Quadtree Spatial Indexing Use to Make Faster Showing Geographical Map in Mobile Geographical Information System Technology Using an Oracle 10g Application Server and MapViewer Applications

[1]Maruto Masserie Sardadi, [2]Mohd Shafry bin Mohd Rahim, [3]Zahabidin Jupri, [4]Daut bin Daman
Faculty of Computer Science and Information System
University Technology of Malaysia,
81310 Skudai, Malaysia

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 slower. One of the reasons that system bottlenecks could happened in this situation is because spatial data is too large. Fine tuning spatial database, however, is an essential issue that must be considered. This paper focuses on a study in quadtree spatial indexing using Oracle spatial database and how to show the geographical map using MapViewer in the web application so we can use in any devices including mobile GIS application. Spatial database systems are now expected to be self-managing, self-healing, and always-up. We researchers and developers have our work cut out for us in sending all these features to optimize delivering data. The purposes of this research are to make an optimization in the speed of spatial data processing for clients with a limited communication bandwidth and to improve the response time.

Keyword:
Quadtree, Indexing, GIS, MapViewer, Oracle Spatial Database.

1. Introduction

Mobile GIS is the expansion of a geographic information system (GIS) from the office into the field. A Mobile GIS enables field based personnel to capture, store, update, manipulate, analyze, and display geographic information. Mobile GIS also integrates one or more technologies, such as: mobile devices, Global Positioning Systems (GPS), and wireless communications for Internet GIS access.

GIS applications and technological trends have over the past years seen a rapid advancement, and this has become so largely due to the emergence and the success of the World Wide Web. We have seen geo-information technology emerging from mainframe computers to stand alone desktop computer GIS through to local networking GIS, to the Web GIS and now Mobile GIS where maps are displayed and processed on small mobile devices like Personal Digital Assistants (PDA) and Mobile Phones [1].

The demand for geographic data access and transfer necessitated the integration together the technologies of the internet and GIS to what is known as internet GIS. The internet is described as a worldwide telecommunication network [2]. With internet connection between client and server, communication between the two is achieved. Field data collected and recorded can be communicated to server for further processing and also information needed to be used on the client to enhance the collection of the data can be accessed with Mobile GIS [3].

The mobile handheld devices may have their limitations like their small screen display, small bandwidth, color resolution and its limited application capabilities, even though most are being overcome by improvements in their technology [4]. However spatial data collection, processing and dissemination of large amount of geographic data are becoming possible due to recent developments we have seen in internet and Mobile GIS technology [5].

Traditionally, the processes of field data collection and editing have been time consuming and error prone. Geographic data has traveled into the field in the form of paper maps. Field edits were performed using sketches and notes on paper maps and forms. Once back in the office, these field edits were deciphered and manually entered into the GIS database. The result has been that GIS data has often not been as up-to-date or accurate as it could have been.

Manuscript received October 5, 2008
Manuscript revised October 20, 2008

The developments in Mobile GIS have enabled GIS to be taken into the field as digital maps on compact, mobile computers, providing field access to enterprise geographic information. This enables organizations to add real-time information to their database and applications, speeding up analysis, display, and decision making by using up-to-date, more accurate spatial data. In this paper we use MapViewer web base application that can be use in the mobile GIS.

Currently, workforce trends appear to be increasing service mobility and seizing opportunities presented by the internet. As the internet usage widely increases, the internet system becomes more complicated and more difficult to deal with. The internet wireless problems usually come from the system bottleneck, the system bottleneck occurs in four main areas: database (27%), network (25%), application server (23%) and web server (20%). The major setback for the system bottleneck originates from the database [6]. As a result, fine-tuning database such as creating spatial index existing databases is an important step to enhance the Mobile GIS performance. Thus we can provide the latest and accurate information for mobile users with a limited communication bandwidth and to improve the response time.

Based on these reasons, this paper will discuss quadtree spatial index database that suitable for Mobile GIS technology with the MapViewer web base application to show geographical map. The discussion will be started by reviewing the spatial indexing. Next, will exploring Quad tree indexing overview with the implementations. Quadtreee limitation will also be described. Spatial index for mobile applications will also be explained. Then MapViewer and Oracle Application Server overview with the implementation will also be discussed. And then we give the performance and evaluation result also. Finally, this paper ends with conclusions.

2. Spatial Indexing

Spatial indexing is based on the physical grouping of items that are indexed. For example, countries can be grouped by continent. With spatial indexing, it is even possible to provide VR (virtual reality) fly through (or fly over, or fly under) to help users maintain the context of their search as the users expand or narrow the scope of their search. Spatial indexing has a richness that parallels indexing by subject or author. The idea of spatial indexing is very powerful, that for records management, records are found based on their association with a place. Many records are tightly coupled to a place. Like other forms of indexing, geographic indexing can be combined with other indices.

There are so many Indexing methods in the spatial Indexing technique. Here we explore and give a little explanation for the most popular Spatial Indexing technique; Kd-tree, Z-order, UB-tree, Quad-tree, Oc-tree, Grid, and R-tree.

2.1 Kd-Tree

In computer science, a Kd-tree (short for k-dimensional tree) is a space-partitioning data structure for organizing points in a k-dimensional space. Kd-trees are a useful data structure for several applications, such as searches involving a multidimensional search key (e.g. range searches and nearest neighbor searches). Kd-trees are a special case of BSP trees. A Kd-tree uses only for splitting planes that are perpendicular to one of the coordinate system axes. This differs from BSP trees, in which arbitrary splitting planes can be used. In addition, in the typical definition every node of a Kd-tree, from the root to the leaves, stores a point. This differs from BSP trees, in which leaves are typically the only nodes that contain points (or other geometric primitives). As a consequence, each splitting plane must go through one of the points in the Kd-tree. Kd-trees are a variant that store data only in leaf nodes. It is worth noting that in an alternative definition of Kd-tree the points are stored in its leaf nodes only, although each splitting plane still goes through one of the points [7].

2.2 Z-Order

Z-order (aka Morton-order, first proposed by G. M. Morton) is a space-filling curve which is often used in computer science: Due to its good locality preserving behavior it is used in data structures for mapping multidimensional data to one dimension. The z-value of a point in multi dimensions is simply calculated by bitwise interleaving its coordinate values. Any one-dimensional data structure can be used such as binary search trees, B-trees, Skip lists or (with low significant bits truncated) Hash tables [8].

2.3UB-Tree

The UB-tree as proposed by Rudolf Bayer and Volker Markl, is a balanced tree for storing and efficiently retrieving multidimensional data. It is basically a B+ tree (information only in the leaves) with records stored according to Z-order (curve), also called Morton order. Z-order is simply calculated by bitwise interlacing the keys. Insertion, deletion, and point query are done as with ordinary B+ trees. To perform range searches in multidimensional point data, however, an algorithm must be provided for calculating, from a point encountered in the data base, the next Z-value which is in the multidimensional search range. The original algorithm to solve this key problem was exponential with the dimensionality and thus not feasible [9] ("GetNextZ-address"). A solution to this "crucial part of the UB-tree range query" linear with the z-address bit length has been described later [10]; this method has already been described in [11] (called "BIGMIN" algorithm) where using Z-order with search trees has first been proposed.

2.4 Quadtree

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. This data structure was named a quadtree by Raphael Finkel and J.L. Bentley in 1974. A similar partitioning is also known as a Q-tree. All forms of Quadtrees share some common features, like: They decompose space into adaptable cells; 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 [12].

2.5 Octree

An octree is a tree data structure in which each internal node has up to eight children. Octrees are most often used to partition a three dimensional space by recursively subdividing it into eight octants. Some common uses of octrees are: Spatial indexing, efficient collision detection in three dimensions, view frustum culling, and Fast Multipole Method. Octrees are the three-dimensional analog of quadtrees. The name is formed from oct + tree, and normally written "octree", not "octtree". Each node in an octree subdivides the space it represents into eight subspaces (octants). In a point region (PR) octree, the node stores an explicit 3-dimensional point, which is the "center" of the subdivision for that node; the point defines one of the corners for each of the eight children. In an MX octree, the subdivision point is implicitly the center of the space the node represents. The root node of a PR quadtree can represent infinite space; the root node of an MX octree must represent a finite bounded space so that the implicit centers are well-defined.

2.6 Grid

In the context of a spatial index, a grid (a.k.a. "mesh", also "global grid" if it covers the entire surface of the Globe) is a regular tessellation of a manifold or 2-D surface that divides it into a series of contiguous cells, which can then be assigned unique identifiers and used for spatial indexing purposes. A wide variety of such grids have been proposed or are currently in use, including grids based on "square" or "rectangular" cells, triangular grids or meshes, hexagonal grids, grids based on diamond-shaped cells, and possibly more [13].

2.7 R-Tress

R-trees are tree data structures that are similar to B-trees, but are used for spatial access methods i.e., for indexing multi-dimensional information; for example, the (X, Y) coordinates of geographical data. A common real-world usage for an R-tree might be: "Find all museums within 2 miles of my current location". The data structure splits space with hierarchically nested, and possibly overlapping, minimum bounding rectangles (MBR, otherwise known as bounding boxes, i.e. "rectangle", what the "R" in R-tree stands for). Each node of an R-tree has a variable number of entries (up to some pre-defined maximum). Each entry within a non-leaf node stores two pieces of data: a way of identifying a child node, and the bounding box of all entries within this child node [14].

3. Quadtree Indexing Overview with the Implementation

So many researchers explored about quadtree spatial indexing, even though quadtree is not recommended to use for particular spatial indexing. However, there are so 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 [15]. 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) [16]. 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 [17].

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 [18]. 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 [19]. 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 [20].

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 [21]. 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 [22]. 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 [23]. 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 [24].

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 [25]. 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 [26]. 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 [27].

In the linear quadtree indexing scheme of oracle spatial 10g, the coordinate space (for the layer where all geometric objects are located) is subjected to a process called tessellation, which defines exclusive and exhaustive cover tiles for all of stored geometry. Tessellation is done by decomposing the coordinate space in a regular hierarchical manner. 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, generating four tiles. Each tile that interacts with the geometry being tessellated is further decomposed into four tiles. This process 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.

Spatial quadtree indexing uses fixed-size tiles to cover geometry. Fixed-size tiles are controlled by tile resolution. If the resolution is the sole controlling factor, then tessellation terminates when the coordinate space has been decomposed a specific number of times. Therefore, each tile is of a fixed size and shape. Fixed-size tile resolution is controlled by a user-selectable keyword named sdo_level. Smaller fixed-size tiles provide better geometry approximations. The smaller the number of tiles, the coarser is the approximations.

In this paper we use USA Base Map as an example for the geographical data (including spatial and attribute data), putted in the oracle spatial database and can be shown in the MapViewer web application. We use a quadtree indexing in the geometry data to improve access and retrieval spatial data processing. This following table (Table. 1) is the table in our spatial database that uses quadtree spatial indexing for the Geometry data.

Table 1: Example of Data Indexing in our Spatial Database

USA BASE MAP Database	
GC_ROAD_SEGMENT_SIN	MAP_ADMIN_AREA1_US_SIN
MAP_ADMIN_AREA4_US_SIN	MAP_ADMIN_AREA5_US_SIN
MAP_CITIES_POINTS_US_SIN	MAP_FERRY_ROADS_US_SIN
MAP_ISLANDS_US_SIN	MAP_LAKES_US_SIN
MAP_OCEANS_US_SIN	MAP_PARKFACILITY_US_SIN
MAP_RIVERS_US_SIN	MAP_SEC_HIGHWAYS_US_SIN
MAP_MAJHWYSHIELD_US_SIN	MAP_SECHWYSHIELD_US_SIN
MAP_LANDMARK_US_SIN	MAP_HAMLET_US_SIN
MAP_RESTRNTS_US_SIN	MAP_ENTERTN_US_SIN
MAP_BUSINESS_US_SIN	MAP_PARKING_US_SIN
MAP_ADMIN_AREA2_US_SIN	MAP_ADMIN_AREA3_US_SIN
MAP_AIRPORTS_PNTS_US_SIN	MAP_AIRPORTS_US_SIN

MAP_HARBORS_US_SIN	MAP_HWY_SIGNS_US_SIN
MAP_MAJOR_HIGHWAYS_US_SIN	MAP_MAJOR_ROADS_US_SIN
MAP_PARKFACIL_PNTS_US_SIN	MAP_RAILWAYS_US_SIN
MAP_STREETS_US_SIN	MAP_TRANSHUBS_US_SIN
MAP_CARTOCOUNTRY_US_SIN	MAP_CARTOSTATE_US_SIN
MAP_TRAVDEST_US_SIN	MAP_SHOPPING_US_SIN
MAP_AUTOSVC_US_SIN	MAP_FININSTS_US_SIN
MAP_BORDCROSS_US_SIN	MAP_NAMEDPLC_US_SIN

1st step in the managing data phase, we need to create *Table Space* and *User* in the oracle spatial database, for this research we use oracle 10g Express Edition (Spatial Extension). All of the map tables (with prefix "MAP_") are the geographical and map data in the USA location. The Node table is the most important geometry data that connected with another map tables. SDO_MAPS, SDO_STYLES, and SDO_THEMES are the tables for background style or theme in the MapViewer web application. However, we need to insert in the view in user MDSYS (Oracle spatial SYSDBA or System table for spatial function).

In our spatial database we use a quadtree indexing structure for the geometry data. Some of the quadtree indexing structure are use partition. Here we explore one of quadtree spatial indexing that use partition in our spatial database.

We created a partitioned spatial index on a partitioned table. This is describes usage considerations specific to quadtree indexing. We specify parameters for individual partitions, the following considerations applied in our spatial database:
- All index partitions must have either R-tree or quadtree indexes. Mixed R-tree and quadtree indexes are not allowed.
- For quadtree indexes, the same sdo_level value must be used for each partition.

In the following example, if the ADCI.MAP_ADMIN_AREA2_US table has two partitions, P1 and P2, we can create a quadtree index as follows:

```
CREATE INDEX MAP_ADMIN_AREA2_US ON
GEOMETRY (geometry)
INDEXTYPE IS MDSYS.SPATIAL_INDEX
PARAMETERS ('sdo_level=6 tablespace=def_tbs')
LOCAL
(PARTITION ip1 PARAMETERS ('sdo_level=6
tablespace=local_tbs1'),
PARTITION ip2 PARAMETERS ('sdo_level=6
tablespace=local_tbs2'),
PARTITION ip3);
```

In the preceding example:
- IP1 is the index partition that corresponds to partition P1 of the ADCI.MAP_ADMIN_AREA2_US table, and IP2 is the index partition that corresponds to partition P2.
- The tablespace parameters are specified for each of the local index partitions as LOCAL_TBS1 and LOCAL_TBS2, respectively.
- If you omit the PARTITION ip2 PARAMETERS... clause (as is done for partition IP3), the default parameters specified before LOCAL are used.
Specifically, the DEF_TBS tablespace is used for storing the index partition (which will have the same name as the second partition of the ADCI.MAP_ADMIN_AREA2_US table, that is, P2). We also use tuning for quadtree spatial indexing in our spatial database. Here one of the tables that uses spatial indexing tuning, and can give an estimation of spatial index performance with creating one column.

```
DECLARE
table_name VARCHAR2(32) :=
'ADCI.MAP_ADMIN_AREA2_US';
column_name VARCHAR2(32) := ' GEOMETRY ';
sample_ratio INTEGER := 15;
tiling_level INTEGER := 4;
num_tiles INTEGER := 10;
window_obj SDO_GEOMETRY :=
SDO_GEOMETRY(
2003, -- two-dimensional polygon
8307,
NULL,
SDO_ELEM_INFO_ARRAY(1,1003,1), -- one polygon
SDO_ORDINATE_ARRAY(-77.03783,………,38.8777)
);
BEGIN
ret_number :=
SDO_TUNE.ESTIMATE_INDEX_PERFORMANCE
(
table_name,
column_name,
sample_ratio,
tiling_level,
num_tiles,
window_obj,
);
END;
```

4. Quadtree Limitations

In this section we discuss the Quadtree spatial data indexing limitations. There are some limitations in Oracle spatial database to use Quadtree indexing; one of the limitations is the significant performance improvements have not been made for spatial Quadtree indexing. We are encouraged not to use Quadtree indexing for spatial applications, but to use R-tree indexes instead, unless we need to continue using quadtree indexes for special situations. These are some of another limitation of Quadtree spatial data indexing as in this case we use Oracle Spatial 10g:

- Tuning is more complex, and setting the appropriate tuning parameter values can affect performance significantly.
- More storage is required.
- If our application workload includes nearest-neighbor queries, quadtree indexes are slower, and we cannot use the sdo_batch_size keyword. SDO_BATCH_SIZE is a tunable parameter that may affect your query's performance.
- Heavy update activity does not affect the performance of a quadtree index.
- 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. LRS is a natural and convenient means system to associate attributes or events to locations or portions of a linear feature.
- A quadtree index is not recommended for Indexing geodetic data if SDO_WITHIN_DISTANCE queries will be used on it. The SDO_WITHIN_DISTANCE operator identifies the set of spatial objects that are within some specified distance of a given object.
- A quadtree index cannot be used for a whole-Earth index.

5. MapViewer and Oracle Application Server Overview with the Implementation

Oracle Application Server is made up of a middle tier and Oracle Application Server Infrastructure. We deploy and run our applications on the middle tiers. The infrastructure provides services that are used by middle tiers. These services can be shared by one or more middle tiers. Oracle Application Server 10g provides Oracle WebCenter Suite, this feature help us to build our applications more effective that weave transactions, process, business intelligence, structured and unstructured content, communication, and Web 2.0-style services into a highly productive and contextually rich on-line work environment. There are two key feature areas within WebCenter Suite that we use in this research; WebCenter Framework that injects the flexibility and power of our portals into JavaServer Faces to create a completely standards-based, declarative development environment for building all types of user interaction in USA Base Map, and WebCenter Services are a collection of horizontal, Web 2.0 services that we need for show USA Base Map in MapViewer web application.

In this research we use MapViewer web application and Oracle Application Server 10. After we install Oracle Application Server and MapViewer web application, there are some important tasks that we need to set and configure. 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.

Oracle Application Server MapViewer (OracleAS MapViewer) is a programmable tool for rendering maps using spatial data managed by Oracle Spatial or Oracle Locator (also referred to as Locator). Oracle Application Server MapViewer provides tools that hide the complexity of spatial data queries and cartographic for rendering our USA Base Map data, while providing customizable options for more advanced users. These tools can be deployed in a platform-independent manner and are designed to integrate with map-rendering applications.

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.

The flow of action with Oracle Application Server in this application is two-step request or response model, whether the client request a map or some Oracle Application Server MapViewer administrative action.

For a map request:
1. The client requests a USA Base Map, passing in the map name, data source, center location, map size, and, optionally, other data to be plotted on top of a map.

2. The server returns the USA Base Map image (or a URL for the image) and the minimum bounding rectangle (MBR) of the map, and the status of the request.

For an Oracle Application Server MapViewer administrative request:
1. The client requests an Oracle Application Server MapViewer administrative action, passing in the specific type of request and appropriate input values.
2. The server returns the status of the request and the requested USA Base Map information.

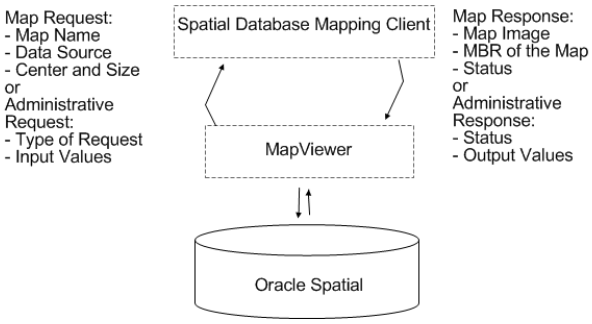

Fig 1. Geographical Map Basic Flow of Action with Oracle Application Server MapViewer

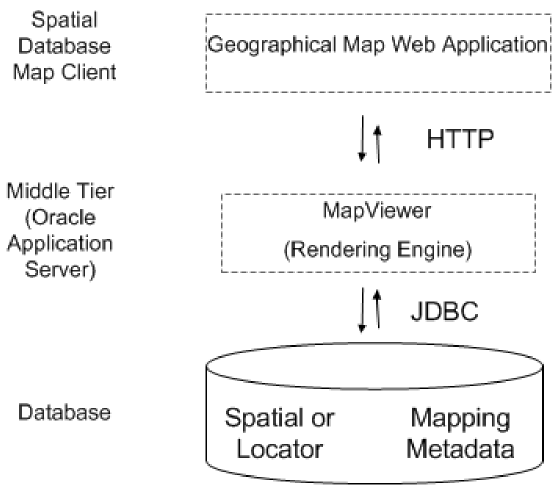

Fig. 2. USA Base Map - Oracle Application Server Architecture

As shown in figure (Fig. 2):
- Oracle Application Server MapViewer is part of the Oracle Application Server middle tier.
- Oracle Application Server MapViewer includes a rendering engine.
- Oracle Application Server MapViewer can communicate with a spatial database Map client Web browser or application using the HTTP protocol.
- Oracle Application Server MapViewer performs spatial data access (reading and writing Oracle Spatial and Oracle Locator data) through JDBC calls to the database.
- The database includes Oracle Spatial or Oracle Locator, as well as mapping metadata.

We can create one of the geographical map 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_NAME>
<NAME_LANGCODE>ENG</NAME_LANGCODE>
<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, main position in the 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,mapCenterL
at,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>
```

6. Performance Evaluation

The model discussed in this paper has been simplified with the following assumptions:

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 4.

Table 2: 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 is depicted in Figure 4, 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. 4). 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.

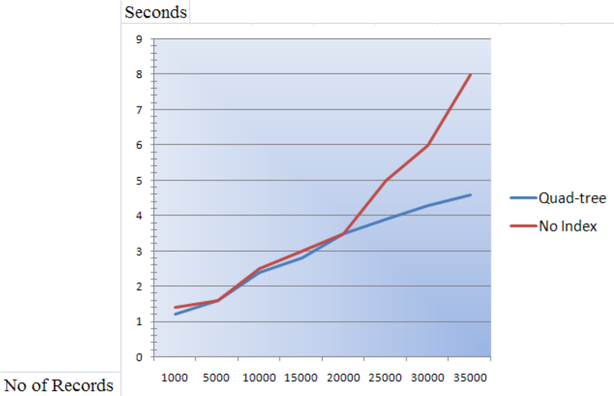

Fig. 4. Average Response Time

We also give a table as shown in table (Table. 3) below, to make it clear see the result.

Table 3: Setting of the Experiments

Number of Records	**No Index (Seconds)**	**Quadtree Index (Seconds)**
1000	1.4	1.2
5000	1.6	1.6
10000	2.5	2.4
15000	3	2.8
20000	3.5	3.5
25000	5	3.9
30000	6	4.3
35000	8	4.6

Other approach has discussed the importance of the spatial component of data items which has been long recognize and gave rise to a successful line of research and development in GIS [28]. They gave the preliminary results on a spatiotemporal database implementation. The proposed system builds on existing database technologies, TimeDB and Oracle Spatial, for temporal and spatial support, respectively. The integration of the spatial and temporal components is achieved with the extension of the TimeDB implementation layer. A set of goals has been established in order to cover both the integration of the spatial support and the enforcement of the temporal requirements in the extended system. However this paper only focuses in the spatial support without discussed about spatiotemporal extended system. It gives deeper exploration of using quadtree spatial data indexing approach to enhance spatial database performance.

7. 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. Quadtree spatial index can be use as spatial database tuning methods to enhance Mobile GIS performance. From our result and the condition that we have discussed, the use of Quadtree spatial index can save the time until 42.5 %. 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.

Acknowledgments

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] Berg, M., Kreveld, M., Overmars, M., Schwarzkopf, O. (1997). *Computational Geometry: Algorithms and Applications*. Springer-Verlag: ISBN 3-540-65620-0.

[8] Morton, G. M. (1996). *A computer Oriented Geodetic Data Base; and a New Technique in File Sequencing*. Technical Report, IBM Ltd. Ottawa, Canada.

[9] Markl, V. (1999). *MISTRAL: Processing Relational Queries using a Multidimensional Access Technique*. Doctoral Thesis University of Munich, Germany.

[10] Ramsak, F., Markl, V., Fenk, R., Zirkel, M., Elhardt, K., Bayer, R. (2000). *Integrating the UB-tree into a Database*

System Kernel. Int. Conf. on Very Large Databases, (VLDB), pp 263-272.

[11] Tropf, H., & Herzog, H. (1981). *Multidimensional Range Search in Dynamically Balanced Trees, Angewandte Informatik, pp 71-77*.

[12] Finkel, R., & Bentley, J. L. (1974). *Quad Trees: A Data Structure for Retrieval on Composite Keys*. Acta Informatica 4 (1): 1-9.

[13] Sahr, K., White, D., Kimerling, A. J. (2003). *Geodesic Discrete Global Grid Systems*. Cartography and Geographic Information Science, 30(2), 121-134.

[14] 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.

[15] 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*.

[16] Geller, S., Talke, J., Krafczyk, M. (2007). *Lattice-Boltzmann Method on Quadtree-Type Grids for Fluid Structure Interaction*. Fluid-Structure Interaction, 270-293.

[17] Tzouramanis, T., Vassilakopoulos, M., Manolopoulos, Y. (2000). *Multiversion Linear Quadtree for Spatio-Temporal Data*. Current Issues in Databases and Information Systems, 279-292.

[18] 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.

[19] Varas, J. (2007). *Mobile Robot Path Planning Among Weighted Regions Using Quadtree Representations*. Computer Aided Systems Theory - EUROCASTâ€™99, 239-249.

[20] Cheng, S. W. & Lee, K. H. (2008). *Quadtree Decomposition, Steiner Triangulation, and Ray Shooting*. Algorithms and Computation, 368-377.

[21] 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.

[22] Varas, J. (2007). *Mobile Robot Path Planning Among Weighted Regions Using Quadtree Representations*. Computer Aided Systems Theory - EUROCASTâ€™99, 239-249.

[23] 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.

[24] Samet, R. & Ozsavas, E. (2007). *Optimization of Quadtree Triangulation for Terrain Models*. Advanced Concepts for Intelligent Vision Systems, 48-59.

[25] Mir, Z. H. & Ko, Y. B. (2007). *A quadtree-based hierarchical data dissemination for mobile sensor networks*. Telecommunication Systems, 117-128.

[26] 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.

[27] 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.

[28] Carvalho, A., Ribeiro, C., Sousa, A. A. (2007). *A Spatio-temporal Database System Based on TimeDB and Oracle Spatial*. IFIP International Federation for Information Processing, 11-20.

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, PM 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.