ITQS: An Integrated Transport Query System

B. Huang[1], Z. Huang[2], H. Li[3], D. Lin[2], H. Lu[2], and Y. Song[2]
[1]Department of Civil Engineering
[2]Department of Computer Science
National University of Singapore, Singapore 117543
[3]Institute of Remote Sensing Applications
Chinese Academy of Sciences, Beijing 100101

1. INTRODUCTION

In this demonstration, we will present the Integrated Transport Query System (ITQS), a system for querying moving objects in a road network. We discuss the system design with a focus on three major aspects: data management, system implementation, and application potential. For data management, we propose a moving object indexing method for road networks: (1) the Voronoi diagram is used together with the R*-tree [1] to represent the regions covered by a road network, and (2) a set of TPR-trees [2] is applied to represent the moving objects in each region. We propose a KNN algorithm for querying moving objects based on the indexing method. It can retrieve the K-nearest moving objects from the current moving object. On system implementation, we introduce the architecture of our system and the communication between the server PC and pocket PCs being the mobile clients. We also implement a graphics user interface (GUI) on the pocket PC for query operations which allows users to view the results of their queries, with different colors used to show different road traffic conditions. Finally, we discuss the application potential of our work for taxi booking. This work is part of a bigger project called SpADE (Spatio-temporal Autonomic Database Engine for location-based services). The details can be found at http://www.comp.nus.edu.sg/~ ooibc/spade/.

We will demonstrate the system using a notebook as the server, and a couple of pocket PCs as clients. We use the wireless network data transmission via Wi-Fi/802.11b or the infrared light data transmission via IrDA.

2. DATA MANAGEMENT

We observe that in a road network in the real world, (1) the roads are static, and (2) queries are not random, i.e., they are usually posed around certain places like shopping malls, schools, hospitals, etc. Based on this observation, we propose a two-layer indexing.

SIGMOD 2004 June 13-18, 2004, Paris, France.

In the upper layer, the R*-tree [1], is used together with the Voronoi Diagram to describe the regions covered by a road network. Each entry of the leaf nodes of the R*-tree represents an elementary region that is a Voronoi cell. Neighboring Voronoi cells are summarized and organized in the internal nodes. The Voronoi Diagram is constructed with *pivots*, the reference points representing the centers of interest in the road network. In the lower layer, in each elementary region, as referenced by a leaf node of the R*-tree, we use a TPR-tree to index moving objects. Each entry of a leaf node represents a moving object with the object identity number, the current position, velocity, and the time when the moving object discloses its position. Each TPR-tree is associated with additional data structures that provide the road network information. Based on the indexing, the KNN query algorithm is listed as follows:

Algorithm knnQuery (q, k, t, result)
1. Find the nearest reference point RP_i of q
   in the upper layer R*-tree
2. // Search the TPR-tree of RP_i
   Invoke **searchTPR**(TPR-tree of RP_i, q, k, t, result)
3. // Search the ADT Tables
   Invoke **searchADT**(q, k, t, RP_i, result)
4. D_{qc} = the distance between q and its
   nearest boundary of the Voronoi cell of RP_i
5. D_{qk} = the network distance from q to the
   k-th candidate nearest neighbor so far
6. if ($D_{qc} < D_{qk}$)
7. Invoke **enlargedSearch**(q, k, t, RP_i, result)

end knnQuery

Given a query point q, we first find its nearest pivot point by searching the R*-tree of the upper level indexing. Then, the TPR-tree relevant to the pivot is searched to get k nearest neighbor candidates. After that, we check whether the query circle (centered at q with radius equal to the network distance from q to its k'th nearest neighbor) is wholly inside the current Voronoi cell. If so, the algorithm stops. Otherwise, the adjacent cells will also be searched: first, all adjacent cells will be sorted in ascending order by the distance from the current pivot to each boundary nearest to the pivot; second, each adjacent cell will be visited until one cell is found to be outside of the query circle (the radius of the query circle might be shortened during the search).

To compute the network distance, we pre-compute the node-

node network distances of the road links and store the results in the *Network Distance Table (NDT)*. For two specified moving objects, their network distance equals to the sum of the distance between the objects and the nodes that they move towards in the links, and the node-node distance of these nodes. The pre-computed node-node network distances are also used as the upper bounds for the network distance of two moving objects in the related links to accelerate the search process.

To insert a moving object, first, the algorithm finds the nearest pivot point and inserts it into the corresponding TPR-tree. Second, if this object may go outside the current Voronoi cell within a certain maximum update interval, we store a pointer to this object in the Voronoi cell that it may enter in the future. Deletion also consists of two steps, which are similar to those for insertion.

3. SYSTEM IMPLEMENTATION

As illustrated in Figure 1, the proposed system embodies a client-server architecture with mobile clients. A mobile client consists of three hardware components: a Trimble GPS receiver, a Compaq iPAQ pocket PC, and a wireless or infrared card. The server is a desktop PC connected to a fixed network. The mobile client communicates with the server via IrDA and Wi-Fi/802.11b for infrared and wireless network cards respectively. Other wireless communication protocols, such as GSM and CDMA, can also be used.

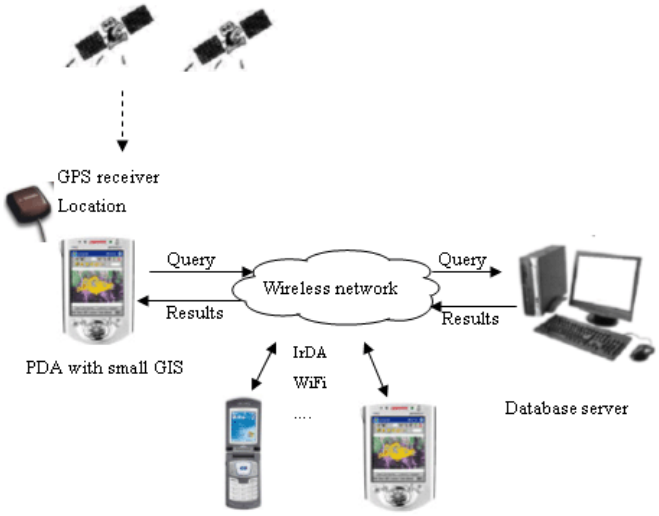

Figure 1: Illustration of the major components of the system and the wireless network.

A Trimble GPS Pathfinder Pocket Receiver is used to locate the position of a mobile client (http://www.trimble.com). It can continuously track up to eight satellites with two- to five-meter accuracy after differential correction and a one-second update rate. The user can get his/her current position in the form of longitude and latitude. The system also can report the current position of the user automatically with a specified period.

The network communication between the Compaq iPAQ pocket PC and the desktop PC serves the purposes of position updates, query requests, and transmissions of query results. On the server side, windows sockets programming is accomplished based on MFC. Each Compaq iPAQ pocket PC comes with 64MB RAM, and we can store the entire Singapore map on each of them. This significantly reduces the network load.

On the client side, a user interacts with the system through a graphics user interface (GUI). There are six pull-down menus: File (GIS layers management), View (visualization), GPS (set up, record and position), Comm (wireless communication IrDA and WiFi), and Query (KNN, circle and rectangle window queries).

Visualization of query results should be fast and intuitive. We use different colors to indicate the different traffic conditions of roads, e.g., red/blue color for slow/smooth traffic. We use icons, points, lines, and polygons to show different static and moving objects. The moving object ID can also be displayed together. Besides the graphics display, data tables are also implemented. When a user needs to know more details, e.g., area, perimeter, street name, starting/ending point, zip code, etc., a table can be displayed on request.

4. APPLICATION POTENTIAL

In this section, we discuss the application potential of the ITQS for taxi booking. Taxis are one of the major means of transport in Singapore. However, the telephone booking fee is S$2.30 – relatively high compared to the taxi fare which starts between S$2.40 and S$2.80 for the first 1km and follows a lower rate for the subsequent distance. For example, from Singapore Changi Airport to City Hall, the total fare is about S$10.00. The major reason for the high booking fee is that in the current system, a booking request is broadcast to all the cabs of the company. It is a waste of network bandwidth. Furthermore, an available cab near the request site may not get the request. The high booking fee encourages taxi drivers to accept a request even if they are not near the request site. Once a taxi driver accepts the request, it blocks others from doing so, including those nearer the request site.

The ITQS can find its role in the new taxi booking service. Efficient indexing for moving objects is necessary considering that Comfort, one among several taxi companies in Singapore, alone owns more than 11,000 cabs. Furthermore, pivot-based indexing can well serve the new booking system, where a request is sent locally. The KNN query is necessary because the nearest cab may be unable to take a call due to a pre-existing request. The road network distance metric should also be used in the query.

5. REFERENCES

[1] N. Beckmann, H.-P. Kriegel, and B. S. R. Schneider. The r*-tree: An efficient and robust access method for points and rectangles. *ACM SIGMOD*, pages 322–331, 1990.

[2] S. Saltenis, C. S. Jensen, S. T. Leutenegger, and M. A. Lopez. Indexing the positions of continuously moving objects. *ACM SIGMOD*, pages 331–342, 2000.