleum.

Mohd Shafry Mohd Rahim obtained his B.Sc. degree in Computer Education, 1999 and M.Sc. degree in Computer Graphics, 2002 from University Technology of Malaysia (UTM). He is currently a Computer Science PhD. research student in University Putra of Malaysia (UPM) and UTM Computer Science lecturer. His research interests include GIS and spatiotemporal database.

Zahabidin Jupri obtained his B.Sc. degree in Mathematics Education from University of Malaya (UM) and M.Sc. degree in Info. Mgmt. from Sheffield, UK. He is currently UTM Computer Science lecturer.

Daut Daman, obtained his B.Sc. degree in Math. Computer Science, 1979 from University Sains of Malaysia (USM) and M.Sc. degree in Applied Computing, 1984 from Cranfield Institute of Technology, UK. He is currently a Head Department of Computer Graphics and Multimedia Faculty of Computer Science UTM.