IR^2 Trees for Spatial Database Keyword Search

NSF Award: CNS-0220562
MII: Infrastructure for Research and Training in Database Management for Web-based Geospatial Data Visualization with Applications to Aviation

PI: Naphtali Rishe
Florida International University
School of Computing and Information Sciences

Co-PI: Boonserm Wongsaroj
Florida Memorial University
Department of Computer Sciences and Mathematics

ABSTRACT

Our MII-provided infrastructure and research support has enabled us to perform basic research in keyword search. Many applications require finding objects closest to a specified location that contains a set of keywords. For example, online yellow pages allow users to specify an address and a set of keywords. In return, the user obtains a list of businesses whose description contains these keywords, ordered by their distance from the specified address. The problems of nearest neighbor search on spatial data and keyword search on text data have been extensively studied separately. However, to the best of our knowledge there is no efficient method to answer spatial keyword queries, that is, queries that specify both a location and a set of keywords. In this work, we present an efficient method to answer top-k spatial keyword queries. To do so, we introduce an indexing structure called IR^2-Tree (Information Retrieval R-Tree) which combines an R-Tree with superimposed text signatures. We present algorithms that construct and maintain an IR^2-Tree, and use it to answer top-k spatial keyword queries. Our algorithms are experimentally and analytically compared to current methods and are shown to have superior performance and excellent scalability.

1. INTRODUCTION

An increasing number of applications require the efficient execution of nearest neighbor queries constrained by the properties of the spatial objects. Due to the popularity of keyword search, particularly on the Internet, many of these applications allow the user to provide a list of keywords that the spatial objects (henceforth referred to simply as objects) should contain, in their description or other attribute. For example, online yellow pages allow users to specify an address and a set of keywords, and return businesses whose description contains these keywords, ordered by their distance to the specified address location. As another example, real estate web sites allow users to search for properties with specific keywords in their description and rank them according to their distance from a specified location. We call such queries *spatial keyword queries*.

A spatial keyword query consists of a query area and a set of keywords. The answer is a list of objects ranked according to a combination of their distance to the query area and the relevance of their text description to the query keywords. A simple yet popular variant, which is used in our running example, is the *distance-first spatial keyword query*, where objects are ranked by distance and keywords are applied as a conjunctive filter to eliminate objects that do not contain them.

Figure 1, which is our running example, displays a dataset of fictitious hotels with their spatial coordinates and a set of descriptive attributes (name, amenities). An example of a spatial keyword query is “find the nearest hotels to point *[30.5, 100.0]* that contain keywords *internet* and *pool*”. The top result of this query is the hotel object H_7.

	Name	Latitude	Longitude	Amenities
H_1	Hotel A	25.4	-80.1	tennis court, gift shop, spa, Internet
H_2	Hotel B	47.3	-122.2	wireless Internet, pool, golf course
H_3	Hotel C	35.5	139.4	spa, continental suites, pool
H_4	Hotel D	39.5	116.2	sauna, pool, conference rooms
H_5	Hotel E	51.3	-0.5	dry cleaning, free lunch, pets
H_6	Hotel F	40.4	-73.5	safe box, concierge, internet, pets
H_7	Hotel G	-33.2	-70.4	Internet, airport transportation, pool
H_8	Hotel H	-41.1	174.4	wake up service, no pets, pool

Figure 1. Sample dataset of hotel objects.

Unfortunately there is no efficient support for top-*k* spatial keyword queries, where a prefix of the results list is required. Instead, current systems use ad-hoc combinations of nearest neighbor (NN) and keyword search techniques to tackle the problem. For instance, an R-Tree is used to find the nearest neighbors and for each neighbor an inverted index is used to check if the query keywords are contained. We show that such two-phase approaches are inefficient.

We present a method to efficiently answer top-*k* spatial keyword queries, which is based on the tight integration of data structures and algorithms used in spatial database search and Information Retrieval (IR). In particular, our method consists of building an Information Retrieval R-Tree (*IR^2-Tree)*, which is a structure based on the R-Tree [Gut84]. At query time an incremental algorithm is employed that uses the IR^2-Tree to efficiently produce the top results of the query.

The IR^2-Tree is an R-Tree where a signature (Faloutsos and Christodoulakis [FC84]) is added to each node *v* of the IR^2-Tree to denote the textual content of all spatial objects in the subtree rooted at *v*. Our top-*k* spatial keyword search algorithm, which is inspired by the work of Hjaltason and Samet [HS99], exploits this information to locate the top query results by accessing a minimal portion of the IR^2-Tree. This work has the following contributions:

- The problem of top-*k* spatial keyword search is defined.
- The IR^2-Tree is proposed as an efficient indexing structure to store spatial and textual information for a set of objects. Efficient algorithms are also presented to maintain the IR^2-Tree, that is, insert and delete objects.
- An efficient incremental algorithm is presented to answer top-*k* spatial keyword queries using the IR^2-Tree. Its performance is analytically and experimentally evaluated and compared to current approaches. Real datasets are used in our experiments that show the significant improvement in execution times.

Note that our method can be applied to arbitrarily-shaped and multi-dimensional objects and not just points on the two dimensions, which are used in our running examples for clarity.

2. IR^2-TREE

The IR^2-Tree is a combination of an R-Tree and signature files. In particular, each node of an IR^2-Tree contains both spatial and keyword information; the former in the form of a minimum bounding area and the latter in the form of a signature. An IR^2-Tree facilitates both top-*k* spatial queries and top-*k* spatial keyword queries as we explain below.

More formally, an IR^2-Tree *R* is a height-balanced tree data structure, where each leaf node has entries of the form *(ObjPtr, A, S)*. *ObjPtr* and *A* are defined as in the R-Tree while *S* is the signature of the object referred by *ObjPtr*. A non-leaf node has entries of the form *(NodePtr, A, S)*. *NodePtr* and *A* are defined as in the R-Tree while *S* is the signature of the node. The signature of a node is the superimposition (OR-ing) of all the signatures of its entries. Thus a signature of a node is equivalent to a signature for all the documents in its subtree. Figure 2 shows an IR^2-Tree for the sample dataset of Figure 1.

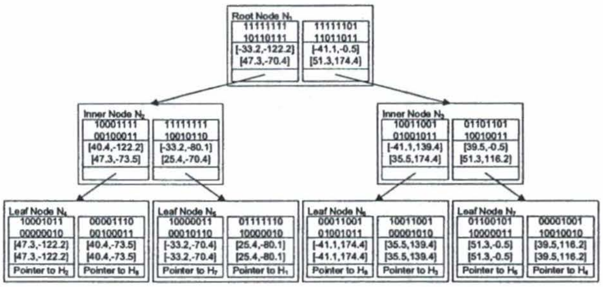


Figure 2. IR2-Tree for dataset of Figure 1.

3. ANSWERING TOP-*K* SPATIAL KEYWORD QUERIES

To the best of our knowledge, there is no specialized algorithm to solve a top-*k* spatial keyword query. However, there are well known ways in which this query can be solved in a two-step fashion by applying Nearest Neighbor search and Information Retrieval techniques. We have developed a distance-first IR^2-Tree algorithm that exploits the structure of the IR^2-Tree to efficiently answer distance-first top-*k* spatial keyword queries. The tree traversal is based on the Incremental Nearest Neighbor algorithm (Figure 3). The key advantage of this algorithm is that it prunes whole subtrees if their root-node signature does not match the query signature *Signature(Q.t)*. This happens because the signature of an IR^2-Tree node is composed from all the signatures of its children. This pruning occurs in addition to the spatial pruning provided by the traditional Incremental Nearest Neighbor. By tightly integrating these two pruning mechanisms, the distance-first IR^2-Tree algorithm accesses a minimal set of IR^2-Tree nodes and objects to answer a distance-first top-*k* spatial keyword query.

4. ANALYSIS

To simplify the analysis we will only consider pruning of IR^2-Tree subtrees on the leaf level of the IR^2-Tree, which is the most significant. For a leaf node *v* to be pruned, it must (a) not contain *w* and (b) not be a false positive. Leaf node *v* of the IR^2-Tree, which has *e* pointers to objects, has probability P_{NK} to not contain any object that contains *w*, where

$$P_{NK} = (1-q)^e.$$

The probability P_{FP} of a false positive is computed [ZMR98] by

$$P_{FP} = (1-(1-\frac{1}{r})^{sd})^s \qquad (1)$$

The probability P_{NFP}' that v is not a false positive is

$$P_{NFP}' = (1-P_{FP})^e$$

where P_{FP} is defined in Equation 1. Hence, the probability P_{PR} that v is pruned is

$$P_{PR} = P_{NK} \cdot P_{NFP}' = ((1-q)(1-P_{FP}))^e \qquad (2)$$

Thus, the number of accesses on the IR^2-Tree will be less that the ones on the R-Tree by a factor of P_{PR}. The number of objects retrieved is $k+P_{FP} \cdot k/q$ since we expect to go up to the k/qth closest object and each one of them is retrieved with probability $P_{FP.}$

For the IR^2-Tree the additional space required is the storage of the signatures of the nodes of the R-Tree. Hence the space requirement is

$$Space_{IR2\text{-}Tree} \approx 6 \cdot n + n \cdot r/word\_length$$

To reduce the space overhead to store the signatures, we can use compression techniques [Sal97] which provide considerable space savings. However, in our experiments we do not use compression because we found that the small space savings are not worth the time overhead for compression/decompression. Compression techniques can also be used in the inverted index storage, which achieve a compression of about 30% [NMN+00]. Our experimental analyses have confirmed the performance of our IR^2-Tree; these results will be published soon.

5. Top-*k* queries

Top-*k* queries [Fag01, BGM02] handle the aggregation of attribute values of objects in the case where the attribute values lie in different sources. For example [BGM02] consider the problem of ordering a set of restaurants by distance and price. They present an optimal sequence of random or sequential accesses on the sources (e.g., Zagat for price and Mapquest for distance) in order to compute the top-*k* restaurants. They view the sources as black boxes in contrast to our work where we assume full access which allows us to build an IR^2-Tree.

Chen et al. [CSM06] tackle the problem of finding web pages (objects) whose area (footprint) intersects the query-specified area (query footprint), and also contain a set of keywords. Our problem is different since objects do not have to intersect the query footprint, but are ranked according to their distance from it. Their work focuses on finding the footprint of a web page and its "closeness" to the query footprint, but they do not consider any elaborate indexing mechanisms, except for R-tree and inverted index, which we study as our baseline algorithms. Vaid et al. [VJJS05] and Martins et al. [MSA05] present techniques to combine the output of a text and a spatial index to answer a spatial keyword query. These techniques are very similar to the baseline algorithms we use. However, they do not consider combining these indexes in a single structure like our IR^2-tree. Further, Vaid et al. [VJJS05] use a grid-based distribution of the spatial objects.

6. CONCLUSIONS

We have proposed a solution which is dramatically faster than current approaches and is based on a combination of R-Trees and signature files techniques. In particular our MII support has allowed us to introduce the IR^2-Tree. We have quantitatively and experimentally evaluated our technique, which proved its superior performance.

REFERENCES

[BGM02] Nicolas Bruno, Luis Gravano, Amélie Marian. Evaluating Top-*k* Queries over Web-Accessible Databases., In Proceedings of the 18th IEEE International Conference on Data Engineering (ICDE '02), 2002.

[CSM06] Yen-Yu Chen, Torsten Suel, Alexander Markowetz. Efficient Query Processing in Geographic Web Search Engines. SIGMOD 2006

[Fag01] Ronald Fagin, Amnon Lotem, Moni Naor: Optimal Aggregation Algorithms for Middleware. In PODS 2001

[FC84] Christos Faloutsos, Stavros Christodoulakis: Signature Files: An Access Method for Documents and Its Analytical Performance Evaluation. In ACM Trans. Inf. Syst. 2(4): 267-288(1984)

[Gut84] A. Guttman. R-Trees: a dynamic index structure for spatial searching. In SIGMOD Conference, 1984.

[HS99] G.R. Hjaltason and H. Samet. Distance browsing in spatial databases. In ACM Transactions on Database Systems, Vol. 24, No. 2, 1999

[MSA05] B. Martins, M. Silva, and L. Andrade. Indexing and ranking in Geo-IR systems. In Proc. of the 2nd Int. Workshop on Geo-IR (GIR), November 2005.

[NMN+00] Gonzalo Navarro, Edleno Silva de Moura, Marden S. Neubert, Nivio Ziviani, Ricardo A. Baeza-Yates: Adding Compression to Block Addressing Inverted Indexes. In Information Retrieval 3(1): 49-77 (2000), 2000

[Sal97] D. Salomon. Data Compression. The Complete Reference. Springer, New York, 1997.

[VJJS05] S. Vaid, C. B. Jones, H. Joho, and M. Sanderson. Spatio-textual indexing for geographical search on the web. In Proc. of the 9th Int. Symp. on Spatial and Temporal Databases (SSTD), 2005.

[ZMR98] Justin Zobel, Alistair Moffat, Kotagiri Ramamohanarao: Inverted Files Versus Signature Files for Text Indexing. In ACM Trans. Database Syst. 23(4): 453-490 (1998)

Project Title: Frameworks for the Development of Efficient and Scalable Knowledge-based Systems

Proposal Number: CNS-0220590

Principal Investigator: Enrico Pontelli

Institution: New Mexico State University

Pathways in Computer Science

Introduction/Background

The MII project is articulated in two inter-twined directions – a *research* direction and an *educational* direction. The overall objectives of the project are to sustain and grow the research capabilities of the Department of Computer Science, with particular emphasis on the development of collaborative research initiatives with underlying themes related to knowledge-based system, and to integrate the research efforts in a comprehensive educational plan, aimed at promoting participation of a diverse student population to Computer Science education and training.

In this report we highlight some of the activities and results accomplished within the educational and outreach component of the project.

The Pathways in Computer Science Project: Motivations

We have adopted the term *Pathways in CS* to generally denote the set of outreach and educational support activities that we developed within the scope of this MII project. The term highlights the intent of creating opportunities and resources that will strengthen the links between the different components of the pipeline that runs from high school to the graduate program.

Program Components

The Pathways program includes the following components:

- *Recruitment/Outreach*, performed towards students in high school and at community and tribal colleges in the region
- *Pre-entrance training*, performed through summer experiences
- *Retention and success*, achieved through mentoring and tutoring throughout the gateway courses of the undergraduate program
- *Advanced studies*, through research involvement and transition from the undergraduate to the graduate program

Identification of Post-translational Modifications via Blind Search of Mass-Spectra. Dekel Tsur, Stephen Tanner, Ebrahim Zandi, Vineet Bafna, Pavel A. Pevzner. Nature Biotechnology 23, 1562-2567 (Dec 2005).

Identification of Post-translational Modifications via Blind Search of Mass-Spectra. Dekel Tsur, Stephen Tanner, Ebrahim Zandi, Vineet Bafna, and Pavel A. Pevzner IEEE Computer Society Bioinformatics Conference (CSB) 2005.

N. Taesombut and A. A. Chien. "Integrated Resource Management for High Application Performance and Resource Efficiency in LambdaGrids, paper in submission (Supercomputing 2007).

N. Taesombut and A. A. Chien. "Evaluating Network Information Models on Resource Efficiency and Application Performance in Lambda-Grids," paper in submission (Supercomputing 2007).

R. Singh, N. Schwarz, N. Taesombut, et al. "Real-time Multi-scale Brain Data Acquisition, Assembly, and Analysis Using End-to-End OptIPuter," *Journal of Future Generation Computer Systems*, Vol. 22(8), October 2006. pp. 1032-1039.

N. Taesombut, X. Wu, A. A. Chien, et al. "Collaborative Data Visualization for Earth Sciences with the OptIPuter," *Journal of Future Generation Computer Systems*, Vol. 22(8), October 2006. pp. 955-963.

Eric Weigle and Andrew A. Chien, "Partial Content Distribution on High Performance Networks", Proceedings of IEEE International Symposium on High Performance Distributed Computing (HPDC), 2007.

Richard Huang, Henri Casanova, and Andrew Chien. Generating Grid Resource Requirement Specifications, submitted to the IEEE International Symposium on High Performance Distributed Computing (HPDC 2007). (June 2007)

Fei Sha and Lawrence K. Saul (2007). Large margin hidden Markov models for automatic speech recognition. To appear in Advances in Neural Information Processing Systems 19 (edited by B. Scholkopf, J. C. Platt and T. Hofmann), MIT Press, Cambridge, MA., 2007. (Best Student Paper Award0

Fei Sha and Lawrence K. Saul (2007). Comparison of large margin training to other discriminative methods for phonetic recognition by hidden Markov models. ICASSP 2007 (Honolulu). (Finalist of Best Student Paper)

Fei Sha (2007). Large margin training of acoustic models for speech recognition. University of Pennsylvania. (Ph.D dissertation)

Honghao Shan, Lingyun Zhang, Garrison W. Cottrell, "Recursive ICA", in preparation.

Piotr Dollár, Vincent Rabaud and Serge Belongie. Non-Isometric Manifold Learning: Analysis and an Algorithm, ICML 2007, Corvallis, Oregon.

Vincent Rabaud, Piotr Dollár and Serge Belongie. Learning Class Specific Image Transformations
ICCV 2007 submission, Rio, Brazil.

VIII. Future and Strategic Directions

We made significant additions we have made to FWGrid's core infrastructure in 2H2006, (doubling the compute power and storage by adding 160 Dell SC1425 3.2 Xeon nodes (4GB Mem and 500GB's storage)). FWGrid's total capacity is now ~1.5 Tflops CPU power, ~1TB RAM and ~160TB of disk storage, all connected over the fast 10Gb network. In 2007, we have added some new platforms, purchasing several Playstation 3's (CELL platform) to allow low-level programming of significant multicore parallelism. In addition, we are developing a visualization cluster, to provide data viz and analysis capabilities to our current computational and storage capabilities. Due to the changing technical directions of our driving researcher constituencies, we will reduce our extension of the system with additional cameras, wireless access points, and other novel input devices as part of the overall FWGrid open testbed system. The purpose of these elements is to enable a broader range of interactive, visual, mobile, and simulation experiments.

Thin Client Technologies for Spatial Data Visualization

NSF Award: CNS-0320956
MRI: Acquisition of Research Instrumentation for Web-based Visualization of Spatio-Temporal Data

PI: Naphtali Rishe
Florida International University
School of Computing and Information Sciences

Co-PI: Ouri Wolfson
University of Illinois at Chicago
Department of Computer Science

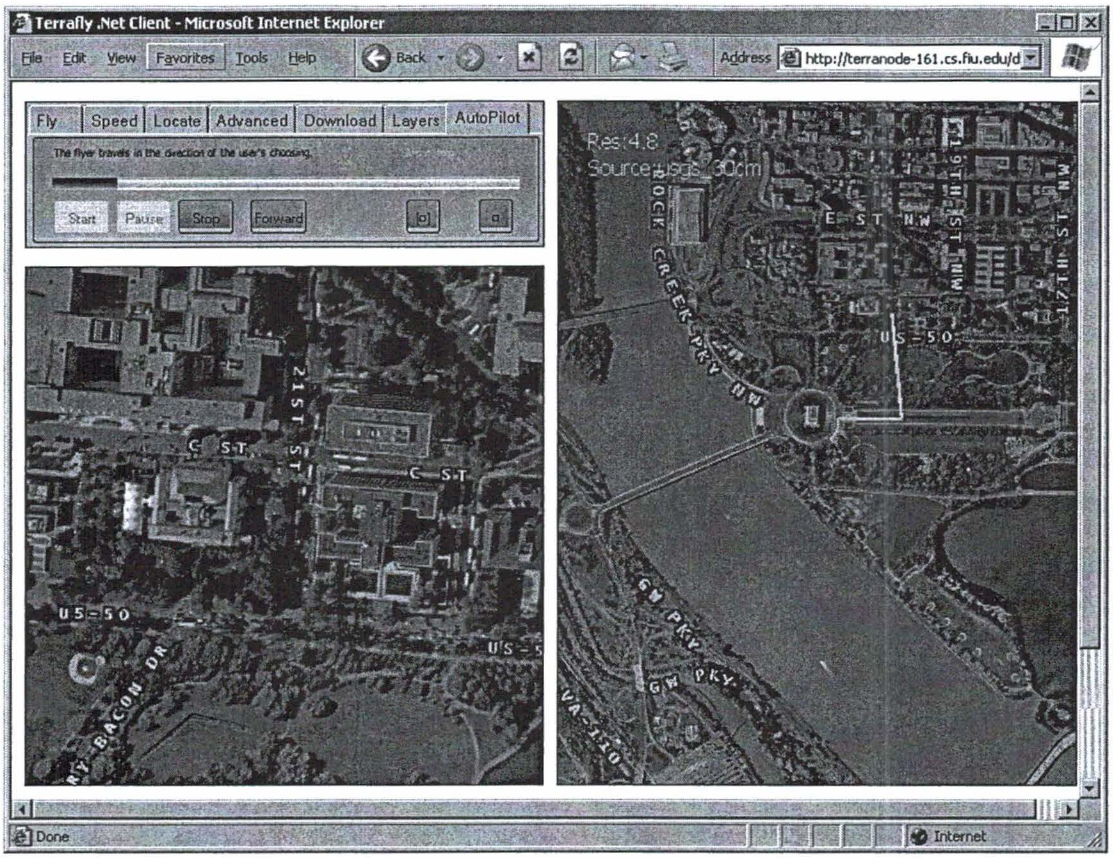
ABSTRACT

FIU's TerraFly spatial data dissemination project (http://terrafly.fiu.edu) has been substantially improved via our NSF MRI funding as well as via IBM-donated equipment. It is enabling several forms of data dissemination and visualization and is serving research projects that are supported by NSF MII and CREST funding. TerraFly is a web-based maps & aerials delivery product with high efficiency of Web data delivery to a thin client, namely a Web browser. TerraFly allows the user to simultaneously visualize multiple datasets in synchronous thin-client windows and drill down and query data, to see results in graphic, tabular and textual forms utilizing the TerraFly PointData and GeoQuery mechanisms. The PointData mechanism presents drill-down information about a point. GeoQuery allows html-client ergonomic point-and-click formulation of complex geospatial queries by unskilled users and efficient computation of results. We have developed and prototyped a classification of thin clients based on capabilities and requirements and will present this classification and our results.

1. TERRAFLY

In preparation for a discussion on thin client technologies, we will first discuss the existing thin client product our team has designed and developed over the past 8 years – that is **TerraFly**. TerraFly is a web-based maps & aerials delivery product with high efficiency of Web data delivery to a thin client, namely a Web browser. TerraFly allows the user to simultaneously visualize multiple datasets in synchronous thin-client windows and drill down and query data, to see results in graphic, tabular and textual forms utilizing the TerraFly PointData and GeoQuery mechanisms. The PointData mechanism presents drill-down information about a point. GeoQuery allows html-client ergonomic point-and-click formulation of complex geospatial queries by unskilled users and efficient computation of results. Figure 1, depicts the TerraFly application on Autopilot (a route identified by one user and sent to another user for viewing). The right zoomed out window of this applet shows the tracking in yellow.

The TerraFly team in conjunction with the NASA Regional Applications Center at Florida International University has conducted extensive research on thick and thin clients as well as 2-tier, 3-tier and N-tier architectures in Web mapping application areas.



The flyer travels in the direction of the user's choosing.

Figure 1. TerraFly on Autopilot

2. CATEGORIZATION OF CLIENTS

In the course of our development of TerraFly, we have experimented with several implementation strategies. This experimentation has resulted in our development of the following hierarchy of client types.

- Group 1, Level 1. The "active server" and "static" client solution allowing full interactivity in the 2D Web-mapping scenario - This technology is applicable to browsers which are not JavaScript, Java, or .NET enabled. In particular this is an important technology for minimal PDA-class devices. Note that the interactive "2D flight," as well as the "2.5D flight," and, most surprisingly, the "3D flight" are feasible in this scenario though the mechanism of our video streaming Web-mapping technology provided the device allows the embedding of a video stream player in the Web page (most PDA devices have video players and allow video streaming).
- Group 1, Level 2. The "active server" and "active client" with JavaScript allows easier and faster interaction with the server, especially for a "2D flight" - A "2.5D flight" is a possibility in this situation, even though we prefer to deploy it in the Group 2 solution via .NET. Let's note that the "2.5 flight" and, importantly, the "3D flight" are also available for these devices through interactive video

streaming. Most PDA devices fall into this level, with JavaScript being restricted typically to its versions available in Microsoft Internet Explorer v.4.0.

- Group 1, Level 3. The "active server" and "active client" with the latest JavaScript as well as XML capabilities (AJAX/ATLAS model) - This has essentially the same capability as Level 2 above with one notable difference: the availability of the latest libraries to the client makes the task of achieving the same results as in Level 2 above to some extent easier for the developer. The "2.5D flight" and "3D flight" interactive video streaming is also feasible at this level.
- Group 2. Java or .NET enabled "active client" with "active server" - This class of devices supports the TerraFly "2.5D flight" solution without the need to resort to interactive video streaming. The interactive video streaming puts considerable load on the server farm and hence, if the client is able to bear most of the computational need for a particular mode of flight, it is desirable to offload computations to the client (even a thin client!), provided the client has idle computation power available and sufficient capacity to perform such computations. We would like to note that since Java and .NET are general purpose languages with rich libraries. They could easily be deployed on the client and re-use existing server-side Web-mapping algorithms to various degrees to reduce the load on the servers and increase the promptness of the interactive response and the Web-map ergonomic features by tapping into the impressive power of the Java/.NET programming languages and the available computational power on the client.
- Group 3. "3D flight" without any need of interactive video-streaming

3. Other Issues Associated with Thin Client Implementations

We now discuss the traffic and servers load, fault tolerance as well as flight prediction and pre-fetching issues which arise with Web-mapping solutions in all the classes defined above.

Load Balancing and Fault Tolerance

We have preformed a study of web-map usage based on various user groups and GIS problem sub-domains with the TerraFly product. Such groups ranged from thousands of subscribers using particular datasets geared toward real estate data, demographic data, etc., to tens of thousands of non-subscribers per day using most of the major subsets of common interest. These studies allowed us to determine the scalability and availability requirements and propose, implement and deploy mechanisms to sustain operations with such requirements. Our approach to load balancing and fault tolerance in Web-mapping took into consideration the JavaScript, Java applet and .NET applet deployment and security models, which allowed smooth virtualization of all server farms participating in the TerraFly Web-mapping data generation and delivery, thus creating a virtual representation most suitable to these deployment and security models. We have profiled execution and implemented algorithms with very fast response times for both raster (bitmap) content of response and text content of response times. In the raster data response area we have developed algorithms for aerial photography and satellite imagery, with efficient storage and fast delivery (dynamic mosaicing) of the rasterization of vector data (maps generation).

Application Security

The TerraFly research project employs various techniques of data security targeting specific needs of optimal network bandwidth, computational resources availability and desired levels of protection. We have researched and deployed a multi-layered architecture for encryption and secure access with a combination of access control based on high strength encryption and digital signatures, which is fast and efficient due to access control resolution dialog where there is rather small amounts of data to be transferred. According to user-defined priorities, the TerraFly system allows various protection levels for the massive data transfer streams, such as:

- Partial encryption of compressed raster data objects, which makes it computationally difficult to reconstitute original image because key data elements are encrypted. This approach allows us to reduce the CPU overhead associated with the encryption and decryption process while still providing some level of data protection. We found this method to be a good application for base maps and base imagery which have rather low levels of confidentiality in most cases.
- Full encryption with symmetric encryption (either fast medium strength or slower high strength methods), which could be applied to raster base maps, as well as to higher sensitivity smaller objects (vector map elements which contain higher confidentiality requirements).
- Public-key/private-key encryption for the highest confidential elements of data and due to specific requirements of deployment where usage of only public-key/private-key deployment is preferred. This is the slowest method and should be used sparingly.

All these various levels of encryption and access could be combined together to achieve optimal response times within limited network bandwidth and limited computational power available with particular devices and systems. The layered structure of Web maps allows for the application of flexible methods of security stratification in communication channels.

Performing Complex Queries that Blend Geospatial Queries with Keyword Search

Many applications require finding the objects closest to a specified location that contains a set of keywords. For example, online yellow pages allow users to specify an address and a set of keywords, and return businesses whose description contains these keywords, ordered by their distance to the specified address location. The problems of nearest neighbor search and keyword search have been extensively studied separately. However, to the best of our knowledge there is no efficient method to answer spatial keyword queries, that is, queries that specify both a location and a set of keywords.

We have developed algorithms to implement combined spatial and keyword queries. In particular, we use a modification of the R-Tree, called IR^2-Tree (Information Retrieval R-Tree), which incorporates the principles of Signature Files to efficiently answer top-k spatial keyword queries. We have developed algorithms to maintain the IR^2-Tree and use it to incrementally search for the nearest neighbors that contain the query keywords. Our algorithms have been experimentally and analytically compared to current state-of-the-art methods and have been shown to have superior performance and excellent scalability.

Fast Main-Memory Resolution of Geospatial Queries

We have introduced and are working on further improvement of, the S-Tree, a high performance spatial point indexing structure. S-Tree is a hybrid structure that uses the quad-tree style space decomposition and a hash table for searching. The performance of the data structure is mostly a hash table performance and is a better than O(log(N)), where N is the size of the tree. S-Tree has an easy to compute hashing function that improves the performance and reduces the index storage requirements by eliminating the need to store the quad-tree keys. S-Tree uses a hash table allowing it to start searching from an arbitrary level of the tree without traversing many levels close to the root or to the leaves of the tree as it is done in other algorithms. S-Tree is also very compact, allowing large spatial indexes to be stored in the main memory, and it is optimized for main-memory index residence.