Use of Graph Databases in Tourist Navigation Application

Anahid Basiri[1], Pouria Amirian[2], and Adam Winstanley[2]

[1] Nottingham Geospatial Institute, the University of Nottingham, UK
Anahid.basiri@nottingham.ac.uk
[2] Department of Computer Science, National University of Ireland, Maynooth, Ireland
pouria.amirian@nuim.ie

Abstract. Navigation services, such as car navigation services, are widely used nowadays. However current car navigation systems are not fully suitable for the navigational needs of tourists. In contrast with drivers, tourists are not constrained by road networks and can walk in places where vehicles are not allowed to move. As current turn-by-turn navigational instructions to be given to vehicle's derivers are mostly based on street network-based algorithms, this way of navigating is not fully suitable for tourists as they do not only move on streets. In addition, Tourists want to see important feature of the area, no matter they take longer path rather than shortest. They want to get navigated through the most touristic path. In order to provide such tourist-specific navigation services, a landmark-based solution was considered. it calculates a route passing more landmarks. This may help user to visit attractive part of a place. It is possible to provide users with the navigational instructions landmark-by-landmark rather than turn-by-turn. In this application, a graph database is used because of having highly connected data and also need to remove the mapping layer between physical storage layer and application logic layer to have more availability and responsiveness.

Keywords: Landmark-based navigation, Graph databases, Tourist navigation, Location Based Services (LBS).

1 Introduction

Nowadays, car navigation has become one of the most widely used examples Location-Based Services (LBSs). But current car navigation systems are not fully suitable for the navigational needs of all categories of users such as tourists and visitors, pedestrians, users with special needs such as handicaps. There is a need to develop different navigation application for different categories of users to consider their specific needs, constraints and preferences. Tourists and visitors can easily go into a building or underground to get to their destination where GPS's signals are unavailable. Seamless Indoor and outdoor navigation is one of the most important features which should be handled in their navigation application and is still topic of many research projects (Karimi, 2011), (Li et al., 2013), (Hansen et al., 2009).

In addition to seamless positioning solutions, another aspect of tourist navigation application which has not been covered in car navigation is non-turn-by-turn navigational instruction delivery. Tourists and visitors would like to get some information about each Point of Interest (POI) while traversing. Instead of having two different applications; one for navigation and the other one for tourist guidance, that would be better to combine them into one application and navigate users through most touristic path (rather than shortest path) and giving navigational instructions landmark-by-landmark (rather than junction-by-junction).

In both mentioned challenges of pedestrian navigation systems; seamless indoor/outdoor positioning and non-turn-by-turn navigational instruction giving, landmarks can help. It is possible to calculate position of a user relative to landmarks' positions. In Landmark-based positioning, positions of user are sensed or calculated respect to landmarks. Relative position of users' devices can be sensed using ultrasound (Holm, 2009), dead-reckoning (Etienne and Séguinot, 1993), collaborative positioning techniques (Lee et al., 2012) or inertial sensors (Vepa, 2011) mostly. Since landmarks can be detected and labeled indoors and outdoors, both, (see figure 1) it is possible to have positions of users seamlessly (Millonig and Schechtner, 2005).

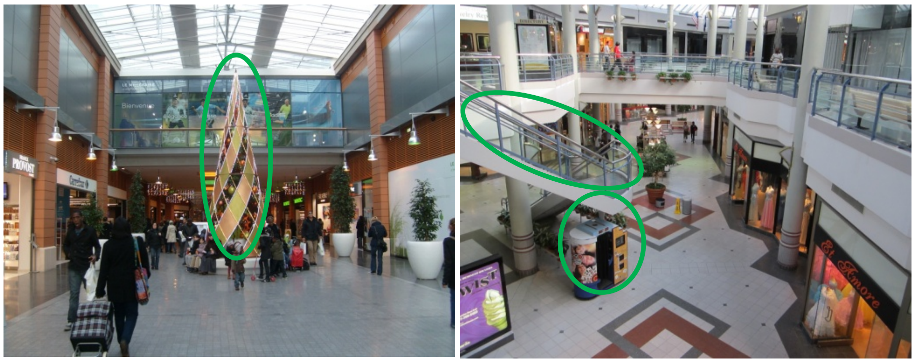


Fig. 1. Indoor landmarks

In addition, it is possible to provide users with the navigational instructions landmark-by-landmark rather than turn-by-turn. This way of navigating is called landmark-based navigation (May et al., 2005). Land mark based navigation is a kind of navigation service in which users are provided with navigational instructions, such as turn right, go straight, turn left, etc. whenever they approach each landmarks (Fang et al., 2011). One of the most important advantages of landmark-based navigation is making user sure that they are on the right way which is one of the tourists' preferences since usually they are not very familiar with the area they are visiting. They may not get lost, since they are seeing the very landmark which was used as a part of navigational instruction. From another point of view this approach is more suitable for tourists and visitors since it is possible to add some descriptive information or image of landmarks while navigating, so they will see more while visiting an area and also being navigated too at the same time (Basiri et al, 2013), (Basiri et al, 2014).

In order to implement such application, a graph database is used. By nature LBS applications and specially navigation applications need to be highly responsive in real time or near real time. In addition, users of such applications usually include huge number of users who receive information with high volume, such as video or image of landmarks so LBS application must provide appropriate scalability and performance measures. In this case, many LBS applications resort to proprietary network processing technologies. Most conventional and some modern storage systems such as Relational Database Management Systems (RDBMS) usually build a network representation on top of the physical storage model and keep the network structure in the main memory. In addition, some of the network processing technologies process widely used network analysis and store the results beside the network data itself. However when additional network elements (such as new land marks or new roads) have to be added to the data, all the representation and built network must be rebuilt and recompiled. This recompilation process would be serious processing task especially when the newly added elements have lots of connection to the existing elements.

In order to prevent facing with such issues, it is better to store connected data in their natural representation and this is where the graph database comes into play (storing data elements as well as their relationships as graphs in the graph database). In summary, graph databases don't need a mapping layer between physical storage layer and application logic layer. The mentioned integration of storage and application layer also results in flexibility for handling consistency of the data when the size of data is very large and as a result cannot be stored in main memory of single server.

Section 2 describes tourist navigation service and related requirements. Section 3 explains graph databases and their use in tourist navigation services. And in the section 4 implementation of the application is shown.

2 Landmark-Based Pedestrian Navigation

A pedestrian has several possible navigational strategies to find a desired goal (Redish, 1999); the individual has no information and is forced to search randomly (random navigation), the individual moves towards a visible cue which leads to the arrival point (taxon navigation), The individual follows a fixed motor program (praxic navigation, e.g. "turn left after 200 meters, then turn right after 150 meters"), the individual associates directions with visual cues (route navigation, e.g. "turn left at the church") and when the individual forms a mental representation of the surroundings and is able to plan routes between any locations within the area (locale navigation).

Landmarks can have an important role in random navigation, taxon navigation, route navigation and local navigation. Several researchers in the field of spatial cognition assert that navigating humans rely on three forms of spatial knowledge: landmark, route and survey knowledge (Siegel and White, 1975), (Werner et al., 1997). Exploring an unfamiliar environment, pedestrians first notice outstanding objects or structures at fixed locations. These unique objects or places are easy to recognize and can be kept in memory without difficulty (Schechtner, 2005).

This shows the importance of the meaning of landmarks for human navigational tasks. Landmarks are stationary, distinct and salient objects or places, which serve as cues for structuring and building a mental representation of the surrounding area. Any object can be perceived as a landmark, if it is unique enough in comparison to the adjacent items. The importance of landmarks for pedestrian navigation and wayfinding instructions is proved by many researches (Hung,. 2012), (Michon and Denis, 2001), (Denis, 2003), (Basiri et al, 2012) and (Raubal and winter, 2002).

This section landmark and landmark-based navigation concepts and related definitions are explained. Since in landmark-based navigation firstly landmarks should be stored in a database and then it becomes possible to localize users or provide them with landmark-based navigational instruction, firstly landmark definition and attributes are discussed in more details. Then landmark-based positioning and landmark-based path finding which are two important components of a landmark-based navigation service are explained.

2.1 What Is a Landmark?

A landmark can be defined as anything which is easily recognizable, such as a monument or a building. Landmarks are one of the interests of tourists probably due to notable physical features or historical significance. Landmarks are often used for casual navigation by ordinary people, such as giving directions.

In urban studies as well as in geography, a landmark is furthermore defined as an external point of reference that helps orienting in a familiar or unfamiliar environment (Lynch, 1960), (Schoier, 2012). Landmarks are also used in verbal route instructions. These two properties of landmarks; being used as references to orient objects and being used in verbal route instructions, are very important and potentially helpful in navigation systems and services.

Landmarks can be geometric shapes, and they may include additional information (e.g., in the form of bar-codes and geo-tags). In general, landmarks have a fixed and known position, relative to which users can localize themselves. Landmarks' data must be stored in a database to be used in landmark-based positioning.

Landmarks should be carefully chosen to be easy to identify; for example, a large building has got priority to a small one. A feature which has got sufficient contrast to the background is a good option to be considered as landmark since its image would be recognizable to users. Such objects have to possess a certain saliency, which makes them remarkable and distinctive. So the surrounding area determines the characteristics a point must have to be perceived as a landmark (e.g. a shopping center may not be very outstanding in urban areas, but becomes a salient landmark when being situated in a rural village). In the third section, the process of landmark extraction is explained in more detail.

After extracting important features as landmarks, best path between current location of user and selected destination should be found. This make the application enable to provide users with navigational instructions on the calculated route. Next subsection is focused on landmark-based localization. Then landmark-based path finding algorithm is explained.

2.2 Landmark-Based Positioning

Positioning is one of the most important components of any navigation system. Tourists and in general Pedestrians need seamless indoor and outdoor wayfinding assistance, so the system needs reliable and accurate positioning techniques in the situation where Globe Navigation Satellite Systems (GNSS) signals are not available. Unlike cars, tourists can easily enter buildings or even their destination is a roofed area such as gallery or museum; so an ideal navigation system should work seamlessly in and out of doors. This paper discusses landmark-based positioning technique which can be implemented as a seamless indoor/outdoor positioning solution since landmarks are available both indoors and outdoors. One of the advantages of using landmarks in tourist navigation services is being sure about availability of landmarks around users since the purpose of their visit is seeing touristic features which usually can be considered as a landmark because of their uniqueness.

In general, two main categories of landmark-based positioning techniques can be imagined; image-based and non-image-based techniques. Some examples of image based landmark positioning are QRCode-based positioning (Basiri et al, 2014) and photo-based positioning. Examples of non-image-based positioning are Radio Frequency Identification (RFID) and Bluetooth network positioning.

This paper focuses on image-based landmark positioning techniques since most mobile phones are equipped with cameras so less hardware requirements are needed. RFID tags and Bluetooth networks are not available ubiquitously. The hardware needs to be installed both on users' handheld devices (e.g. RFID readers) and also on the landmarks (e.g. RFID tags). This means extra cost for both service provider and users. In addition, in most image-based positioning approaches the computation and processing phase is done on the server side so such approaches have got less power consumption in comparison with Bluetooth positioning and similar techniques where users need to keep their mobile phones' Bluetooth on all the time. In addition, many landmarks are historically registered features, or absolutely huge objects. In such cases, it is almost impossible to install or affix a tag or any signal transmitter for positioning purposes. Being much cheaper, having less power consumption and ability of being used using available mobile devices may make image-based positioning more accepted by users.

In Image-based or camera-based positioning user can be viewed, identified and tracked by a network of cameras (such as CCTVs). Another approach which is categorized in image-based positioning techniques is user takes a photo of a registered landmark and then send/upload it for further image processing, feature matching and finally to find his/her location. Since second approach needs less hardware infrastructure, it has been implemented in this project. In this part image-based positioning using mobile devices' cameras is explained in more detailed.

In image-based positioning using mobile devices' cameras, user can take a photo of a registered landmark and then send/upload it for further image processing and feature extraction to find relative location respect to the landmark. Usually this process is handled on server-side machines. Based on feature extraction and image matching techniques, it is possible to find landmark of which the photo has taken. As landmarks are usually unique and distinctive objects, the feature matching process is usually accurate enough for positioning purposes. Then scale of the photo and angel of view (rotation) can be easily calculated since the absolute sizes of different façades of the landmark are stored in a database. Based on scale and angel of view, relative location of user respect to the landmark can be calculated.

Positioning of a user based on camera positioning system has two main steps generally; image processing to identify the landmark (feature detection) and finding scale and rotation (localization). In the first step, i.e. landmark detection, uniqueness of landmark make it much easier to find matching image in the database. Then using actual size and shape of the corresponding landmark which has been stored in a database, it is possible to calculate scale and rotation of photo taken by user. This piece of information is used to calculate relative position of user respect to the landmark. Since absolute positions of landmarks are available in the database, absolute position of user can easily be calculated and used in path finding and navigation service. In next subsection, landmark-based path finding using calculated location of user and specified destination is explained.

2.3 Landmark-Based Path Finding

In order to find the shortest path between two points, path finding algorithms are looking for minimum distance traversed, or to find the fastest path, the route with minimum time is preferable. In the tourist navigation the most touristic path should be calculated whose output is a path with maximum landmark features on the way to the destination. Users of a tourist navigation application want to see monuments and landmarks which may need deviation from the shortest path. Landmark-based path finding algorithm is providing more attractive and at the same time more reliable path. Since users are seeing more landmarks on the way, they are surer that they are taking the right way.

In landmark-based path finding we look for a path which traversing less distance to get the destination passing more landmarks on the way. So landmark-based path finding algorithms are trying to maximize the result of:

$$\text{Number of landmarks of each edge / length of that edge.} \tag{1}$$

The same shortest path algorithm can be implemented but the cost or distance, which is usually called weight and supposed to be optimized, will be replaced by the value of *number of landmark/length*. Based on landmark-based path finding algorithm, the route is calculated then it is possible to navigate user providing image, informatics text of landmarks can be viewed from this route (Basiri et al., 2014).

As it explained in the section 2.2, the application uses image-based positioning technique to localise users. Also the tourist navigation application sends some descriptive information as well as image of landmarks to be seen on the way.

Exchange of such high volume data is an issue which should be solved to maintain responsiveness of the application. Next section explains how graph database can handle such issue.

3 Graph Databases

Highly connected network data are at the heart of most LBS applications. Elements in Location-based networks, road networks, junctions, landmarks, moving users and in general any producer and consumer network constitute highly connected data. Storage and processing of such highly connected and interrelated data can be problematic for most conventional and modern storage systems such as Relational Database Management Systems (RDBMS) and most types of NoSQL databases. The mentioned issue gets worse if the LBS applications need to provide responsive interactivity to the end users.

By nature LBS applications need to be highly responsive in real time or near real time. In addition, since users of tourist navigation applications include huge number of users who send and receive high volume data such as images, such application must provide appropriate scalability and performance measures. In this case, many navigation applications resort to proprietary network processing technologies. The mentioned technologies usually build a network representation on top of the physical storage model and keep the network structure in the main memory. In addition, some of the network processing technologies process widely used network analysis and store the results beside the network data itself. However when additional network elements (such as new landmarks or new roads) have to be added to the data, all the representation and built network must be rebuilt and recompiled. This recompilation process would be serious processing task especially when the newly added elements have lots of connection to the existing elements. In order to prevent facing with such issues, it is good idea to store connected data in their natural representation and this is where the graph database comes into play (storing data elements as well as their relationships as graphs in the graph database). In summary, graph databases don't need a mapping layer between physical storage layer and application logic layer. The mentioned integration of storage and application layer also results in flexibility for handling consistency of the data when the size of data is very large and as a result cannot be stored in main memory of single server.

Graph theory was pioneered by Euler in the 18th century, and has been actively researched and improved by mathematicians, sociologists, anthropologists, and others ever since. However, it is only in the past few years that graph theory and graph thinking have been applied to information management. The graph databases can be categorized as one of several models of modern NoSQL databases. All the other models of modern NoSQL databases lack capabilities to handle huge volume of highly connected data (Amirian et al. 2013). At the other hand, the conventional RDBMS systems can handle relationship, but they are not designed with highly connected data in mind.

This section tries to explain the use of graph databases for handling highly connected data (landmark networks) in comparison with relational (SQL) and NoSQL databases. In this paper relational DBMS are called SQL databases.

3.1 SQL Database and Handling Highly Connected Data

It has been more than thirty years since relational DBMSs (SQL DBMSs) became the major solution for storing all kinds of data in many types of applications. They manage data using relational algebra and relational calculus as their theoretical foundation. They use tables, relationships, keys and Structured Query Language (SQL) to perform all sorts of functions with data. One of the important advantages of SQL systems is the normalization process which ensure about storage of data in separate tables and only once in whole database. They usually are the best solutions when the schema of data is fixed and predefined. In other words, RDBMSs are ideal solutions to managing structured data. The SQL systems can be effectively used in many common Geospatial-related workflows. Since they support transaction and locking features, they provide robust consistency and backend for enterprise GIS systems. Usually geospatial data have a fixed schema and in most cases they are not used in isolation. That is a join of two or more datasets and connecting data through spatial operations is needed in most GIS workflows. For this reason managing fixed schema geospatial data with limited connectivity and using them in GIS workflows can usually be done effectively through SQL systems.

The SQL databases handle connected data using relationship and they retrieve connected data using joins. However joins are one of the most computationally expensive processes for SQL databases (Amirian et al., 2010). In most cases, joins are the bottleneck of SQL databases. In order to avoid many joins (which is needed in handling highly connected data in SQL databases), denormalization process can be used to store data items several times. But there are several issues associated with denormalization process especially with providing consistency in large datasets (Amirian et al., 2013). In addition to issues related to handling highly connected data, some other problems arise with SQL systems when scalability is needed by adding more servers and technologies to bind them together. With more loads on a SQL system, vertical partitioning, denormalization and removal of the relational constraints comes into play. In summary, to achieve high scalability in SQL systems the normalized relational model of data storage has to be compromised and deviated from relational model.

3.2 NoSQL Databases and Handling Highly Connected Data

The NoSQL (Not only SQL) DBMSs are a broad class of DBMSs identified by non-adherence to the SQL (relational) model. There are different types of NoSQL databases, each with distinct set of characteristics but they all can deal with large amounts of (semi-structured and unstructured) data and are able to support a large set of read and write operations and they are designed with scalability and distribution of data in mind. For this reason NoSQL and relational models are not in contrast with each other rather they complement each other. The most widely accepted taxonomy of NoSQL databases are: key-value, document, columnar and graph (Tiwari, 2011).

The key-value database is the simplest type of NoSQL databases. As the name implies, this type of database stores schema-less data using keys. The key is usually a string and the stored values can be any valid type such as a primitive programming data type (string, integer, etc.) or a BLOB (Binary Large Object) without any predefined schema. It provides a simple API to access stored data (Fowler and Sadalage, 2012). In most cases, this type of NoSQL database solution provides very little functionality beyond key-value storage. There is no support for relationships (Xiang et al. 2010). In terms of concurrency, they usually provide eventual consistency but don't provide optimistic concurrency especially in highly scalable environments. Queries are just limited to accessing values using keys but since there is one request to access the value, the queries are executed quickly. Transactions are limited to a single key. The database contains no semantic model. In other words, the client is in charge of interpreting and understanding values. Key-value databases can be utilized to store geospatial data but their complexity hinders spatial searches especially for polylines and polygons. For this reason, it needs to be spatially indexed for fast data retrieval which, in most cases, gives lower performance than a relational database. Many researchers and developers recommend using indexing techniques such as grids or tiles, quadkeys, space filling curves and similar approaches. In summary, key-value data stores are ideal for inserting, deleting and searching huge amount of data items using their unique identifiers (keys). In the case of highly connected data or spatial searches they are not good solutions.

A document database in its simplest form is a key-value database in which the database understands its values (Hecht and Jablonski, 2011). In other words, values inside the database are based on predefined formats such as XML, JSON or BSON (Binary JSON). This feature of document databases provides many advantages over key-value databases. Instead of putting too much logic in the application layer, many operations can be done by the database itself. Queries in this type of NoSQL databases are quite flexible and some document databases even have their own query languages. Similar to key-value databases, there is no need to adhere to a predefined schema to insert data. There is only limited support for relationships and joins as each document is stand alone. However, more concurrency options, such as optimistic concurrency and eventual consistency are available (Chang, 2006). Transaction integrity is supported for one document or document fragment .

The document databases can be used for managing geospatial data more effectively than key-value databases. Since geospatial data inside the document database can be retrieved using flexible queries, they can be used for storing and managing geospatial data in multiple use cases. In fact many document databases support geospatial data natively or through extensions. Some applications of LBS such as proximity queries can be efficiently implemented using these document databases. As mentioned before, relationships and joins are not supported the way they are supported in relational databases. Often in common GIS workflows relationships and joins have to be used. However, the document-oriented nature of the system has some major effects on the way that data can be retrieved. For example if the application needs data items from the same documents it would be very fast. However whenever the data items are part of different types of documents there is no efficient approach to reduce the number of index-lookups. In summary indexing is just based on documents and there is no notion of relationships in document databases.

The columnar (or column family) databases store data in set of columns and distribute data based on columns (rather than rows in SQL databases). The column is the smallest unit of data and it is a triplet that contains a key, value and timestamp (Hecht and Jablonski, 2011). Columnar databases store all values beside the name of the columns and stores null values simply by ignoring the column. Usually, related columns compose a column-family. All the data in a single column family will be stored on the same physical set of files (Xiang et al., 2010). This feature provides higher performance for search, data retrieval and replication operations. A super column is a column that contains other columns but it cannot contain other super columns ((Hecht and Jablonski, 2011). Most columnar databases use a distributed file-system to store data to disk and so provide a horizontally scalable system. In fact columnar databases are designed to run on a large number of machines. Queries in this type of NoSQL databases are limited to keys and in most cases they don't provide a way to query by column or value. By limiting queries to just keys, columnar databases ensure that procedure to find the machine containing actual data is quite fast. There is no join capability and, as in other types of NoSQL databases, there is limited support for transactions.

Columnar databases are ideal for storing huge amounts of data when high availability is needed. Similar to document databases, there are many columnar databases which support the management and simple analysis of geospatial data. Any Geospatial Information System (GIS) related application which needs heavy data insertion and fast data retrieval with simple queries can efficiently make use of columnar databases. As an example, an Automatic Vehicle Location (AVL) application which needs to track the location of many vehicles simultaneously can store the incoming data from vehicles in a columnar database and respond to queries efficient. In summary this type of databases doesn't support relationships and in order to handle highly connected data, there is a need for mapping layer to create network structure (which is not efficient).

As the name implies graph databases are based on graph theory and employ nodes, properties and edges as their building blocks. The nodes and edges can have properties. In the graph databases various nodes might have different properties. The graph databases are well suited for data which can be modeled as networks such as road networks. Their main feature is the fact that each node contains a direct pointer to its adjacent node, so no index lookups are necessary for traversing connected data (which is really valuable). As a result they can manage huge amount of highly connected data since there is no need for expensive join operations. Some of graph databases support transactions in the way that relational databases support them. In other words the graph database allows the update of a section of the graph in an isolated environment, hiding changes from other processes until the transaction is committed. Geospatial data can be modeled as graphs. Since graph databases support topology natively, topological relationship (especially connectivity) between geospatial data can be easily managed by this type of NoSQL databases. In most GIS workflows, topological relationships play a major role. In addition, since graph databases are ideal for managing data with evolving schema, they can be effectively used in Volunteered Geographic Information (VGI) and crowd sourcing applications. Also, graph databases are the best choice for managing huge linear networks (such as roads) and for routing and navigation applications (Amirian et al., 2013). In addition since each edge in graph database can have different set of properties, they provide flexibility in traversal of network based on various properties. For example it is possible to combine time; distance, number of points of interest and user preferences in finding best path and the mentioned path would be unique for each user. This allows us to optimize number of landmarks/length of path.

4 Implementation

The landmark-based navigation system is intended to provide navigation services to tourist in Maynooth, a small town where National University of Ireland is located. From general point of view, there are mainly four steps in our landmark-based navigation system design. Firstly, Landmarks must be defined, extracted and stored in a database. In this step both geometrical and non-geometrical characters of each landmark is stored in the graph database.

Another component of this system is positioning component, see figure 2. The positioning component is responsible for calculating users' positions using image-based positioning using photo taken by users' cameras where GPS signals are not available wherever GPS signals are available, then GPS gives the location of users. In both situations; availability and unavailability of GPS signals, it is possible to use image-based positioning service however it is not recommended to use it where GPS signals are received. This is recommended to users to make requests as minimum as possible to reduce firstly user's devices' battery consumption and also network data exchange and secondly prevent any potential problem to positioning component.

Then based on a routing algorithm, the most touristic path is calculated. Finally, navigational instructions which help users to get their destination are provided using landmarks' information and photos on the way.

Using landmarks stored in the database, it is possible to find the best path and navigate users to get their destination. The architecture used to provide user with landmark-based navigational instructions is illustrated in figure 2. The landmark-based navigation system implemented in National University of Ireland is consists of four main components; positioning component, service and data database, navigation service calculation engine and users and their mobile devices.

Positioning component is responsible of calculation position of the user using GPS or image-based positioning techniques which explained previously and also tracking them. It delivers its output, position of the user, to the navigation service calculation engine.

The navigation service calculation engine uses user's position as input of two other services; first it calculates the best path using user's position and selected destination and secondly, based on user's position, it can calculate visible landmarks on the way. In order to do both of these tasks, the navigation service calculation engine needs to have access to the spatial database where landmarks' information, such as location, size, etc is stored. In this project, a graph data base is used. Elements in landmark-based navigation services are changing with quite high degree of frequency, since landmark which has got one of the essential role can be visible from one user but not another one. Processing of such highly connected and interrelated data can be problematic for most conventional and modern storage systems such as Relational Database Management Systems (RDBMS). Also when additional network elements (such as new land marks or new roads) have to be added to the data, all the representation and built network should not be rebuilt and recompiled over and over. In order to prevent facing with such issues, it is better to store connected data in their natural representation using a graph database.

The navigation service calculation engine uses landmarks, edges and nodes data to find more reliable path. This is calculated based on landmark-based path finding algorithm explained in previous section.

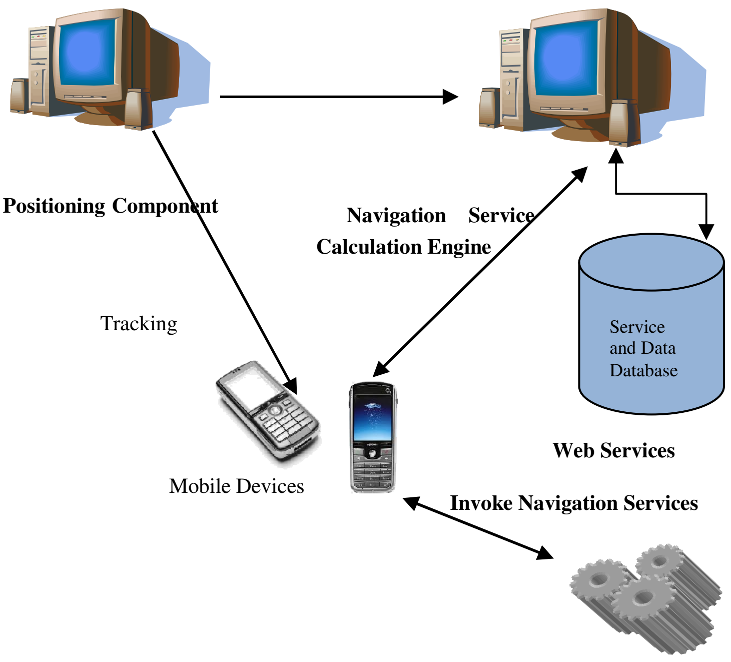
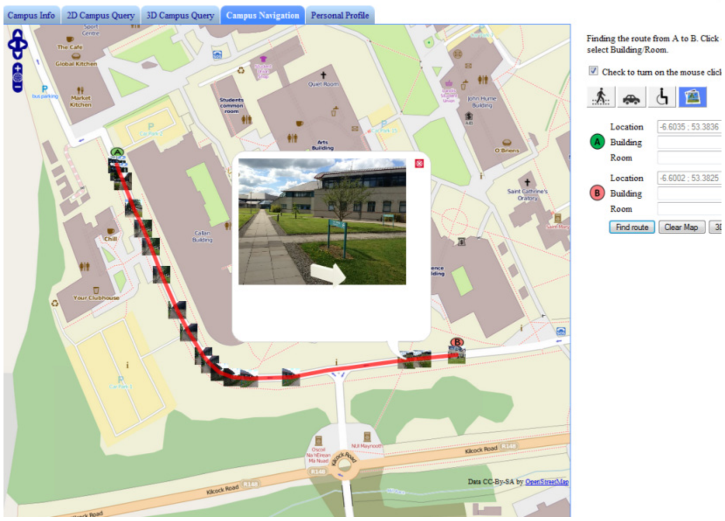

Positioning Component
Navigation Service Calculation Engine
Tracking
Service and Data Database
Mobile Devices
Web Services
Invoke Navigation Services

Fig. 2. NUIM Campus Landmark-based Navigation System Arcitecture

After route calculation, user should be provided with navigational instructions to follow calculated route. In this step, the navigation service calculation engine uses positional information provided by positioning engine and also data stored in the database such as information of landmarks to calculate the landmarks to be seen from user's location. Whenever user's location changes or new image-based positioning is requested by user, this process will be reseated and a new set of navigational instruction is provided.

1. Figure 3 shows, the web application interface of the system which provides image of landmarks as a part of navigation service. As it is shown in figure 3, three modes of travel; pedestrians, cars and wheelchair can use this service.



Finding the route from A to B. Click
select Building/Room.

☑ Check to turn on the mouse click

Location	-6.6035 ; 53.3836
Building	
Room	
Location	-6.6002 ; 53.3825
Building	
Room	

Fig. 3. Navigation services in eCampus web application

5 Conclusion

Landmarks are often used for casual navigation, such as giving directions, by ordinary people. Also they might be interests of tourists and visitors because of their uniqueness. So landmarks can be applied in tourist navigation services to navigate tourists and visitors to get their destinations and at the same time give additional information about landmarks passing by. Landmark-based navigation is a kind of navigation service in which positions of users are calculated based on the nearest landmark. Then based on current position of user which can be calculated using image of nearest landmark and selected destination, a landmark-based path finding algorithm, which calculates a route with maximum landmarks on the way and minimum distance passed, calculates most reliable path. Then users are provided with some information about interesting or important features, landmarks, around themselves to make sure they are on the correct way. In order to implement this application a graph database has been used to store landmark data and network information to avoid unavailability of the system due to high volume data exchange for positioning purpose. This paper explains these three steps in detail and implemented a landmark-based navigation application.

Acknowledgement. Research presented in this paper was funded by a Strategic Research Cluster grant (07/SRC/I1168) by Science Foundation Ireland under the National Development Plan. The authors gratefully acknowledge this support.

This work was financially supported by EU FP7 Marie Curie Initial Training Network MULTI-POS (Multi-technology Positioning Professionals) under grant nr. 316528.

References

2. Amirian, P., Alesheikh, A.A., Basiri, A.: Standard-based, interoperable services for accessing urban services data. Computer Environment and Urban Systems 34(4), 309–321 (2010)
3. Amirian, P., Winstanley, A.C., Basiri, A.: Using Graph databases in LBS applications: Storing and Processing Navigational and Tracking data. In: Mobile Gehnt, Belgium (2013)
4. Basiri, A., Amirian, P., Winstanley, A.C.: The Use of Quick Response (QR) Codes in Landmark-Based Pedestrian Navigation. International Journal of Navigation and Observation (2014)
5. Basiri, A., Amirian, P., Winstanley, A.C., Kuntzsch, C., Sester, M.: Uncertainty han-dling in navigation services using rough and fuzzy set theory. In: Proceedings of the Third ACM SIGSPATIAL International Workshop on Querying and Mining Uncertain Spatio-Temporal Data, pp. 38–41 (2012)
6. Basiri, A., Winstanley, A.C., Amirian, P.: Landmark-based pedestrian navigation. In: 21st GIS Research UK (GISRUK) Conference, UK (2013)
7. Chang, F., Dean, J., Ghemawat, S., Hsieh, W., Gruber, R.: Bigtable: A distributed stor-age system for structured data. In: Seventh Symposium on Operating System Design and Implementation (2006)
8. Tom, A., Denis, M.: Referring to Landmark or Street Information in Route Directions: What Difference Does It Make? In: Kuhn, W., Worboys, M.F., Timpf, S. (eds.) COSIT 2003. LNCS, vol. 2825, pp. 362–374. Springer, Heidelberg (2003)
9. Elias, B.: Determination of Landmarks and Reliability Criteria for Landmarks. Technical Paper, ICA Commission on Map Generalization, 5th Workshop on Progress in Automated Map Generalization. IGN, Paris, France (2003)
10. Etienne, S., Séguinot, V.: Navigation by Dead Reckoning and Local Cues. Journal of Navigation 46, 364–370 (1993), doi:10.1017/S0373463300011802.
11. Fang, Z., Li, Q., Zhang, X., Shaw, S.L.: A GIS data model for landmark-based pe-destrian navigation. International Journal of Geographical Information Science (2011), doi:10.1080/13658816.2011.615749
12. Fontaine, S., Denis, M.: The Production of Route Instructions in Underground and Urban Environments. In: Freksa, C., Mark, D.M. (eds.) COSIT 1999. LNCS, vol. 1661, pp. 83–94. Springer, Heidelberg (1999)
13. Fowler, M., Sadalage, P.: NoSQL distilled: a brief guide to the emerging world of polyglot persistence. Addison-Wesley Publication (2012)
14. Gaisbauer, C., Frank, A.U.: Wayfinding Model for Pedestrian Navigation. In: The AGILE International Conference on Geographic Information Science, pp. 1–9 (2008)
15. Hecht, R., Jablonski, S.: NoSQL Evaluation A Use Case Oriented Survey. In: International Conference on Cloud and Service Computing, pp. 336–341 (2011)
16. Hansen, R., Wind, R., Jensen, C.S., Thomsen, B.: Seamless Indoor/Outdoor Positioning Handover for Location-Based Services in Streamspin. In: Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, pp. 267–272 (2009)

17. Holm, S.: Hybrid ultrasound-RFID indoor positioning: Combining the best of both worlds. In: IEEE Int. Conf. RFID, Orlando, FL, pp. 155–162 (2009)
18. Hung, J.C.: The smart-travel system: utilising cloud services to aid traveller with personalised requirement. IJWGS 8(3), 279–303 (2012)
19. Karimi, H.: Universal Navigation on Smartphones. Springer (2011) ISBN-10: 1441977406
20. Lee, J.K., Grejner-Brzezinska, D.A., Toth, C.: Network-based Collaborative Navigation in GPS-Denied Environment. Journal of Navigation 65, 445–457 (2012), doi:10.1017/S0373463312000069.
21. Li, X., Wang, J., Li, T.: Seamless Positioning and Navigation by Using Geo-Referenced Images and Multi- Sensor Data. Journal of Sensors 13(7), 9047–9069 (2013)
22. Lynch, K.: The image of the city, p. 48. MIT Press (1960)
23. May, A.J., Ross, T., Bayer, S.H.: Incorporating Landmarks in Driver Navigation System Design: An Overview of Results from the REGIONAL Project. Journal of Navigation 58, 47–65 (2005), doi:10.1017/S0373463304003054.
24. Michon, P.-E., Denis, M.: When and Why Are Visual Landmarks Used in Giving Directions? In: Montello, D.R. (ed.) COSIT 2001. LNCS, vol. 2205, pp. 292–305. Springer, Heidelberg (2001)
25. Millonig, A., Schechtner, K.: Developing Landmark-based Pedestrian Navigation Systems. In: Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems, pp. 196–202 (2005) 0-7803-9215-9/05
26. Pielot, M., Boll, S.: "In Fifty Metres Turn Left": Why Turn-by-turn Instructions Fail Pedestrians. In: Haptic, Audio and Visual Interfaces for Maps and Location Based Services (2010)
27. Raubal, M., Winter, S.: Enriching Wayfinding Instructions with Local Landmarks. In: Egenhofer, M., Mark, D.M. (eds.) GIScience 2002. LNCS, vol. 2478, pp. 243–259. Springer, Heidelberg (2002)
28. Redish, D.: Beyond the cognitive map: from place cells to episodic memory. MIT, Cambridge (1999)
29. Schechtner, M.K.: Developing Landmark-based Pedestrian Navigation Systems. In: Proceedings of the 8th International IEEE Conference on Intelligent Transportation Systems, Vienna (2005)
30. Siegel, W., White, S.H.: The Development of Spatial Representations of Large-scale Environments. In: Reese, H.W. (ed.) Advances in Child Development and Behaviour, vol. 10, pp. 9–55. Academic Press, New York (1975)
31. Schoier, G., Borruso, G.: Spatial Data Mining for Highlighting Hotspots in Personal Navigation Routes. IJDWM 8(3), 45–61 (2012)
32. Tiwari, S.: Professional NoSQL. Wrox Publication (2011)
33. Vepa, R.: Ambulatory Position Tracking of Prosthetic Limbs Using Multiple Satellite Aided Inertial Sensors and Adaptive Mixing. Journal of Navigation 64, 295–310 (2011), doi:10.1017/S0373463310000494
34. Werner, S., Krieg-Brückner, B., Mallot, H., Schweizer, K., Freksa, C.: Spatial Cogni-tion: The Role of Landmark, Route and Survey Knowledge in Human and Robot Navigation. In: Jarke, M., Pasedach, K., Pohl, K. (eds.) Informatik aktuell, pp. 41–50. Springer, Berlin (1997)
35. Xiang, P., Hou, R., Zhou, Z.: Cache and consistency in NoSQL. In: 3rd IEEE International Conference on Computer Science and Information Technology, pp. 117–120 (2010)